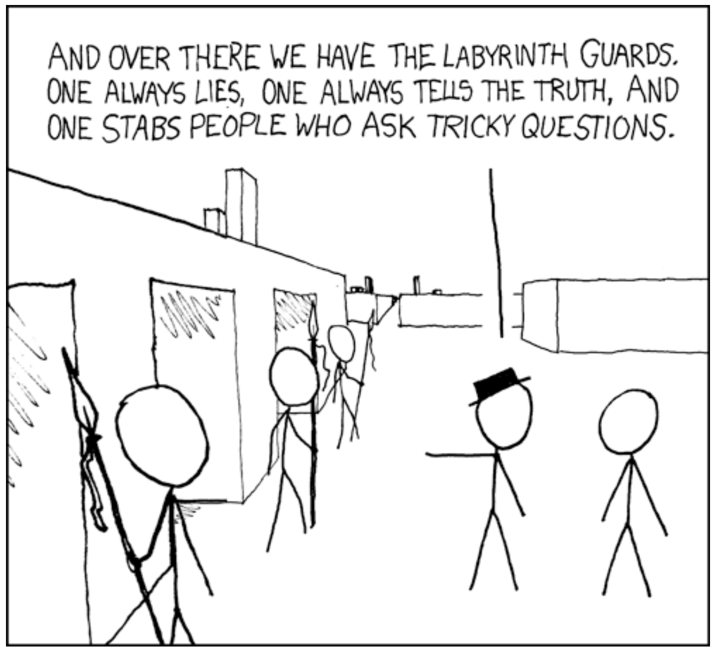
# CSE 311: Foundations of Computing

## Topic 2: Equivalence



# Tautologies!

Terminology: A compound proposition is a…

- *Tautology* if it is always true
- *Contradiction* if it is always false
- *Contingency* if it can be either true or false

$p \vee \neg p$

$p \oplus p$

$(p \rightarrow r) \wedge p$

# Tautologies!

Terminology: A compound proposition is a…

- *Tautology* if it is always true
- *Contradiction* if it is always false
- *Contingency* if it can be either true or false

$p \vee \neg p$

This is a tautology. It's called the "law of the excluded middle".
If p is true, then $p \vee \neg p$ is true. If p is false, then $p \vee \neg p$ is true.

$p \oplus p$

This is a contradiction. It's always false no matter what truth value p takes on.

$(p \rightarrow r) \wedge p$

This is a contingency. When p=T, r=T, $(T \rightarrow T) \wedge T$ is true.
When p=T, r=F, $(T \rightarrow F) \wedge T$ is false.

# Mapping Truth Tables to Logic Gates

Given a truth table:

1. Write the output in a table
2. Write the Boolean expression
3. Draw as gates
4. Map to available gates

| A | B | C | F |
|---|---|---|---|
| 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 |

This will give us *some* circuit.
But is it the <u>best</u> circuit?

# Logical Equivalence

A = B means A and B are the same thing written twice:

- $p \wedge r = p \wedge r$

- $p \wedge r \neq r \wedge p$

# Logical Equivalence

A = B means A and B are same thing written twice:

- $p \wedge r = p \wedge r$

  These are equal, because they are character-for-character identical.

- $p \wedge r \neq r \wedge p$

  These are NOT equal, because they are different sequences of characters. They “mean” the same thing though.

in more detail, “=” means same parse tree (see week 8), so we can ignore differences in whitespace etc.

# Logical Equivalence

A = B means A and B are same thing written twice:

- $p \wedge r = p \wedge r$

  These are equal, because they are character-for-character identical.

- $p \wedge r \neq r \wedge p$

  These are NOT equal, because they are different sequences of characters. They "mean" the same thing though.

$A \equiv B$ means A and B have identical truth values:

- $p \wedge r \equiv p \wedge r$

- $p \wedge r \equiv r \wedge p$

- $p \wedge r \not\equiv r \vee p$

# Logical Equivalence

A = B means A and B are same thing written twice:

- $p \wedge r = p \wedge r$

  These are equal, because they are character-for-character identical.

- $p \wedge r \neq r \wedge p$

  These are NOT equal, because they are different sequences of characters. They "mean" the same thing though.

A ≡ B means A and B have identical truth values:

- $p \wedge r \equiv p \wedge r$

  Two formulas that are **equal** also are equivalent.

- $p \wedge r \equiv r \wedge p$

  These two formulas have the same truth table!

- $p \wedge r \not\equiv r \vee p$

  When $p$=T and $r$=F, $p \wedge r$ is false, but $p \vee r$ is true!

## $A \leftrightarrow B$ vs. $A \equiv B$

$A \leftrightarrow B$ is a **proposition** that may be true or false depending on the truth values of A and B.

$A \equiv B$ is an **assertion** over all possible truth values that A and B always have the same truth values.

$A \equiv B$ and $(A \leftrightarrow B) \equiv T$ have the same meaning as does "$A \leftrightarrow B$ is a tautology"

# Logical Equivalence $A \equiv B$

$A \equiv B$ is an assertion that ***two propositions*** A and B always have the same truth values.

$A \equiv B$ and $(A \leftrightarrow B) \equiv T$ have the same meaning.

$p \wedge r \equiv r \wedge p$

| $p$ | $r$ | $p \wedge r$ | $r \wedge p$ | $(p \wedge r) \leftrightarrow (r \wedge p)$ |
|---|---|---|---|---|
| T | T | T | T | T |
| T | F | F | F | T |
| F | T | F | F | T |
| F | F | F | F | T |

# Examples from Last Lecture

## We previously saw 3 different ways of writing XOR

| *a* | *b* | *a’* | *b’* | *a’b* | *ab’* | *a’b + b’a* |
|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 |

sum of products

| *a* | *b* | *a + b* | *a’* | *b’* | *a’ + b’* | *(a+b)(a’+b’)* |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 | 1 | 1 | 0 |

product of sums

# Exercise: XOR

## We previously saw 3 different ways of writing XOR

| *a* | *b* | *a’* | *b’* | *a’b* | *ab’* | *a’b + b’a* |
|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 |

sum of products

| *a* | *b* | *a + b* | *ab* | *(ab)’* | *(a+b)(ab)’* |
|---|---|---|---|---|---|
| 1 | 1 | | | | |
| 1 | 0 | | | | |
| 0 | 1 | | | | |
| 0 | 0 | | | | |

original definition

# Exercise: XOR

## We previously saw 3 different ways of writing XOR

| *a* | *b* | *a'* | *b'* | *a'b* | *ab'* | *a'b + b'a* |
|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 |

sum of products

| *a* | *b* | *a + b* | *ab* | *(ab)'* | *(a+b)(ab)'* |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 |

original definition

# De Morgan's Laws

$$\neg(p \wedge r) \equiv \neg p \vee \neg r$$
$$\neg(p \vee r) \equiv \neg p \wedge \neg r$$

Negate the statement:
"My code compiles or there is a bug."

To negate the statement,
ask "when is the original statement false".

It's false when not(my code compiles) AND not(there is a bug).

Translating back into English, we get:
My code doesn't compile and there is not a bug.

# De Morgan's Laws

Example: $\neg(p \wedge r) \equiv \neg p \vee \neg r$

| $p$ | $r$ | $\neg p$ | $\neg r$ | $\neg p \vee \neg r$ | $p \wedge r$ | $\neg(p \wedge r)$ |
|---|---|---|---|---|---|---|
| T | T | F | F | F | T | F |
| T | F | F | T | T | F | T |
| F | T | T | F | T | F | T |
| F | F | T | T | T | F | T |

# De Morgan's Laws

$$\neg(p \wedge r) \equiv \neg p \vee \neg r$$

$$\neg(p \vee r) \equiv \neg p \wedge \neg r$$

```
if (!(front != null && value > front.data)) {
    front = new ListNode(value, front);
} else {
    ListNode current = front;
    while (current.next != null && current.next.data < value))
        current = current.next;
    current.next = new ListNode(value, current.next);
}
```

# De Morgan's Laws

$$\neg(p \wedge r) \equiv \neg p \vee \neg r$$

$$\neg(p \vee r) \equiv \neg p \wedge \neg r$$

```
!(front != null && value > front.data)
```

$$\equiv$$

```
front == null || value <= front.data
```

# Law of Implication

$$p \to r \equiv \neg p \lor r$$

| $p$ | $r$ | $p \to r$ | $\neg p$ | $\neg p \lor r$ |
|---|---|---|---|---|
| T | T | | | |
| T | F | | | |
| F | T | | | |
| F | F | | | |

# Law of Implication

$$p \rightarrow r \equiv \neg p \vee r$$

| $p$ | $r$ | $p \rightarrow r$ | $\neg p$ | $\neg p \vee r$ |
|---|---|---|---|---|
| T | T | T | F | T |
| T | F | F | F | F |
| F | T | T | T | T |
| F | F | T | T | T |

# Biconditional: $p \leftrightarrow r$

- *p* if and only if *r* (*p* iff *r*)
- *p* implies *r* and *r* implies *p*
- *p* is necessary and sufficient for *r*

| $p$ | $r$ | $p \leftrightarrow r$ | $p \rightarrow r$ | $r \rightarrow p$ | $(p \rightarrow r) \wedge (r \rightarrow p)$ |
|---|---|---|---|---|---|
| T | T | T | T | T | |
| T | F | F | F | T | |
| F | T | F | T | F | |
| F | F | T | T | T | |

# Biconditional: $p \leftrightarrow r$

- *p* if and only if *r* (*p* iff *r*)
- *p* implies *r* and *r* implies *p*
- *p* is necessary and sufficient for *r*

| $p$ | $r$ | $p \leftrightarrow r$ | $p \rightarrow r$ | $r \rightarrow p$ | $(p \rightarrow r) \wedge (r \rightarrow p)$ |
|---|---|---|---|---|---|
| T | T | T | T | T | T |
| T | F | F | F | T | F |
| F | T | F | T | F | F |
| F | F | T | T | T | T |

# Some Familiar Properties of Arithmetic

- $x + y = y + x$ **(Commutativity)**
- $x \cdot (y + z) = x \cdot y + x \cdot z$ **(Distributivity)**
- $(x + y) + z = x + (y + z)$ **(Associativity)**

# Important Equivalences

- Identity
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- Domination
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- Idempotent
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- Commutative
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- Associative
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- Distributive
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- Absorption
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- Negation
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

# Some Familiar Properties of Arithmetic

- $x \cdot 1 = x$ **(Identity)**
- $x + 0 = x$

- $x \cdot 0 = 0$ **(Domination)**

# Important Equivalences

- **Identity**
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- **Domination**
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- **Idempotent**
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- **Commutative**
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- **Associative**
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- **Distributive**
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- **Absorption**
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- **Negation**
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

## Some Familiar Properties of Arithmetic

- Usual properties hold under relabeling:
  - 0, 1 becomes F, T
  - "+" becomes "$\vee$"
  - " $\cdot$ " becomes "$\wedge$"
- But there are some new facts:
  - Distributivity works for both "$\wedge$" and "$\vee$"
  - Domination works with T
- There are some other facts specific to logic…

# Important Equivalences

- Identity
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- Domination
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- Idempotent
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- Commutative
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- Associative
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- Distributive
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- Absorption
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- Negation
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

# Important Equivalences

- **Identity**
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- **Domination**
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- **Idempotent**
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- **Commutative**
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- **Associative**
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- **Distributive**
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- **Absorption**
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- **Negation**
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

# Using Equivalences

- Note that p, q, and r can be **any** propositions (not just atomic propositions)

- Ex: $(r \rightarrow s) \wedge (\neg t) \equiv (\neg t) \wedge (r \rightarrow s)$

  – apply commutativity: $p \wedge q \equiv q \wedge p$
  with $p := r \rightarrow s$
  and $q := \neg t$

# One more easy equivalence

## Double Negation

$$p \equiv \neg\neg p$$

| $p$ | $\neg p$ | $\neg\neg p$ |
| --- | --- | --- |
| T | F | T |
| F | T | F |

# Understanding logic and circuits

When do two logic formulas mean the same thing?

When do two circuits compute the same function?

What logical properties can we infer from other ones?

# Basic rules of reasoning and logic

- Working with logical formulas
  - Simplification
  - Testing for equivalence
- Applications
  - Query optimization
  - Search optimization and caching
  - Artificial Intelligence
  - Program verification

# Computing Equivalence

Given two propositions, can we write an algorithm to determine if they are equivalent?

What is the runtime of our algorithm?

# Computing Equivalence

Given two propositions, can we write an algorithm to determine if they are equivalent?

Yes! Generate the truth tables for both propositions and check if they are the same for every entry.

What is the runtime of our algorithm?

Every atomic proposition has two possibilities (T, F). If there are $n$ atomic propositions, there are $2^n$ rows in the truth table.

# Another approach: Logical Proofs

## To show A is equivalent to B

– Apply a series of logical equivalences to sub-expressions to convert A to B

## To show A is a tautology

– Apply a series of logical equivalences to sub-expressions to convert A to **T**

# Another approach: Logical Proofs

## To show A is equivalent to B

– Apply a series of logical equivalences to sub-expressions to convert A to B

Example:

Let A be "$p \lor (p \land p)$", and B be "$p$".

Our general equivalence proof looks like:

$$p \lor (p \land p) \equiv (\qquad\qquad\qquad\qquad)$$
$$\equiv p$$

# Another approach: Logical Equivalences

- **Identity**
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- **Domination**
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- **Idempotent**
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- **Commutative**
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- **Associative**
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- **Distributive**
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- **Absorption**
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- **Negation**
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

**De Morgan's Laws**

$$\neg(p \wedge q) \equiv \neg p \vee \neg q$$
$$\neg(p \vee q) \equiv \neg p \wedge \neg q$$

**Law of Implication**

$$p \rightarrow q \equiv \neg p \vee q$$

**Contrapositive**

$$p \rightarrow q \equiv \neg q \rightarrow \neg p$$

**Biconditional**

$$p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$$

**Double Negation**

$$p \equiv \neg \neg p$$

## Example:

Let A be "$p \vee (p \wedge p)$", and B be "$p$".

Our general equivalence proof looks like:

$$p \vee (p \wedge p) \equiv (\qquad\qquad\qquad\qquad)$$
$$\equiv p$$

# Logical Equivalences

- Identity
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- Domination
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- Idempotent
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- Commutative
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- Associative
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- Distributive
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- Absorption
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- Negation
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

De Morgan's Laws

$$\neg(p \wedge q) \equiv \neg p \vee \neg q$$
$$\neg(p \vee q) \equiv \neg p \wedge \neg q$$

Law of Implication

$$p \rightarrow q \equiv \neg p \vee q$$

Contrapositive

$$p \rightarrow q \equiv \neg q \rightarrow \neg p$$

Biconditional

$$p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$$

Double Negation

$$p \equiv \neg\neg p$$

**Example:**

**Let A be "$p \vee (p \wedge p)$", and B be "$p$".**

**Our general equivalence proof looks like:**

$$p \vee (p \wedge p) \equiv ( \quad p \vee p \quad ) \qquad \text{Idempotent}$$
$$\equiv p \qquad \text{Idempotent}$$

# Logical Equivalences

## To show A is a tautology

- Apply a series of logical equivalences to sub-expressions to convert A to **T**

Example:

Let A be "$\neg p \vee (p \vee p)$".

Our general equivalence proof looks like:

$$\begin{aligned} \neg p \vee (p \vee p) &\equiv (\qquad\qquad\qquad\qquad) \\ &\equiv (\qquad\qquad\qquad\qquad) \\ &\equiv \mathbf{T} \end{aligned}$$

# Logical Equivalences

- **Identity**
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- **Domination**
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- **Idempotent**
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- **Commutative**
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$

- **Associative**
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- **Distributive**
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- **Absorption**
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- **Negation**
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

**De Morgan's Laws**

$$\neg(p \wedge q) \equiv \neg p \vee \neg q$$
$$\neg(p \vee q) \equiv \neg p \wedge \neg q$$

**Law of Implication**

$$p \rightarrow q \equiv \neg p \vee q$$

**Contrapositive**

$$p \rightarrow q \equiv \neg q \rightarrow \neg p$$

**Biconditional**

$$p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$$

Example:
Let A be "$\neg p \vee (p \vee p)$".
Our general equivalence proof looks like:

**Example:**

**Let A be "$\neg p \vee (p \vee p)$".**

**Our general equivalence proof looks like:**

$$\neg p \vee (p \vee p) \equiv (\qquad\qquad\qquad\qquad)$$
$$\equiv (\qquad\qquad\qquad\qquad)$$
$$\equiv \mathrm{T}$$

# Logical Equivalences

- **Identity**
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- **Domination**
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- **Idempotent**
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- **Commutative**
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$

- **Associative**
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- **Distributive**
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- **Absorption**
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- **Negation**
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

**De Morgan's Laws**

$$\neg(p \wedge q) \equiv \neg p \vee \neg q$$
$$\neg(p \vee q) \equiv \neg p \wedge \neg q$$

**Law of Implication**

$$p \rightarrow q \equiv \neg p \vee q$$

**Contrapositive**

$$p \rightarrow q \equiv \neg q \rightarrow \neg p$$

**Biconditional**

$$p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$$

Example:
Let A be "$\neg p \vee (p \vee p)$".
Our general equivalence proof looks like:


**Example:**

Let A be "$\neg p \vee (p \vee p)$".

Our general equivalence proof looks like:


$$\neg p \vee (p \vee p) \equiv (\neg p \vee p) \quad \text{Idempotent}$$
$$\equiv (p \vee \neg p) \quad \text{Commutative}$$
$$\equiv \mathbf{T} \quad \text{Negation}$$

# Prove these propositions are equivalent: Option 1

**Prove:** $p \wedge (p \rightarrow r) \equiv p \wedge r$

**Make a Truth Table and show:**

$$(p \wedge (p \rightarrow r)) \leftrightarrow (p \wedge r) \equiv \mathbf{T}$$

| $p$ | $r$ | $p \rightarrow r$ | $(p \wedge (p \rightarrow r))$ | $p \wedge r$ | $(p \wedge (p \rightarrow r)) \leftrightarrow (p \wedge r)$ |
|---|---|---|---|---|---|
| T | T | T | T | T | T |
| T | F | F | F | F | T |
| F | T | T | F | F | T |
| F | F | T | F | F | T |

# Prove these propositions are equivalent: Option 2

## Prove: $p \wedge (p \rightarrow r) \equiv p \wedge r$

$$p \wedge (p \rightarrow r) \equiv$$
$$\equiv$$
$$\equiv$$
$$\equiv$$
$$\equiv p \wedge r$$

- **Identity**
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- **Domination**
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- **Idempotent**
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- **Commutative**
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- **Associative**
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- **Distributive**
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- **Absorption**
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- **Negation**
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

**De Morgan's Laws**

$\neg(p \wedge q) \equiv \neg p \vee \neg q$
$\neg(p \vee q) \equiv \neg p \wedge \neg q$

**Law of Implication**

$p \rightarrow q \equiv \neg p \vee q$

**Contrapositive**

$p \rightarrow q \equiv \neg q \rightarrow \neg p$

**Biconditional**

$p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$

**Double Negation**

$p \equiv \neg\neg p$

# Prove these propositions are equivalent: Option 2

## Prove: $p \wedge (p \rightarrow r) \equiv p \wedge r$

| | |
|---|---|
| $p \wedge (p \rightarrow r) \equiv p \wedge (\neg p \vee r)$ | Law of Implication |
| $\equiv (p \wedge \neg p) \vee (p \wedge r)$ | Distributive |
| $\equiv \mathbf{F} \vee (p \wedge r)$ | Negation |
| $\equiv (p \wedge r) \vee \mathbf{F}$ | Commutative |
| $\equiv p \wedge r$ | Identity |

- **Identity**
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- **Domination**
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- **Idempotent**
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- **Commutative**
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- **Associative**
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- **Distributive**
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- **Absorption**
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- **Negation**
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

**De Morgan's Laws**

$$\neg(p \wedge q) \equiv \neg p \vee \neg q$$
$$\neg(p \vee q) \equiv \neg p \wedge \neg q$$

**Law of Implication**

$p \rightarrow q \equiv \neg p \vee q$

**Contrapositive**

$p \rightarrow q \equiv \neg q \rightarrow \neg p$

**Biconditional**

$p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$

**Double Negation**

$$p \equiv \neg\neg p$$

# Prove this is a Tautology: Option 1

$$(p \wedge r) \rightarrow (r \vee p)$$

**Make a Truth Table and show:**

$$(p \wedge r) \rightarrow (r \vee p) \equiv \mathbf{T}$$

| $\boldsymbol{p}$ | $\boldsymbol{r}$ | $\boldsymbol{p \wedge r}$ | $\boldsymbol{r \vee p}$ | $\boldsymbol{(p \wedge r) \rightarrow (r \vee p)}$ |
|---|---|---|---|---|
| **T** | **T** | | | |
| **T** | **F** | | | |
| **F** | **T** | | | |
| **F** | **F** | | | |

# Prove this is a Tautology: Option 1

$$(p \wedge r) \rightarrow (r \vee p)$$

**Make a Truth Table and show:**

$$(p \wedge r) \rightarrow (r \vee p) \equiv \mathbf{T}$$

| $\boldsymbol{p}$ | $\boldsymbol{r}$ | $\boldsymbol{p \wedge r}$ | $\boldsymbol{r \vee p}$ | $\boldsymbol{(p \wedge r) \rightarrow (r \vee p)}$ |
|---|---|---|---|---|
| **T** | **T** | **T** | **T** | **T** |
| **T** | **F** | **F** | **T** | **T** |
| **F** | **T** | **F** | **T** | **T** |
| **F** | **F** | **F** | **F** | **T** |

# Prove this is a Tautology: Option 2

**Associative**
- $(p \vee q) \vee r \equiv p \vee (q \vee r)$
- $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$

**Distributive**
- $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
- $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$

**Absorption**
- $p \vee (p \wedge q) \equiv p$
- $p \wedge (p \vee q) \equiv p$

**Negation**
- $p \vee \neg p \equiv \mathrm{T}$
- $p \wedge \neg p \equiv \mathrm{F}$

$$(p \wedge r) \rightarrow (r \vee p)$$

Use a series of equivalences like so:

$(p \wedge r) \rightarrow (r \vee p) \equiv$

$\equiv$

$\equiv$

$\equiv$

$\equiv$

$\equiv$

$\equiv$

$\equiv$

$\equiv$ T

**Identity**
- $p \wedge \mathrm{T} \equiv p$
- $p \vee \mathrm{F} \equiv p$

**Domination**
- $p \vee \mathrm{T} \equiv \mathrm{T}$
- $p \wedge \mathrm{F} \equiv \mathrm{F}$

**Idempotent**
- $p \vee p \equiv p$
- $p \wedge p \equiv p$

**Commutative**
- $p \vee q \equiv q \vee p$
- $p \wedge q \equiv q \wedge p$

# Prove this is a Tautology: Option 2

$$(p \wedge r) \rightarrow (r \vee p)$$

**Associative**
- $(p \vee q) \vee r \equiv p \vee (q \vee r)$
- $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$

**Distributive**
- $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
- $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$

**Absorption**
- $p \vee (p \wedge q) \equiv p$
- $p \wedge (p \vee q) \equiv p$

**Negation**
- $p \vee \neg p \equiv \mathrm{T}$
- $p \wedge \neg p \equiv \mathrm{F}$

Use a series of equivalences like so:

| | | |
|---|---|---|
| $(p \wedge r) \rightarrow (r \vee p)$ | $\equiv \neg(p \wedge r) \vee (r \vee p)$ | Law of Implication |
| | $\equiv (\neg p \vee \neg r) \vee (r \vee p)$ | De Morgan |
| | $\equiv \neg p \vee (\neg r \vee (r \vee p))$ | Associative |
| | $\equiv \neg p \vee ((\neg r \vee r) \vee p)$ | Associative |
| | $\equiv \neg p \vee (p \vee (\neg r \vee r))$ | Commutative |
| | $\equiv (\neg p \vee p) \vee (\neg r \vee r)$ | Associative |
| | $\equiv (p \vee \neg p) \vee (r \vee \neg r)$ | Commutative (twice) |
| | $\equiv \mathrm{T} \vee \mathrm{T}$ | Negation (twice) |
| | $\equiv \mathrm{T}$ | Domination/Identity |

**Identity**
- $p \wedge \mathrm{T} \equiv p$
- $p \vee \mathrm{F} \equiv p$

**Domination**
- $p \vee \mathrm{T} \equiv \mathrm{T}$
- $p \wedge \mathrm{F} \equiv \mathrm{F}$

**Idempotent**
- $p \vee p \equiv p$
- $p \wedge p \equiv p$

**Commutative**
- $p \vee q \equiv q \vee p$
- $p \wedge q \equiv q \wedge p$

# Logical Proofs of Equivalence/Tautology

- Not smaller than truth tables when there are only a few propositional variables...

- ...but usually ***much shorter*** than truth table proofs when there are many propositional variables

- A big advantage will be that we can extend them to a more in-depth understanding of logic for which truth tables don't apply.

# Recall: Corollaries of Circuit Construction

- **$\neg$, $\wedge$, $\vee$ can implement any Boolean function**

  we didn't need any others to do this

- **Actually… just $\neg$, $\wedge$ (or $\neg$, $\vee$) are enough**

  follows by De Morgan's laws

- **Actually… just NAND (or NOR)**

# Boolean Algebra

- Usual notation used in circuit design
- Boolean algebra
  - a set of elements B containing {0, 1}
  - binary operations { + , • }
  - and a unary operation { ' }
  - such that the following axioms hold:

For any a, b, c in B:

| | | |
|---|---|---|
| 1. closure: | a + b is in B | a • b is in B |
| 2. commutativity: | a + b = b + a | a • b = b • a |
| 3. associativity: | a + (b + c) = (a + b) + c | a • (b • c) = (a • b) • c |
| 4. distributivity: | a + (b • c) = (a + b) • (a + c) | a • (b + c) = (a • b) + (a • c) |
| 5. identity: | a + 0 = a | a • 1 = a |
| 6. complementarity: | a + a' = 1 | a • a' = 0 |
| 7. null: | a + 1 = 1 | a • 0 = 0 |
| 8. idempotency: | a + a = a | a • a = a |
| 9. involution: | (a')' = a | |

# Simplification using Boolean Algebra

**uniting:**

10. $a \bullet b + a \bullet b' = a$
10D. $(a + b) \bullet (a + b') = a$

**absorption:**

11. $a + a \bullet b = a$
11D. $a \bullet (a + b) = a$
12. $(a + b') \bullet b = a \bullet b$
12D. $(a \bullet b') + b = a + b$

**factoring:**

13. $(a + b) \bullet (a' + c) = a \bullet c + a' \bullet b$
13D. $a \bullet b + a' \bullet c = (a + c) \bullet (a' + b)$

**consensus:**

14. $(a \bullet b) + (b \bullet c) + (a' \bullet c) = a \bullet b + a' \bullet c$
14D. $(a + b) \bullet (b + c) \bullet (a' + c) = (a + b) \bullet (a' + c)$

**de Morgan's:**

15. $(a + b + \ldots)' = a' \bullet b' \bullet \ldots$
15D. $(a \bullet b \bullet \ldots)' = a' + b' + \ldots$

# Proving Theorems

| | | |
|---|---|---|
| 2. commutativity: | a + b = b + a | a • b = b • a |
| 3. associativity: | a + (b + c) = (a + b) + c | a • (b • c) = (a • b) • c |
| 4. distributivity: | a + (b • c) = (a + b) • (a + c) | a • (b + c) = (a • b) + (a • c) |
| 5. identity: | a + 0 = a | a • 1 = a |
| 6. complementarity: | a + a' = 1 | a • a' = 0 |
| 7. null: | a + 1 = 1 | a • 0 = 0 |
| 8. idempotency: | a + a = a | a • a = a |
| 9. involution: | (a')' = a | |

## Using the laws of Boolean Algebra:

**XOR variants:** $(A + B)(AB)' = (A + B)(A' + B')$

original — product of sums

De Morgan

# Proving Theorems

| | | |
|---|---|---|
| 2. commutativity: | a + b = b + a | a • b = b • a |
| 3. associativity: | a + (b + c) = (a + b) + c | a • (b • c) = (a • b) • c |
| 4. distributivity: | a + (b • c) = (a + b) • (a + c) | a • (b + c) = (a • b) + (a • c) |
| 5. identity: | a + 0 = a | a • 1 = a |
| 6. complementarity: | a + a' = 1 | a • a' = 0 |
| 7. null: | a + 1 = 1 | a • 0 = 0 |
| 8. idempotency: | a + a = a | a • a = a |
| 9. involution: | (a')' = a | |

## Using the laws of Boolean Algebra:

**XOR variants:** $(A + B)(A' + B') = AB' + A'B$

product of sums    sum of products

| | | |
|---|---|---|
| $(A + B)(A' + B')$ | $= (A + B)A' + (A + B)B'$ | distributivity |
| | $= A'(A + B) + B'(A + B)$ | commutativity |
| | $= A'A + A'B + B'A + B'B$ | distributivity |
| | $= 0 + A'B + B'A + 0$ | complementarity |
| | $= A'B + AB'$ | identity |
| | $= AB' + A'B$ | commuativity |

# Proving Theorems

| | | |
|---|---|---|
| 2. commutativity: | $a + b = b + a$ | $a \bullet b = b \bullet a$ |
| 3. associativity: | $a + (b + c) = (a + b) + c$ | $a \bullet (b \bullet c) = (a \bullet b) \bullet c$ |
| 4. distributivity: | $a + (b \bullet c) = (a + b) \bullet (a + c)$ | $a \bullet (b + c) = (a \bullet b) + (a \bullet c)$ |
| 5. identity: | $a + 0 = a$ | $a \bullet 1 = a$ |
| 6. complementarity: | $a + a' = 1$ | $a \bullet a' = 0$ |
| 7. null: | $a + 1 = 1$ | $a \bullet 0 = 0$ |
| 8. idempotency: | $a + a = a$ | $a \bullet a = a$ |
| 9. involution: | $(a')' = a$ | |

## Using the laws of Boolean Algebra:

**XOR variants:** $(A + B)(A' + B') = AB' + A'B$

product of sums sum of products

| | |
|---|---|
| $(A + B)(A' + B') = (A + B)A' + (A + B)B'$ | distributivity |
| $= A'(A + B) + B'(A + B)$ | commutativity |
| $= A'A + A'B + B'A + B'B$ | distributivity |
| $= 0 + A'B + B'A + 0$ | complementarity |
| $= A'B + AB'$ | identity |
| $= AB' + A'B$ | commuativity |

# Sum-of-Products Canonical Form

Product term (or minterm)

- ANDed product of literals – input combination for which output is true
- each variable appears exactly once, true or inverted (but not both)

| A | B | C | minterms |
|---|---|---|---|
| 0 | 0 | 0 | A′B′C′ |
| 0 | 0 | 1 | A′B′C |
| 0 | 1 | 0 | A′BC′ |
| 0 | 1 | 1 | A′BC |
| 1 | 0 | 0 | AB′C′ |
| 1 | 0 | 1 | AB′C |
| 1 | 1 | 0 | ABC′ |
| 1 | 1 | 1 | ABC |

F in canonical form:

F(A, B, C) = A′B′C + A′BC + AB′C + ABC′ + ABC

canonical form ≠ minimal form

F(A, B, C) = A′B′C + A′BC + AB′C + ABC + ABC′
= (A′B′ + A′B + AB′ + AB)C + ABC′
= ((A′ + A)(B′ + B))C + ABC′
= C + ABC′
= ABC′ + C
= AB + C

# Product-of-Sums Canonical Form

**Sum term (or maxterm)**

- **ORed sum of literals – input combination for which output is false**
- **each variable appears exactly once, true or inverted (but not both)**

| A | B | C | maxterms |
|---|---|---|---|
| 0 | 0 | 0 | A+B+C |
| 0 | 0 | 1 | A+B+C′ |
| 0 | 1 | 0 | A+B′+C |
| 0 | 1 | 1 | A+B′+C′ |
| 1 | 0 | 0 | A′+B+C |
| 1 | 0 | 1 | A′+B+C′ |
| 1 | 1 | 0 | A′+B′+C |
| 1 | 1 | 1 | A′+B′+C′ |

F in canonical form:

$$F(A, B, C) = (A + B + C)\,(A + B' + C)\,(A' + B + C)$$

canonical form ≠ minimal form

$$\begin{aligned} F(A, B, C) &= (A + B + C)\,(A + B' + C)\,(A' + B + C) \\ &= (A + B + C)\,(A + B' + C) \\ &\quad\ (A + B + C)\,(A' + B + C) \\ &= (A + C)\,(B + C) \end{aligned}$$

# Recall: Truth Table to Logic

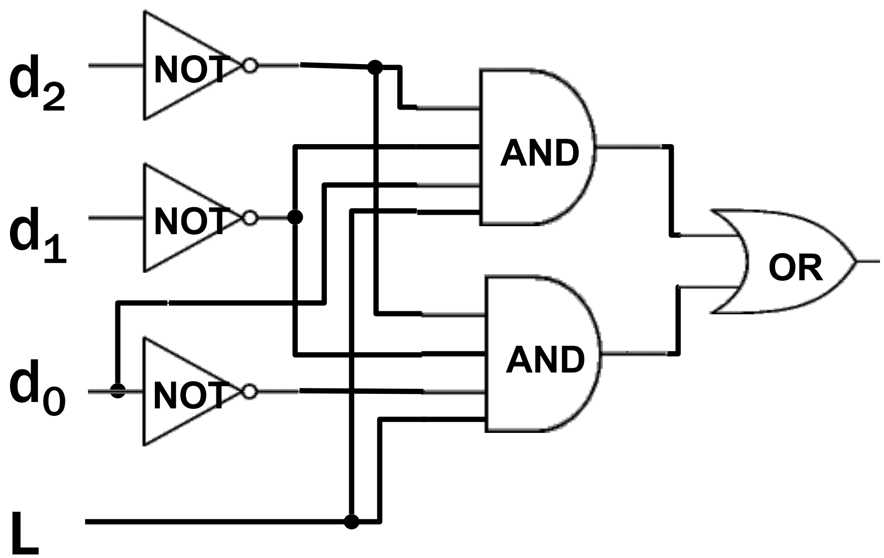
$c_0 = d_2 \cdot d_1' \cdot d_0 \cdot L' + d_2 \cdot d_1 \cdot d_0' + d_2 \cdot d_1 \cdot d_0$

$c_1 = d_2' \cdot d_1' \cdot d_0' \cdot L' + d_2' \cdot d_1' \cdot d_0 \cdot L' + d_2' \cdot d_1 \cdot d_0' \cdot L' + d_2' \cdot d_1 \cdot d_0 \cdot L' + d_2 \cdot d_1' \cdot d_0' + d_2 \cdot d_1' \cdot d_0 \cdot L$

$c_2 = d_2' \cdot d_1 \cdot d_0' \cdot L + d_2' \cdot d_1 \cdot d_0 \cdot L$

$c_3 = d_2' \cdot d_1' \cdot d_0' \cdot L + d_2' \cdot d_1' \cdot d_0 \cdot L$

Here's $c_3$ as a circuit:



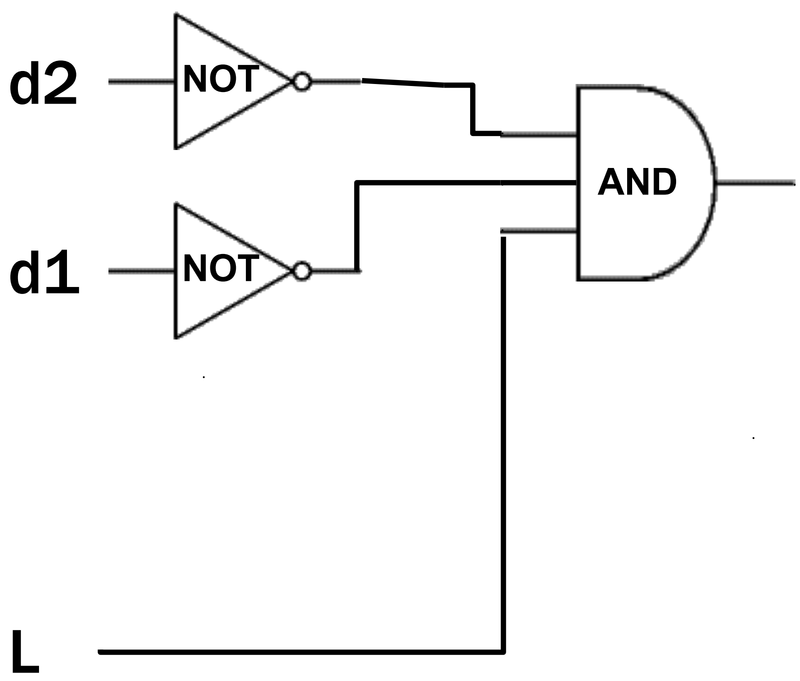
# Simplifying using Boolean Algebra

c3 = d2’•d1’•d0’•L + d2’•d1’•d0•L

= d2’•d1’•(d0’ + d0)•L

= d2’•d1’•1•L

= d2’•d1’•L



# 1-bit Binary Adder

| | |
|---|---|
| $A$ | $0 + 0 = 0$ (with $C_{OUT} = 0$) |
| $+ B$ | $0 + 1 = 1$ (with $C_{OUT} = 0$) |
| $S$ | $1 + 0 = 1$ (with $C_{OUT} = 0$) |
| ($C_{OUT}$) | $1 + 1 = 0$ (with $C_{OUT} = 1$) |

# 1-bit Binary Adder

|   |   |
|---|---|
| A | $0 + 0 = 0$ (with $C_{OUT} = 0$) |
| + B | $0 + 1 = 1$ (with $C_{OUT} = 0$) |
| S | $1 + 0 = 1$ (with $C_{OUT} = 0$) |
| ($C_{OUT}$) | $1 + 1 = 0$ (with $C_{OUT} = 1$) |

**Idea**: chain these together to add larger numbers

Recall from elementary school:

$$\begin{array}{r} 2\ 4\ 8 \\ +\ 3\ 7\ 5 \\ \hline \end{array}$$

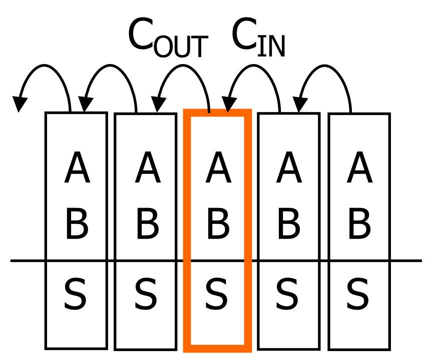
# 1-bit Binary Adder

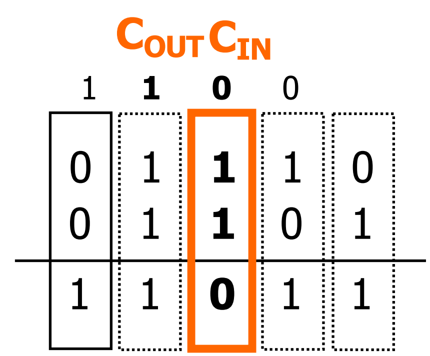
A
+ B
S
($C_{OUT}$)

0 + 0 = 0 (with $C_{OUT}$ = 0)
0 + 1 = 1 (with $C_{OUT}$ = 0)
1 + 0 = 1 (with $C_{OUT}$ = 0)
1 + 1 = 0 (with $C_{OUT}$ = 1)

**Idea**: These are chained together with a carry-in

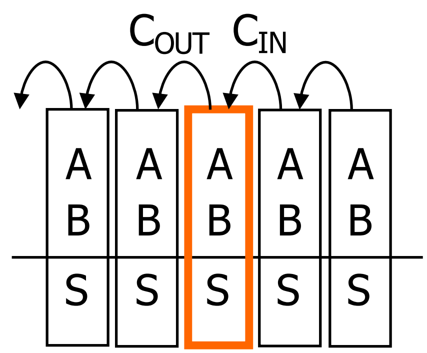
($C_{IN}$)
A
+ B
S
($C_{OUT}$)



# 1-bit Binary Adder

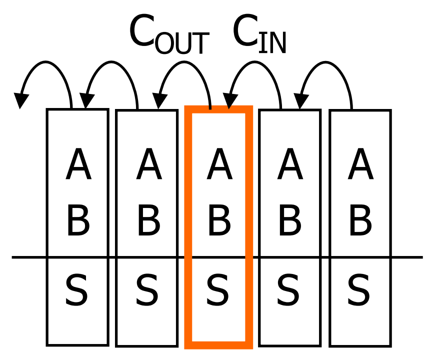
- **Inputs:** A, B, Carry-in
- **Outputs:** Sum, Carry-out




| A | B | $C_{IN}$ | $C_{OUT}$ | S |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

# 1-bit Binary Adder

- **Inputs:** A, B, Carry-in
- **Outputs:** Sum, Carry-out




| A | B | $C_{IN}$ | $C_{OUT}$ | S |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

**Derive an expression for S**

$A' \cdot B' \cdot C_{IN}$

$A' \cdot B \cdot C_{IN}'$

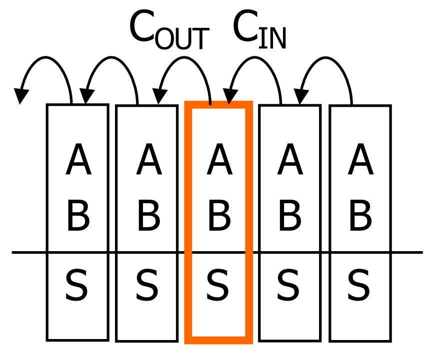
$A \cdot B' \cdot C_{IN}'$

$A \cdot B \cdot C_{IN}$

$$S = A' \cdot B' \cdot C_{IN} + A' \cdot B \cdot C_{IN}' + A \cdot B' \cdot C_{IN}' + A \cdot B \cdot C_{IN}$$

# 1-bit Binary Adder

- **Inputs:** A, B, Carry-in
- **Outputs:** Sum, Carry-out




| A | B | $C_{IN}$ | $C_{OUT}$ | S |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

**Derive an expression for $C_{OUT}$**

$A' \cdot B \cdot C_{IN}$

$A \cdot B' \cdot C_{IN}$

$A \cdot B \cdot C_{IN}'$

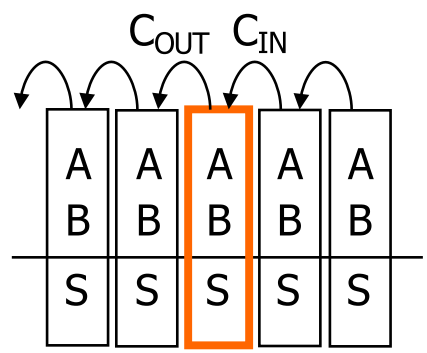
$A \cdot B \cdot C_{IN}$

$$C_{OUT} = A' \cdot B \cdot C_{IN} + A \cdot B' \cdot C_{IN} + A \cdot B \cdot C_{IN}' + A \cdot B \cdot C_{IN}$$

$$S = A' \cdot B' \cdot C_{IN} + A' \cdot B \cdot C_{IN}' + A \cdot B' \cdot C_{IN}' + A \cdot B \cdot C_{IN}$$

# 1-bit Binary Adder

- **Inputs:** A, B, Carry-in
- **Outputs:** Sum, Carry-out




| A | B | $C_{IN}$ | $C_{OUT}$ | S |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

$$S = A' \cdot B' \cdot C_{IN} + A' \cdot B \cdot C_{IN}' + A \cdot B' \cdot C_{IN}' + A \cdot B \cdot C_{IN}$$

$$C_{OUT} = A' \cdot B \cdot C_{IN} + A \cdot B' \cdot C_{IN} + A \cdot B \cdot C_{IN}' + A \cdot B \cdot C_{IN}$$

# Apply Theorems to Simplify Expressions

The theorems of Boolean algebra can simplify expressions

– e.g., full adder's carry-out function

$$
\begin{aligned}
\text{Cout} &= A'\,B\,Cin + A\,B'\,Cin + A\,B\,Cin' + A\,B\,Cin \\
&= A'\,B\,Cin + A\,B'\,Cin + A\,B\,Cin' + A\,B\,Cin + A\,B\,Cin \\
&= A'\,B\,Cin + A\,B\,Cin + A\,B'\,Cin + A\,B\,Cin' + A\,B\,Cin \\
&= (A' + A)\,B\,Cin + A\,B'\,Cin + A\,B\,Cin' + A\,B\,Cin \\
&= (1)\,B\,Cin + A\,B'\,Cin + A\,B\,Cin' + A\,B\,Cin \\
&= B\,Cin + A\,B'\,Cin + A\,B\,Cin' + A\,B\,Cin + A\,B\,Cin \\
&= B\,Cin + A\,B'\,Cin + A\,B\,Cin + A\,B\,Cin' + A\,B\,Cin \\
&= B\,Cin + A\,(B' + B)\,Cin + A\,B\,Cin' + A\,B\,Cin \\
&= B\,Cin + A\,(1)\,Cin + A\,B\,Cin' + A\,B\,Cin \\
&= B\,Cin + A\,Cin + A\,B\,(Cin' + Cin) \\
&= B\,Cin + A\,Cin + A\,B\,(1) \\
&= B\,Cin + A\,Cin + A\,B
\end{aligned}
$$

# Apply Theorems to Simplify Expressions

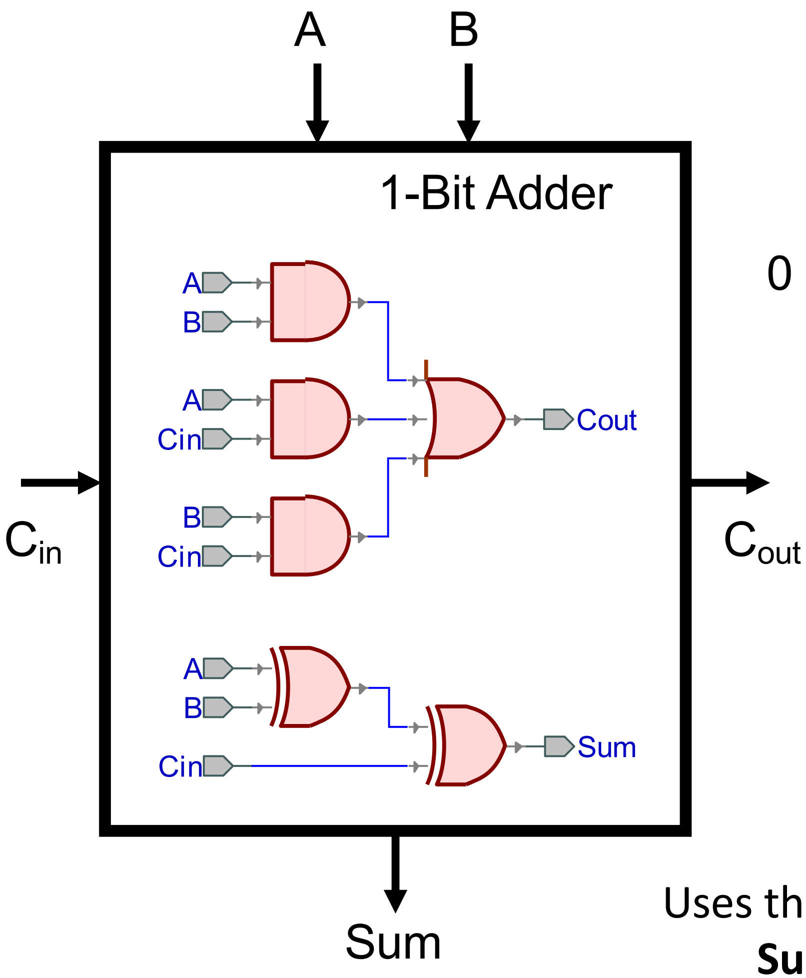
**The theorems of Boolean algebra can simplify expressions**

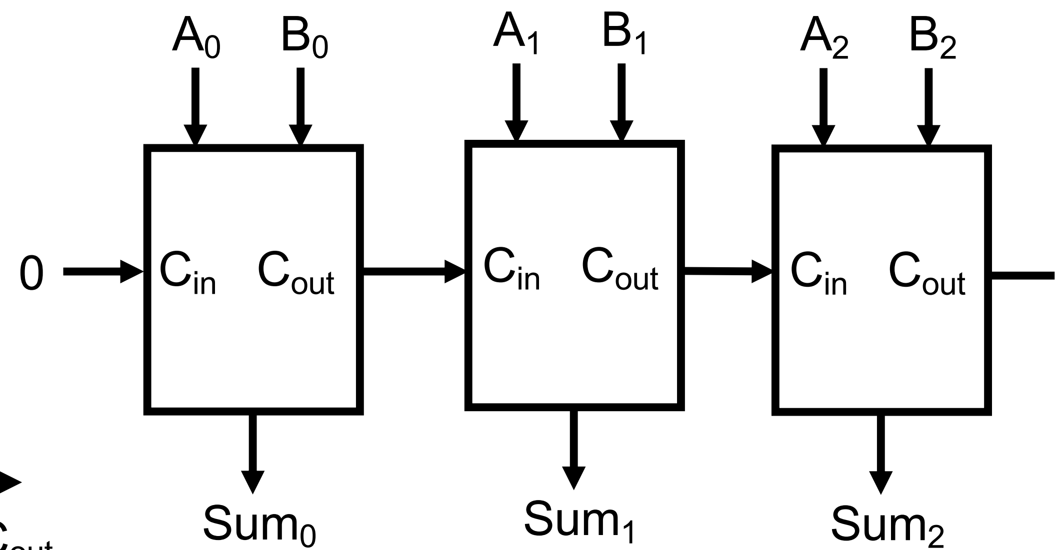
– **e.g., full adder's carry-out function**

$$
\begin{aligned}
Cout &= A' B\ Cin + A\ B' Cin + A\ B\ Cin' + A\ B\ Cin \\
&= A' B\ Cin + A\ B' Cin + A\ B\ Cin' + \boxed{A\ B\ Cin + A\ B\ Cin} \\
&= A' B\ Cin + A\ B\ Cin + A\ B' Cin + A\ B\ Cin' + A\ B\ Cin \\
&= (A' + A)\ B\ Cin + A\ B' Cin + A\ B\ Cin' + A\ B\ Cin \\
&= (1)\ B\ Cin + A\ B' Cin + A\ B\ Cin' + A\ B\ Cin \\
&= B\ Cin + A\ B' Cin + A\ B\ Cin' + \boxed{A\ B\ Cin + A\ B\ Cin} \\
&= B\ Cin + A\ B' Cin + A\ B\ Cin + A\ B\ Cin' + A\ B\ Cin \\
&= B\ Cin + A\ (B' + B)\ Cin + A\ B\ Cin' + A\ B\ Cin \\
&= B\ Cin + A\ (1)\ Cin + A\ B\ Cin' + A\ B\ Cin \\
&= B\ Cin + A\ Cin + A\ B\ (Cin' + Cin) \\
&= B\ Cin + A\ Cin + A\ B\ (1) \\
&= B\ Cin + A\ Cin + A\ B
\end{aligned}
$$

adding extra terms creates new factoring opportunities

# A 2-bit Ripple-Carry Adder




Uses the fact that

**$Sum = A' \bullet B' \bullet C_{IN} + A' \bullet B \bullet C_{IN}' + A \bullet B' \bullet C_{IN}' + A \bullet B \bullet C_{IN}$**

is equivalent to **$Sum = (A \oplus B) \oplus C_{IN}$**

# Mapping Truth Tables to Logic Gates (Revised)

**Given a truth table:**

1. Write the output in a table
2. Write the Boolean expression
3. Minimize the Boolean expression
4. Draw as gates
5. Map to available gates

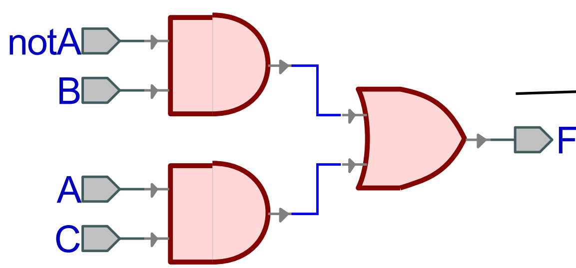
| A | B | C | F |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

(2)

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

(3)

(4)




(5)