

Lecture 2:

Combinational Logic

CSE 140: Components and Design Techniques for Digital Systems

Diba Mirza

Dept. of Computer Science and Engineering
University of California, San Diego

Outline

- What is a combinational circuit?
- Combinational Logic
 1. Scope
 2. Specification : Boolean algebra, truth tables
 3. Synthesis: circuits

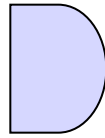
What you should know at the end of this lecture....

1. What is a combinational circuit?
2. What are the inputs and outputs of a combinational circuit?
3. What does a combinational circuit achieve?
4. What are the different ways of representing combinational circuits?
5. Why is the role of Boolean Algebra in designing combinational circuits?
6. What is our ultimate goal when designing a combinational circuit? (Segway to next lecture)

What is a combinational circuit?

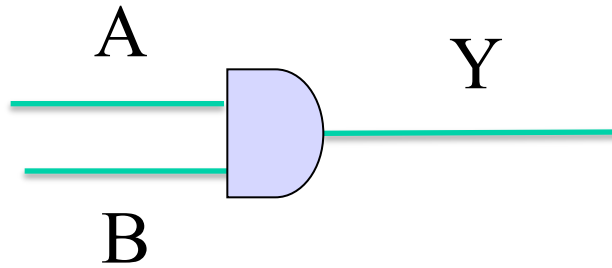
- [illegible]

Inputs and Outputs



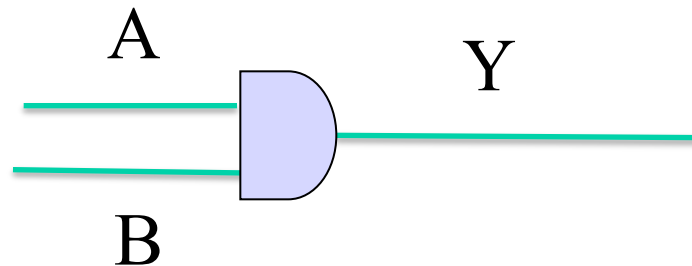
- Inputs and outputs can only have two discrete values:
 - Physical domain (usually, voltages)
 - Schematic and mathematical (Boolean variables)

Representations of combinational circuits: The Schematic



Representations of combinational circuits: Truth Table

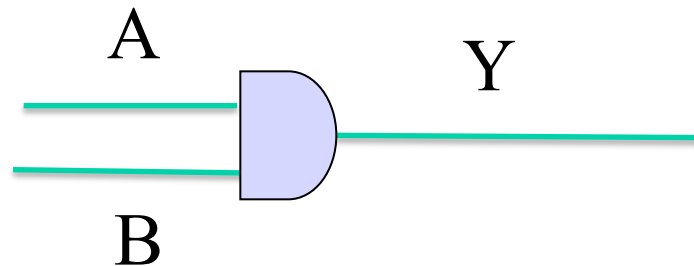
AND	A B	Y
	0 0	0
	0 1	0
	1 0	0
	1 1	1



Representations of combinational circuits: Boolean Expression/Equation

$$Y = AB$$

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



Boolean Algebra

Similar to regular algebra but defined on a set (B) with only three basic 'logic' operations:

1. Intersection: AND (2-input) , Operator: $\cdot$
2. Union: OR (2 input), Operator: $+$
3. Complement: NOT (1 input) Operator:

Boolean Algebra

Q: If X and Y are Boolean variables with $X=1$, $Y=0$, what is $X+X+Y$?

A. 0

B. 1

C. 2

Boolean Algebra

Boolean operations satisfy the following laws:

- 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: $a+0=a$, $a \cdot 1=a$

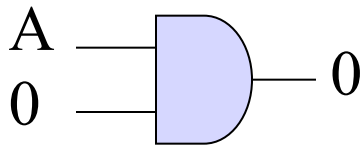
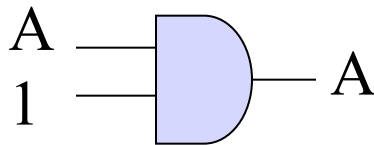
- Complement laws: $a+a'=1$, $a \cdot a'=0$

Boolean algebra and switching functions:

Operators and Digital Logic Gates

Two-input operator
AND ($\cdot$)

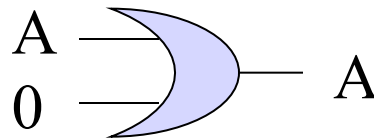
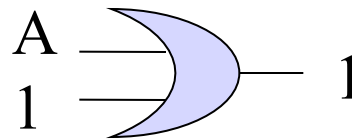
AND	A B	Y
	0 0	0
	0 1	0
	1 0	0
	1 1	1



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

Two-input operator
OR ($+$)

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



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

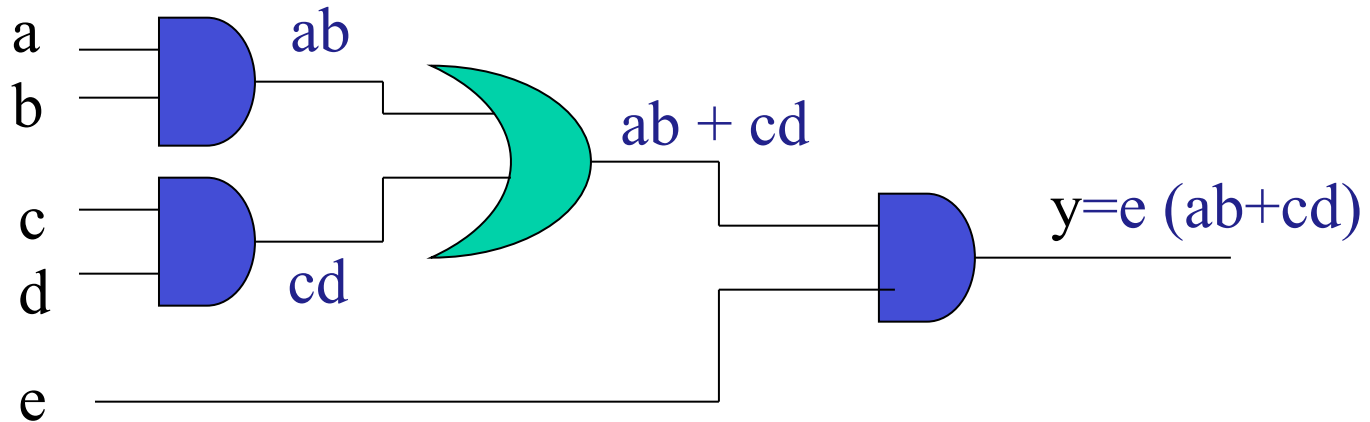
One-input operator
NOT (Complement, ')

NOT	A	Y
	0	1
	1	0

So, what is the point?

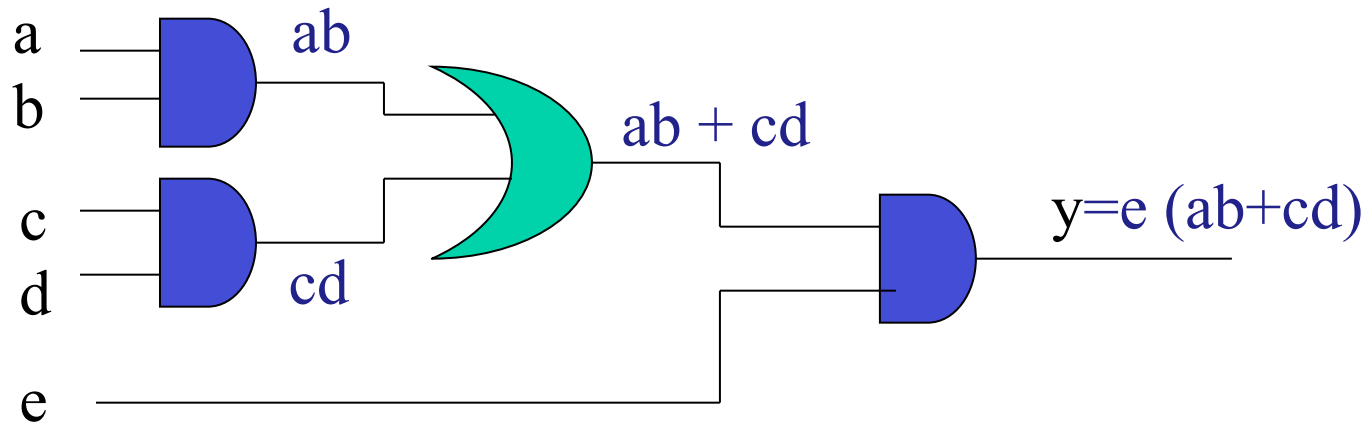
So, what is the point of representing gates as symbols and Boolean expressions?

- Given the Boolean expression, we can draw the circuit it represents by cascading gates (and vice versa)



Logic circuit vs. Boolean Algebra Expression

Logic circuit vs. Boolean Algebra Expression



Schematic (circuit) Diagram: Boolean Algebra:

5 primary inputs

5 literals

4 components

4 operators

9 signal nets

12 pins

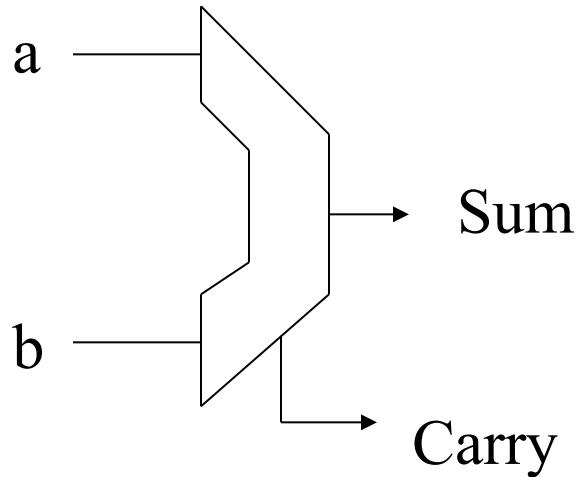
So, what is the point?

Combinational Logic: Scope

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

Specifying Logic Problems: Truth tables

Example: Half Adder



Truth Table

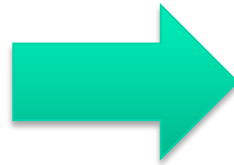
a	b	carry	sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

The process of designing the circuit

Truth Table

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

Switching Expressions or
Boolean Equation:



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

Switching Function

Draw the circuit from the Boolean Equation

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

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

Truth Table

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

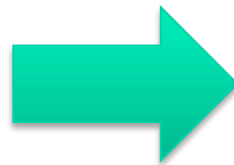
Switching Function

How do we get a Boolean Equation from the truth table?

Truth Table

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

How?



$$\begin{aligned}\text{Sum (a, b)} &= a' b + ab' \\ \text{Carry (a, b)} &= ab\end{aligned}$$

Switching Function

How do we get a Boolean Equation from the truth table? (Intuitive)

Truth Table

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

Switching Function

Need 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

$A\bar{B}C, A\bar{C}, BC$

- Implicate: sum of literals

$(\bar{A}+B+C), (\bar{A}+\bar{C}), (B+C)$

- Minterm: product that includes all input variables

$A\bar{B}C, AB\bar{C}, ABC$

- Maxterm: sum that includes all input variables

$(A+B+C), (A+B+\bar{C}), (A+\bar{B}+C)$

Specifying input combinations as Boolean expressions

A	B	Min term
0	0	
0	1	
1	0	
1	1	

Sum of Product Canonical Form

Minterm	A	B	Carry	Sum
A'B'	0	0	0	0
A'B	0	1	0	1
AB'	1	0	0	1
AB	1	1	1	0

I. $\text{sum}(A,B) =$

II. $\text{carry}(A,B) =$

Ultimate goal...

- Optimize the circuit to be the simplest possible
- Reduction of canonical expressions
(Next lecture)