

9/30/26

# COMP 311:

# *Computer Organization!*

Lecture 12: Introduction to Gate Minimization



The University  
of North Carolina  
at Chapel Hill

<https://tinyurl.com/comp311-fa26>

# Logistics Update

---



- Lab 2 Released
- WA2 due today
  - I put other WA dates up on assignments tab.
- Quiz 3 on Wednesday

# Quiz Topics

---



- Transistors (pMOS vs. nMOS)
- Tracing CMOS gates
- Going between circuit diagrams, boolean equations, and truth tables
- Be able to draw basic circuit diagrams made of logic gates!
- Multiplexers
- Counting the number of transistors
- Boolean algebra simplification
  - Whatever we get through today! 😊

# Plan for Class Today

---



- 3:35-4:00: Finish discussion of multiplexers
- 4:00-4:15: Boolean Algebra Laws
- 4:15-4:20: Break
- 4:20-4:50: Gate Optimization w/ Boolean Algebra!

## Guiding Question 😊

*How do start  
optimizing our circuits?*

# Exercise 3

| $A_1$ | $A_0$ | $B_1$ | $B_0$ | Y |
|-------|-------|-------|-------|---|
| 0     | 0     | 0     | 0     |   |
| 0     | 0     | 1     | 0     |   |
| 0     | 0     | 0     | 1     |   |
| 0     | 0     | 1     | 1     |   |
| 1     | 0     | 0     | 0     |   |
| 1     | 0     | 1     | 0     |   |
| 1     | 0     | 0     | 1     |   |
| 1     | 0     | 1     | 1     |   |
| 0     | 1     | 0     | 0     |   |
| 0     | 1     | 1     | 0     |   |
| 0     | 1     | 0     | 1     |   |
| 0     | 1     | 1     | 1     |   |
| 1     | 1     | 0     | 0     |   |
| 1     | 1     | 1     | 0     |   |
| 1     | 1     | 0     | 1     |   |
| 1     | 1     | 1     | 1     |   |

I want to build a circuit that will output 1 if  $A > B$ , and 0 if  $A \leq B$ . A and B are both two-bit unsigned numbers.

Additionally, define  $A_1$  and  $B_1$  as the left bits of A and B, and  $A_0$  and  $B_0$  as the right bits.

# Exercise 3

| $A_1$ | $A_0$ | $B_1$ | $B_0$ | Y |
|-------|-------|-------|-------|---|
| 0     | 0     | 0     | 0     | 0 |
| 0     | 0     | 1     | 0     | 0 |
| 0     | 0     | 0     | 1     | 0 |
| 0     | 0     | 1     | 1     | 0 |
| 1     | 0     | 0     | 0     | 1 |
| 1     | 0     | 1     | 0     | 0 |
| 1     | 0     | 0     | 1     | 1 |
| 1     | 0     | 1     | 1     | 0 |
| 0     | 1     | 0     | 0     | 1 |
| 0     | 1     | 1     | 0     | 0 |
| 0     | 1     | 0     | 1     | 0 |
| 0     | 1     | 1     | 1     | 0 |
| 1     | 1     | 0     | 0     | 1 |
| 1     | 1     | 1     | 0     | 1 |
| 1     | 1     | 0     | 1     | 1 |
| 1     | 1     | 1     | 1     | 0 |

I want to build a circuit that will output 1 if  $A > B$ , and 0 if  $A \leq B$ . A and B are both two-bit unsigned numbers.

Additionally, define  $A_1$  and  $B_1$  as the left bits of A and B, and  $A_0$  and  $B_0$  as the right bits.

# Let's Try it Out

| $A_1$ | $A_0$ | $B_1$ | $B_0$ | Y |
|-------|-------|-------|-------|---|
| 0     | 0     | 0     | 0     | 0 |
| 0     | 0     | 1     | 0     | 0 |
| 0     | 0     | 0     | 1     | 0 |
| 0     | 0     | 1     | 1     | 0 |
| 1     | 0     | 0     | 0     | 1 |
| 1     | 0     | 1     | 0     | 0 |
| 1     | 0     | 0     | 1     | 1 |
| 1     | 0     | 1     | 1     | 0 |
| 0     | 1     | 0     | 0     | 1 |
| 0     | 1     | 1     | 0     | 0 |
| 0     | 1     | 0     | 1     | 0 |
| 0     | 1     | 1     | 1     | 0 |
| 1     | 1     | 0     | 0     | 1 |
| 1     | 1     | 1     | 0     | 1 |
| 1     | 1     | 0     | 1     | 1 |
| 1     | 1     | 1     | 1     | 0 |

I want to build a circuit that will output 1 if  $A > B$ , and 0 if  $A \leq B$ . A and B are both two-bit unsigned numbers.

Additionally, define  $A_1$  and  $B_1$  as the left bits of A and B, and  $A_0$  and  $B_0$  as the right bits.

$$\bar{A}_1 A_0 \bar{B}_1 \bar{B}_0 + A_1 \bar{A}_0 \bar{B}_1 \bar{B}_0 + A_1 \bar{A}_0 \bar{B}_1 B_0 + A_1 A_0 \bar{B}_1 \bar{B}_0 + A_1 A_0 \bar{B}_1 B_0 + A_1 A_0 B_1 \bar{B}_0$$

*Is the simplest possible form?? 🤔*

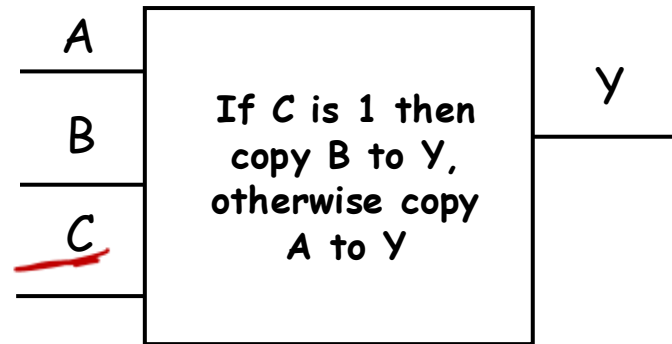


# Multiplexers! 😊

# An interesting 3-input gate...!



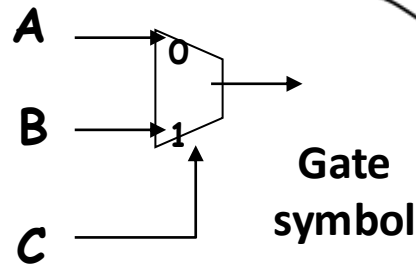
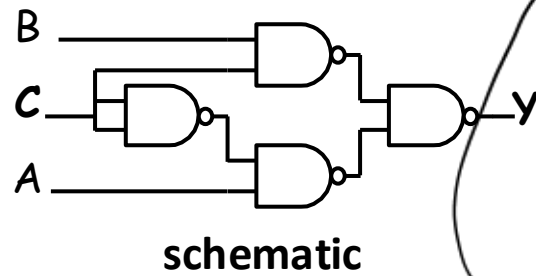
Based on C, select the A or B input to be copied to Y.



## Truth Table

| C | B | A | Y |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

## 2-input Multiplexer



# A Circuit for If-Statements! 😊



<sup>C</sup>  
if (S == 0)  
    Y = A  
elseif (S == 1)  
    Y = B

# In Assembly...



```
beq  x10, x0, S_is_zero      # if S == 0, jump
addi x14, x0, 1
beq  x10, x14, S_is_one     # if S == 1, jump
```

→ S\_is\_zero:  
add x13, x11, x0

# Y = A

→ S\_is\_one:  
add x13, x12, x0           # Y = B

$x10 = S$   
 $x11 = A$

$x12 = B$   
 $x13 \leftarrow Y$

# Multiplexers



Truth Table

Equation

Diagram

↓

| S | A | B | Y |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

# Multiplexers



Truth Table

| S | A | B | Y |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |
| S | A | B | Y |
| 0 | 0 | X | 0 |
| 0 | 1 | X | 1 |
| 1 | X | 0 | 0 |
| 1 | X | 1 | 1 |

Equation

Diagram

$$Y = \overline{A}S + BS$$

(1)                      (2)

# Multiplexers



Truth Table

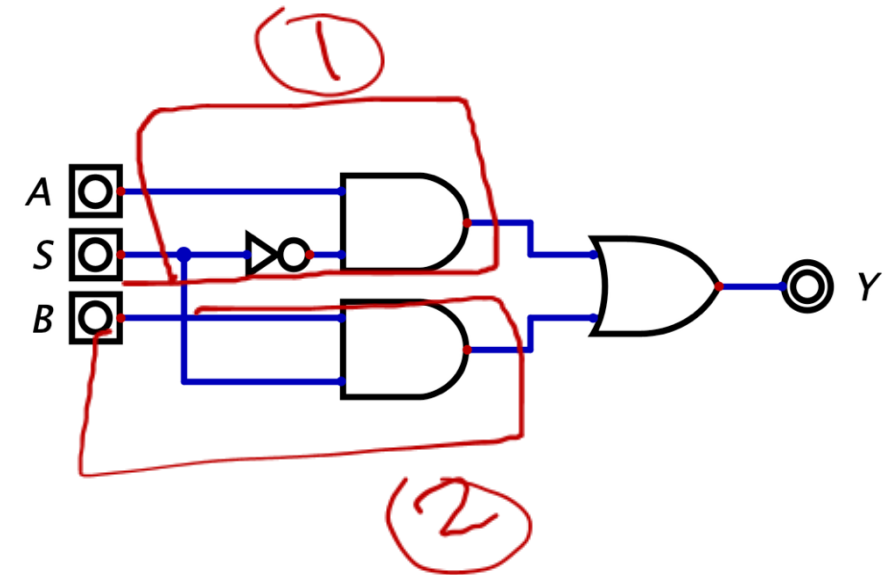
| S | A | B | Y |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

| S | A | B | Y |
|---|---|---|---|
| 0 | 0 | X | 0 |
| 0 | 1 | X | 1 |
| 1 | X | 0 | 0 |
| 1 | X | 1 | 1 |

Equation

$$Y = \overset{(1)}{A}\bar{\underset{(2)}{S}} + \underset{(2)}{B}S$$

Diagram



# Multiplexers



Truth Table

| S | A | B | Y |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

| S | A | B | Y |
|---|---|---|---|
| 0 | 0 | X | 0 |
| 0 | 1 | X | 1 |
| 1 | X | 0 | 0 |
| 1 | X | 1 | 1 |

Equation

Diagram

***Don't-cares can be used to simplify logic!***

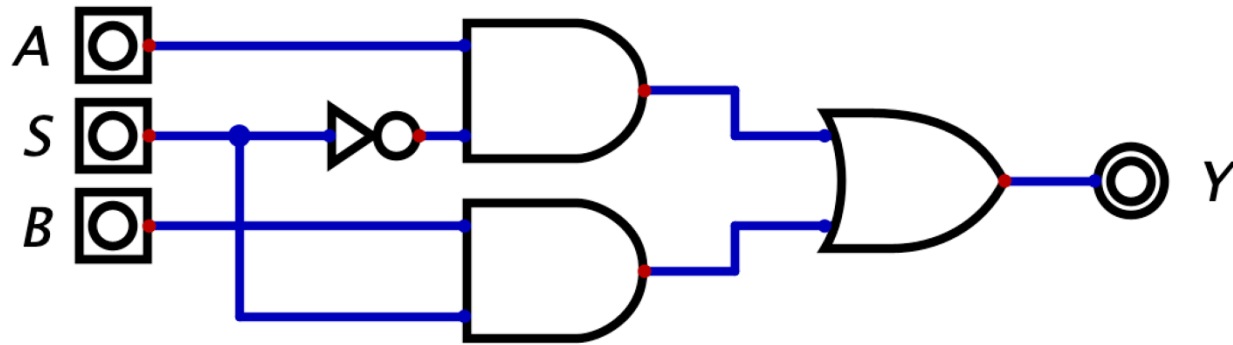
For input combinations that never occur, the designer may assume either 1 or 0, whichever is more useful to achieving a minimum-cost implementation.

Y

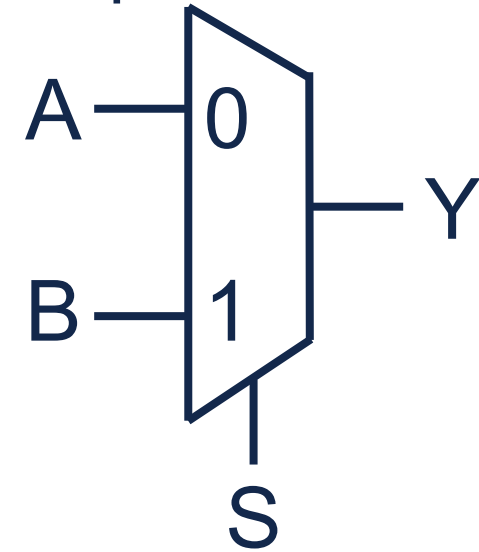
## 2:1 Multiplexer (2:1 mux)



- A circuit that chooses an output from among two inputs based on the value of a select signal

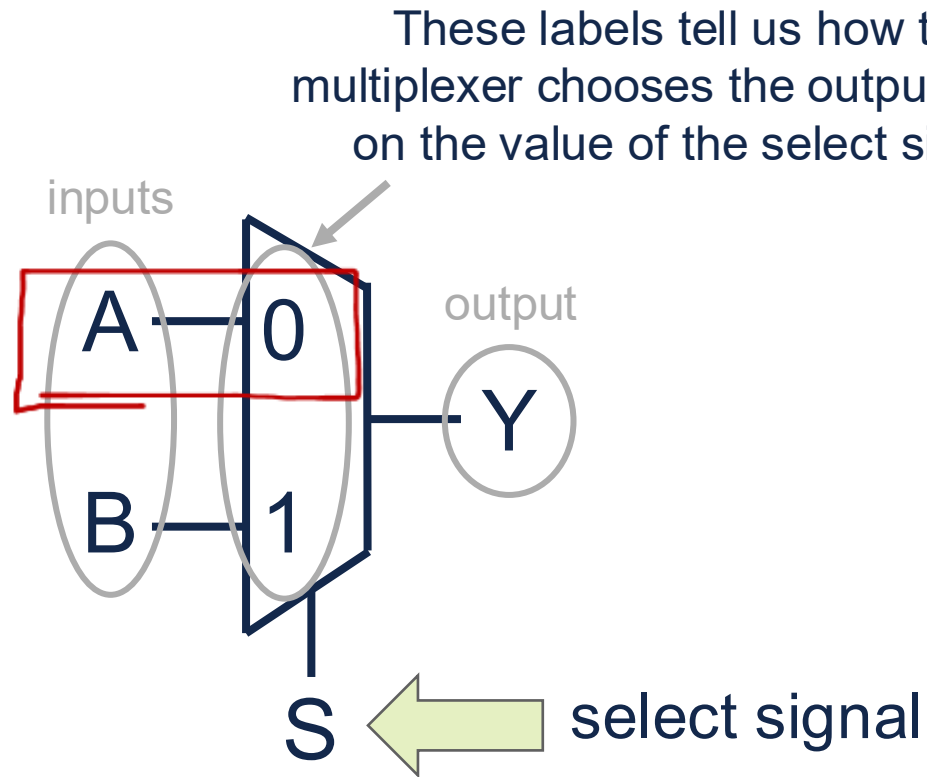


2:1 Multiplexer Schematic  
(Gate Diagram)



2:1 Multiplexer Symbol

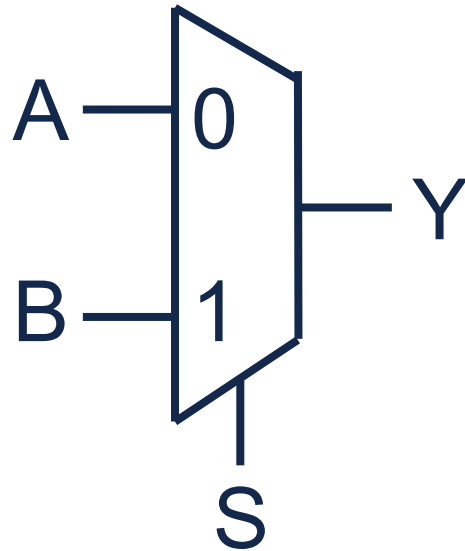
## 2:1 Multiplexer (2:1 mux)



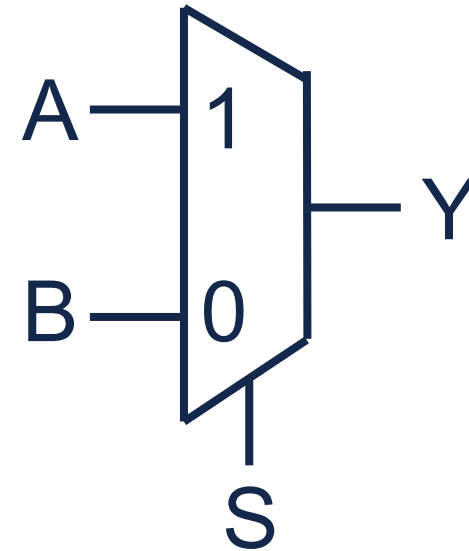
When  $S = 0$ ,  $Y = A$   
When  $S = 1$ ,  $Y = B$

## 2:1 Multiplexer (2:1 mux)

---



When  $S = 0$ ,  $Y = A$   
When  $S = 1$ ,  $Y = B$

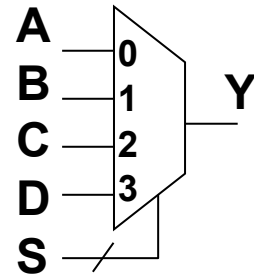
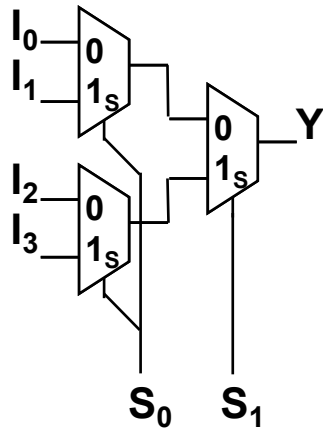


When  $S = 0$ ,  $Y = B$   
When  $S = 1$ ,  $Y = A$

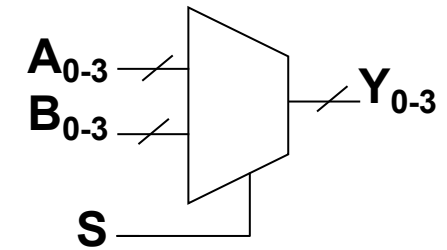
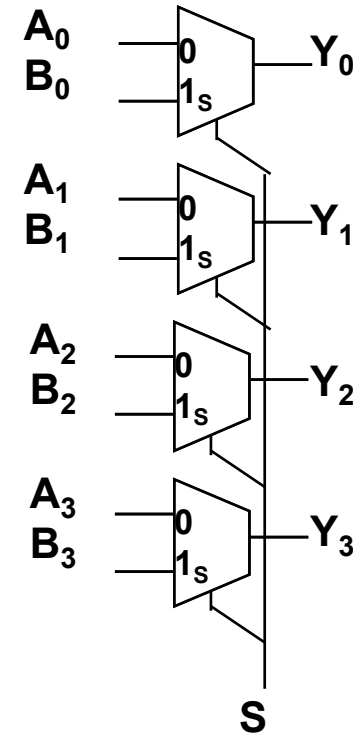
# MUX Compositions and Shortcuts



A 4-input Mux  
(implemented as a tree)



A 4-bit wide 2-input Mux

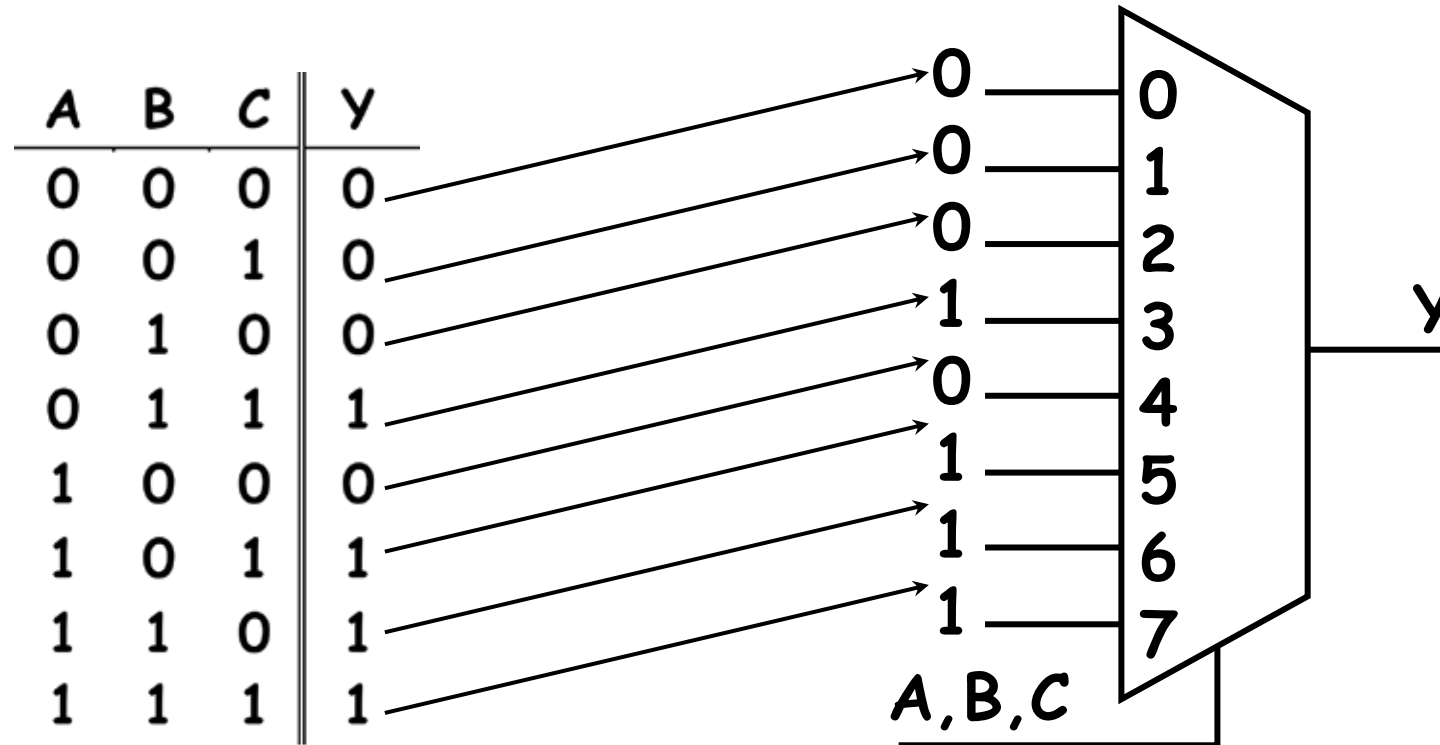


# MUX Function Synthesis

A mux plus constants is enough to implement *any* Boolean function

Consider implementation of some arbitrary Combinational function,  $F(A,B,C)$ ... using a MULTIPLEXER as the only circuit element:

Mux: An example “configurable” logic element



8:1 MUX

# MUX Logic Tricks

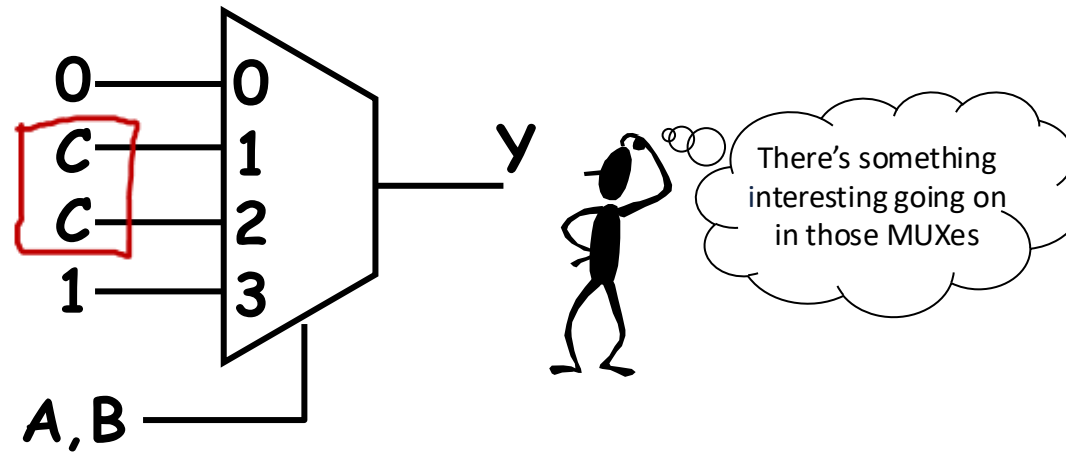


We can apply certain optimizations to MUX Function synthesis

Desired Logic  
Function

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

- Largely by inspection or exhaustive search



- N-input function == N-1 input MUX & 1 inverter

If adjacent truth-table rows differ by just one bit, that bit can become a mux input



# Gate Minimization



# Optimizing our circuits...

- Fewer transistors means....
  - Smaller area!
  - Lower power consumption!
  - Shorter time from input to output!
  - Lower cost!
- How will we do this?
  - Visual inspection
  - Boolean algebra simplification
  - In practice, → advanced synthesis algorithms to create **logically equivalent**, more efficient circuits! 😊

# Recall: Transistor Count



| Gate | Number of Transistors |
|------|-----------------------|
| NOT  | 2                     |
| AND  | 6                     |
| OR   | 6                     |
| NAND | 4                     |
| NOR  | 4                     |
| XOR  | 12                    |
| XNOR | 12                    |

# Boolean Algebra Laws



| Name         | OR Form                                       | AND Form                                    |
|--------------|-----------------------------------------------|---------------------------------------------|
| Identity     | $A + 0 = A$                                   | $A \cdot 1 = A$                             |
| Null         | $A + 1 = 1$                                   | $A \cdot 0 = 0$                             |
| Idempotent   | $A + A = A$                                   | $A \cdot A = A$                             |
| Complement   | $A + \bar{A} = 1$                             | $A \cdot \bar{A} = 0$                       |
| Commutative  | $A + B = B + A$                               | $A \cdot B = B \cdot A$                     |
| Associative  | $(A + B) + C = A + (B + C)$                   | $(A \cdot B) \cdot C = A \cdot (B \cdot C)$ |
| Distributive | $A \cdot (B + C) = (A \cdot B) + (A \cdot C)$ | $A + (B \cdot C) = (A + B) \cdot (A + C)$   |
| De Morgan's  | $\overline{A + B} = \bar{A} \cdot \bar{B}$    | $\overline{A \cdot B} = \bar{A} + \bar{B}$  |

# Boolean Algebra Practice



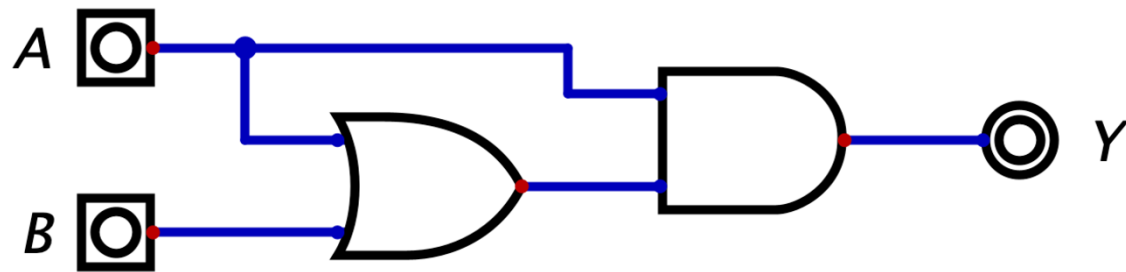
Using the laws that you have just learned, simplify the following Boolean expression:

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

# Comparing Number of Transistors Used



Original Circuit



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

Number of Transistors = 12

Simplified Circuit



$$Y = A$$

Number of Transistors = 0

# Boolean Algebra Practice



Simplify the following Boolean expression:  $B + \bar{B}C$

| Name         | OR Form                                       | AND Form                                    |
|--------------|-----------------------------------------------|---------------------------------------------|
| Identity     | $A + 0 = A$                                   | $A \cdot 1 = A$                             |
| Null         | $A + 1 = 1$                                   | $A \cdot 0 = 0$                             |
| Idempotent   | $A + A = A$                                   | $A \cdot A = A$                             |
| Complement   | $A + \bar{A} = 1$                             | $A \cdot \bar{A} = 0$                       |
| Commutative  | $A + B = B + A$                               | $A \cdot B = B \cdot A$                     |
| Associative  | $(A + B) + C = A + (B + C)$                   | $(A \cdot B) \cdot C = A \cdot (B \cdot C)$ |
| Distributive | $A \cdot (B + C) = (A \cdot B) + (A \cdot C)$ | $A + (B \cdot C) = (A + B) \cdot (A + C)$   |
| De Morgan's  | $\overline{A + B} = \bar{A} \cdot \bar{B}$    | $\overline{A \cdot B} = \bar{A} + \bar{B}$  |
| Absorption   | $A \cdot (A + B) = A$                         | $A + (A \cdot B) = A$                       |

# Recall: De Morgan's Theorem



$$\overline{A + B} = \overline{A} \cdot \overline{B}$$

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

De Morgan's Theorem:  
express conjunctions &  
disjunctions (ANDS, ORS)  
solely in terms of negation!

# Applying DeMorgan's



$$\overline{AB + CD}$$

- How would you apply DeMorgan's to this expression?
  - We group our terms based on the **lowest operator precedence** under the bar
  - The order of operations for Boolean Algebra is **NOT, then AND, then OR**. Expressions inside brackets are always evaluated first.
  - In this case, the operator with the lowest precedence is OR

$$(\overline{AB})(\overline{CD})$$

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

# DeMorgan's Practice



- How would you apply DeMorgan's to this expression?

$$\overline{\bar{A}B + \bar{C}}$$

# DeMorgan's Practice

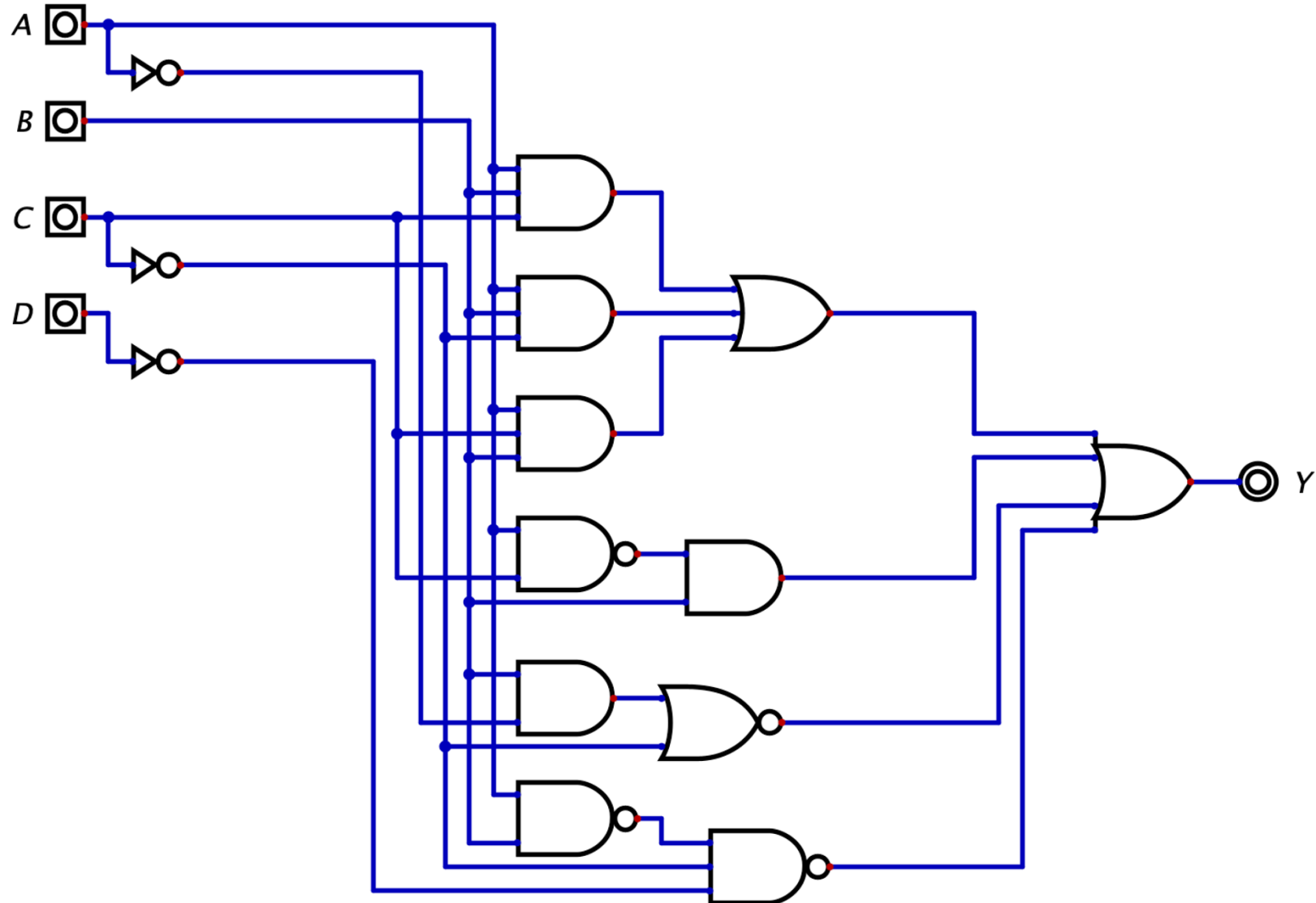


- How would you apply DeMorgan's to this expression?

$$\overline{(\bar{A} + B)(C + D)\overline{(E + F)} + G}$$

$$\begin{aligned}\overline{X + Y} &= \overline{X} \cdot \overline{Y} \\ \overline{XY} &= \overline{X} + \overline{Y}\end{aligned}$$

# Can we optimize this?

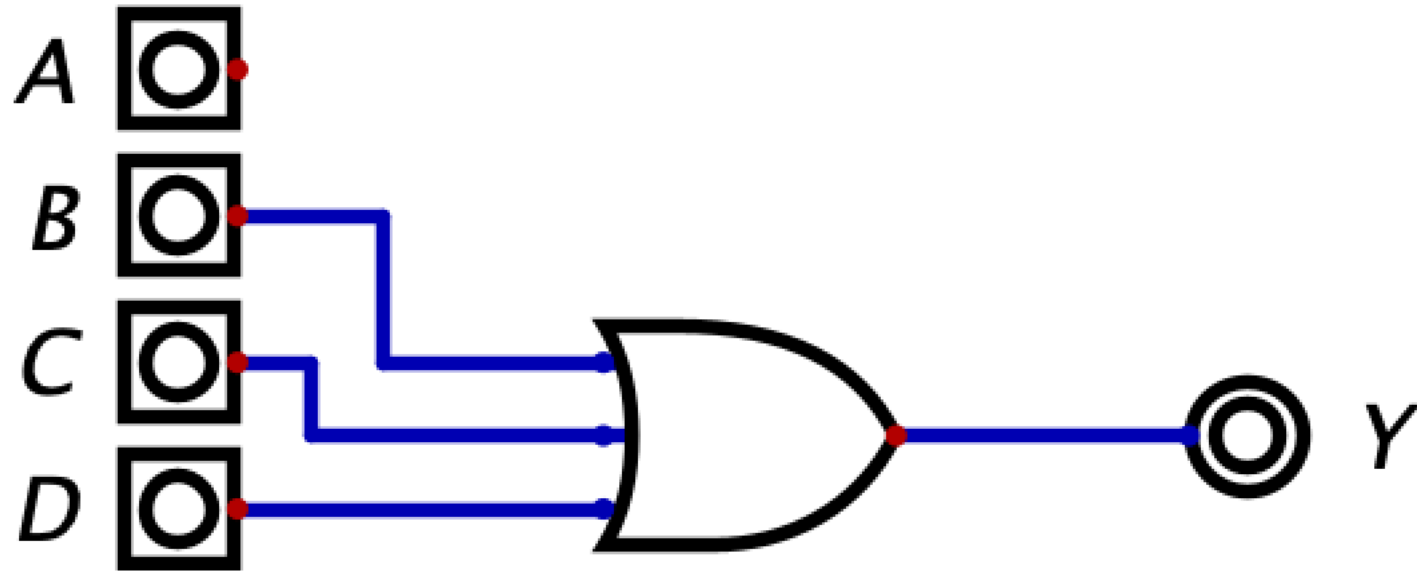


# Simplify the following Boolean expression



$$ABC + AB\bar{C} + \overline{AC}B + ACB + \overline{\overline{AB}\overline{C}\overline{D}} + \overline{\overline{AB} + \bar{C}}$$

# Optimizing Our Gates



Yay! We did it!

# Recall: Our Problem From Earlier

| $A_1$ | $A_0$ | $B_1$ | $B_0$ | Y |
|-------|-------|-------|-------|---|
| 0     | 0     | 0     | 0     | 0 |
| 0     | 0     | 1     | 0     | 0 |
| 0     | 0     | 0     | 1     | 0 |
| 0     | 0     | 1     | 1     | 0 |
| 1     | 0     | 0     | 0     | 1 |
| 1     | 0     | 1     | 0     | 0 |
| 1     | 0     | 0     | 1     | 1 |
| 1     | 0     | 1     | 1     | 0 |
| 0     | 1     | 0     | 0     | 1 |
| 0     | 1     | 1     | 0     | 0 |
| 0     | 1     | 0     | 1     | 0 |
| 0     | 1     | 1     | 1     | 0 |
| 1     | 1     | 0     | 0     | 1 |
| 1     | 1     | 1     | 0     | 1 |
| 1     | 1     | 0     | 1     | 1 |
| 1     | 1     | 1     | 1     | 0 |

I want to build a circuit that will output 1 if  $A > B$ , and 0 if  $A \leq B$ . A and B are both two-bit unsigned numbers.

Additionally, define  $A_1$  and  $B_1$  as the left bits of A and B, and  $A_0$  and  $B_0$  as the right bits.

$$\bar{A}_1 A_0 \bar{B}_1 \bar{B}_0 + A_1 \bar{A}_0 \bar{B}_1 \bar{B}_0 + A_1 \bar{A}_0 \bar{B}_1 B_0 + A_1 A_0 \bar{B}_1 \bar{B}_0 + A_1 A_0 \bar{B}_1 B_0 + A_1 A_0 B_1 \bar{B}_0$$

*Is the simplest possible form??* 🤔

# Simplify?



$$\bar{A}_1 A_0 \bar{B}_1 \bar{B}_0 + A_1 \bar{A}_0 \bar{B}_1 \bar{B}_0 + A_1 \bar{A}_0 \bar{B}_1 B_0 + A_1 A_0 \bar{B}_1 \bar{B}_0 + A_1 A_0 \bar{B}_1 B_0 + A_1 A_0 B_1 \bar{B}_0$$

# Recall: Our Problem From Earlier

| $A_1$ | $A_0$ | $B_1$ | $B_0$ | Y |
|-------|-------|-------|-------|---|
| 0     | 0     | 0     | 0     | 0 |
| 0     | 0     | 1     | 0     | 0 |
| 0     | 0     | 0     | 1     | 0 |
| 0     | 0     | 1     | 1     | 0 |
| 1     | 0     | 0     | 0     | 1 |
| 1     | 0     | 1     | 0     | 0 |
| 1     | 0     | 0     | 1     | 1 |
| 1     | 0     | 1     | 1     | 0 |
| 0     | 1     | 0     | 0     | 1 |
| 0     | 1     | 1     | 0     | 0 |
| 0     | 1     | 0     | 1     | 0 |
| 0     | 1     | 1     | 1     | 0 |
| 1     | 1     | 0     | 0     | 1 |
| 1     | 1     | 1     | 0     | 1 |
| 1     | 1     | 0     | 1     | 1 |
| 1     | 1     | 1     | 1     | 0 |

I want to build a circuit that will output 1 if  $A > B$ , and 0 if  $A \leq B$ . A and B are both two-bit unsigned numbers.

Additionally, define  $A_1$  and  $B_1$  as the left bits of A and B, and  $A_0$  and  $B_0$  as the right bits.

$$A_1\bar{B}_1 + A_0\bar{B}_1\bar{B}_0 + A_1A_0\bar{B}_0$$