

CSE 140, Lecture 2

Combinational Logic

CK Cheng

CSE Dept.

UC San Diego

Combinational Logic Outlines

1. Introduction

- Scope
- Boolean Algebra (Review)
 - Definition
 - Duality
 - AND/OR Gates
 - Expressions vs Circuits
- Handy Tools
 - DeMorgan's Theorem
 - Consensus Theorem
 - Shannon's Expansion

2. Specification

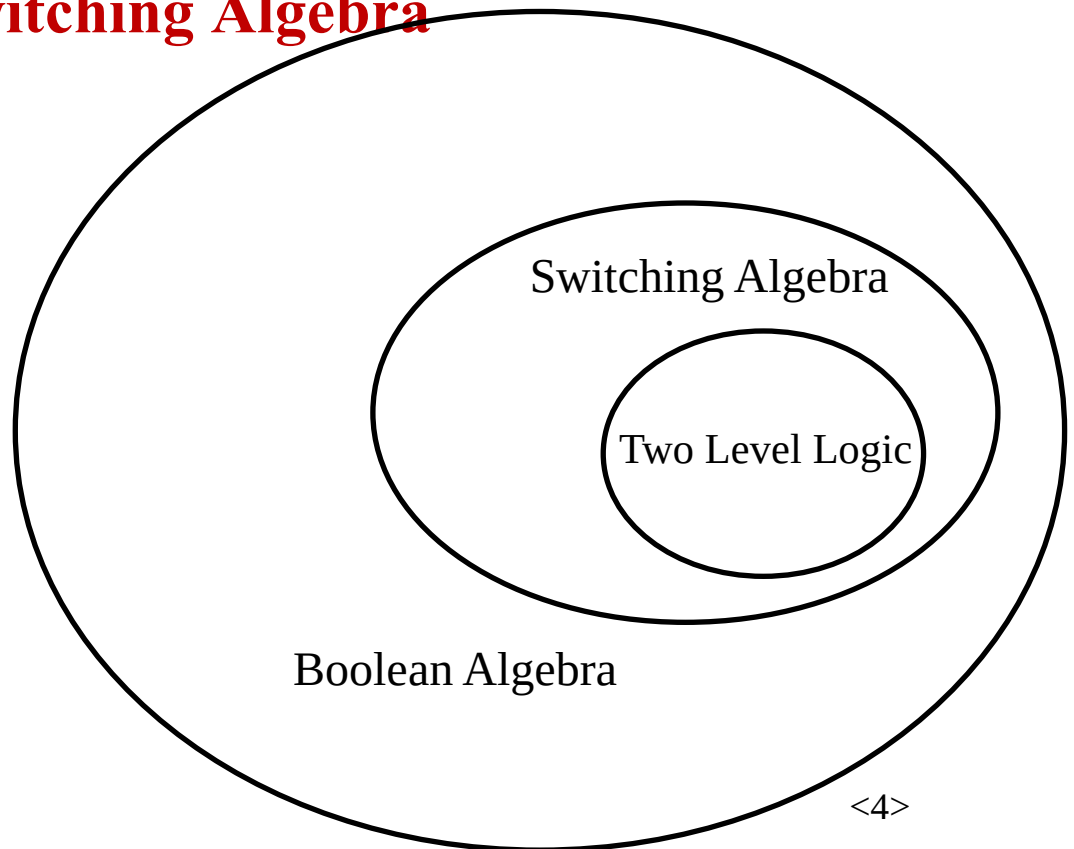
3. Synthesis

1.1 Combinational Logic: Scope

- Description
 - **Language:** e.g. C Programming, **BSV**, Verilog, VHDL
 - **Boolean algebra**
 - **Truth table: Powerful engineering tool**
- Design
 - Schematic Diagram
 - Inputs, Gates, Nets, Outputs
- Goal
 - Validity: **correctness**, turnaround time
 - Performance: power, timing, **cost**
 - Testability: yield, diagnosis, robustness

Scope: Boolean algebra, switching algebra, logic

- Boolean Algebra: multiple-valued logic, i.e. each variable have multiple values.
- Switching Algebra: binary logic, i.e. each variable can be either 1 or 0.
- **Boolean Algebra $\neq$ Switching Algebra**



Scope: Switching Algebra (Binary Values)

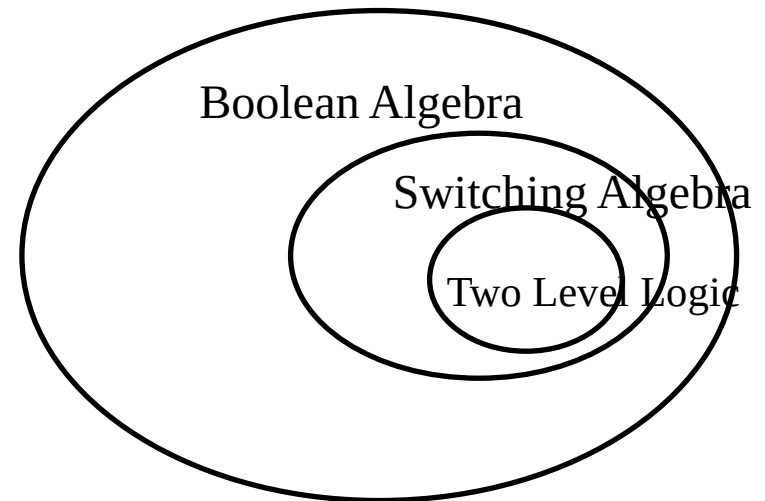
- Typically consider only two discrete values:
 - 1's and 0's
 - 1, TRUE, HIGH
 - 0, FALSE, LOW
- 1 and 0 can be represented by specific voltage levels, rotating gears, fluid levels, etc.
- Digital circuits usually depend on specific voltage levels to represent 1 and 0
- *Bit: Binary digit*

Scope: Levels of Logic

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

Features of Digital Logic Design

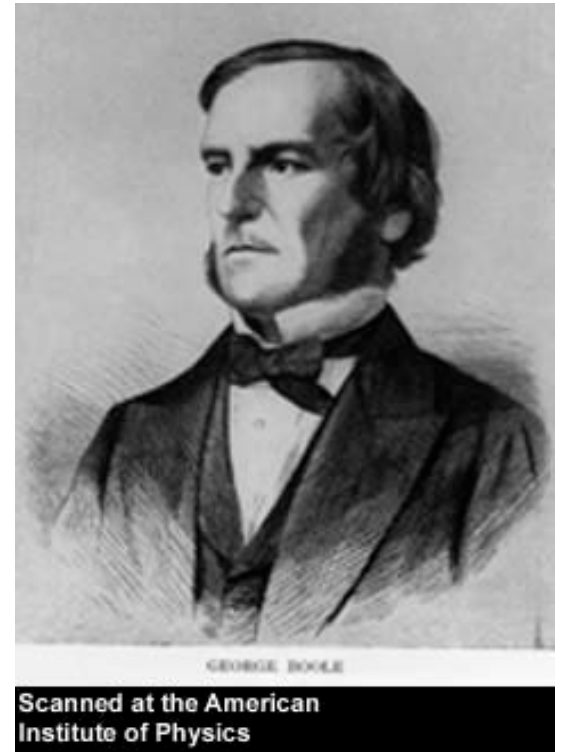
- Multiple Outputs
- Don't care sets



1.2 Boolean Algebra (Review)

George Boole, 1815-1864

- Born to working class parents: Son of a shoemaker
- Taught himself mathematics and joined the faculty of Queen's College in Ireland.
- Wrote *An Investigation of the Laws of Thought* (1854): **systematize Aristotle's logic**
- 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 2-input operations, a 1-input operation $\bar{}$ (complement), and two distinct elements 0 and 1. Then B is called a Boolean algebra if the following axioms hold.

- Associative laws: $(a+b)+c=a+(b+c)$, $(a \cdot b) \cdot c=a \cdot (b \cdot c)$
- 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: $a+a'=1$, $a \cdot a'=0$

Review of Boolean Algebra: Duality

Associative laws	$(a+b)+c=a+(b+c)$	$(a \cdot b) \cdot c=a \cdot (b \cdot c)$
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	$a+a'=1$	$a \cdot a'=0$

Duality: We swap all operators between (+,.) and interchange all elements between (0,1).

For a theorem if the statement can be proven with the laws of Boolean algebra, then the duality of the statement is also true.

1.3 Switching functions: Operators and Digital Logic Gates

id	A	B	Y
0	0	0	0
1	0	1	0
2	1	0	0
3	1	1	1

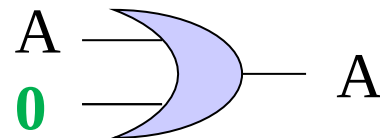
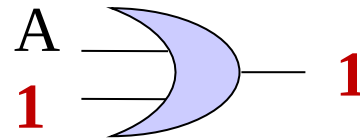
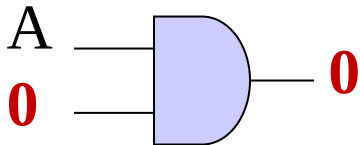
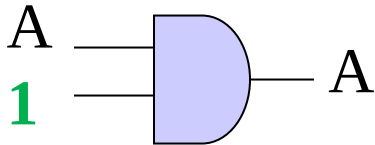
AND
 $Y=AB$

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

OR
 $Y=A+B$

A	Y
0	1
1	0

NOT
 $Y=A'$



Input 0 dominates Y
0 blocks the output
1 passes signal A

Input 1 dominates Y
0 passes signal A
1 blocks the output

1.3 Switching functions: Operators and Digital Logic Gates

id	A	B	C	Y
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	0
4	1	0	0	0
5	1	0	1	0
6	1	1	0	0
7	1	1	1	1

AND
 $Y=ABC$

**0 blocks
the output**
**1 passes
signal A**
**For AND,
only one
row is true
(minterm)**

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

OR
 $Y=A+B+C$

**0 passes
signal A**
**1 blocks
the output**
**For OR,
only one
row is false
(maxterm)**

1.3 Switching functions: Example on AND and OR

AND

$$Y = A'B'C$$

id	A	B	C	Y
0	0	0	0	
1	0	0	1	
2	0	1	0	
3	0	1	1	
4	1	0	0	
5	1	0	1	
6	1	1	0	
7	1	1	1	

OR

$$Y = A' + B' + C$$

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

Switching Expressions vs Logic Diagrams

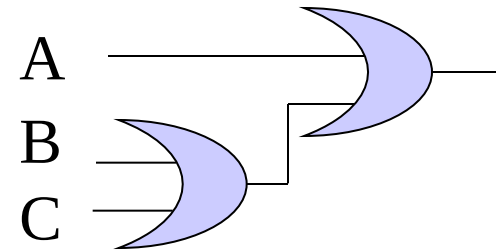
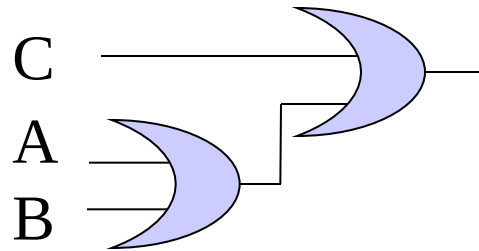
Switching expression is related to logic implementation

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

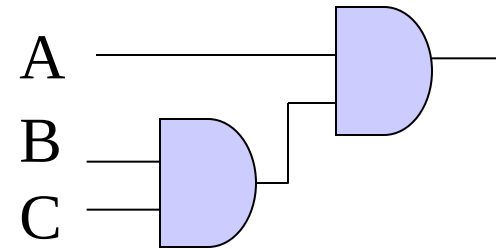
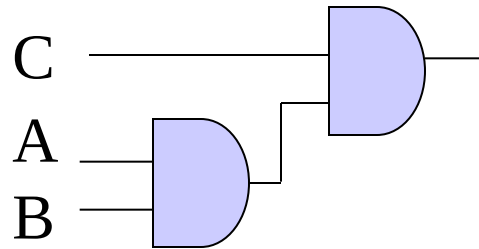
Laws and Logic Diagrams

Associativity Laws

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



$$(AB)C = A(BC)$$



Laws and Logic Diagrams

Identity Laws

$$A \cdot 1 = A$$

$$A + 1 = 1$$

$$A \cdot 0 = 0$$

$$A + 0 = A$$

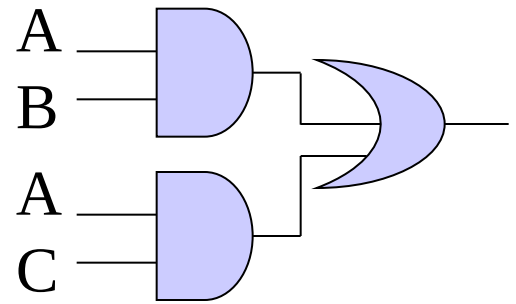
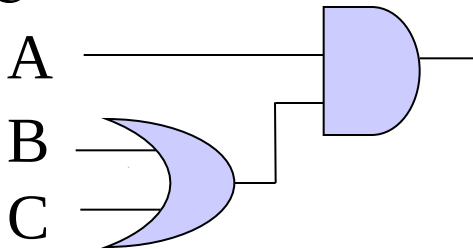
Complement Laws

$$A + A' = 1$$

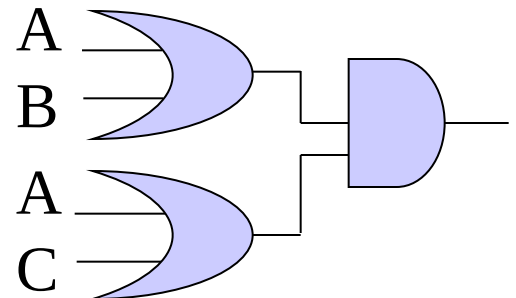
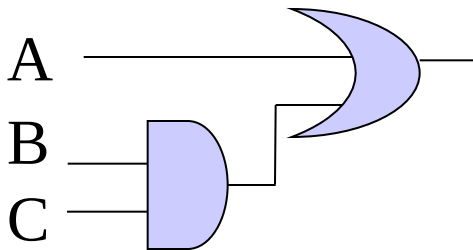
$$A \cdot A' = 0$$

Distributive Laws

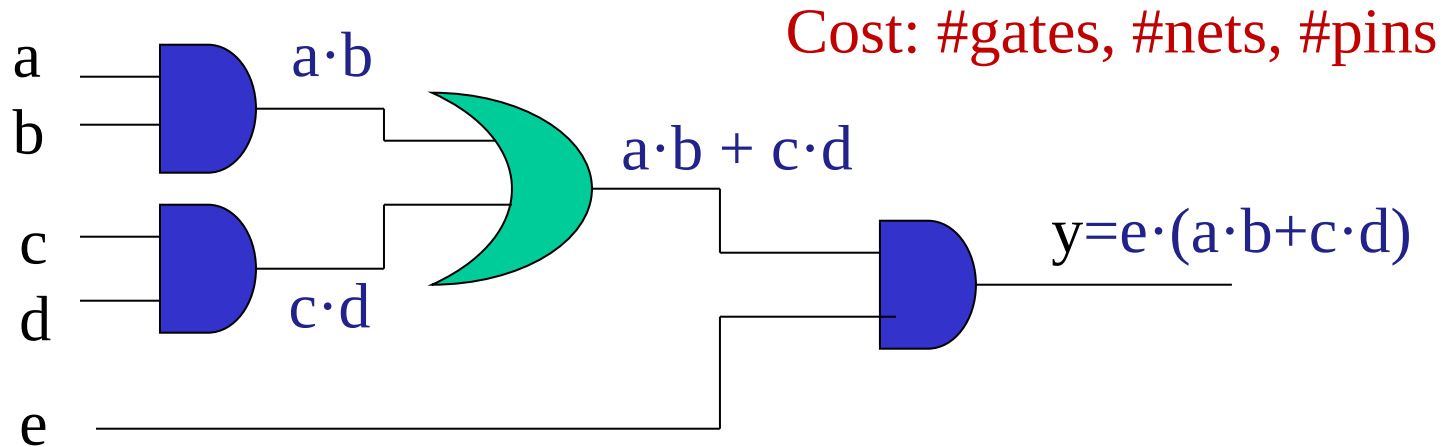
$$A \cdot (B + C) = A \cdot B + A \cdot C$$



$$A + B \cdot C = (A + B) \cdot (A + C)$$



Switching Expression and Logic



Schematic Diagram:

5 primary inputs

1 primary output

4 gates (3 ANDs, 1 OR)

9 signal nets

12 pins

Boolean Algebra:

5 variables

1 expression

4 operators (3 ANDs, 1 OR)

5 literals

Switching Expression and Logic

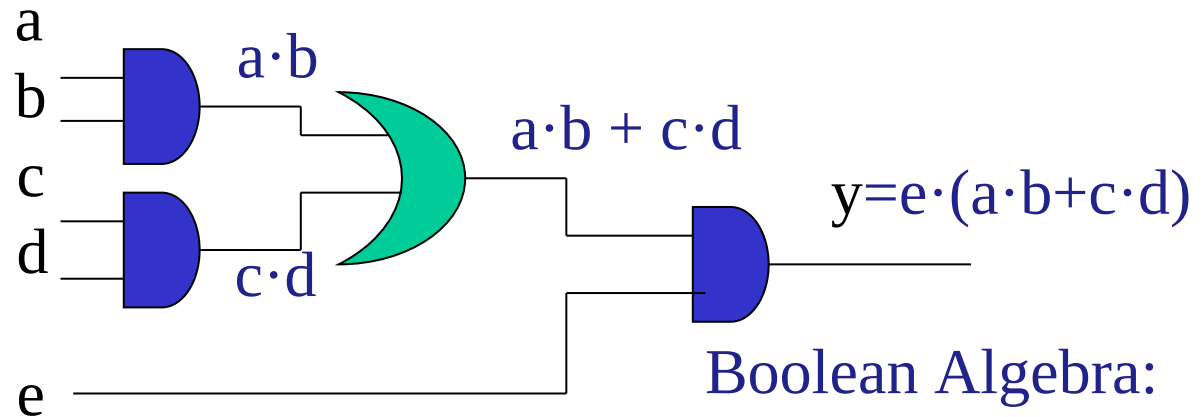
Schematic Diagram:

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. #variables

II. #operators

III. #variables + #operators

IV. #literals + 2 #operators - 1

Example: $f(a,b,c) = ab + a'c + a'b'$

Example: $f(a,b,c) = (a' + c)(a + b')(a + b + c)$

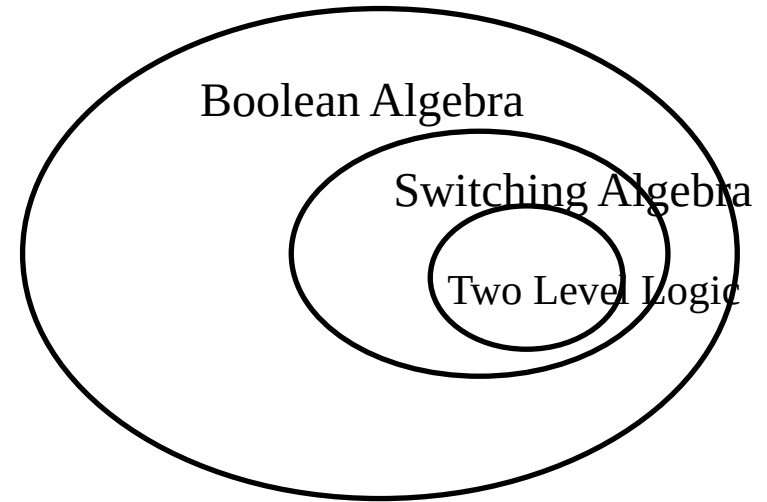
1.4 Handy Tools

Boolean Algebra

- DeMorgan's Law: Complements
- Consensus Theorem

Switching Logic

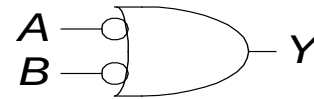
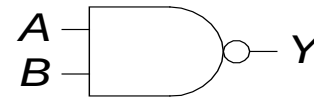
- Shannon's Expansion
- Truth Table
- Karnaugh Map (single output, two level logic)



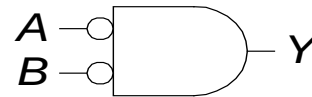
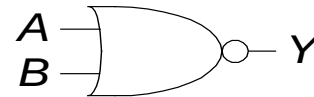
DeMorgan's Theorem and Digital Logic

T12. DeMorgan's Theorem $(A+B)' = A'B'$ $(AB)' = A' + B'$

- $Y = (AB)' = A' + B'$



- $Y = (A + B)' = A'B'$



DeMorgan's Theorem: 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

- $AB + AC + B'C$
 $= AB + B'C$

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

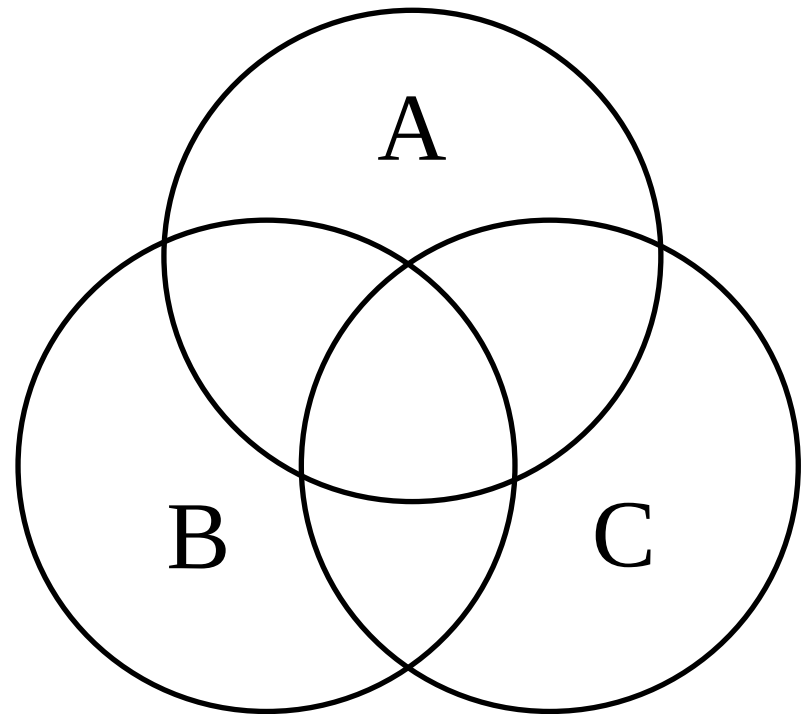
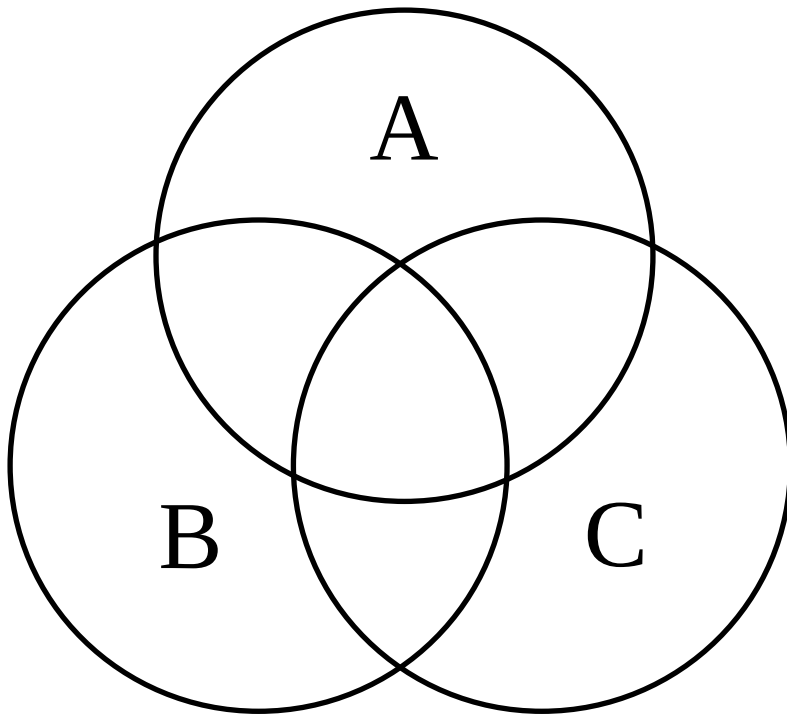
The consensus of AB , $B'C$ is: ?

Exercise: to prove the reduction using

- (1) Venn Diagrams,
- (2) Boolean algebra,
- (3) Logic simulation and
- (4) Shannon's expansion

Consensus Theorem: Venn Diagrams

$$AB+AC+B'C : AB+B'C$$



Consensus Theorem: Boolean Algebra

- $AB+AC+B'C$
 $=AB+B'C$

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

$$\begin{aligned} &AB+AC+B'C \\ &=AB+AC\mathbf{1}+B'C \\ &=AB+AC(\mathbf{B+B'})+B'C \\ &=AB+ABC+AB'C+B'C \\ &=AB(\mathbf{1+C})+(\mathbf{A+1})B'C \\ &=AB+B'C \end{aligned}$$

Consensus Theorem: Logic Simulation

$$f(A,B,C) = AB + AC + B'C$$

$$g(A,B,C) = AB + B'C$$

Index	A	B	C	AB	AC	B'C	f	g
0	0	0	0	0	0	0		
1	0	0	1	0	0	1		
2	0	1	0	0	0	0		
3	0	1	1	0	0	0		
4	1	0	0	0	0	0		
5	1	0	1	0	1	1		
6	1	1	0	1	0	0		
7	1	1	1	1	1	0		

Consensus Theorem (Examples)

iClicker: Which one is not a consensus of the expressions on the right.

A. B

1. $A + A'B$

B. BC

2. $A + A'BC$

C. $(B+C)D$

3. $A(B+C) + A'D$

D. BCD

E. $BCDE$

4. $ABC + A'D + BCDE$

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: $x=1$ and $x=0$
 - $f(x,y,z,\dots) = xf(x=1,y,z,\dots) + x'f(x=0,y,z,\dots)$
- For example
 - $f(x) = xf(1) + x'f(0)$
 - $f(x,y) = xf(1,y) + x'f(0,y)$

Shannon's Expansion (Example)

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

Shannon's expansion:

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

Shannon's Expansion: Example

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

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

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

Shannon's Expansion

- Decompose the switching function into minterms

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

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

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

id	x	y	f(x,y)
0	0	0	f(0,0)
1	0	1	f(0,1)
2	1	0	f(1,0)
3	1	1	f(1,1)

Shannon's expansion can decompose a switching function into a truth table.

Shannon's Expansion vs Truth Table

Example: $f(x,y) = x + x'y$

$$\begin{aligned} f(x,y) &= xf(1,y) + x'f(0,y) \\ &= x(yf(1,1) + y'f(1,0)) + x'(yf(0,1) + y'f(0,0)) \\ &= xyf(1,1) + xy'f(1,0) + x'yf(0,1) + x'y'f(0,0). \end{aligned}$$

id	x	y	f(x,y)
0	0	0	f(0,0)
1	0	1	f(0,1)
2	1	0	f(1,0)
3	1	1	f(1,1)

Shannon's expansion can decompose a switching function into a truth table.

Shannon's Expansion

- Decompose the switching function into minterms

$$\begin{aligned}f(x, y) &= xf(1, y) + x'f(0, y) \\&= x(yf(1,1) + y'f(1,0)) + x'(yf(0,1) + y'f(0,0)) \\&= xyf(1,1) + xy'f(1,0) + x'yf(0,1) + x'y'f(0,0).\end{aligned}$$

- Decompose the switching function into maxterms

$$\begin{aligned}f(x, y) &= (x' + f(1, y)) \cdot (x + f(0, y)) \\&= (x' + (y' + f(1,1)) \cdot (y + f(1,0))) \cdot (x + (y' + f(0,1)) \cdot (y + f(0,0))) \\&= (x' + y' + f(1,1))(x' + y + f(1,0))(x + y' + f(0,1))(x + y + f(0,0))\end{aligned}$$

Shannon's Expansion: Example

Which variable in $ab' + ac + bc$ can be used for expansion?

- A. a
- B. b
- C. c
- D. None of the above

Shannon's Expansion: Example

$$F(A,B,C)=AB'+AC+BC$$

Remark: The choice of the variable for expansion is a nontrivial question.

Review Summary: Switching Algebra and Karnaugh Map

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 a visual aid.

Part I. Combinational Logic

II) Specification

1. Language
2. Boolean Algebra

Canonical Expression: Sum of minterms and
Product of maxterms

3. Truth Table: **minterms and maxterms**
4. Incompletely Specified Function

II. Specification (use addition as an example)

Decimal Addition

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

Carry Sum

Binary Addition

$$\begin{array}{r} 1 \quad 1 \quad 1 \\ 1 \quad 0 \quad 1 \\ + 1 \quad 1 \quad 1 \\ \hline 1 \quad 1 \quad 0 \quad 0 \end{array}$$

Carry bits

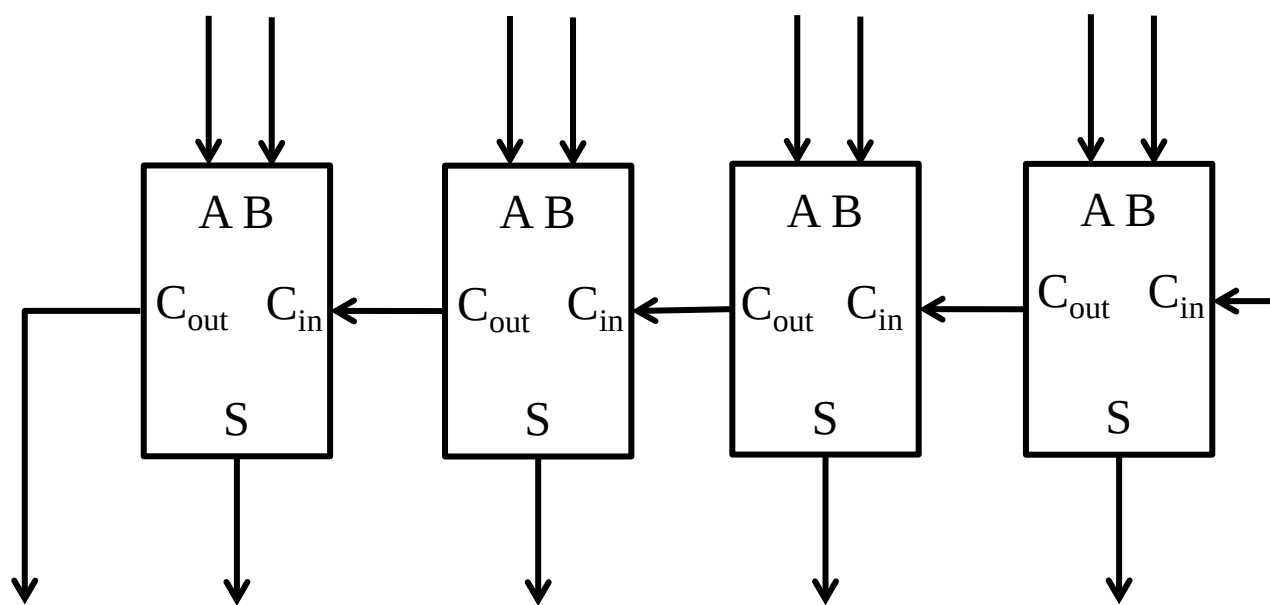
Carryout Sums

II. Specification (use addition as an example)

Decimal Addition

Binary Addition

II. Specification (use addition as an example)

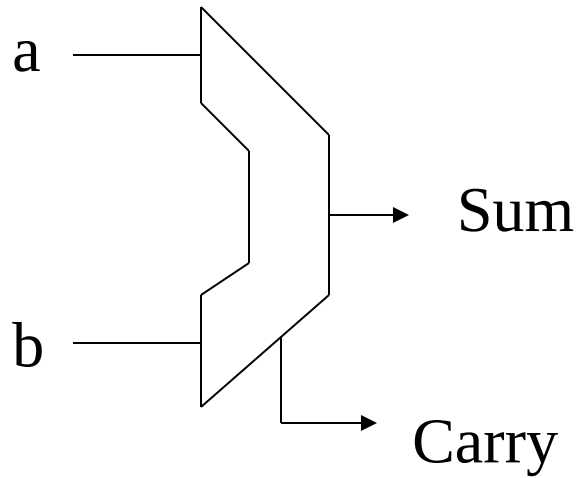


Binary Addition: Hardware

- Half Adder: Two inputs (a, b) and two outputs (carry, sum).
- Full Adder: Three inputs (a, b, c_{in}) and two outputs (carry, sum).

Half Adder

Truth Table



id	a	b	carry	sum
0	0	0	0	0
1	0	1	0	1
2	1	0	0	1
3	1	1	1	0

Switching Function

Switching Expressions:

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

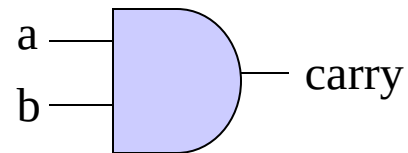
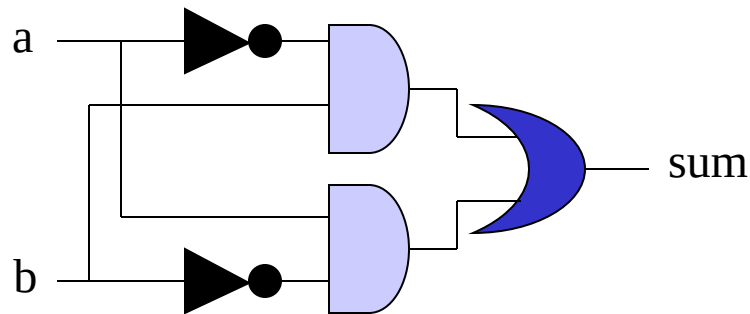
$$\text{Carry (a, b)} = a \cdot b$$

Ex:

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

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

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

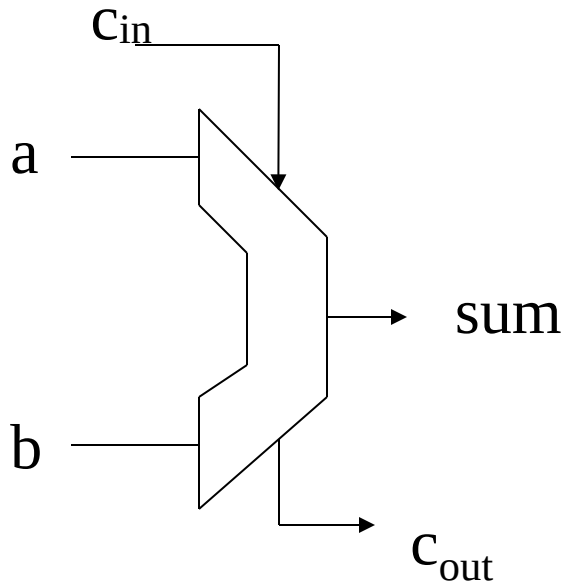


BSV notes

```
function Bit#(2) ha(Bit#(1) a, Bit#(1) b);  
  Bit#(1) s = (~a & b) | (a & ~b);  
  Bit#(1) c = a & b;  
  return {c, s};  
endfunction
```

- {c, s} represents bit concatenation

Full Adder



Arithmetic:

$$2C_{out} + sum = a + b + C_{in}$$

id	a	b	C_{in}	C_{out}	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

Minterms

A product of all variables in the function.

A minterm is equal to 1 on exactly one row of the truth table.

id	a	b	c	carry	sum
0	0	0	0	0	0
1	0	0	1	0	$a'b'c$
2	0	1	0	0	$a'bc'$
3	0	1	1	$a'bc$	0
4	1	0	0	0	$ab'c'$
5	1	0	1	$ab'c$	0
6	1	1	0	abc'	0
7	1	1	1	abc	abc

Maxterms

A sum of all variables in the function.

A maxterm is equal to 0 on exactly one row of the truth table.

id	a	b	c	carry	sum
0	0	0	0	$a+b+c$	$a+b+c$
1	0	0	1	$a+b+c'$	1
2	0	1	0	$a+b'+c$	1
3	0	1	1	1	$a+b'+c'$
4	1	0	0	$a'+b+c$	1
5	1	0	1	1	$a'+b+c'$
6	1	1	0	1	$a'+b'+c$
7	1	1	1	1	1

Minterms and Maxterms

id	a	b	c	minterm	maxterm
0	0	0	0	$m_0 = a'b'c'$	$M_0 = a+b+c$
1	0	0	1	$m_1 = a'b'c$	$M_1 = a+b+c'$
2	0	1	0	$m_2 = a'bc'$	$M_2 = a+b'+c$
3	0	1	1	$m_3 = a'bc$	$M_3 = a+b'+c'$
4	1	0	0	$m_4 = ab'c'$	$M_4 = a'+b+c$
5	1	0	1	$m_5 = ab'c$	$M_5 = a'+b+c'$
6	1	1	0	$m_6 = abc'$	$M_6 = a'+b'+c$
7	1	1	1	$m_7 = abc$	$M_7 = a'+b'+c'$

Minterms cover all the outputs which are **true** (1).

Maxterms cover all the outputs which are **false** (0).

Minterms

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

$$a'bc = 1 \text{ iff } (a,b,c) = (0,1,1)$$

$$ab'c = 1 \text{ iff } (a,b,c) = (1,0,1)$$

$$abc' = 1 \text{ iff } (a,b,c) = (1,1,0)$$

$$abc = 1 \text{ iff } (a,b,c) = (1,1,1)$$

$$f_1(a,b,c) = 1 \text{ iff } (a,b,c) = (0,1,1), (1,0,1), (1,1,0), \text{ or } (1,1,1)$$

$$\text{Ex: } f_1(1,0,1) = 1'01 + 10'1 + 101' + 101 = 1$$

$$f_1(1,0,0) = 1'00 + 10'0 + 100' + 100 = 0$$

Maxterms

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

$$a + b + c = 0 \text{ iff } (a,b,c) = (0,0,0)$$

$$a + b + c' = 0 \text{ iff } (a,b,c) = (0,0,1)$$

$$a + b' + c = 0 \text{ iff } (a,b,c) = (0,1,0)$$

$$a' + b + c = 0 \text{ iff } (a,b,c) = (1,0,0)$$

$$f_2(a,b,c) = 0 \text{ iff } (a,b,c) = (0,0,0), (0,0,1), (0,1,0), (1,0,0)$$

$$\text{Ex: } f_2(1,0,1) = (1+0+1)(1+0+1')(1+0'+1)(1'+0+1) = 1$$

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

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

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

$$f_1(a, b, c) = m_3 + m_5 + m_6 + m_7 = \Sigma m(3,5,6,7)$$

$$f_2(a, b, c) = M_0M_1M_2M_4 = \Pi M(0, 1, 2, 4)$$

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.

Minterms and Maxterms: Summary

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

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

Canonical presentation of logic functions

Conversion between truth tables and switching functions

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
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

$$\begin{aligned}g_1(a,b,c) &= a'b'c + a'bc + ab'c' + abc \\ &= m_1 + m_3 + m_4 + m_7 \\ &= \sum m(1,3,4,7)\end{aligned}$$

$$\begin{aligned}g_2(a,b,c) &= (a+b+c)(a'+b'+c) \\ &= M_0 M_6 \\ &= \prod M(0,6)\end{aligned}$$

Incompletely Specified Function

id	a	b	c	g
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

$$g_1(a,b,c) = \sum m(1,3,4,7)$$

$$g_2(a,b,c) = \prod M(0,6)$$

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

A: Yes

B: No