ÇUKUROVA ÜNİVERSİTESİ

Department
of
Computer Engineering

CEN 263
Digital Design

Autumn 2024

Lecture 2

Announcements

- **Website**:

Week 2 Outlines

- Combinational logic
- Boolean algebra
- Simplification of Boolean expressions

Where are we now?

Processor (CPU)
- Datapath
- Control Logic
- Pipelining

Synchronous Digital Systems
- Sequential Logic
- **Combinational Logic**

Transistors

Note: Transistors will not be in scope for exams.

Logic Gates

- Operators with:
  - One or more 1-bit inputs
  - One 1-bit output
- Can be built out of transistors
- Can be represented as:
  - A block in a circuit diagram
  - A truth table, listing the output for every possible input
  - A Boolean algebra expression
- Used to perform bitwise operations
  - Recall: We saw bitwise operations (NOT, OR, AND, XOR)

NOT

- One 1-bit inputs, labeled A
- One 1-bit output, labeled Out
- Can be represented as:

A diagram

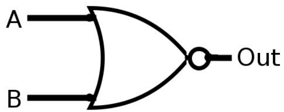
$$Out = \bar{A}$$

An algebraic expression

A	Out
0	1
1	0

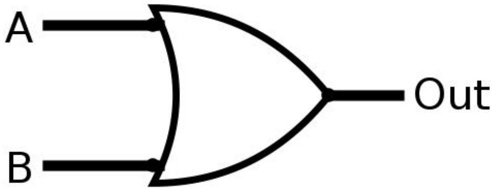
A truth table

OR



NOR gate looks like this.

- Two 1-bit inputs, labeled A and B
- One 1-bit output, labeled Out
- Can be represented as:



A diagram

Out = A + B

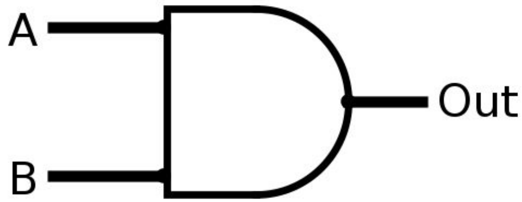
An algebraic expression

A	B	Out
0	0	0
0	1	1
1	0	1
1	1	1

A truth table

AND

- Two 1-bit inputs, labeled A and B
- One 1-bit output, labeled Out
- Can be represented as:



A diagram

Out = AB

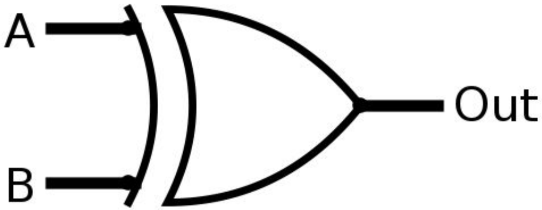
An algebraic expression

A	B	Out
0	0	0
0	1	0
1	0	0
1	1	1

A truth table

XOR

- Two 1-bit inputs, labeled A and B
- One 1-bit output, labeled Out
- Can be represented as:



A diagram

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

An algebraic expression

A	B	Out
0	0	0
0	1	1
1	0	1
1	1	0

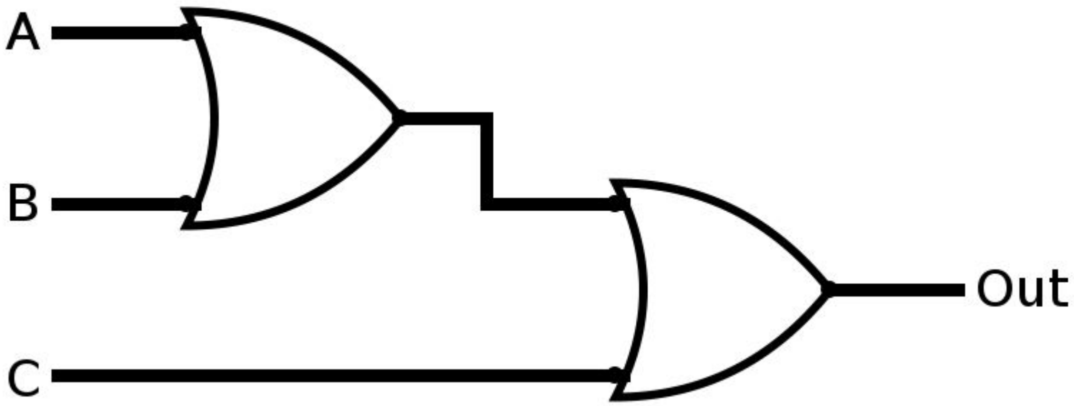
A truth table

Combinational Logic

- We can combine logic gates to make larger circuits
  - One or more inputs
  - One or more outputs
  - Perform more complicated logic operations
- These circuits can also be represented as
  - A block in a circuit diagram
  - A truth table, listing the output for every possible input
  - A Boolean algebra expression

Mystery Circuit #1

- Three 1-bit inputs, labeled A, B, C
- One 1-bit output, labeled Out

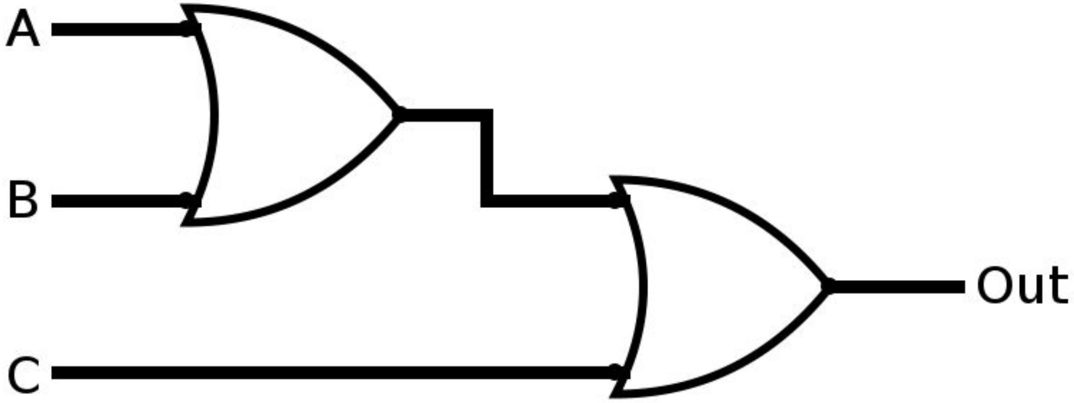


Let's try to write the truth table from the circuit diagram.

How many rows are in the truth table?
(Hint: How many unique inputs are there?)

Mystery Circuit #1

- Three 1-bit inputs, labeled A, B, C
- One 1-bit output, labeled Out

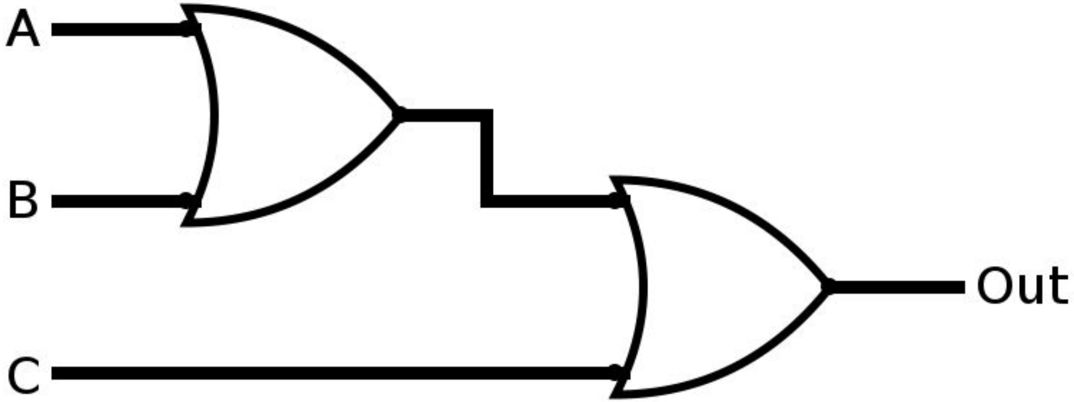


A	B	C	Out
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Try inputs on the circuit to fill out the truth table!

Mystery Circuit #1

- Three 1-bit inputs, labeled A, B, C
- One 1-bit output, labeled Out



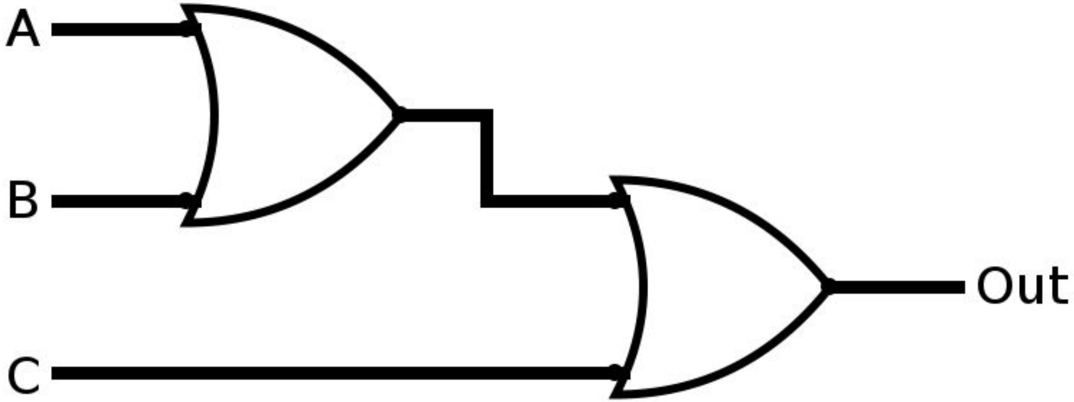
A	B	C	Out
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Let's write the algebraic expression from the circuit.

Mystery Circuit #1

- Three 1-bit inputs, labeled A, B, C
- One 1-bit output, labeled Out

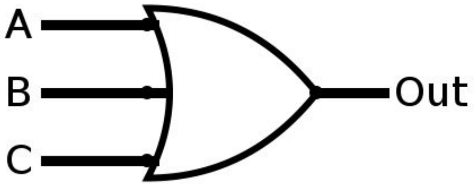
A	B	C	Out
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

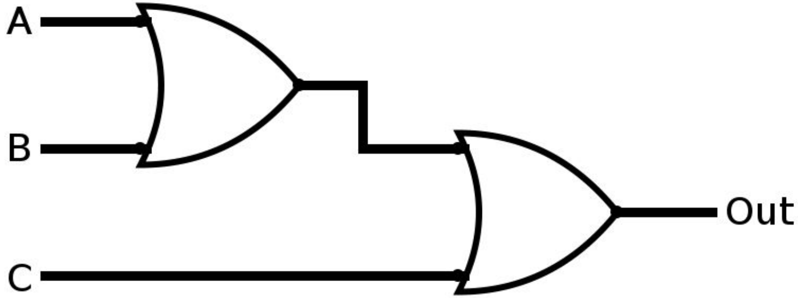


Out = (A + B) + C

Mystery Circuit #1: 3-way OR

- Intuitively: Outputs 1 when at least one of the inputs is 1
- Sometimes drawn like this:

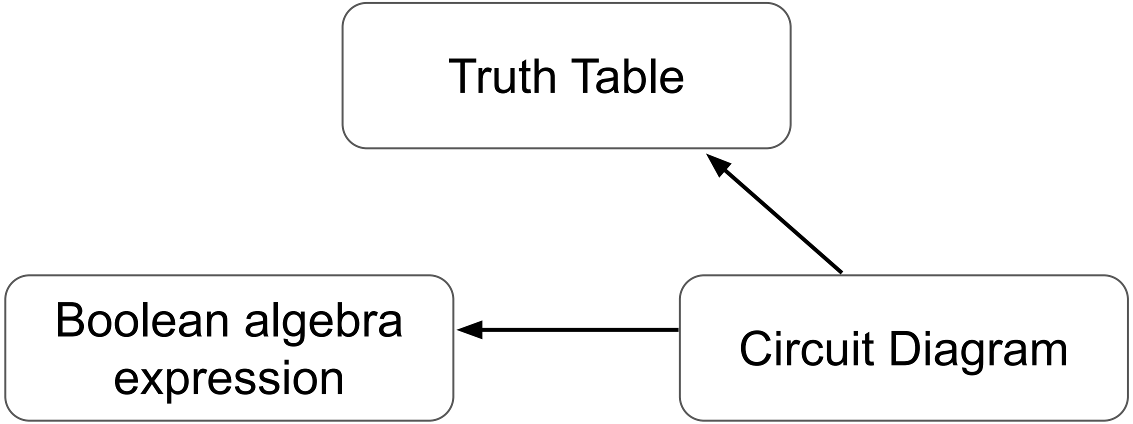




A	B	C	Out
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Out = (A + B) + C

Converting Between Representations



We converted a circuit diagram to a truth tables and algebraic expression.

Can we convert between the other representations too?

Mystery Circuit #2

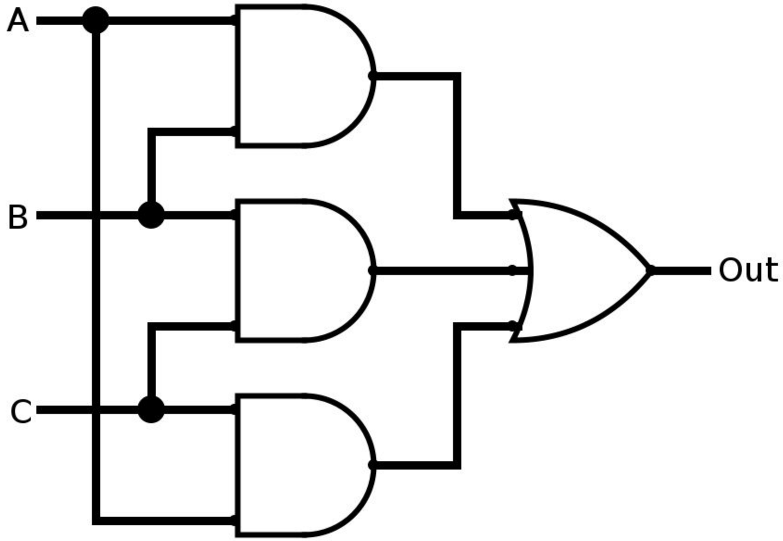
- Three 1-bit inputs, labeled A, B, C
- One 1-bit output, labeled Out

Given the algebraic expression, let's draw the circuit diagram.

Out = AB + BC + AC

Mystery Circuit #2

- Three 1-bit inputs, labeled A, B, C
- One 1-bit output, labeled Out



Out = AB + BC + AC

Mystery Circuit #2

- Three 1-bit inputs, labeled A, B, C
- One 1-bit output, labeled Out

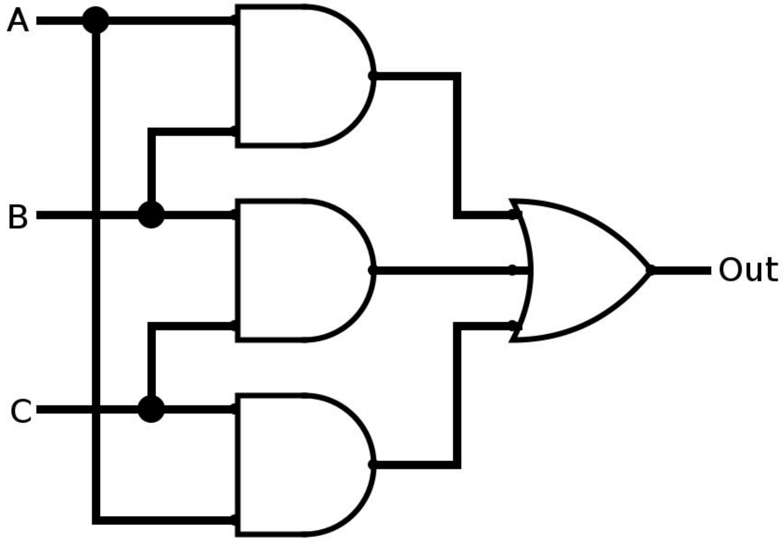
A	B	C	Out
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Given the algebraic expression, let's write the truth table.

Out = AB + BC + AC

Mystery Circuit #2

- Three 1-bit inputs, labeled A, B, C
- One 1-bit output, labeled Out

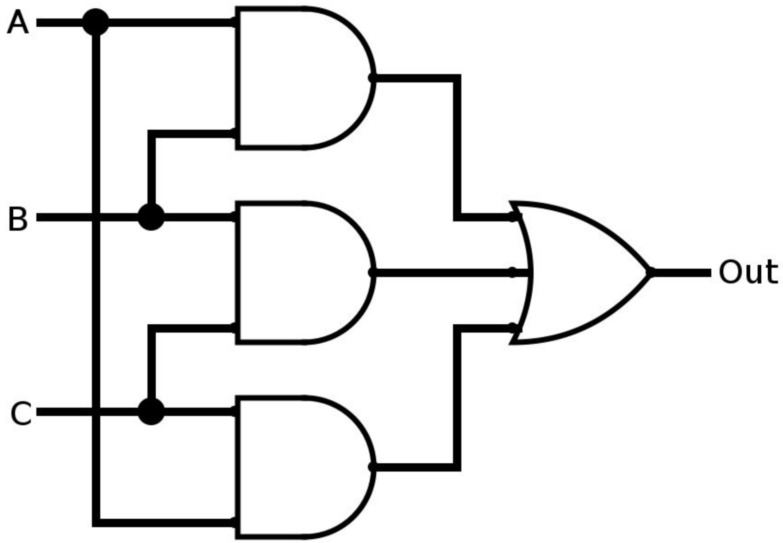


A	B	C	Out
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Out = AB + BC + AC

Mystery Circuit #2: Majority Circuit

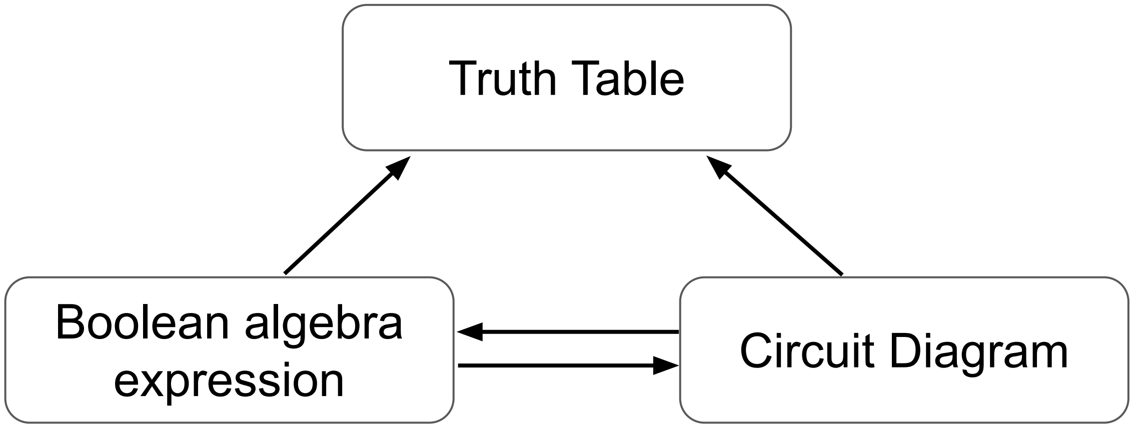
- Intuitively: Outputs the most common value (0 or 1) among the three inputs



A	B	C	Out
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

$$Out = AB + BC + AC$$

Converting Between Representations



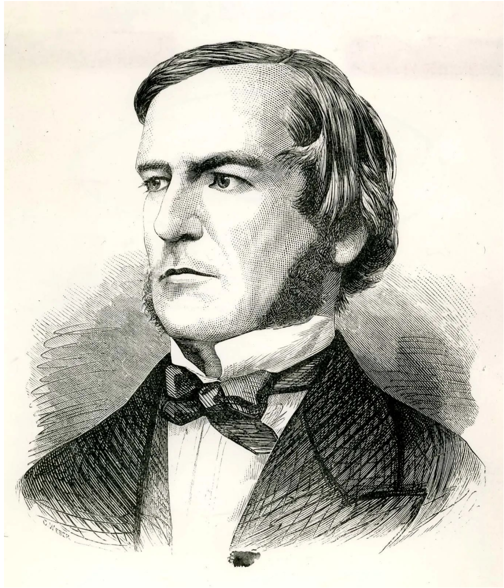
We just converted algebraic expressions to truth tables and circuit diagrams.

Before we see the next conversion, let's talk a bit more about Boolean algebra.

Boolean Algebra

History of Boolean Algebra

- Early computer designers built ad hoc circuits from switches
- Began to notice common patterns in their work: ANDs, ORs, …
- Master's thesis (by Claude Shannon, 1940) made link between work and 19th century mathematician George Boole
- Called it "Boolean Algebra" in his honor



George Boole

Boolean Algebra Operations

- Recall the Boolean algebra operations from earlier
  - We can compute these operations with logic gates

Operation	**Notation**
A and B	AB
A or B	A + B
not A	$\bar{A}$

Note: There are a variety of accepted symbols used for these operations.

Simplifying Boolean Algebra

- Given a complicated Boolean algebra expression, we can use some rules to simplify them
    - Useful way to simplify circuits: convert to Boolean algebra, simplify, and convert back
    - Simplifying circuits = less hardware

Laws of Boolean Algebra

Name	AND Form	OR form
Commutative	$AB = BA$	$A + B = B + A$
Associative	$AB(C) = A(BC)$	$A + (B + C) = (A + B) + C$
Identity	$1A = A$	$0 + A = A$
Null	$0A = 0$	$1 + A = 1$
Absorption	$A(A + B) = A$	$A + AB = A$
Distributive	$(A + B)(A + C) = A + BC$	$A(B + C) = AB + AC$
Idempotent	$A(A) = A$	$A + A = A$
Inverse	$A(\overline{A}) = 0$	$A + \overline{A} = 1$
De Morgan's	$\overline{AB} = \overline{A} + \overline{B}$	$\overline{A + B} = \overline{A}(\overline{B})$

Boolean Algebra Simplification Example

y	$= ab + a + c$	
	$= ab + a(1) + c$	Identity
	$= a(b+1) + c$	Distributive
	$= a(1) + c$	Null
	$= a + c$	Identity

Sum-of-Products

- Given a truth table, how can we convert it to a Boolean algebra expression?
- Idea: Look for every row where the output is 1
  - For Out=1, we either have:
  - (A=0 and B=0 and C=1), or
  - (A=1 and B=0 and C=1), or
  - (A=1 and B=1 and C=1)
- In Boolean algebra: $(\overline{A})(\overline{B})(C) + (A)(\overline{B})(C) + (A)(B)(C)$
- This is called the sum-of-products form
  - For each row where Out=1, create one product (AND the inputs together)
  - Then take the sum of all the products (OR the products together)

A	B	C	Out
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Mystery Circuit #3

- Three 1-bit inputs, labeled A, B, S
- One 1-bit output, labeled Out

Given the truth table, let's write the algebraic expression.

Strategy: Start with sum-of-products form, then simplify with Boolean algebra laws.

A	B	S	Out
0	0	0	0
1	0	0	1
0	0	1	0
1	0	1	0
0	1	0	0
1	1	0	1
0	1	1	1
1	1	1	1

Mystery Circuit #3

- Three 1-bit inputs, labeled A, B, S
- One 1-bit output, labeled Out

$$\text{Out} = (a)(\overline{b})(\overline{s}) + (a)(b)(\overline{s}) + (\overline{a})(b)(s) + abs$$

$$= (\ (a)(\overline{b}) + ab\ )(\overline{s}) + (\ (\overline{a})(b) + ab\ )s$$

$$= (\ (a)(\overline{b} + b)\ )(\overline{s}) + (\ (\overline{a} + a)(b)\ )s$$

$$= (a(1))(\overline{s}) + ((1)b)s$$

$$= a(\overline{s}) + bs$$

A	B	S	Out
0	0	0	0
1	0	0	1
0	0	1	0
1	0	1	0
0	1	0	0
1	1	0	1
0	1	1	1
1	1	1	1

Mystery Circuit #3

- Three 1-bit inputs, labeled A, B, S
- One 1-bit output, labeled Out

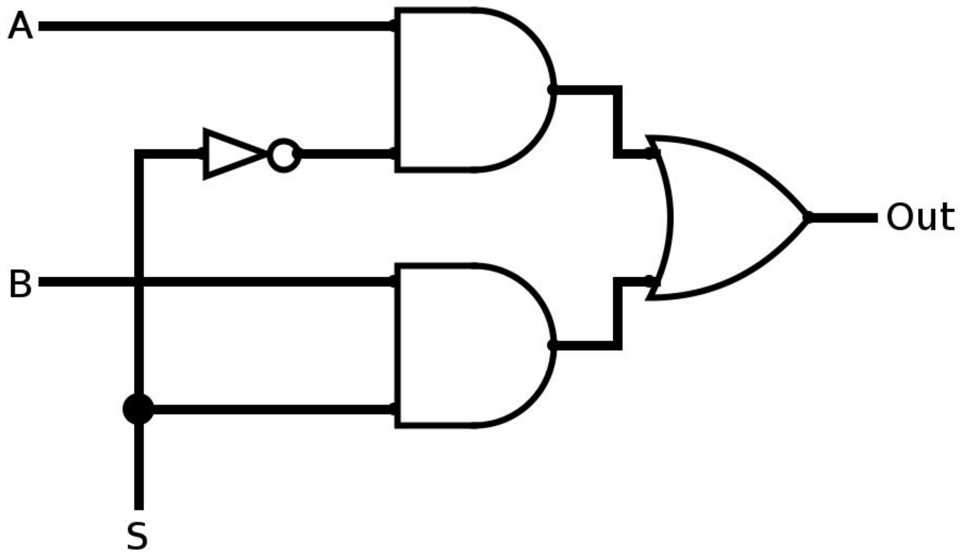
Given the simplified algebraic expression, we can draw a circuit diagram with fewer logic gates (compared to drawing a diagram from sum-of-products form).

A	B	S	Out
0	0	0	0
1	0	0	1
0	0	1	0
1	0	1	0
0	1	0	0
1	1	0	1
0	1	1	1
1	1	1	1

$$Out = (A)(\overline{S}) + (B)(S)$$

Mystery Circuit #3

- Three 1-bit inputs, labeled A, B, S
- One 1-bit output, labeled Out

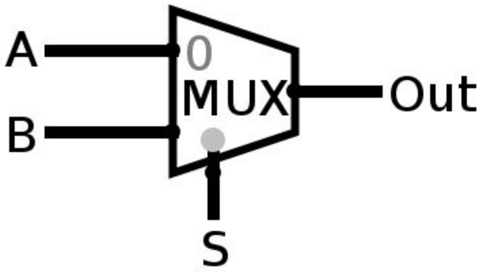


A	B	S	Out
0	0	0	0
1	0	0	1
0	0	1	0
1	0	1	0
0	1	0	0
1	1	0	1
0	1	1	1
1	1	1	1

$$Out = (A)(\overline{S}) + (B)(S)$$

Mystery Circuit #3: Multiplexer (MUX)

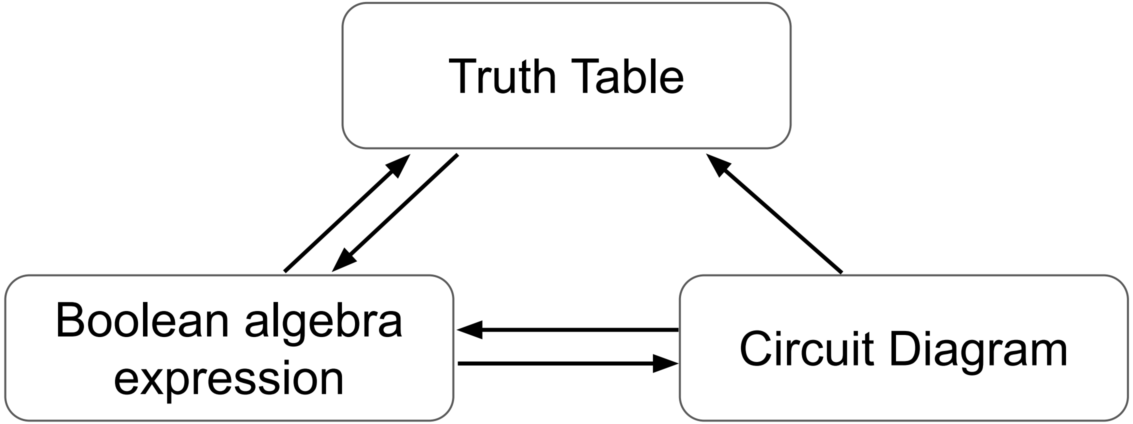
- Intuitively:
  - If S is 0, output the value in A.
  - If S is 1, output the value in B.
  - S is the “select” bit
- Usually drawn like this:



A	B	S	Out
0	0	0	0
1	0	0	1
0	0	1	0
1	0	1	0
0	1	0	0
1	1	0	1
0	1	1	1
1	1	1	1

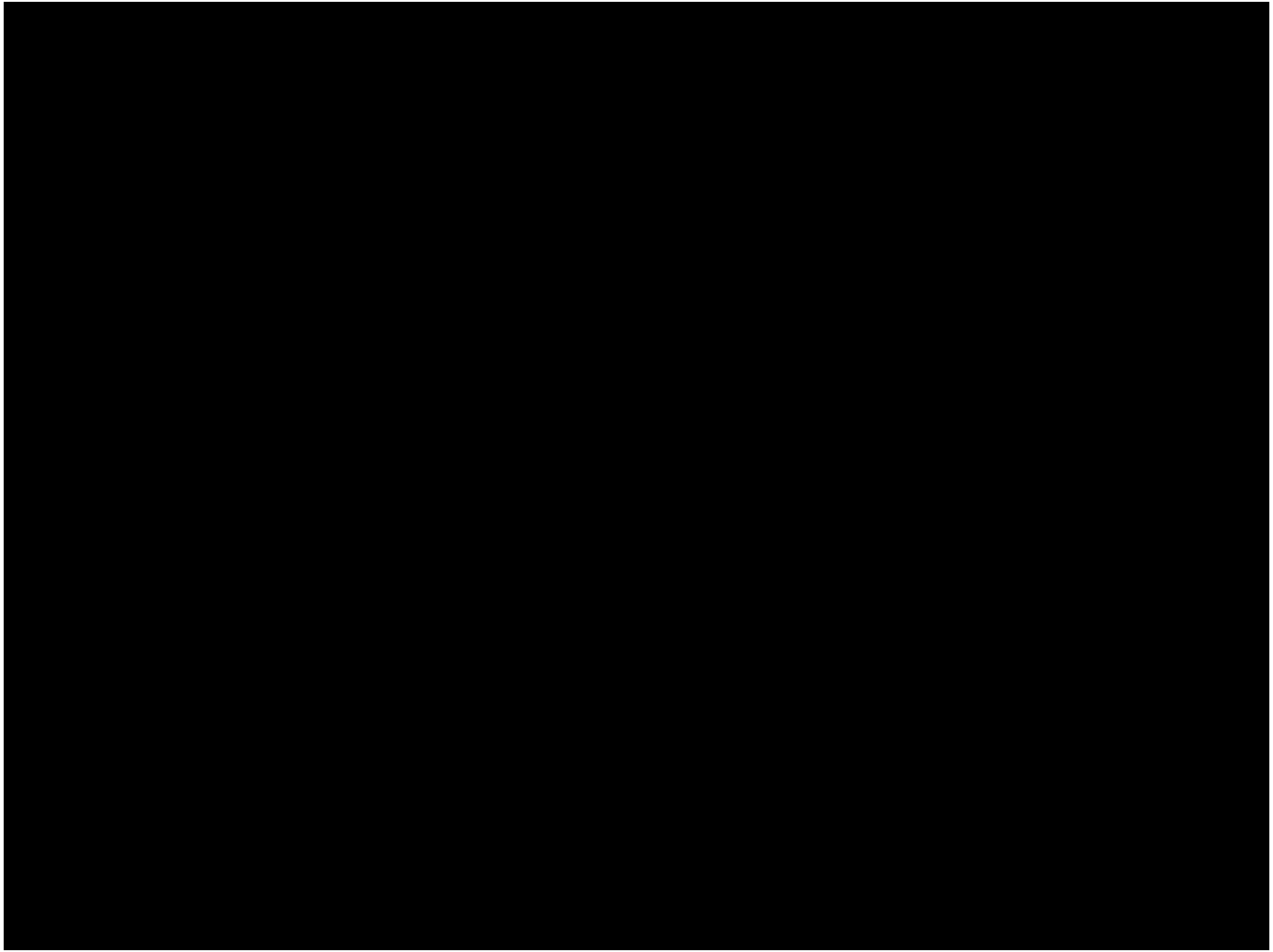
$$Out = (A)(\overline{S}) + (B)(S)$$

Converting Between Representations



Now, we can convert truth tables to algebraic expressions.

Adders



32-bit Adder

- Inputs:
  - 32-bit input A
  - 32-bit input B
- Outputs:
  - 32-bit output Sum
- How many rows in the truth table?
  - 2^{32} possible inputs for each of A and B
  - 2^{64} possible inputs in total
- Can we build this more efficiently?
  - Yes - connect smaller building blocks together!

1-bit Adder

How do we add binary numbers by hand?

	0	0	1	0
+	0	1	1	1

1-bit Adder

Consider the inputs and outputs in one column:

	(1)	(1)	(0)	
	0	0	1	0
+	0	1	1	1
	1	0	0	1

Inputs:

- 0, 1: Bits being added
- (1): Carry-in bit

Outputs:

- 0: Sum
- (1): Carry-out bit

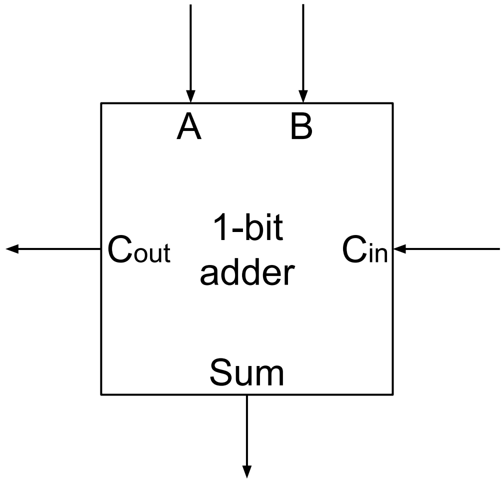
1-bit Adder

- Three 1-bit inputs, labeled A, B, C_{in}
- Two 1-bit outputs, labeled Sum and C_{out}

A	B	C_{in}	Sum	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

1-bit Adder

- Three 1-bit inputs, labeled A, B, C_{in}
- Two 1-bit outputs, labeled Sum and C_{out}



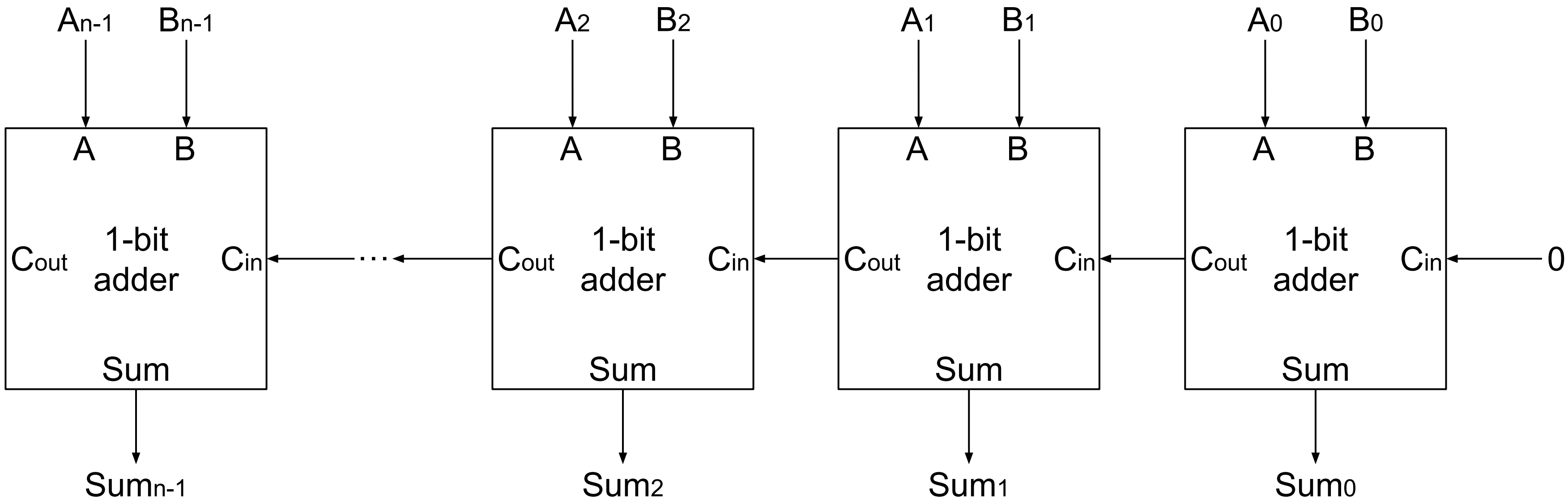
A	B	C_{in}	Sum	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$C_{out} = BC_{in} + AC_{in} + AB$

$\text{Sum} = A \oplus B \oplus C_{in}$

n-bit Adder

- Idea: Chain *n* 1-bit adders together to add *n*-bit numbers together



End of Week 2!

Do not go home, we’ll have another lecture…