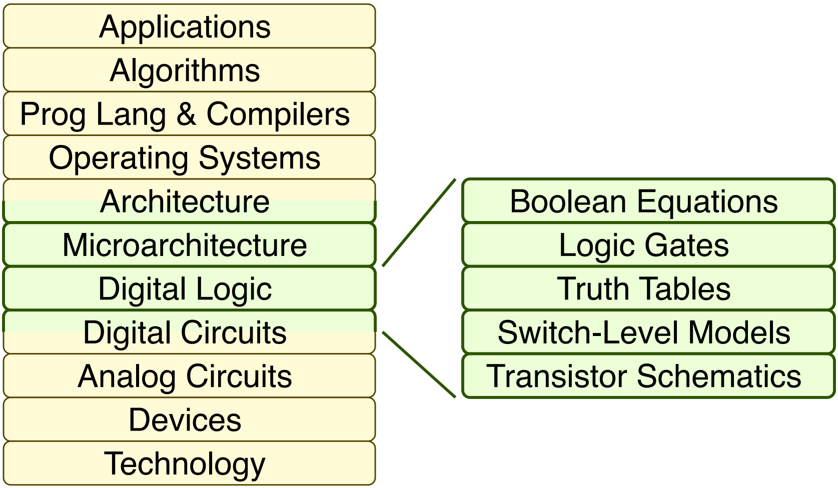
ECE 2300 Digital Logic and Computer Organization
Fall 2024

Topic 3: Boolean Equations

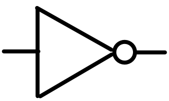
School of Electrical and Computer Engineering
Cornell University

revision: 2024-09-10-10-35

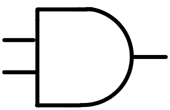

Applications
Algorithms
Prog Lang & Compilers
Operating Systems
Architecture
Microarchitecture
Digital Logic
Digital Circuits
Analog Circuits
Devices
Technology
Boolean Equations
Logic Gates
Truth Tables
Switch-Level Models
Transistor Schematics

1. From Logic Gates to Boolean Equations

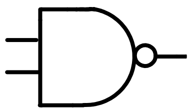
- Start by defining some terminology
  - ____________________ : equation where all variables are TRUE or FALSE
  - ____________________ : variable or its inverse (complement of A is $\overline{A}$)
  - ____________________ : variable or its complement (A, B, $\overline{A}$, $\overline{B}$)
  - ____________________ : AND of one or more literals ($\overline{A}B$, $A\overline{BC}$, B)
  - ____________________ : another name for a product
  - ____________________ : product involving all inputs to function ($A\overline{B}C$)
  - ____________________ : OR of one or more literals ($A + \overline{B}$, B)
  - ____________________ : sum involving all inputs to function ($A + \overline{B} + C$)
- We can represent our primitive logic gates as Boolean equations
  - Higher level of abstraction makes it easier to understand, manipulate, and optimize gate-level networks



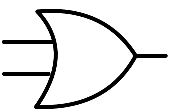
$Y = \overline{A}$



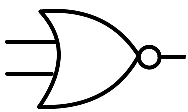
$Y = AB$



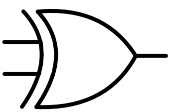
$Y = \overline{AB}$



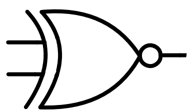
$Y = A + B$



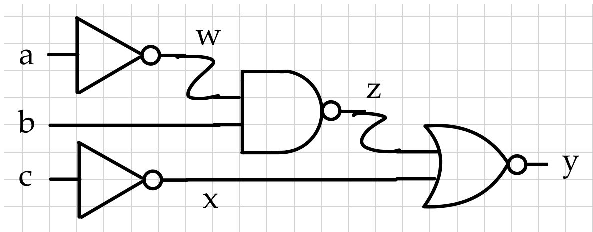
$Y = \overline{A + B}$



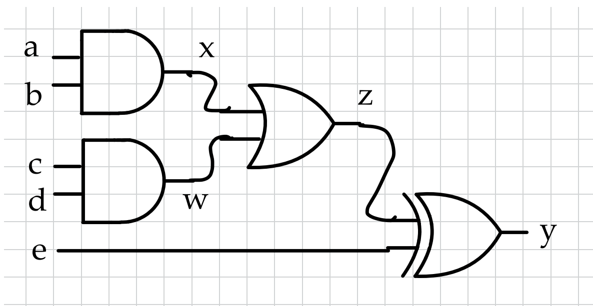
$Y = A \oplus B$



$Y = \overline{A \oplus B}$



Corresponding Boolean equation



Corresponding Boolean equation

1.1. Writing Boolean Equations in Verilog

NOT	$Y = \overline{A}$	`assign Y =  ~A;`
AND	$Y = AB$	`assign Y =   A & B;`
NAND	$Y = \overline{AB}$	`assign Y = ~(A & B);`
OR	$Y = A + B$	`assign Y =   A | B;`
NOR	$Y = \overline{A + B}$	`assign Y = ~(A | B);`
XOR	$Y = A \oplus B$	`assign Y =   A ^ B;`
XNOR	$Y = \overline{A \oplus B}$	`assign Y = ~(A ^ B);`

$$\begin{aligned} W &= \overline{A} \\ Z &= \overline{W} \\ X &= \overline{C} \\ Y &= \overline{Z + X} \end{aligned}$$

```
module BooleanEqEx1
(
  input  wire a,
  input  wire b,
  input  wire c,
  output wire y
);

  wire w;
  wire x;
  wire z;

  assign w = ~a;
  assign z = ~( w & b );
  assign x = ~c;
  assign y = ~( z | x );

  // assign y = ~( ~( ~a & b ) | ~c )

endmodule
```

2. From Truth Tables to Boolean Equations

- Truth table for N inputs has 2^N rows, one for every possible set of input values
- We can systematically transform any truth table into a corresponding Boolean equation

2.1. Sum-of-Products Canonical Form

- Each row in the truth table is associated with a minterm
- We can write a Boolean equation for any truth table as the OR of the minterms for which the output is TRUE
- Called sum-of-products (SOP) canonical form

A	B	Y	minterm	name
0	1	0	$\overline{A}\,\overline{B}$	m_0
0	1	1	$\overline{A}\,B$	m_1
1	0	0	$A\,\overline{B}$	m_2
1	1	0	$A\,B$	m_3

Corresponding Boolean equation

A	B	Y	minterm	name
0	1	0	$\overline{A}\,\overline{B}$	m_0
0	1	1	$\overline{A}\,B$	m_1
1	0	0	$A\,\overline{B}$	m_2
1	1	1	$A\,B$	m_3

Corresponding Boolean equation

2.2. Product-of-Sums Canonical Form

- We can write a Boolean equation for any truth table as the AND of the maxterms for which the output is FALSE minterms for which the output is TRUE
- Called product-of-sums (POS) canonical form

A	B	Y	maxterm	name
0	1	1	$A+B$	M_0
0	1	0	$A+\overline{B}$	M_1
1	0	1	$\overline{A}+B$	M_2
1	1	1	$\overline{A}+\overline{B}$	M_3

Corresponding Boolean equation

A	B	Y	maxterm	name
0	1	0	$A+B$	M_0
0	1	1	$A+\overline{B}$	M_1
1	0	0	$\overline{A}+B$	M_2
1	1	1	$\overline{A}+\overline{B}$	M_3

Corresponding Boolean equation

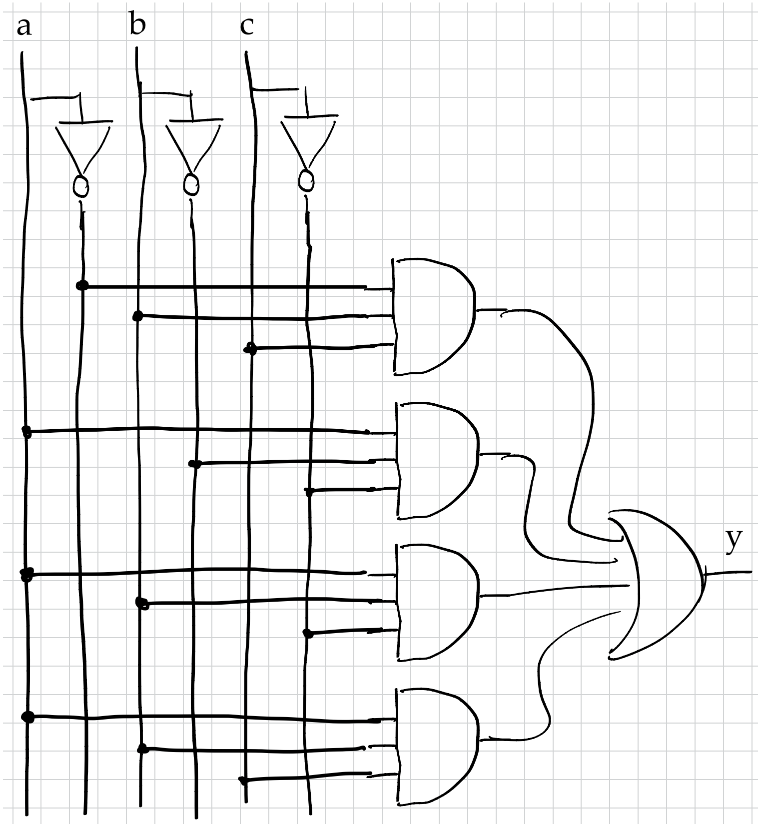
2.3. Using Canonical Forms in Verilog

- Let's revisit a truth table from the previous topic
- Create a wire for every minterm
- Assign output as the OR of every minterm for which output is TRUE

a	b	c	y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

```
module SumOfProductsEx
(
  input  wire a,
  input  wire b,
  input  wire c,
  output wire y
);

  assign min0 = ~a & ~b & ~c;
  assign min1 = ~a & ~b &  c;
  assign min2 = ~a &  b & ~c;
  assign min3 = ~a &  b &  c;

  assign min4 =  a & ~b & ~c;
  assign min5 =  a & ~b &  c;
  assign min6 =  a &  b & ~c;
  assign min7 =  a &  b &  c;

  assign y = min3 | min4 | min6 | min7;

endmodule
```

Corresponding Boolean equation


a
b
c
y

3. Boolean Algebra

- Algebra enables logically manipulating mathematical equestions
- Boolean algebra enables logically manipulating Boolean equestions

3.1. Boolean Axioms, Theorems, and Proofs

- The five axioms of Boolean algebra and their duals define Boolean variables and meaning of NOT, AND, OR

Number	Axiom	Dual	Name
A1	$B = 0$ if $B \neq 1$	$B = 1$ if $B \neq 0$	Binary Field
A2	$\overline{0} = 1$	$\overline{1} = 0$	NOT
A3	$0 \bullet 0 = 0$	$1 + 1 = 1$	AND/OR
A4	$1 \bullet 1 = 1$	$0 + 0 = 0$	AND/OR
A5	$0 \bullet 1 = 1 \bullet 0 = 0$	$1 + 0 = 0 + 1 = 1$	AND/OR

- Following theorems describe how to simplify equations involving one variable

Number	Theorem	Dual	Name
T1	$B \bullet 1 = B$	$B + 0 = B$	Identity
T2	$B \bullet 0 = 0$	$B + 1 = 1$	Null Element
T3	$B \bullet B = B$	$B + B = B$	Idempotency
T4	$\overline{\overline{B}} = B$		Involution
T5	$B \bullet \overline{B} = 0$	$B + \overline{B} = 1$	Complements

- Following theorems describe how to simplify equations involving more than one variable

#	Theorem	Dual	Name
T6	$B \bullet C = C \bullet B$	$B+C = C+B$	Commutativity
T7	$(B \bullet C) \bullet D = B \bullet (C \bullet D)$	$(B + C) + D = B + (C + D)$	Associativity
T8	$B \bullet (C + D) = (B \bullet C) + (B \bullet D)$	$B + (C \bullet D) = (B+C)\ (B+D)$	Distributivity
T9	$B \bullet (B+C) = B$	$B + (B \bullet C) = B$	Covering
T10	$(B \bullet C) + (B \bullet \overline{C}) = B$	$(B+C) \bullet (B+\overline{C}) = B$	Combining
T11	$(B \bullet C) + (\overline{B} \bullet D) + (C \bullet D) =$ $(B \bullet C) + (\overline{B} \bullet D)$	$(B+C) \bullet (\overline{B}+D) \bullet (C+D) =$ $(B+C) \bullet (\overline{B}+D)$	Consensus
T12	$\overline{B \bullet C \bullet D \ldots} = \overline{B}+\overline{C}+\overline{D}\ldots$	$\overline{B+C+D\ldots} = \overline{B} \bullet \overline{C} \bullet \overline{D}\ldots$	De Morgan's

- Two methods to prove a theorem
  - Method 1: Perfect induction
  - Method 2: Use other theorems and axioms to make one side of the equation look like the other

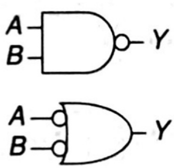
$$\text{T9:} \quad B \bullet (B + C) = B$$

$$\text{T10:} \quad (B \bullet C) + (B \bullet \overline{C}) = B$$

De Morgan's Theorem

$$\text{T12:} \quad \overline{B_0 \bullet B_1 \bullet B_2 \ldots} = \overline{B_0} + \overline{B_1} + \overline{B_2} \ldots$$

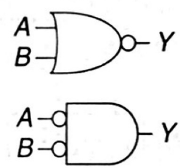
NAND



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

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

NOR



$Y = \overline{A + B} = \overline{A}\,\overline{B}$

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

Derive Product-of-Sum from Sum-of-Product

$$Y = \overline{A}\,\overline{B} + \overline{A}B$$

3.2. Simplifying with Boolean Algebra

$$\begin{aligned} Y &= \overline{A}B + AB \\ Y &= B && \text{T10: Combining} \end{aligned}$$

$$\begin{aligned} Y &= \overline{A}B + AB \\ Y &= B(A + \overline{A}) && \text{T8: Distributivity} \\ Y &= B(1) && \text{T5': Complements} \\ Y &= B && \text{T1: Identity} \end{aligned}$$

$$Y = \overline{A}B\overline{C} + \overline{A}BC + ABC$$

Attempt #1

Attempt #2

3.3. Simplifying with Karnaugh Maps

- K-maps are a systematic way to simplify Boolean equations

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

$$Y = \overline{A}B\overline{C} + \overline{A}BC + ABC$$

C \ AB	00	01	11	10
0				
1				

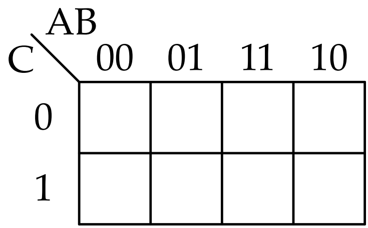
A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

$$Y = \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}\,\overline{C} + A\overline{B}C$$

C \ AB	00	01	11	10
0				
1				

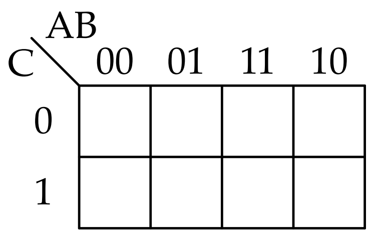
A	**B**	**C**	**Y**
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

$$Y = \overline{A}B\overline{C} + \overline{A}BC + A\overline{B}\,\overline{C} + AB\overline{C} + ABC$$



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

$$Y = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,\overline{B}C + A\overline{B}\,\overline{C} + A\overline{B}C + AB\overline{C}$$



- Extending k-maps to equations of four variables

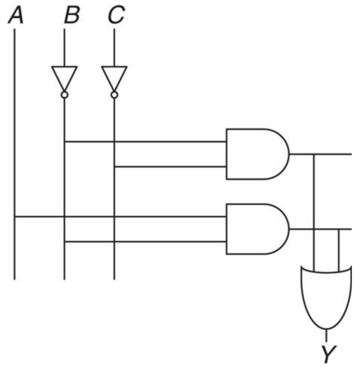
A	B	C	D	Y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

$$
\begin{aligned}
Y &= \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}C\overline{D} + \overline{A}\,\overline{B}CD \\
&+ \overline{A}B\overline{C}D + \overline{A}\,BC\overline{D} + \overline{A}\,BCD \\
&+ A\overline{B}\,\overline{C}\,\overline{D} + A\overline{B}\,\overline{C}D + A\overline{B}C\overline{D}
\end{aligned}
$$

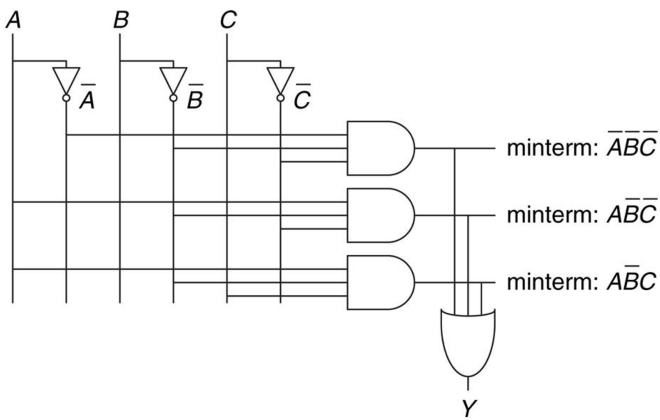
CD \ AB	00	01	11	10
00				
01				
11				
10				

4. From Boolean Equations to Logic Gates

$$Y = \overline{B}\,\overline{C} + A\overline{B}$$

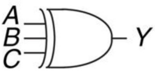


$$Y = \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}\,\overline{C} + A\overline{B}C$$



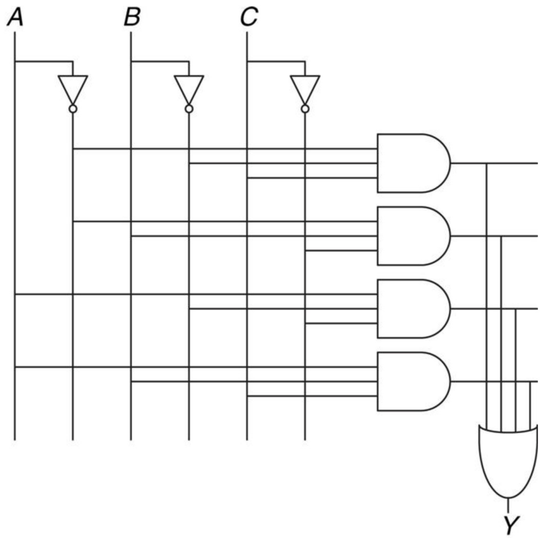
$$Y = \overline{A}\,\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + ABC$$

XOR3



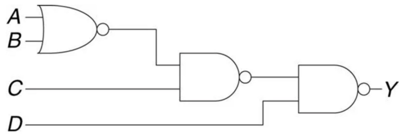
$Y = A \oplus B \oplus C$

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

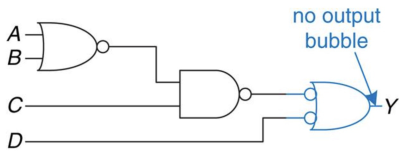


Bubble pushing

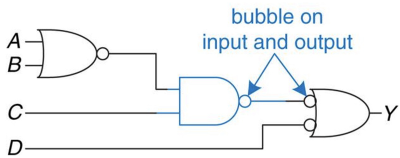
$$Y = \overline{(\overline{(\overline{A+B})C})D}$$



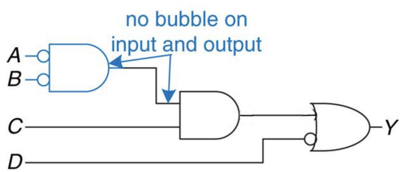
$$Y = \overline{\overline{(\overline{A+B})C}} + \overline{D}$$



$$Y = (\overline{A+B})C + \overline{D}$$



$$Y = \overline{A}\,\overline{B}C + \overline{D}$$



5. From Boolean Equations to Transistor Schematics

$$Y = \overline{(A+B)C}$$

$$Y = \overline{AB+CD}$$

6. Summary of Abstractions