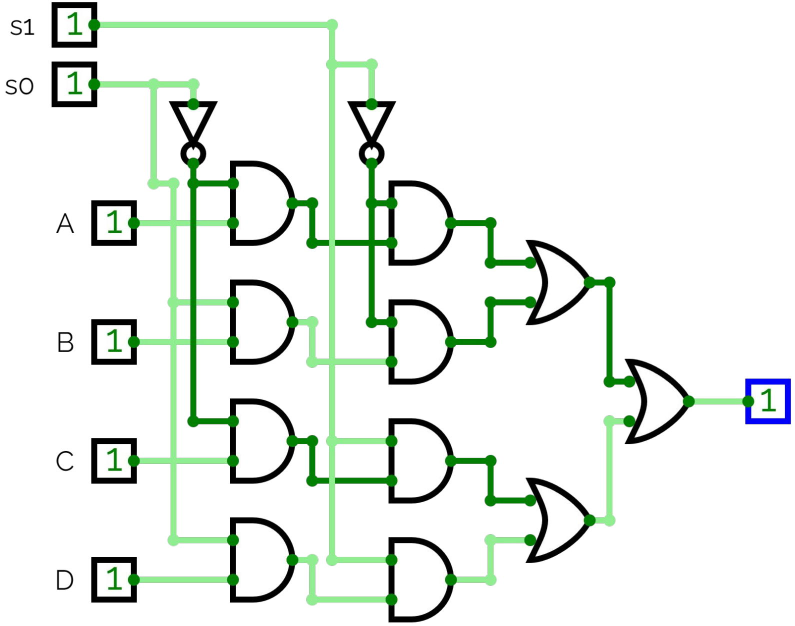
Recitation 12

Computer Architecture (section 1)

Combinatorial Circuits

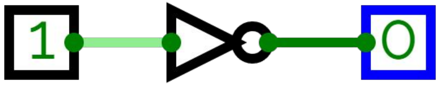
- An arrangement of electronic components that perform operations on a set of inputs.
- Combinatorial logic is:
  - Time invariant
    - No memory
  - Composed of *logic gates*



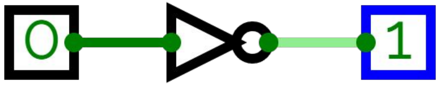
Basic Logic Gates

- Common logic gates are: **NOT**, **AND**, **OR**, **NAND**, **NOR**, **XOR**.
- These gates can be combined together to form a logic circuit.
- Boolean algebra is a theoretical framework used to analyze logic circuits.

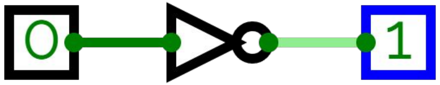
NOT Gate



NOT Gate



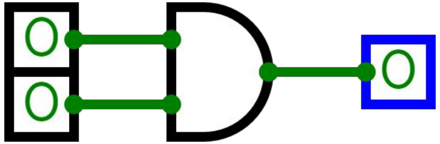
NOT Gate



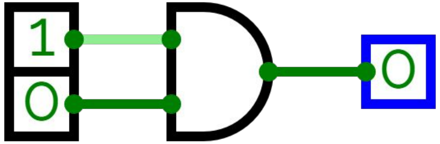
Truth Table

In A	Out B
0	1
1	0

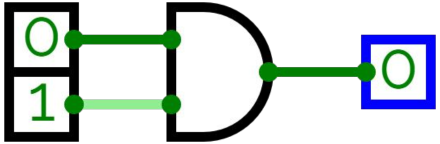
AND Gate



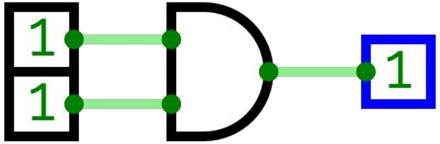
AND Gate



AND Gate



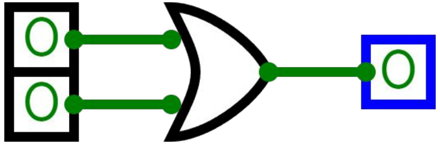
AND Gate



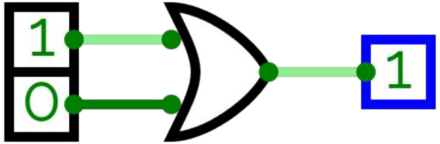
Truth Table

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

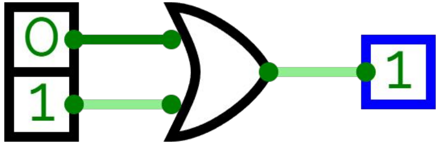
OR Gate



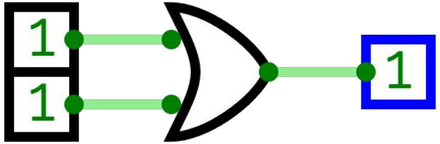
OR Gate



OR Gate



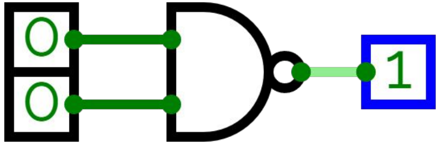
OR Gate



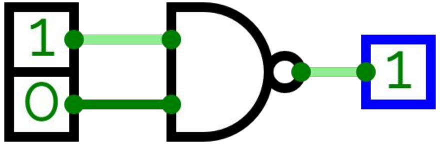
Truth Table

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

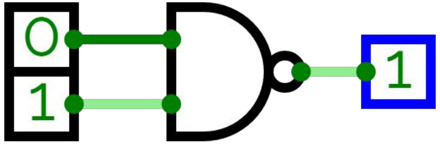
NAND Gate



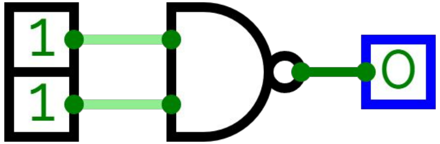
NAND Gate



NAND Gate



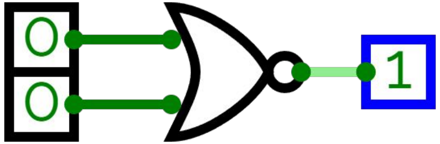
NAND Gate



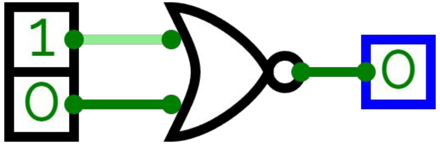
Truth Table

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

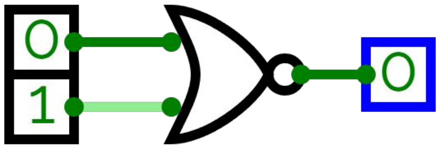
NOR Gate



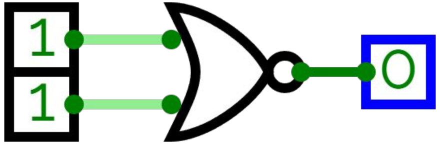
NOR Gate



NOR Gate



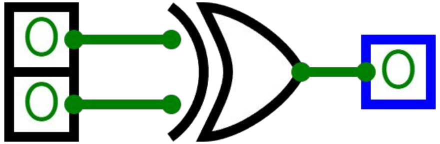
NOR Gate



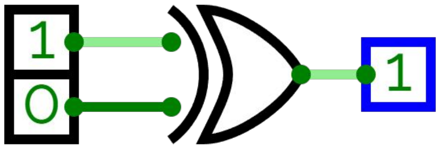
Truth Table

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

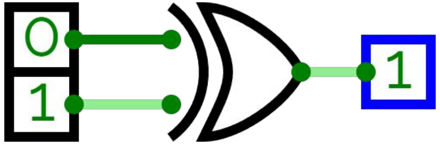
XOR Gate



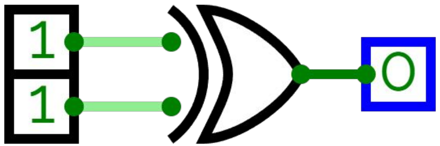
XOR Gate



XOR Gate



XOR Gate

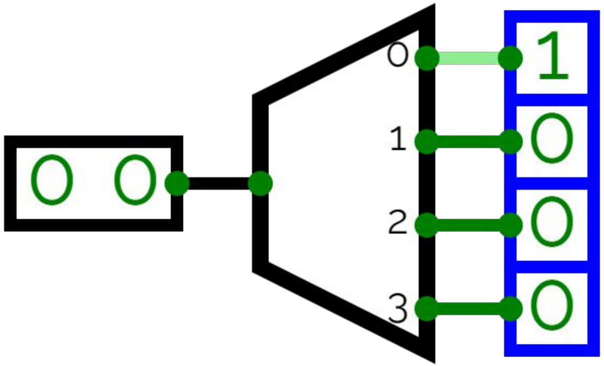


Truth Table

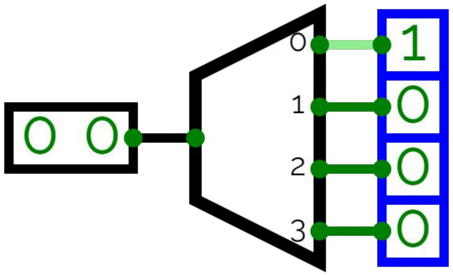
In A	In B	Out C
0	0	0
0	1	1
1	0	1
1	1	0

Decoder

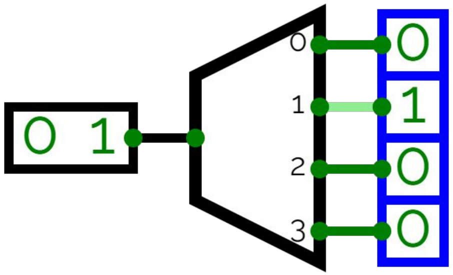
- A logic circuit with *n* inputs and 2^n outputs.
- Used to select the index of the represented binary input.
- Only one output bit can be set for any input.



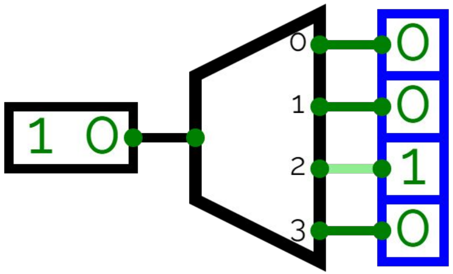
Decoder



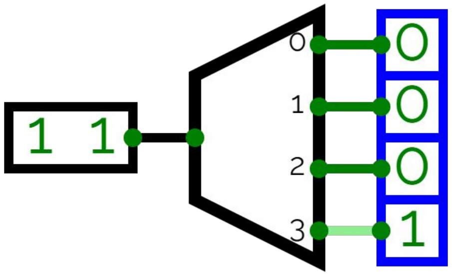
Decoder



Decoder



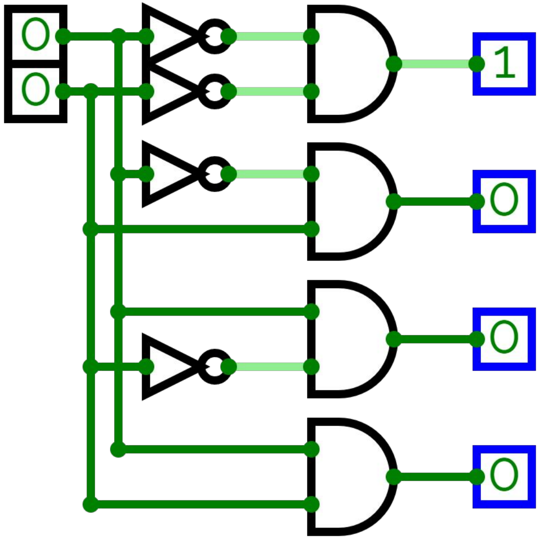
Decoder



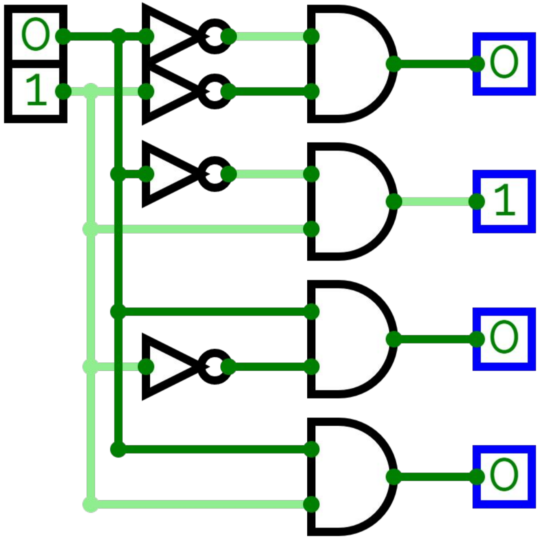
Truth Table

In A	In B	Out C	Out D	Out E	Out F
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

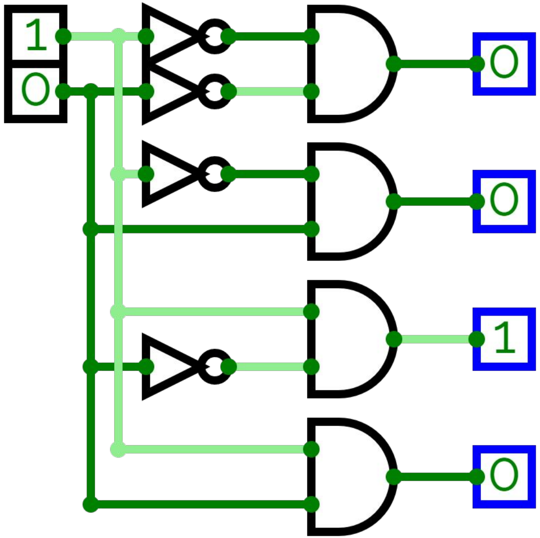
Building a 2-bit Decoder



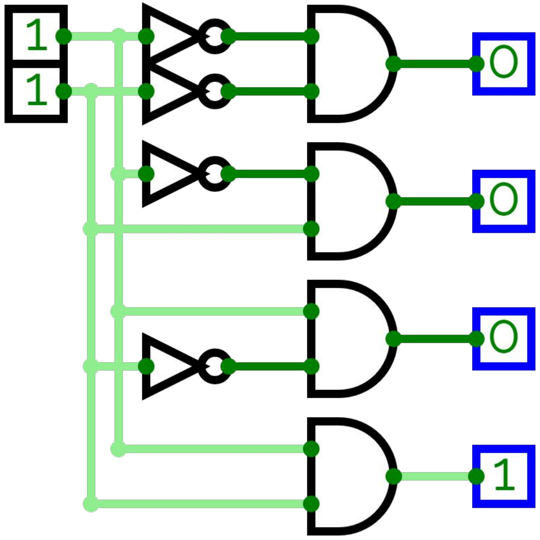
Building a 2-bit Decoder



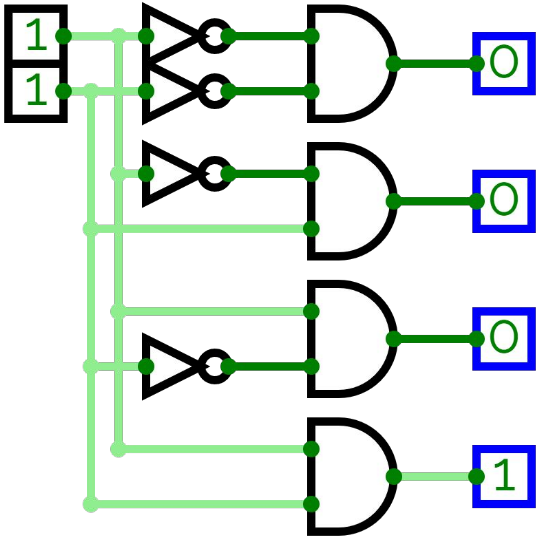
Building a 2-bit Decoder



Building a 2-bit Decoder



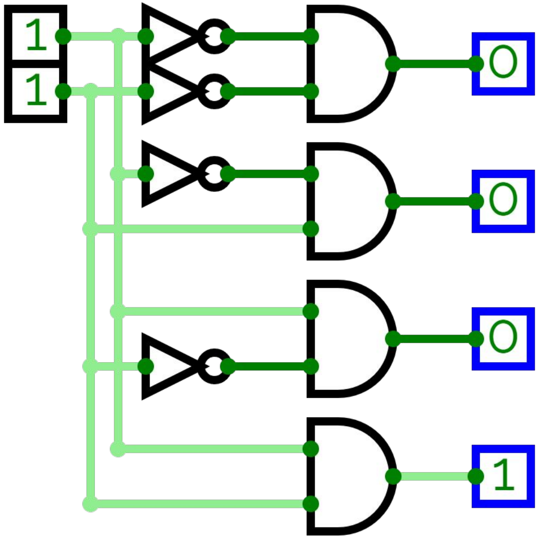
Building a 2-bit Decoder



Truth Table

In A	In B	Out C	Out D	Out E	Out F
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Building a 2-bit Decoder

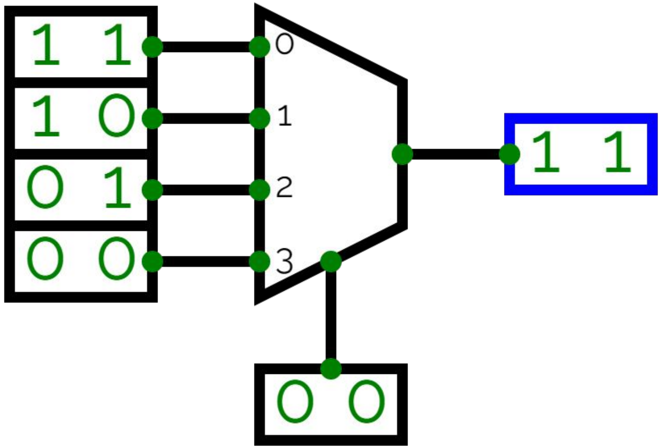


Try it out yourself

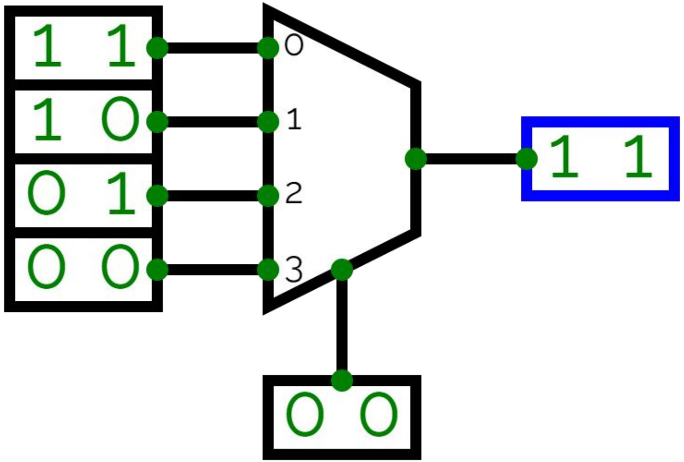
https://circuitverse.org/simulator

Multiplexer

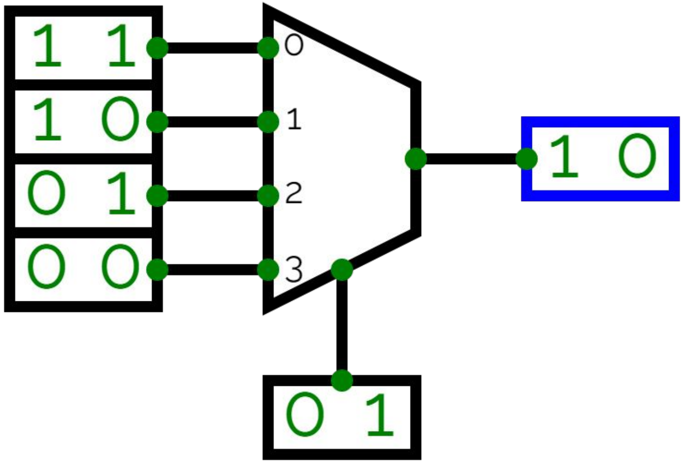
- A logic circuit with 2^n signal inputs, *n* select inputs, and 1 signal output .
- Used to select which signal to output.
- Output is always one of the inputs.



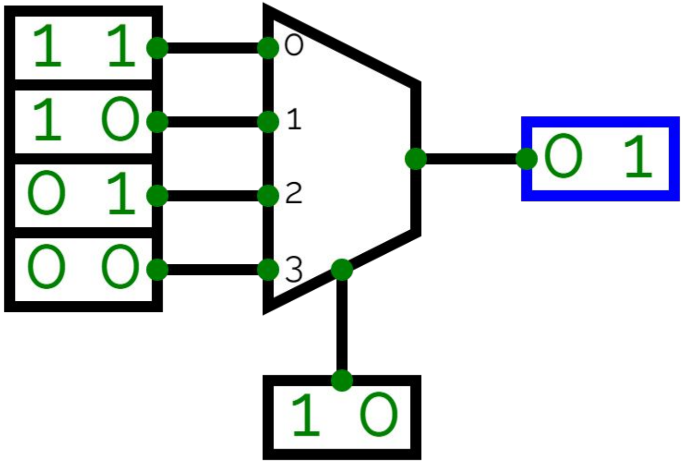
Multiplexer



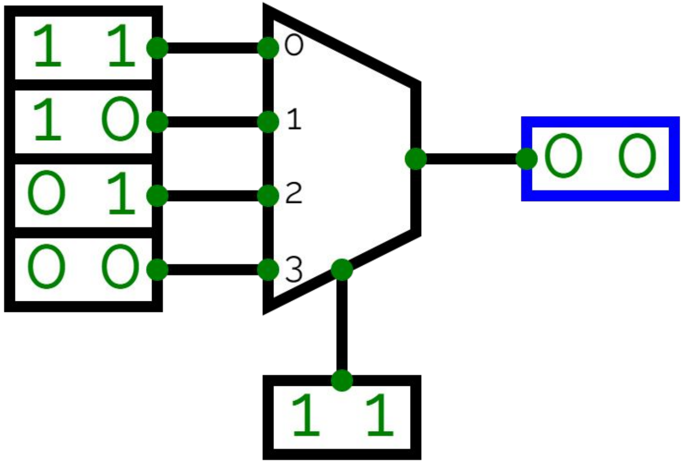
Multiplexer



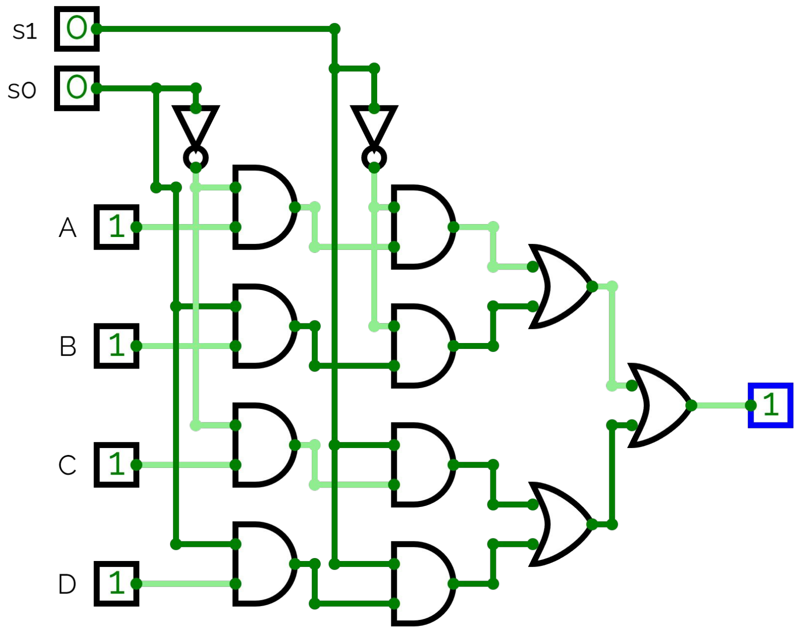
Multiplexer



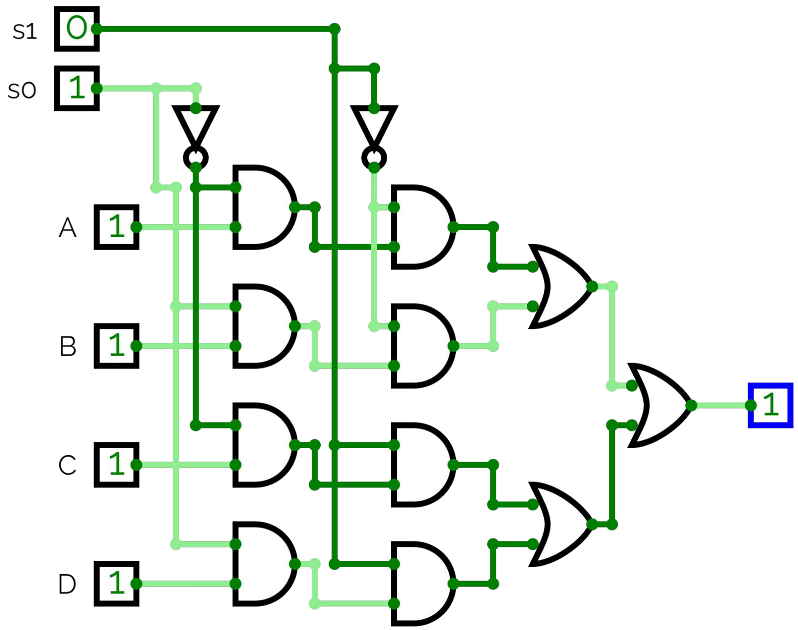
Multiplexer



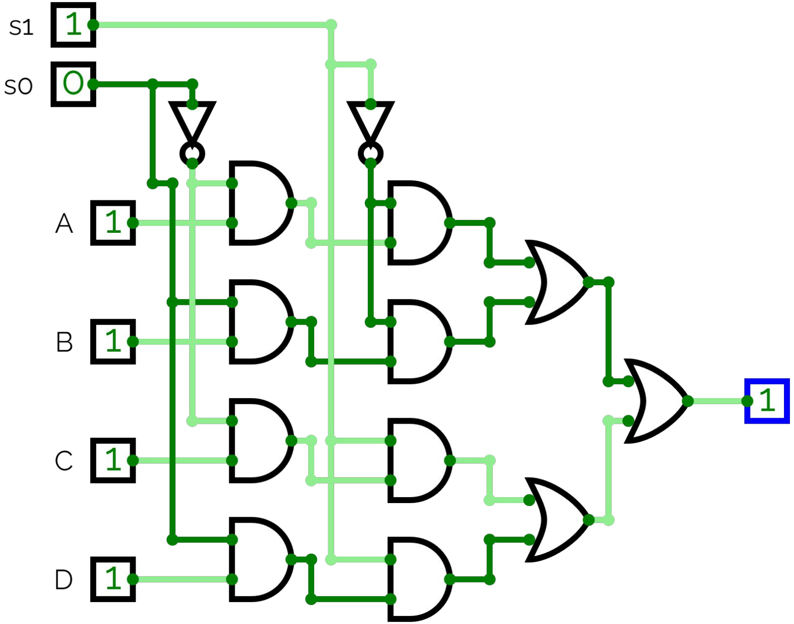
Building a 4-to-1 Multiplexer



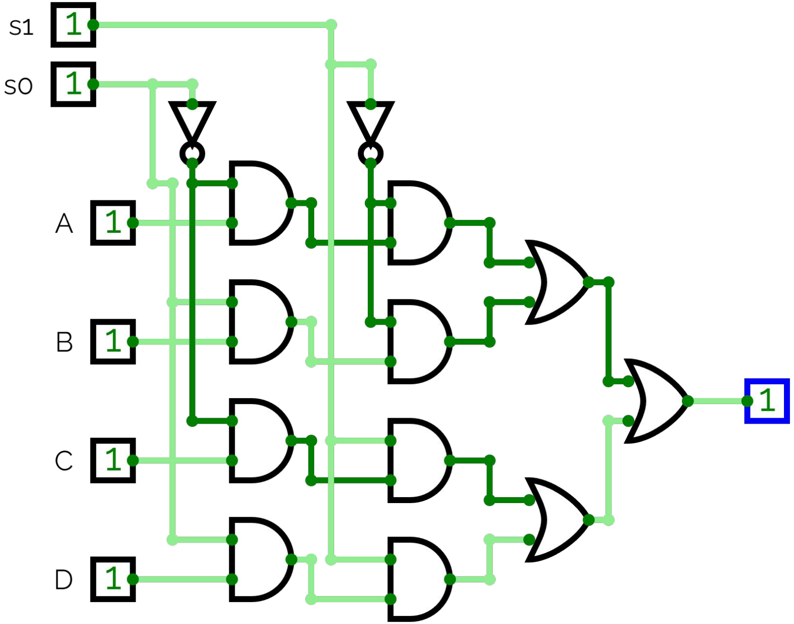
Building a 4-to-1 Multiplexer



Building a 4-to-1 Multiplexer



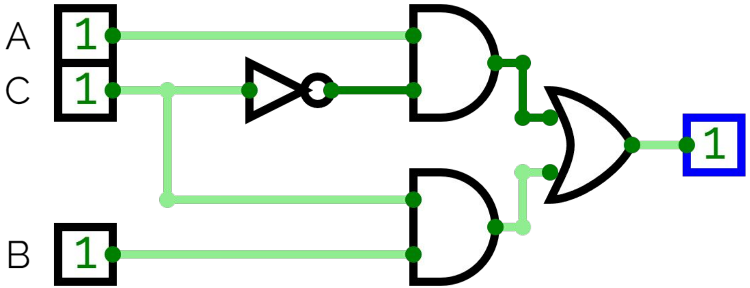
Building a 4-to-1 Multiplexer



Boolean Algebra

We can define logic circuits with boolean algebra.

$$F = A\overline{C} + BC$$



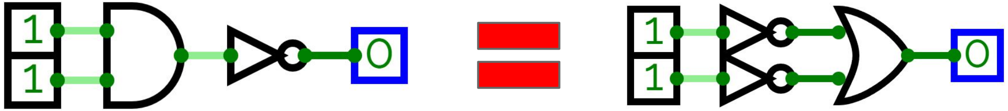
Boolean Algebra Identities

1.	Law of Identity	$A = A$ $\overline{A} = \overline{A}$
2.	Commutative Law	$A \cdot B = B \cdot A$ $A + B = B + A$
3.	Associative Law	$A \cdot (B \cdot C) = A \cdot B \cdot C$ $A + (B + C) = A + B + C$
4.	Idempotent Law	$A \cdot A = A$ $A + A = A$
5.	Double Negative Law	$\overline{\overline{A}} = A$
6.	Complementary Law	$A \cdot \overline{A} = 0$ $A + \overline{A} = 1$
7.	Law of Intersection	$A \cdot 1 = A$ $A \cdot 0 = 0$
8.	Law of Union	$A + 1 = 1$ $A + 0 = A$
9.	DeMorgan's Theorem	$\overline{AB} = \overline{A} + \overline{B}$ $\overline{A + B} = \overline{A}\,\overline{B}$
10.	Distributive Law	$A \cdot (B + C) = (A \cdot B) + (A \cdot C)$ $A + (BC) = (A + B) \cdot (A + C)$
11.	Law of Absorption	$A \cdot (A + B) = A$ $A + (AB) = A$
12.	Law of Common Identities	$A \cdot (\overline{A} + B) = AB$ $A + (\overline{A}B) = A + B$

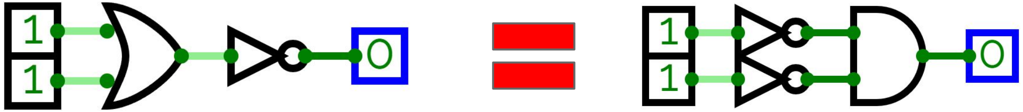
We can prove equality with truth tables.

De Morgan's Law

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



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



Expressing Boolean Functions

We can express boolean functions in canonical forms known and **sum-of-products** and **product-of-sums**.

$$PoS = (\overline{A}+B)(A+\overline{B})$$

A minterm is the product of all literals in the expression where each literal may be negated. A minterm can only output 1 for a single input.

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

Boolean Algebra Exercise

Convert the truth table to a boolean expression.

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

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

Min terms

$\overline{A}BC = 1$

$A\overline{B}C = 1$

$AB\overline{C} = 1$

$ABC = 1$

Simplifying Boolean Expressions

Often times a boolean expression has an equivalent more concise form.

One way to simplify an expression is via identities.

$$\bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$$

↓ Factoring **BC** out of 1st and 4th terms

$$BC(\bar{A} + A) + A\bar{B}C + AB\bar{C}$$

↓ Applying identity $A + \bar{A} = 1$

$$BC(1) + A\bar{B}C + AB\bar{C}$$

↓ Applying identity $1A = A$

$$BC + A\bar{B}C + AB\bar{C}$$

↓ Factoring **B** out of 1st and 3rd terms

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

↓ Applying rule $A + \bar{A}B = A + B$ to the $C + A\bar{C}$ term

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

↓ Distributing terms

$$BC + AB + A\bar{B}C$$

↓ Factoring **A** out of 2nd and 3rd terms

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

↓ Applying rule $A + \bar{A}B = A + B$ to the $B + \bar{B}C$ term

$$BC + A(B + C)$$

↓ Distributing terms

$$BC + AB + AC$$

or — Simplified result

$$AB + BC + AC$$