Digital Logic I

Outline

- Transistors
- Logic Gates
  - NOT, OR, NOR, AND, NAND
  - DeMorgan's Law
  - Larger Gates
- Combinational Logic Circuits
  - Decoder, MUX, Full Adder, PLA,
  - Logical Completeness
- Simplification
  - Boolean
  - Karnaugh Maps
  - PLA/PGA

Nomenclature

Open Switch
No current can flow

Closed Switch
Current can flow

Our Story Begins

Boolean Algebra

George Boole

library.thinkquest.org

George Boole was an English mathematician, philosopher and logician. His work was in the fields of differential equations and algebraic logic, and he is now best known as the author of The Laws of Thought. Wikipedia

Born: November 2, 1815, Lincoln

Died: December 8, 1864, Ballintemple, Cork

Spouse: Mary Everest Boole (m. 1855)

Children: Alicia Boole Stott, Ethel Lilian Voynich, Lucy Everest Boole, Mary Ellen Boole Hinton, Margaret Taylor

Awards: Royal Medal

Telegraph

Samuel Morse

en.wikipedia.org

Samuel Finley Breese Morse was an American contributor to the invention of a single-wire telegraph system based on European telegraphs, co-inventor of the Morse code, and an accomplished painter. Wikipedia

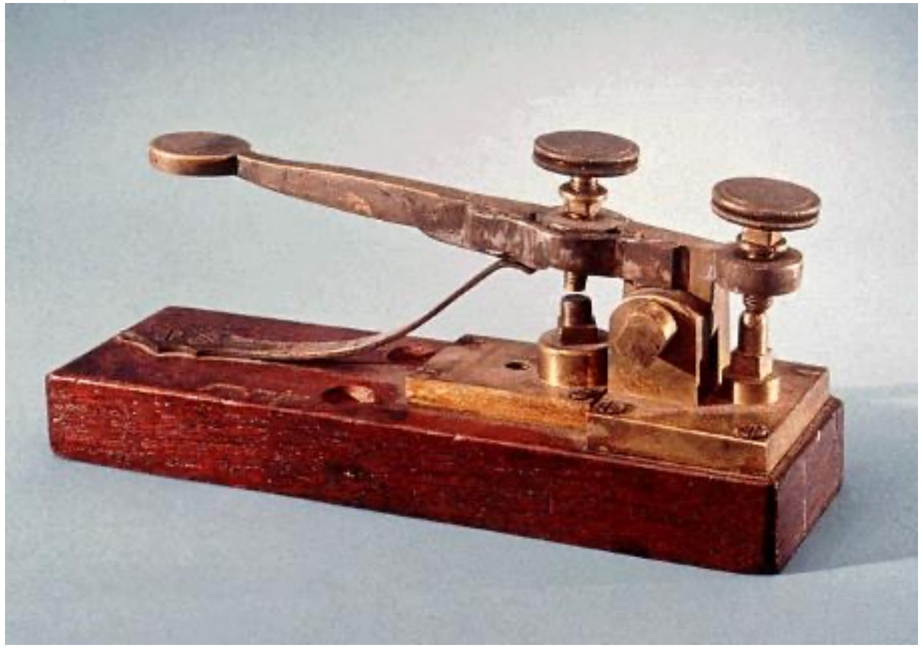
Born: April 27, 1791, Charlestown

Died: April 2, 1872, New York City

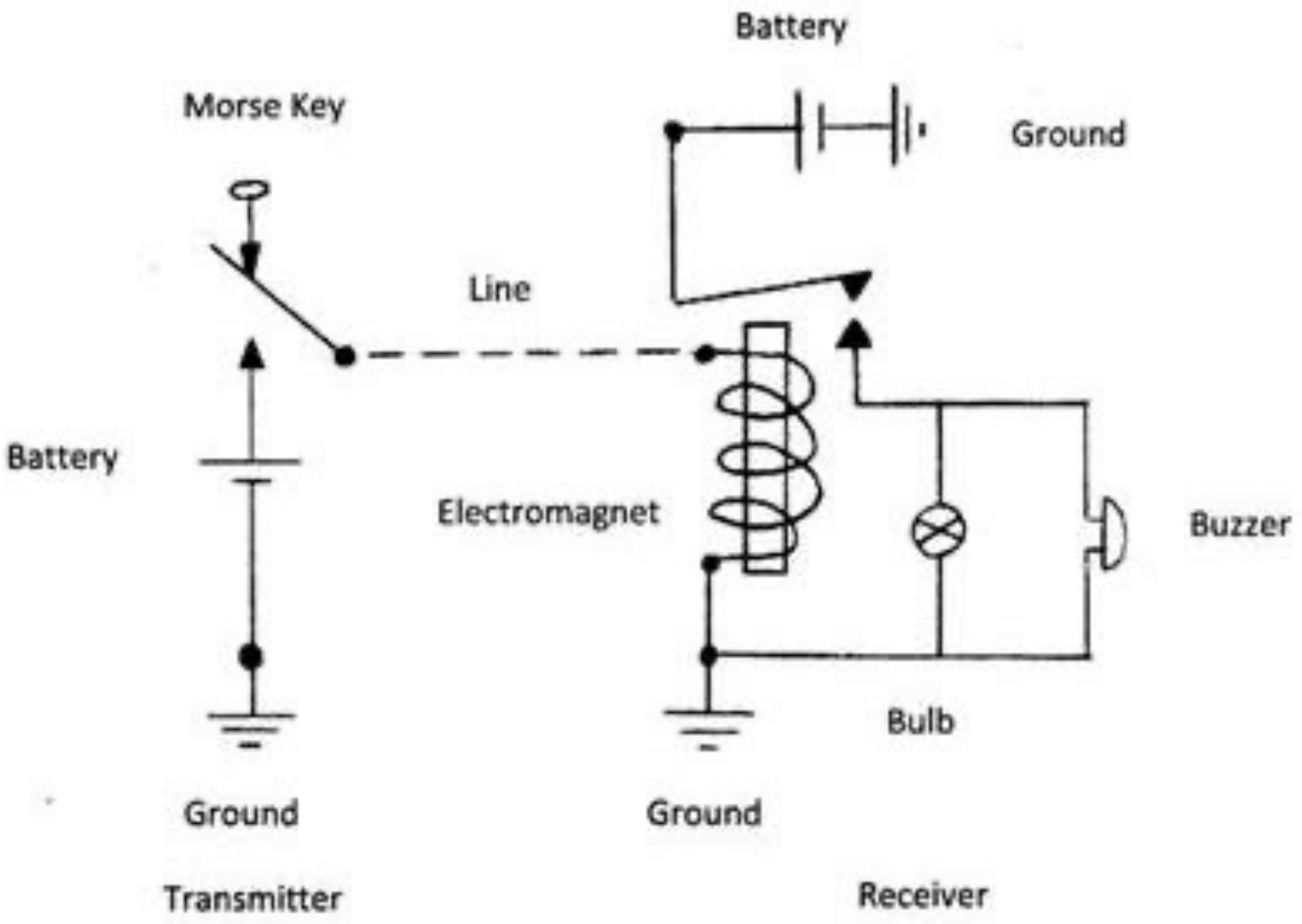
Education: Yale University, Phillips Academy

Parents: Jedidiah Morse, Elizabeth Ann Finley Breese

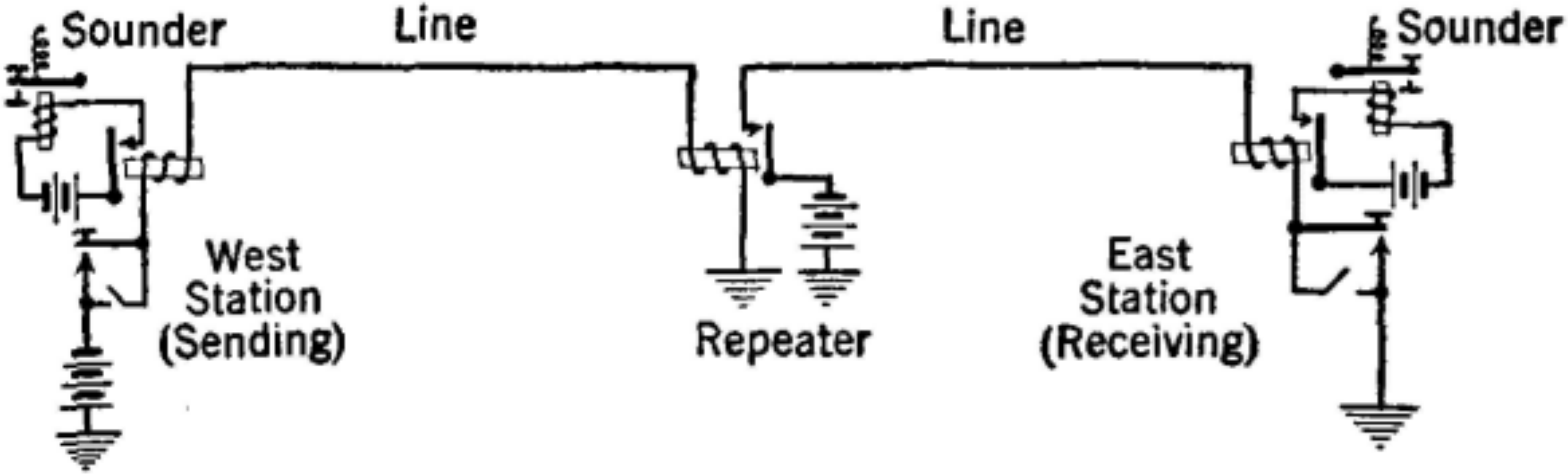
Siblings: Sidney Edwards Morse



Telegraph Schematic

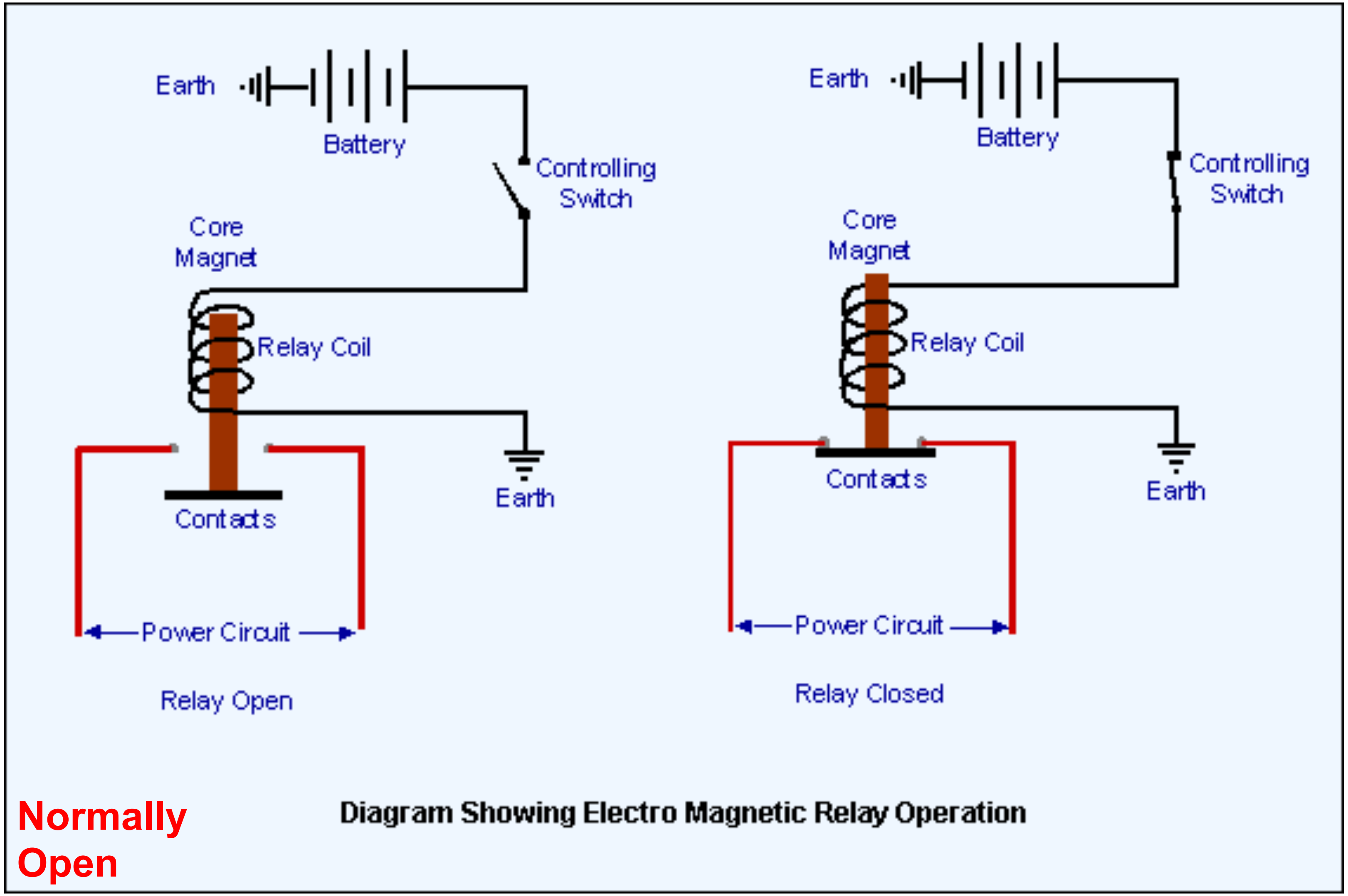


Repeater



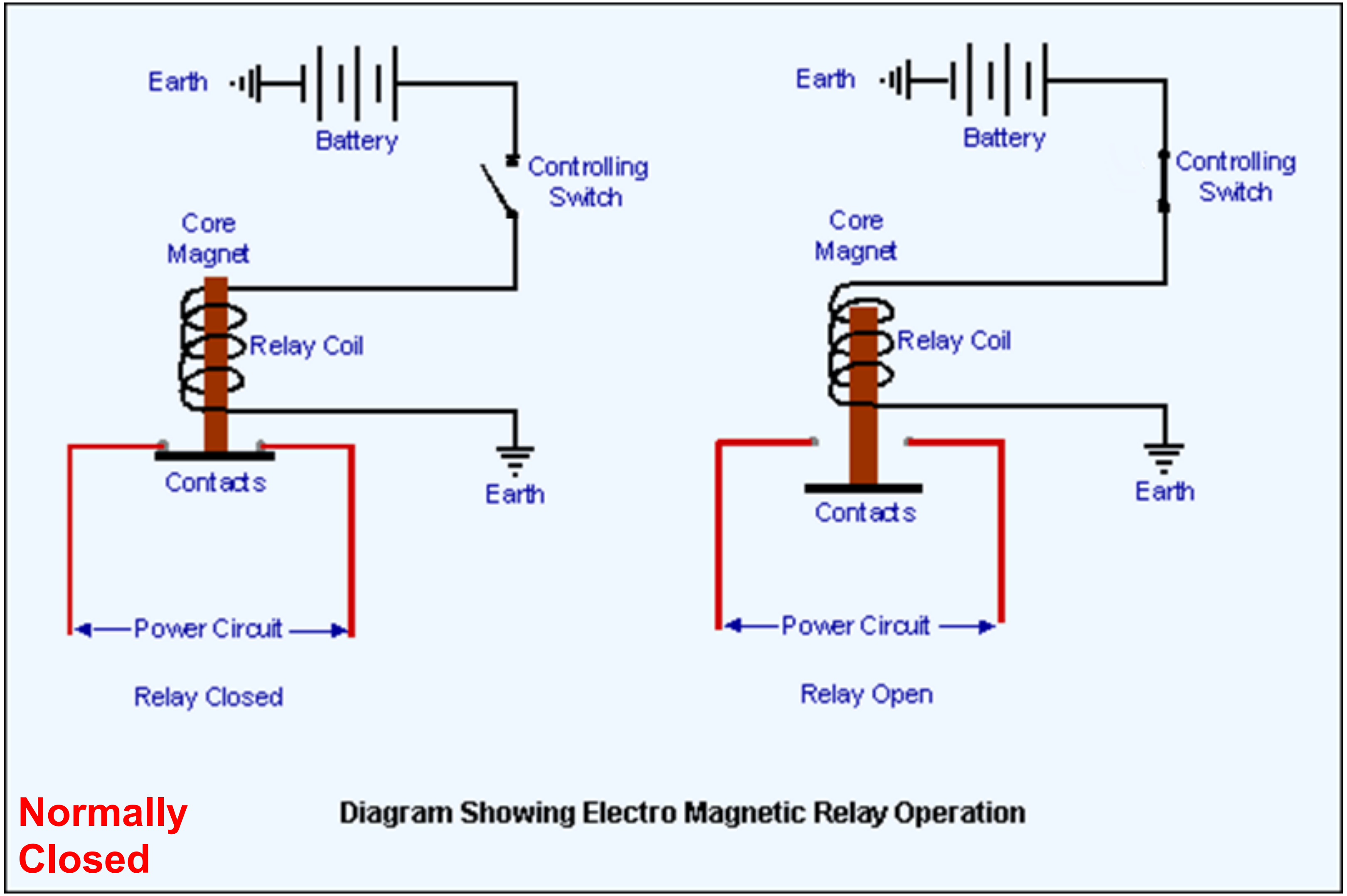
Sounder
Line
Line
Sounder
West
Station
(Sending)
Repeater
East
Station
(Receiving)

Relay



Earth
Battery
Controlling Switch
Core Magnet
Relay Coil
Contacts
Earth
Power Circuit
Relay Open
Earth
Battery
Controlling Switch
Core Magnet
Relay Coil
Contacts
Earth
Power Circuit
Relay Closed
Normally Open
Diagram Showing Electro Magnetic Relay Operation

Relay



Earth
Battery
Controlling Switch
Core Magnet
Relay Coil
Contacts
Earth
Power Circuit
Relay Closed
Earth
Battery
Controlling Switch
Core Magnet
Relay Coil
Contacts
Earth
Power Circuit
Relay Open
Normally Closed
Diagram Showing Electro Magnetic Relay Operation

Claude Shannon

Claude Elwood Shannon

en.wikipedia.org

Claude Elwood Shannon was an American mathematician, electronic engineer, and cryptographer known as "the father of information theory". Shannon is famous for having founded information theory with one landmark paper published in 1948. Wikipedia

Born: April 30, 1916, Petoskey

Died: February 24, 2001, Medford

Books: The mathematical theory of communication, Claude Elwood Shannon

Education: Massachusetts Institute of Technology, University of Michigan

Awards: IEEE Medal of Honor, National Medal of Science for Engineering, More

Claude Shannon's Masters Thesis

A SYMBOLIC ANALYSIS

OF

RELAY AND SWITCHING CIRCUITS

by

Claude Elwood Shannon

B.S., University of Michigan

1936

Submitted in Partial Fulfillment of the

Requirements for the Degree of

MASTER OF SCIENCE

from the

Massachusetts Institute of Technology

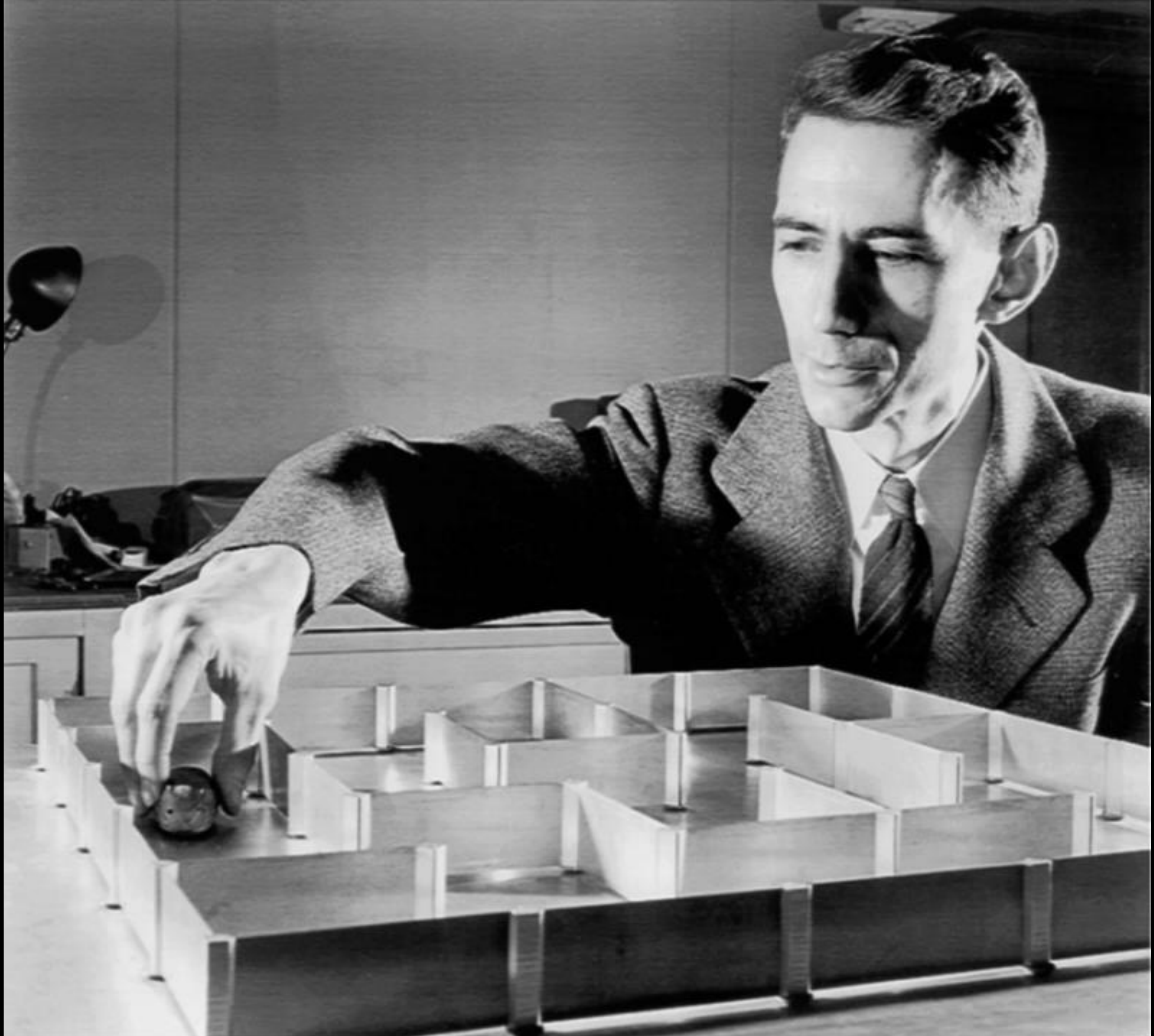
1940

Signature of Author ______________________________

Department of Electrical Engineering, August 10, 1937

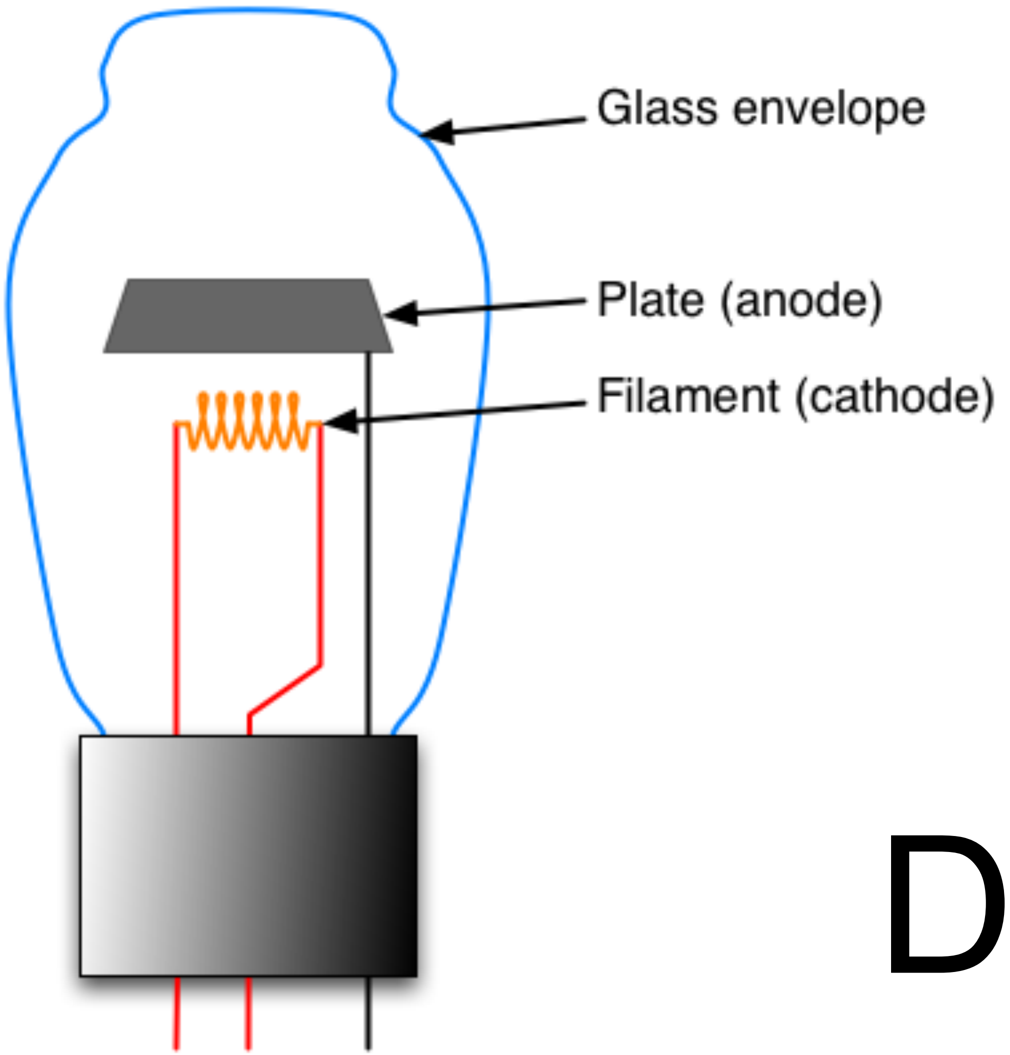
Signature of Professor
in Charge of Research ______________________________

Signature of Chairman of Department,
Committee on Graduate Students ______________________________



But…

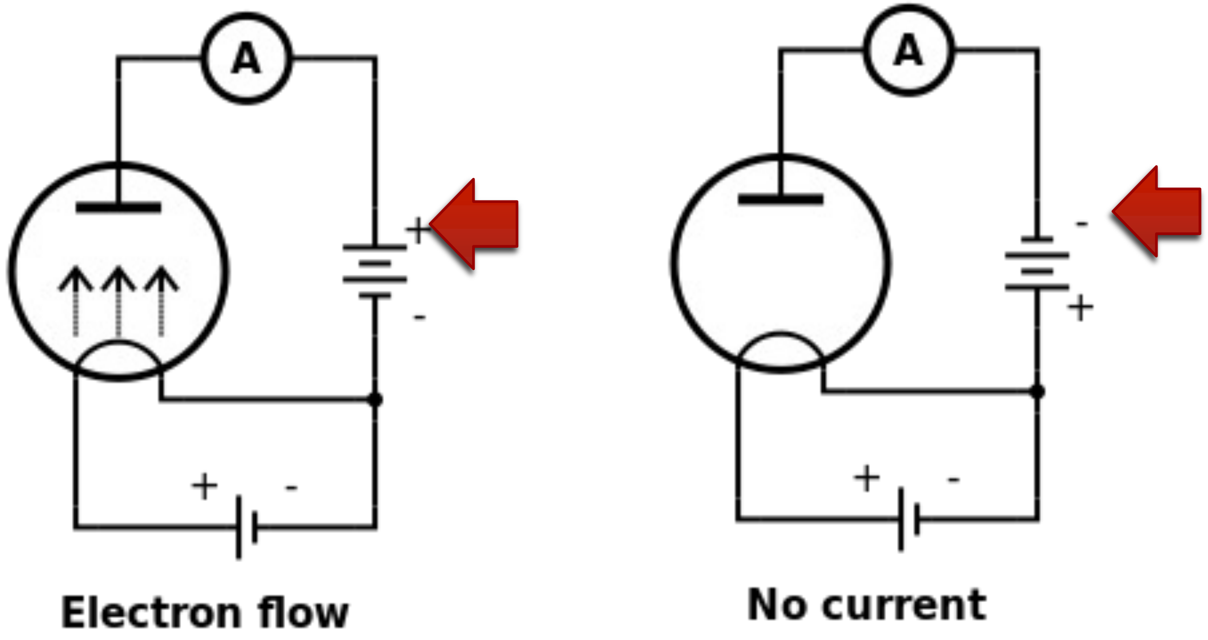
- Relays are
  - Slow
  - Good for hundreds of thousands of cycles
  - Wear out
  - Big
  - Noisy
- So…



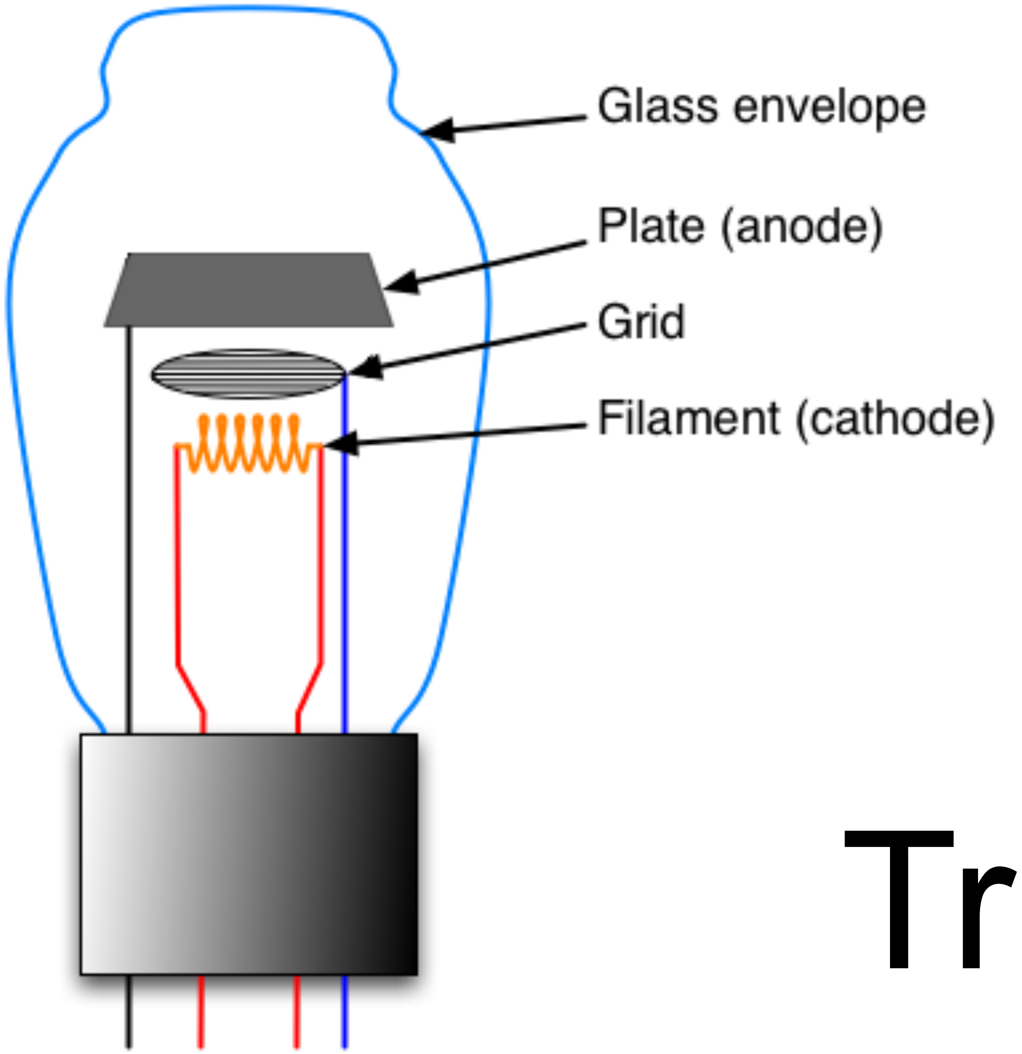
Diode

Edison Effect (Thermionic Emission)

- Note that the electrons will only flow in one direction!
- If an AC power source replaces the anode battery, the current changes polarity twice each cycle
- Current will only flow during the parts of the sine wave where the charge is negative
- The resulting current is thus DC with current flowing every half-cycle
- We call this diode a **rectifier** when it's used to convert AC to DC power



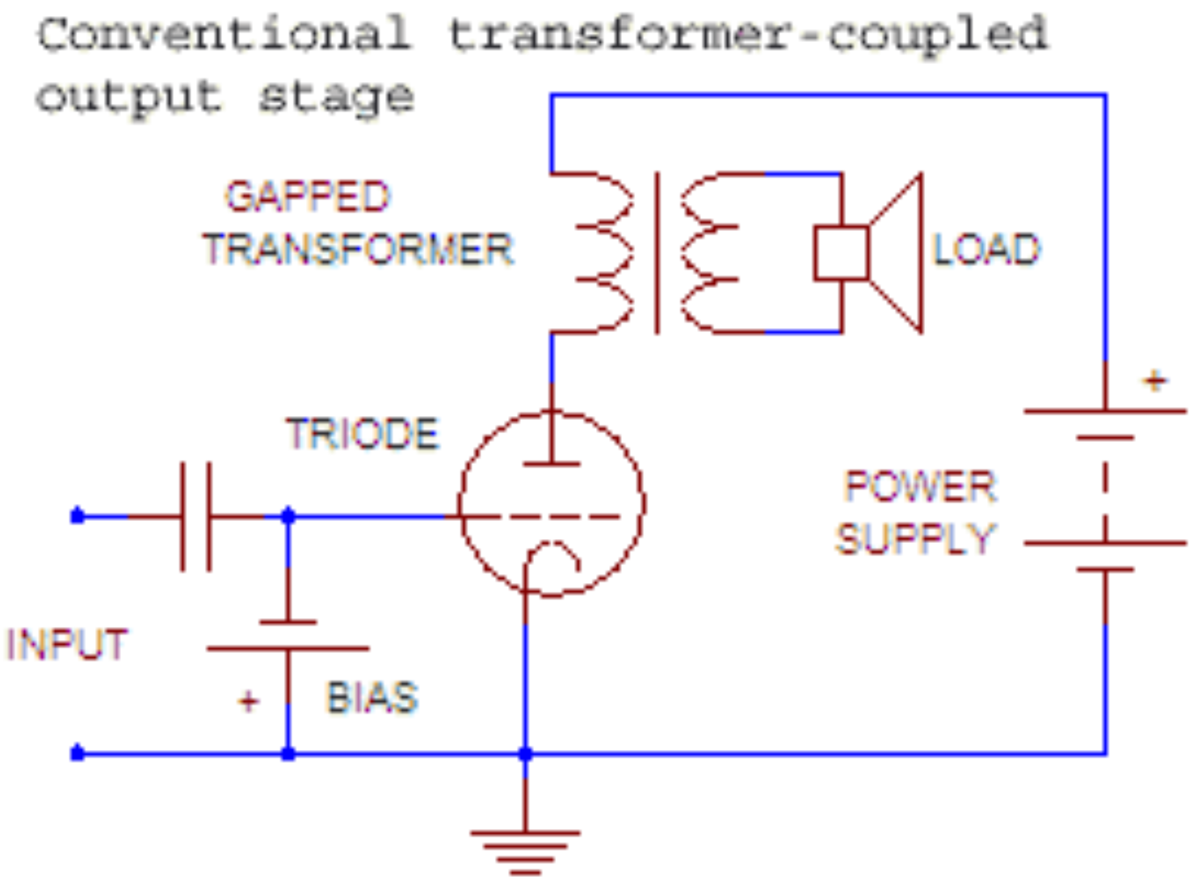
User:Omegatron - https://secure.wikimedia.org/wikipedia/commons/wiki/File:Edisoneffect.png



Triode

Triode as Amplifier

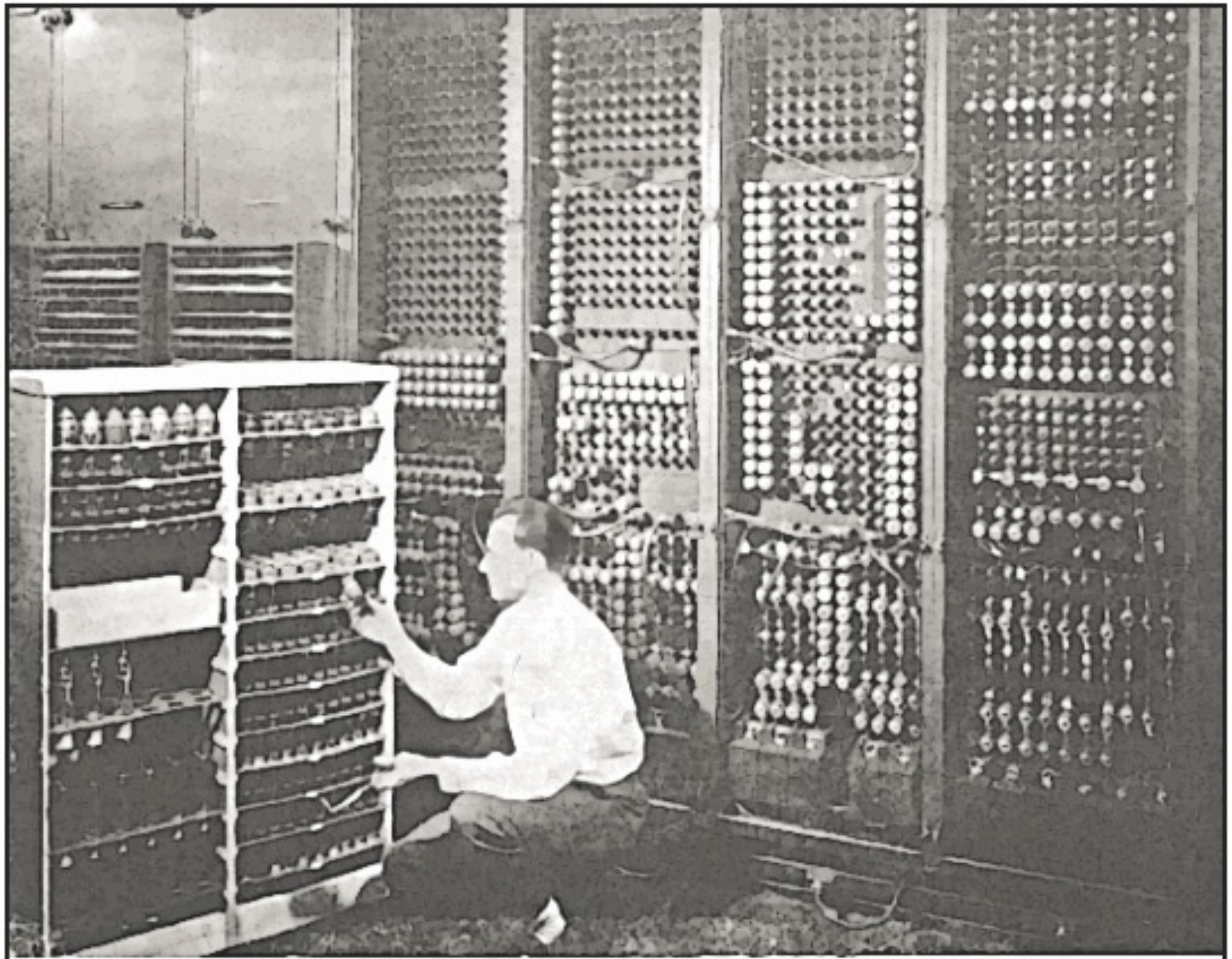
- The input signal is a current from a microphone that represents the sound waves hitting the microphone
- The signal modulates the charge on the triode grid
- The charge applied to the grid permits or inhibits the flow of a larger current from the filament to the anode
- The output signal to the speaker is a higher current but proportional to the input signal



Vacuum Tubes

- Used as
  - Switches
  - Amplifiers
- In
  - Radio
  - Television
  - Stereo
  - Radar
  - Computers

eniac vacuum tubes



Replacing a bad tube meant checking among ENIAC's 19,000 possibilities.

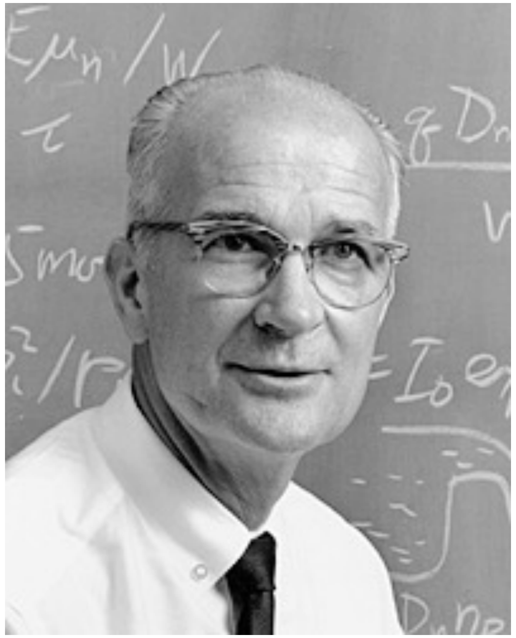
ENIAC

But…

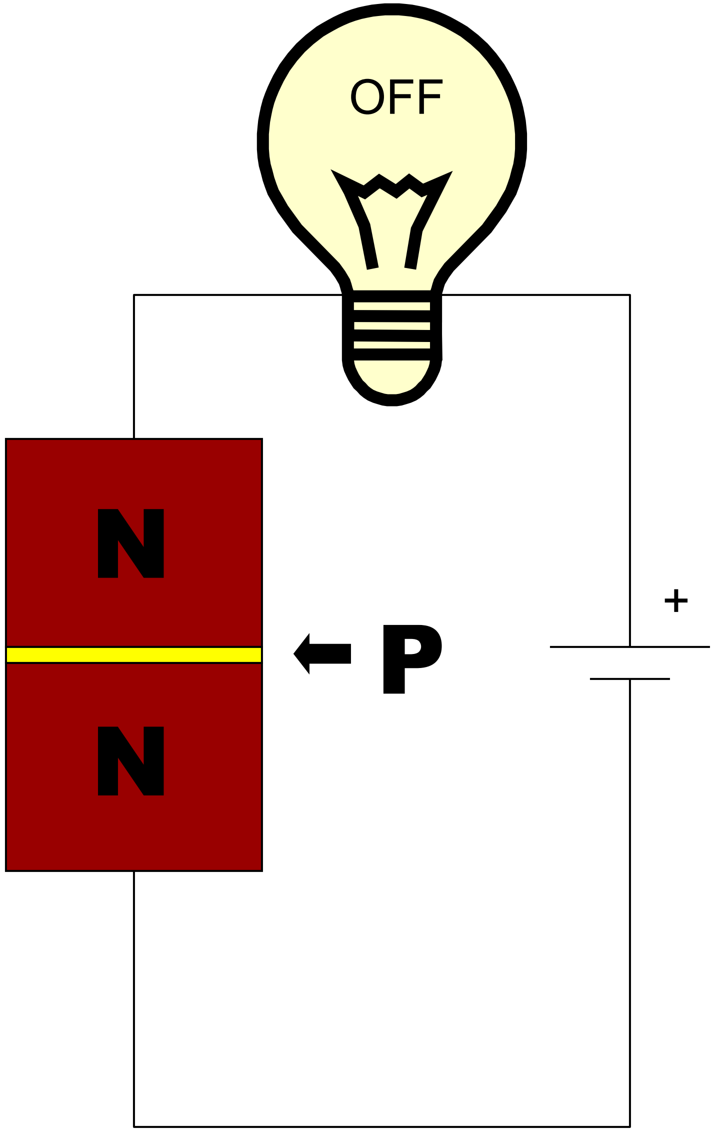
- Tubes are
  - Hot
  - High power consumption
  - High voltages
  - Unreliable
  - Expensive
  - Consist of many individual components
- So...

William Shockley

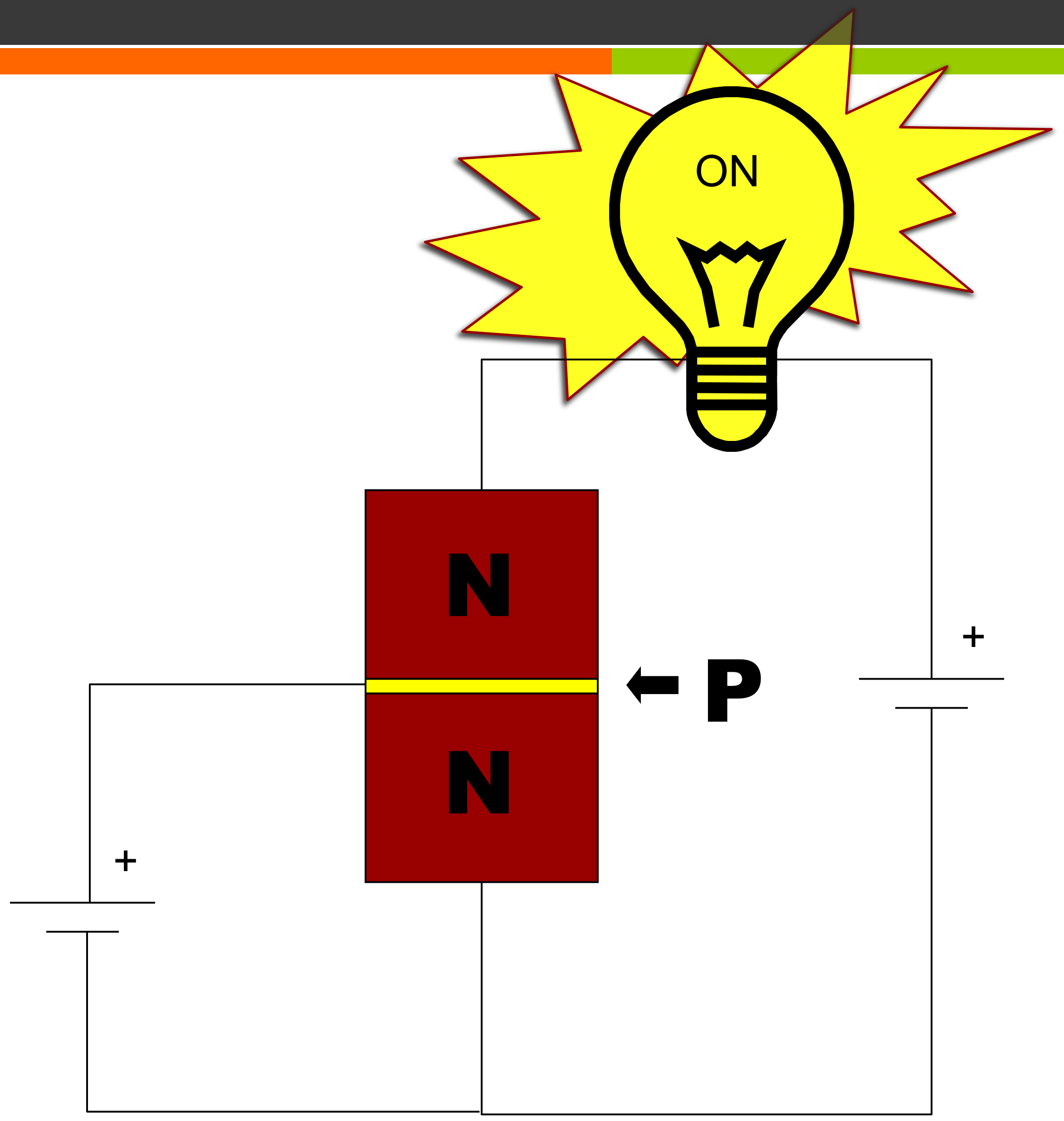
- Born February 13, 1910 Greater London, England, UK
- Died August 12, 1989 (aged 79) Stanford, California, US
- Nationality American
- Alma mater MIT, Caltech
- Known for
  - Point-contact transistor and BJT
  - Shockley diode equation
  - Read-Shockley equation
  - Shockley–Ramo theorem
  - Haynes-Shockley experiment
  - Shockley–Queisser limit
- Awards
  - Nobel Prize in Physics (1956)
  - Comstock Prize in Physics (1953)
  - Oliver E. Buckley Condensed Matter Prize (1953)
  - Wilheln Exner Medal (1963)
  - IEEE Medal of Honor (1980)



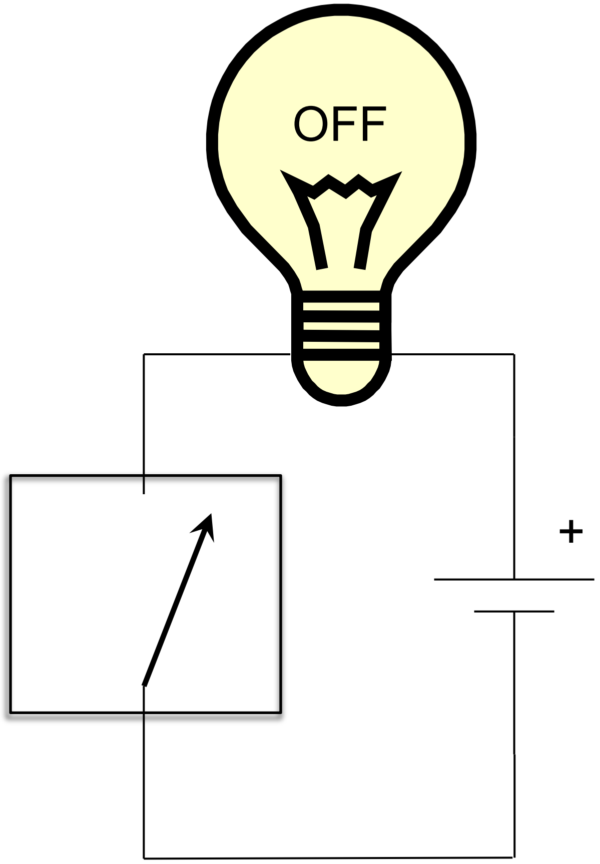
So to save money...



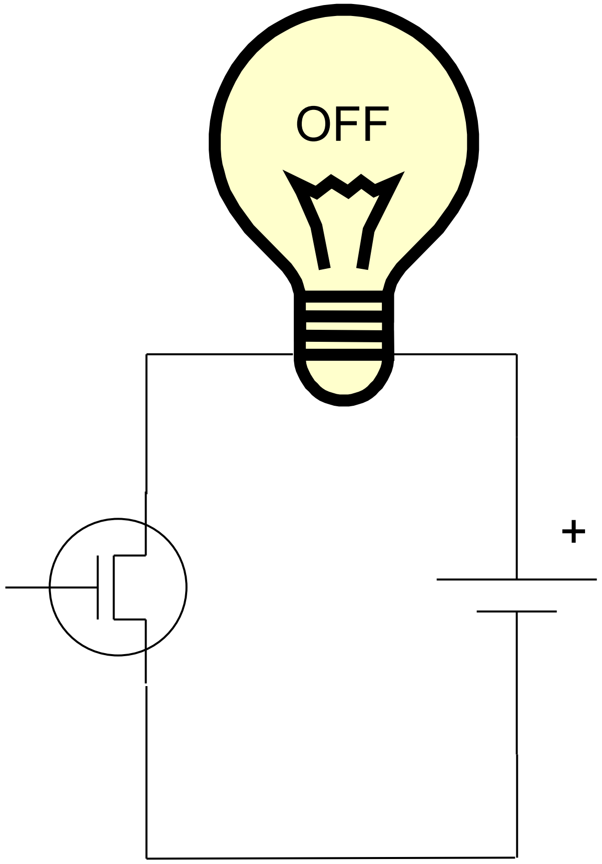
Transistors



Different Switches



A physical switch



An electronic switch

Who made the connection that Boolean algebra expressions could be implemented by electric circuits?

A. William Shockley
B. Thomas Edison
C. Claude Shannon
D. John Ambrose Fleming

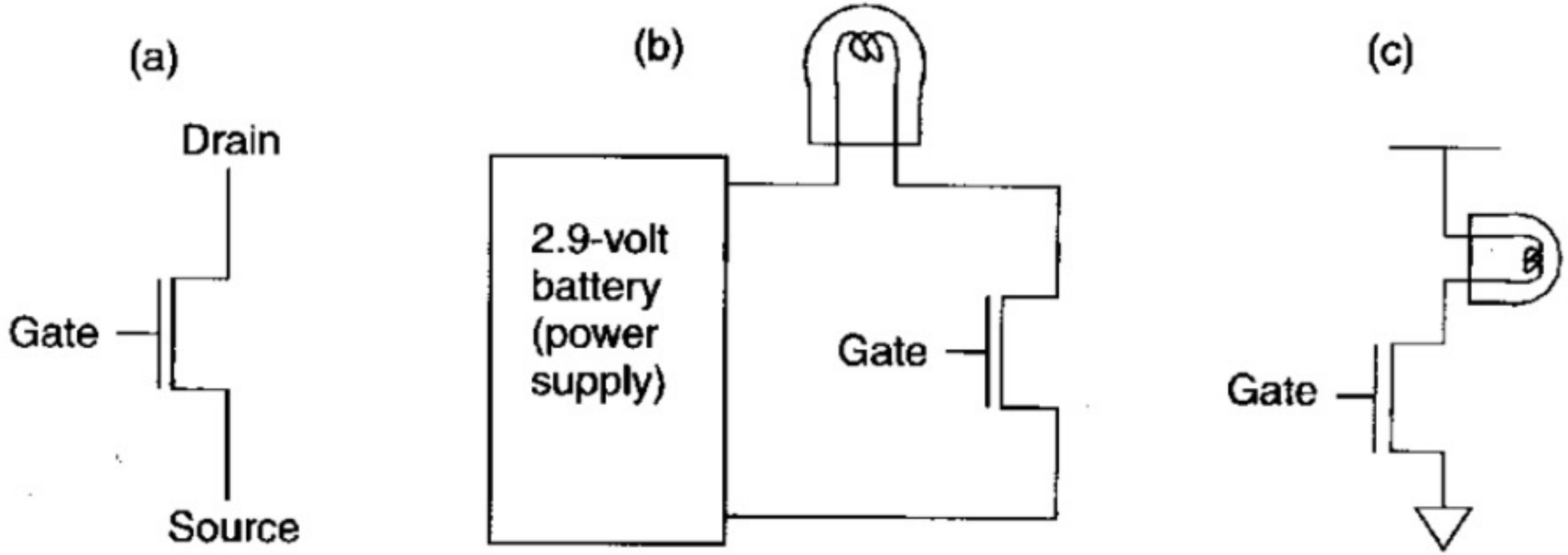
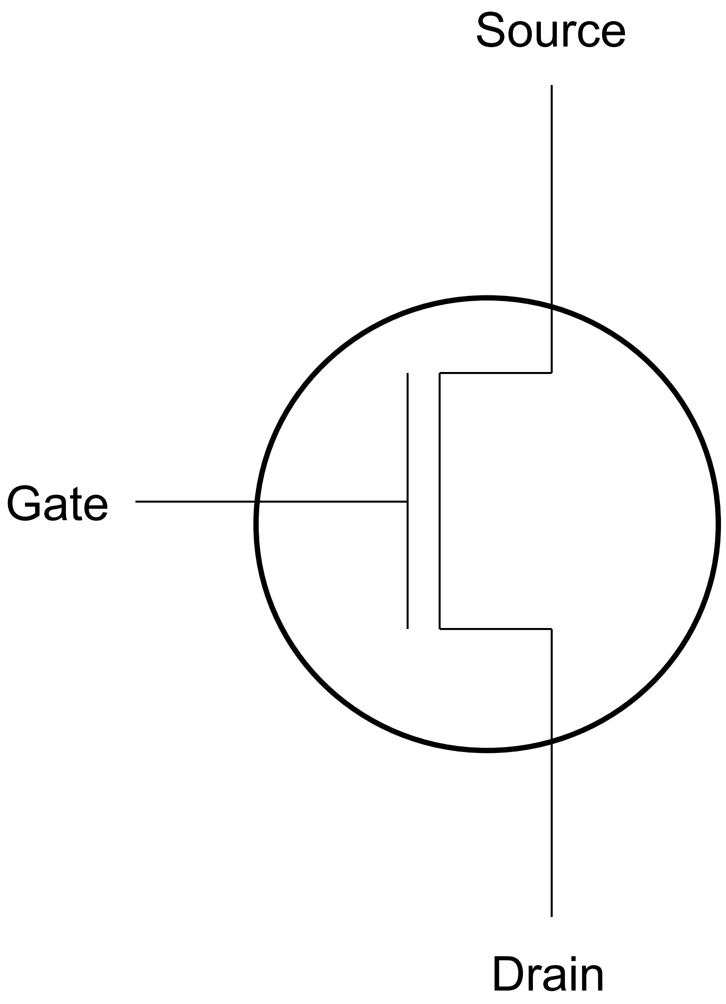
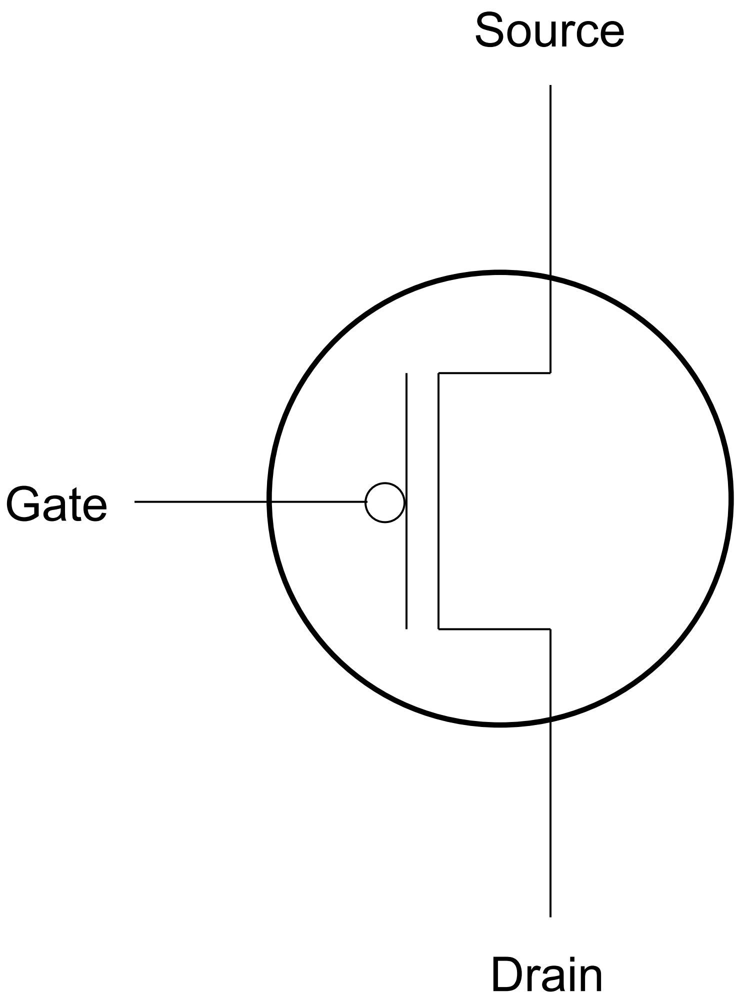


Figure 3.2 The n-type MOS transistor

Complementary Transistors



n-Type MOS



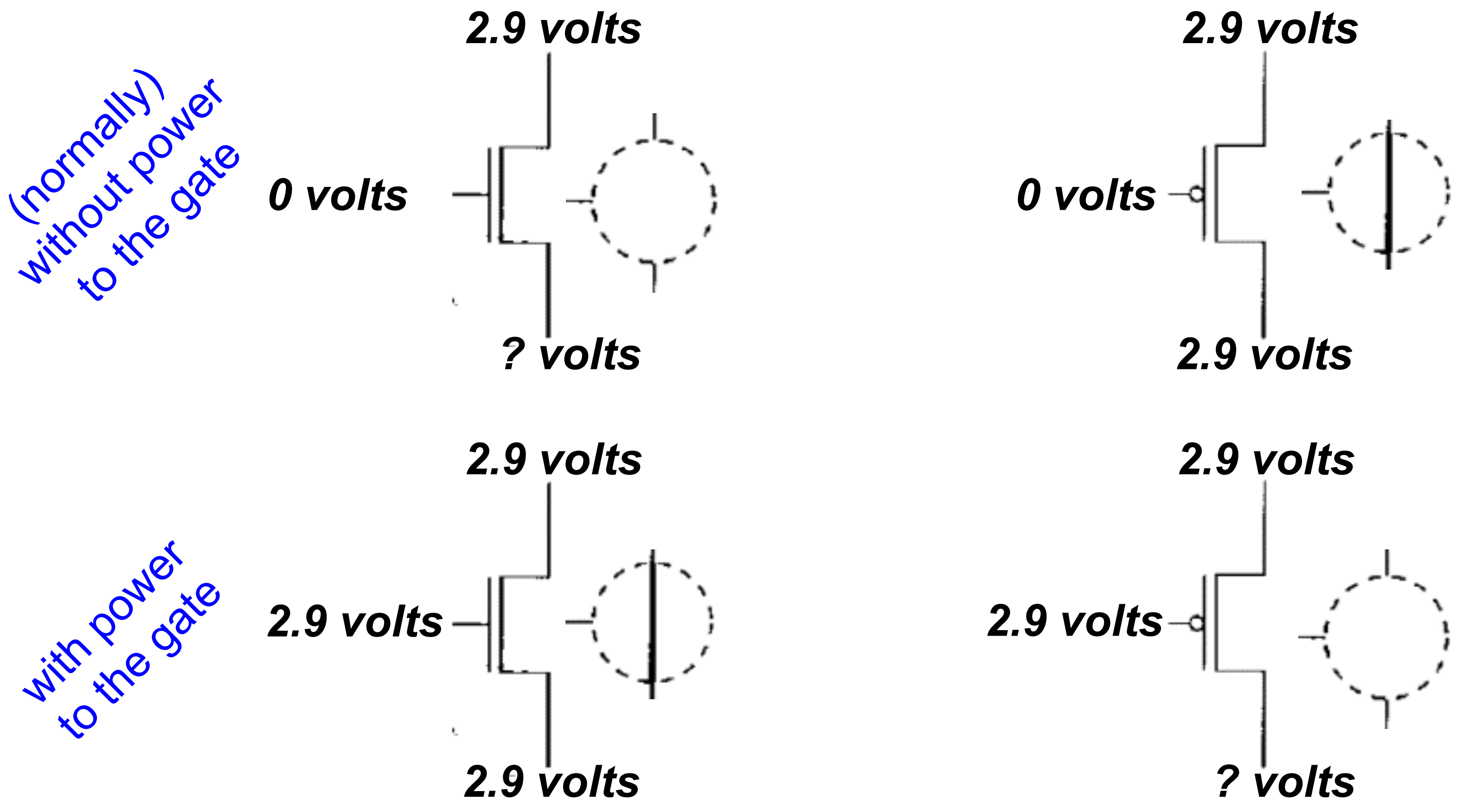
p-Type MOS

Common Misconception: Digital Wires Have 3 (not 2) States!

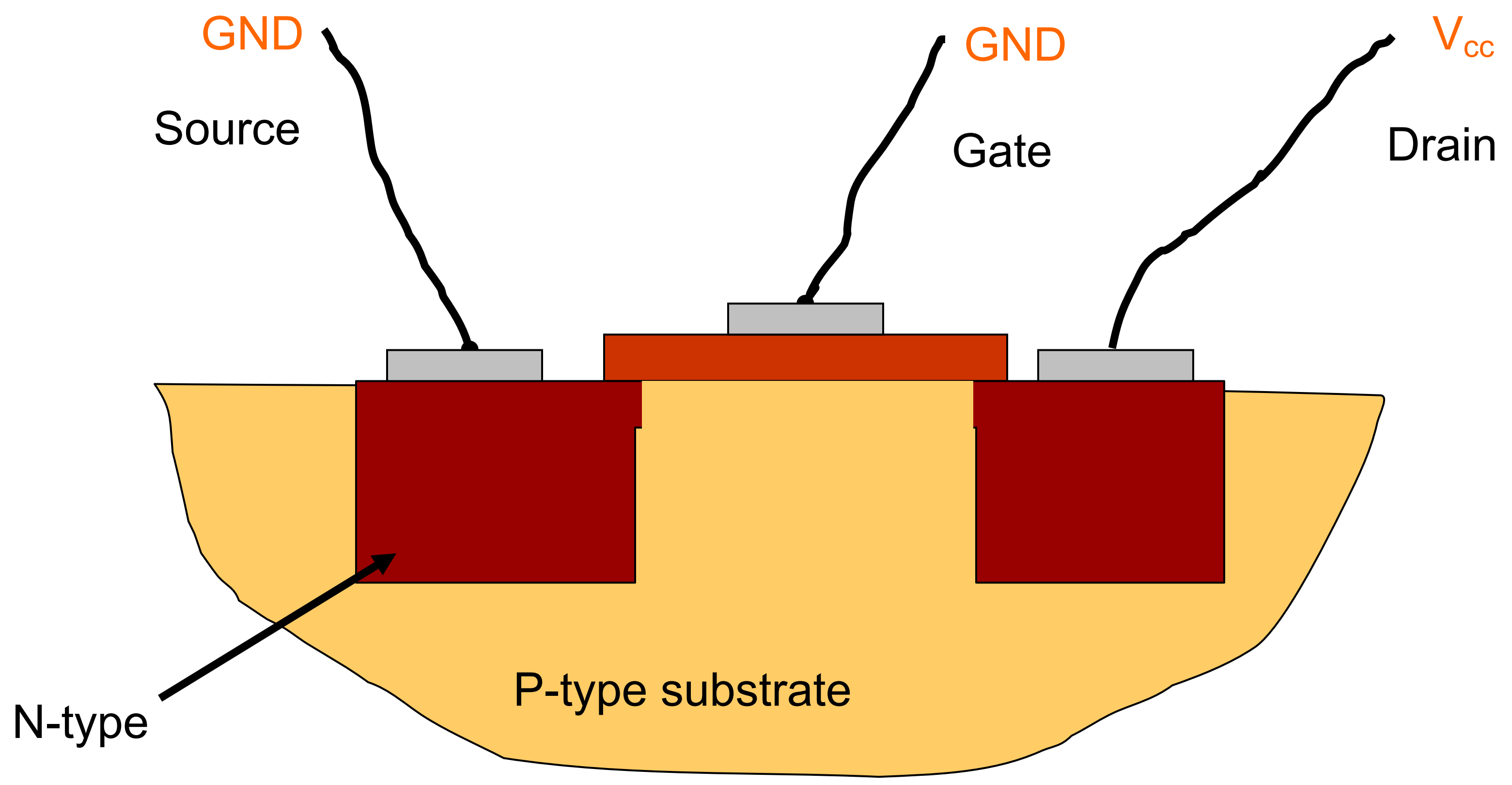
- A wire with some designated voltage (e.g. +2.9 volts) can represent a logical 1.
- A wire with some designated voltage (e.g. 0 volts or ground) can represent a logical 0.
- A wire that is not connected to 2.9 volts or ground is said to be **floating** or in a **high impedance state** and its value can randomly vary from a logical 0 to 1.

Transistor Comparison

n-Type (normally open) **p-Type** (normally closed)

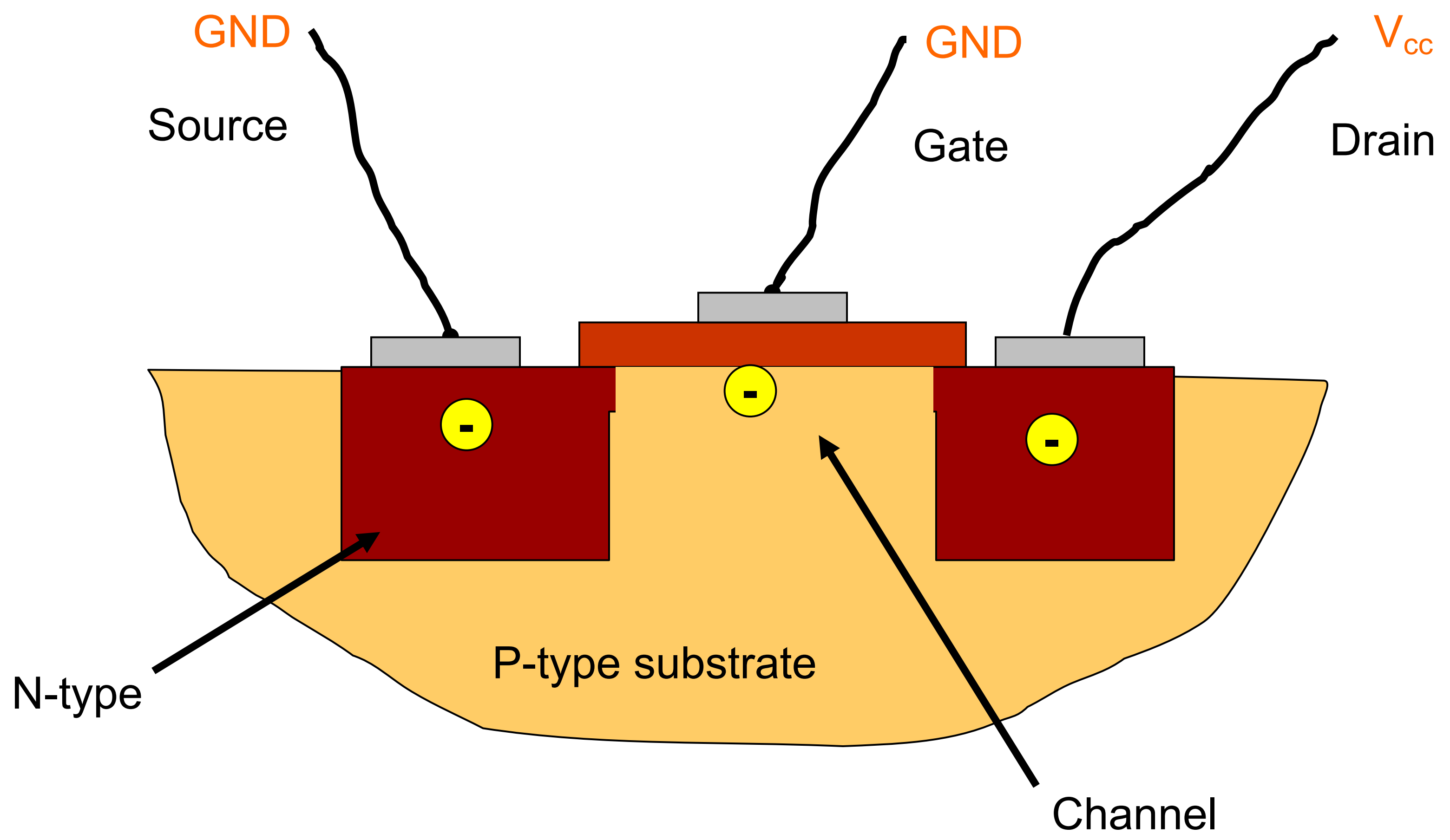


N-type MOS FET*



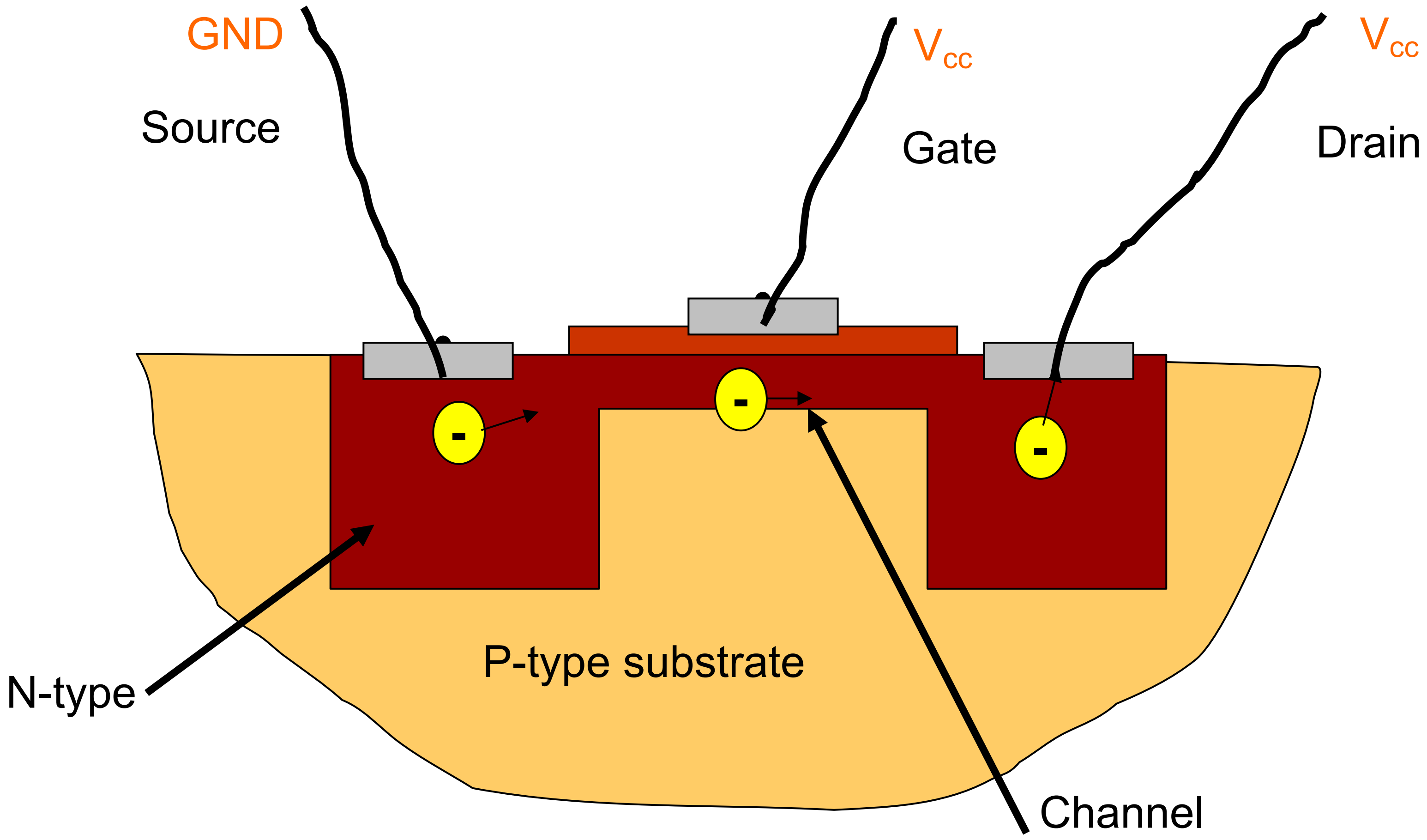
*Metal Oxide Semiconductor Field Effect Transistor

Gate Open

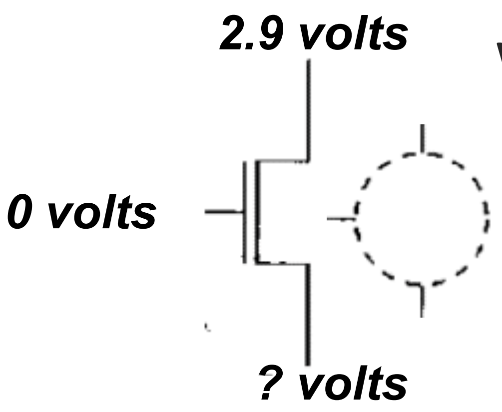


GND
Source
GND
Gate
V_{cc}
Drain
-
-
-
N-type
P-type substrate
Channel

Gate Closed


GND
Source
V_{cc}
Gate
V_{cc}
Drain
-
-
-
N-type
P-type substrate
Channel

n-Type (normally open)



You have an n-type transistor connected to 2.9v and 0v as shown. What voltage value will we measure at the drain?

A. 0v

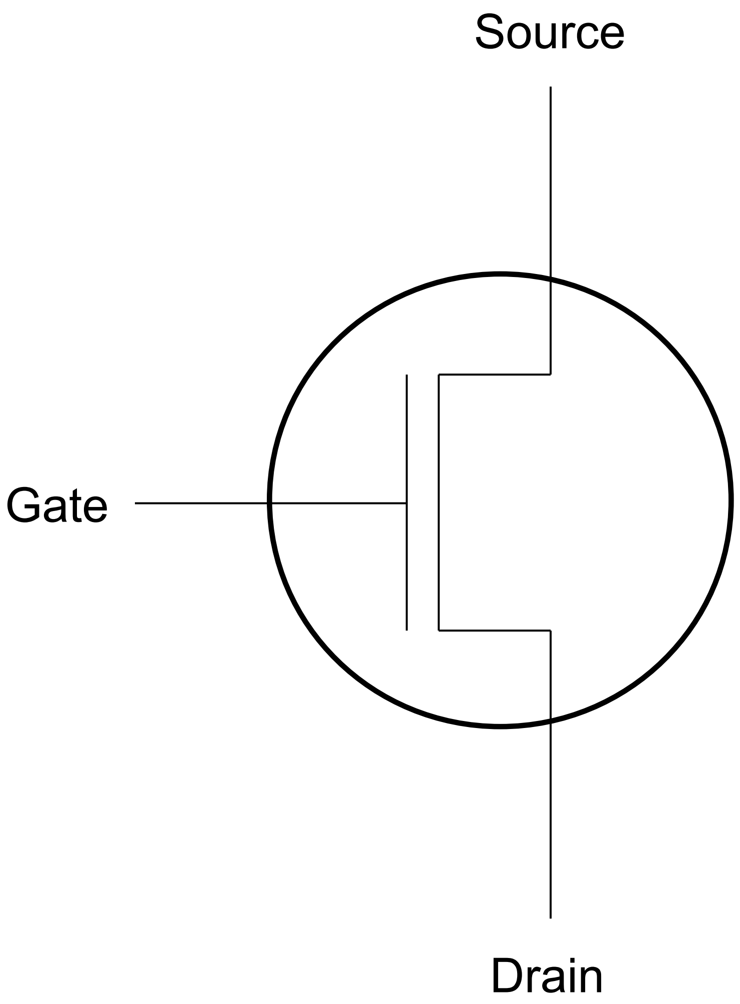
B. 2.9v

C. 1.45v

D. Unknown

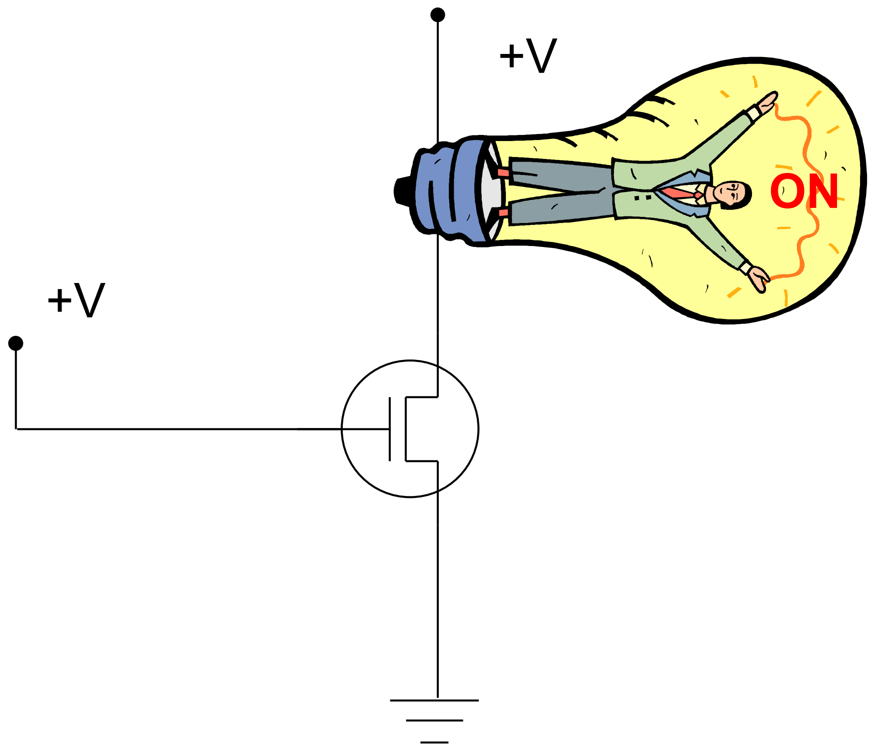
Because the gate is at 0v, the drain is disconnected from the source and is floating. We can't tell what value will be measured.

N-Type FET



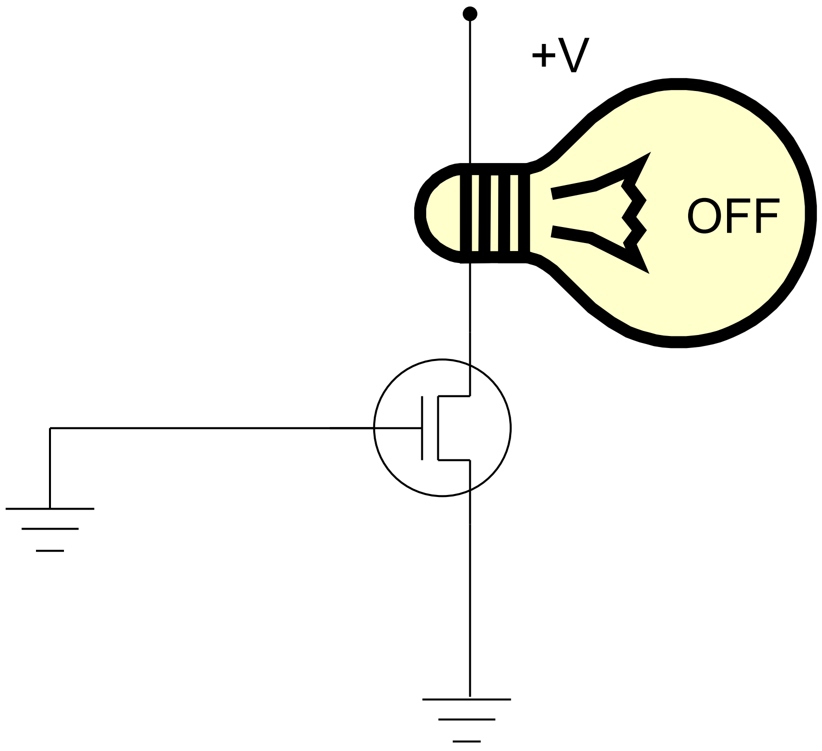
CircuitSim N-Type MOS FET

N-Type with Gate at +V



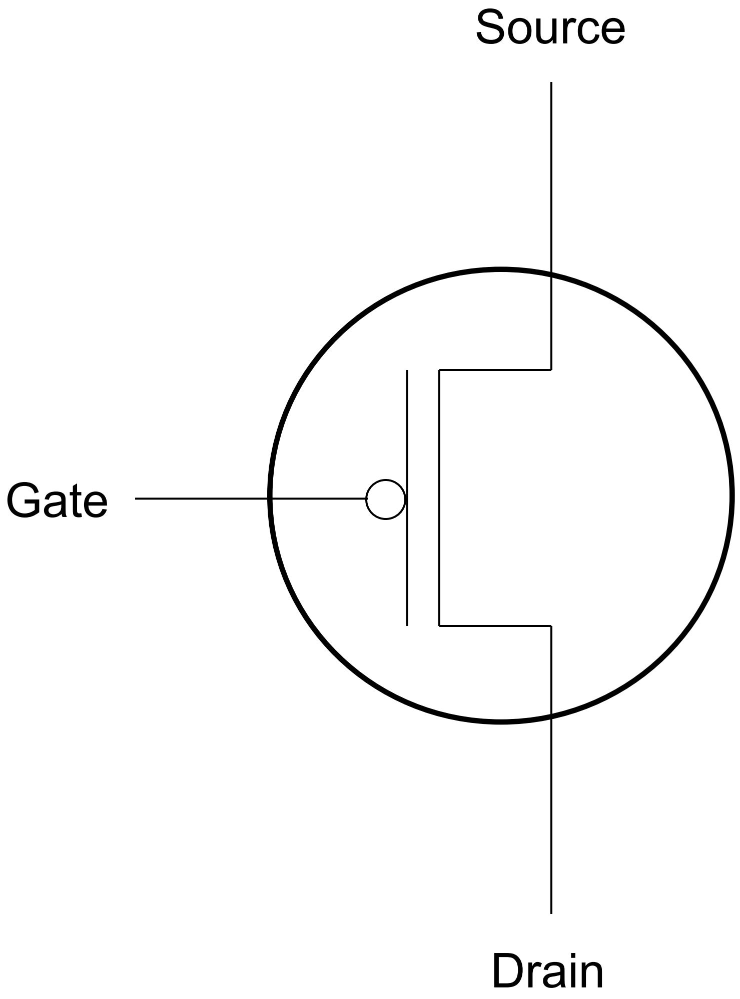
With the supply voltage applied to the device the transistor acts like a closed or connected switch

N-Type with Gate at Ground



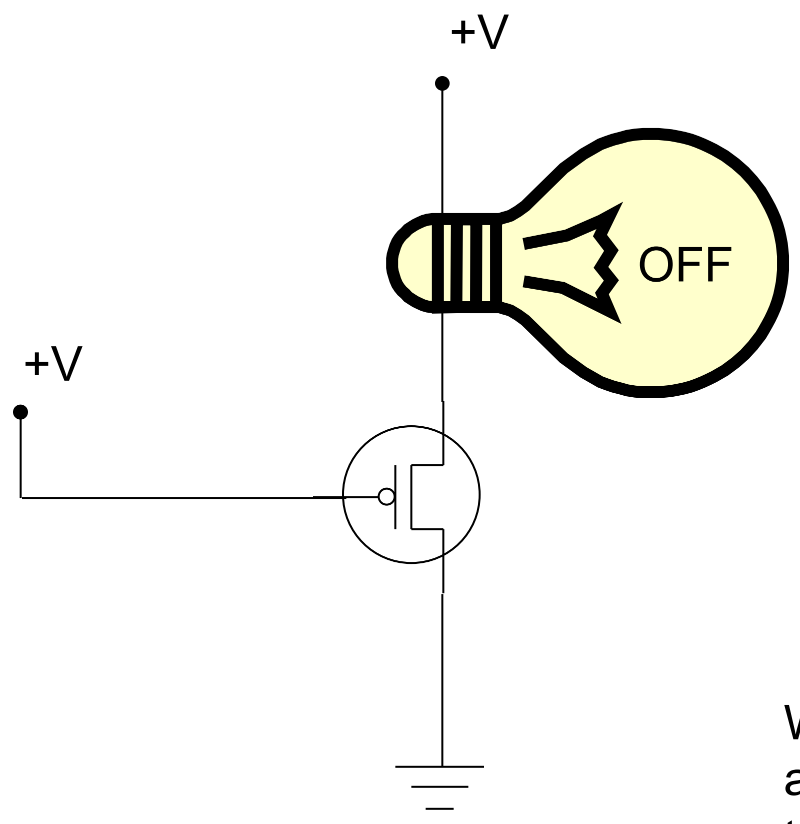
Open

P-Type MOS FET



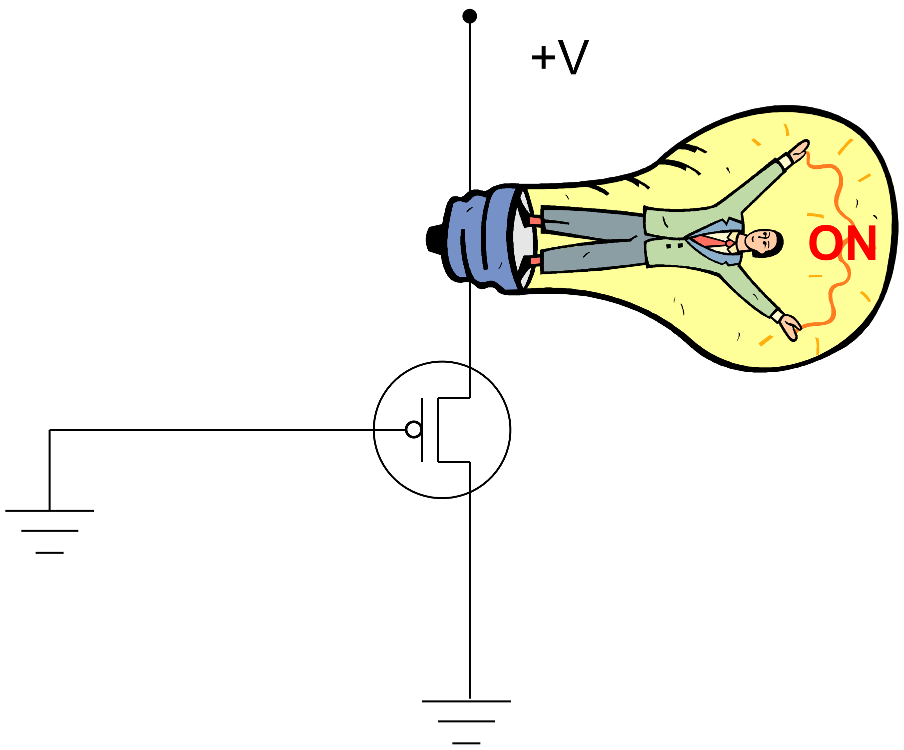
CircuitSim P-Type MOS FET

P-Type with Gate at +V



With the supply voltage applied to the device the transistor acts like a open or disconnected switch

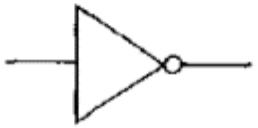
P-Type with Gate at Ground



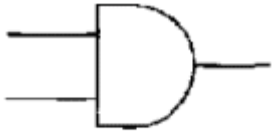
Closed

Logical Operations Revisited

A	NOT
0	1
1	0



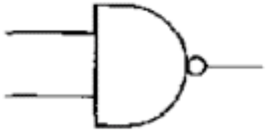
A	B	AND
0	0	0
0	1	0
1	0	0
1	1	1



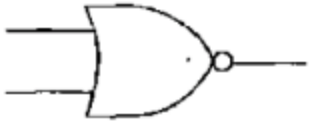
A	B	OR
0	0	0
0	1	1
1	0	1
1	1	1

A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	0

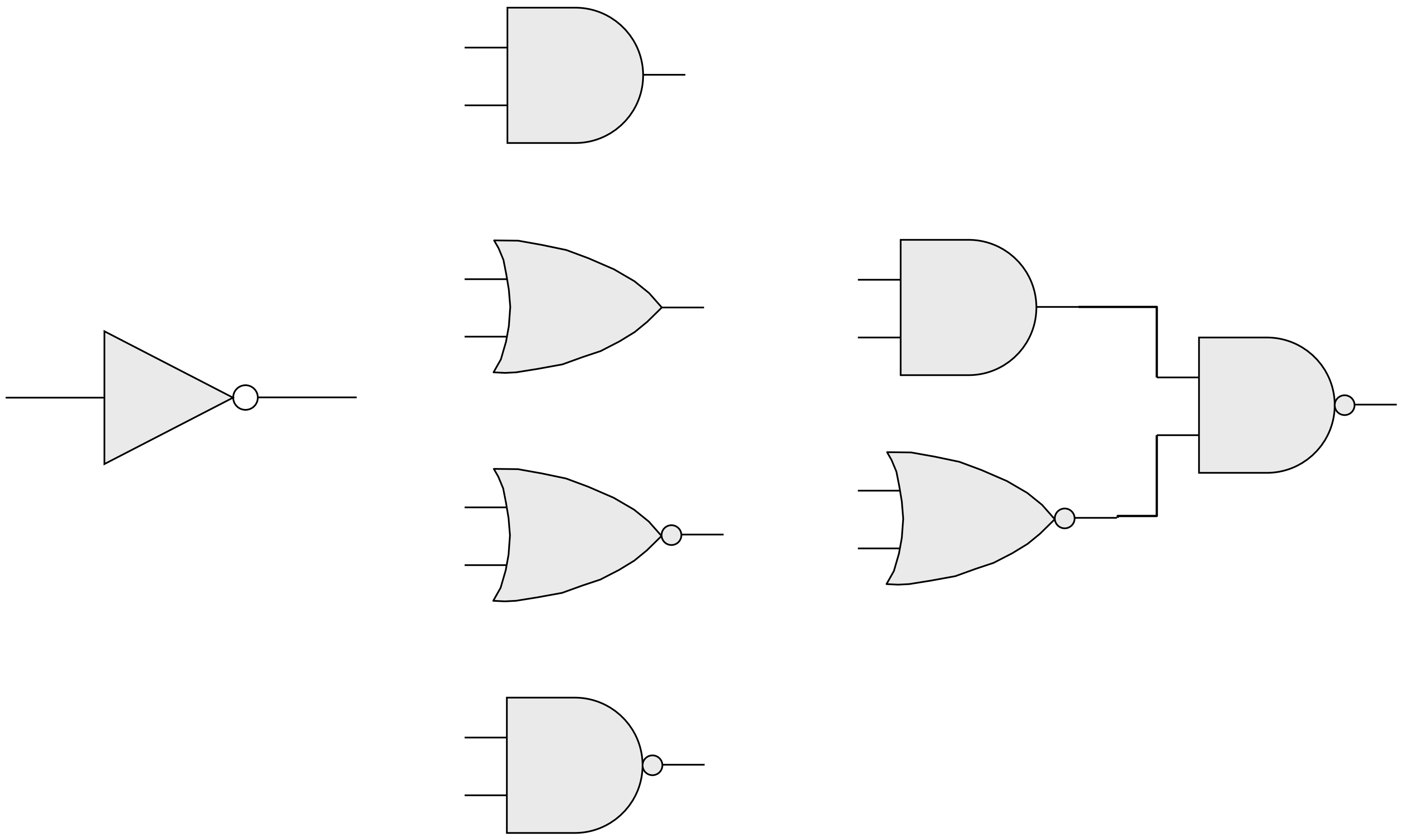
A	B	NAND
0	0	1
0	1	1
1	0	1
1	1	0



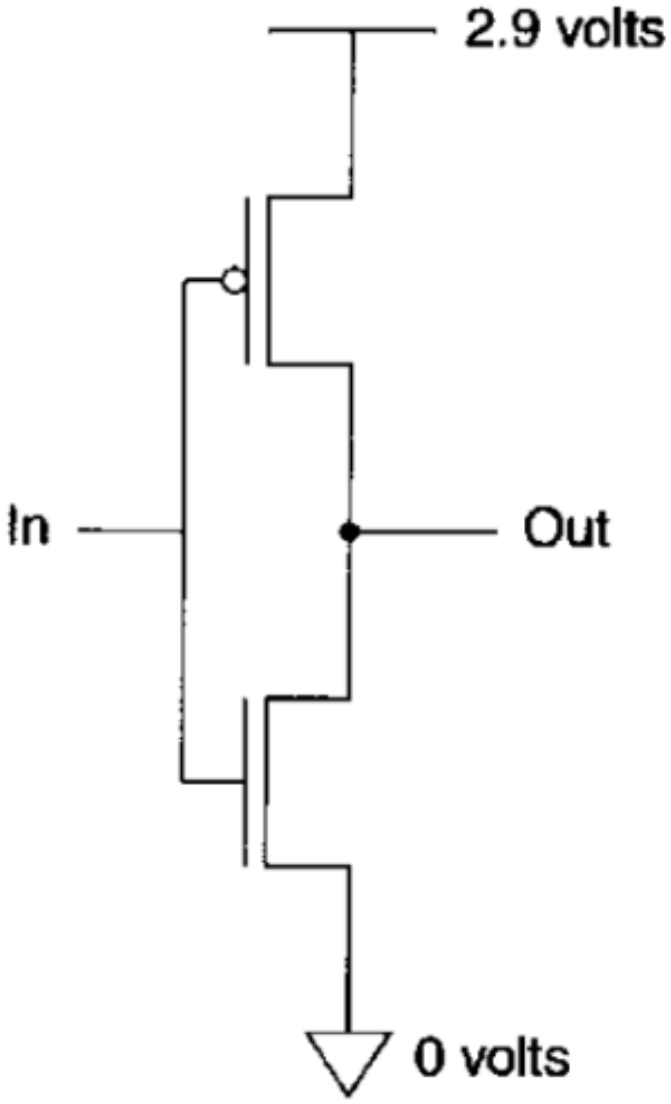
A	B	NOR
0	0	1
0	1	0
1	0	0
1	1	0



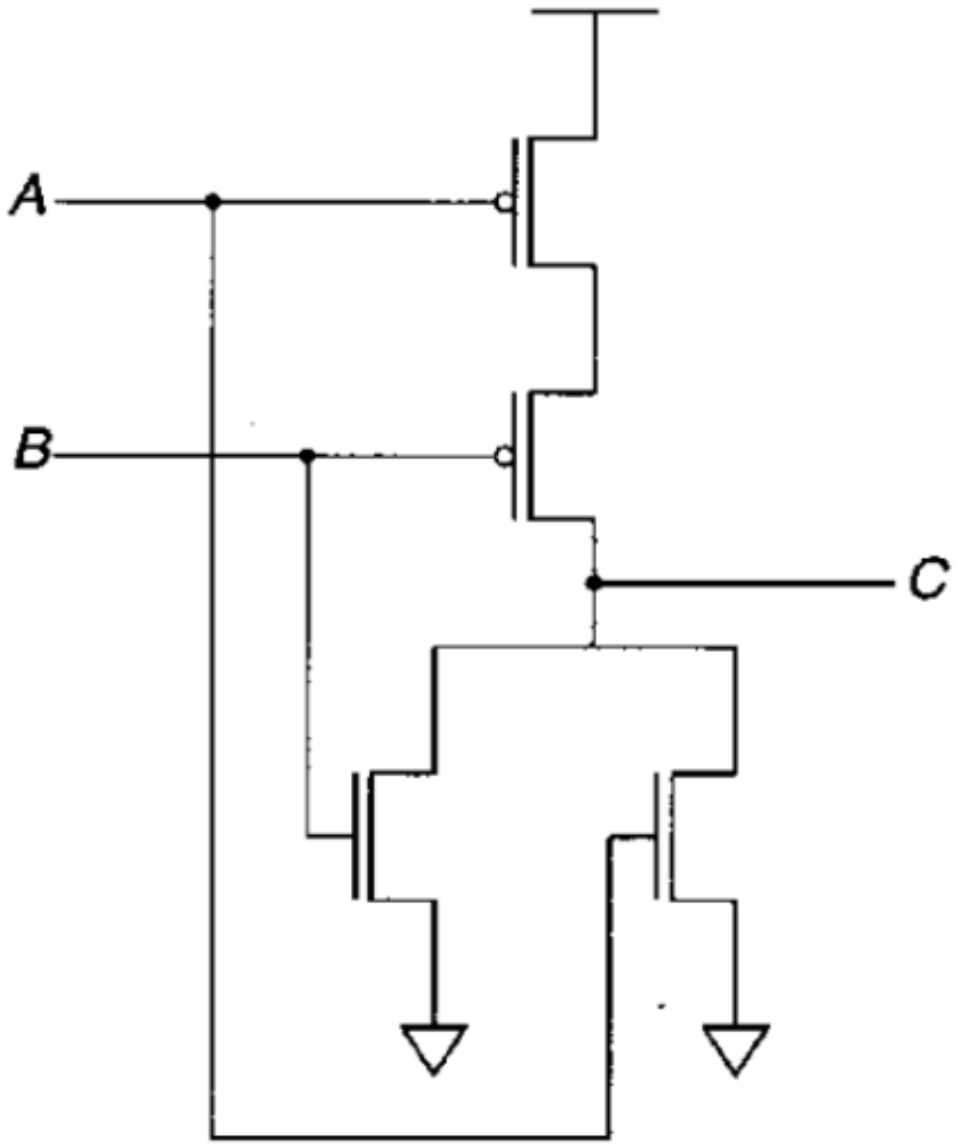
Boolean Gate Representation



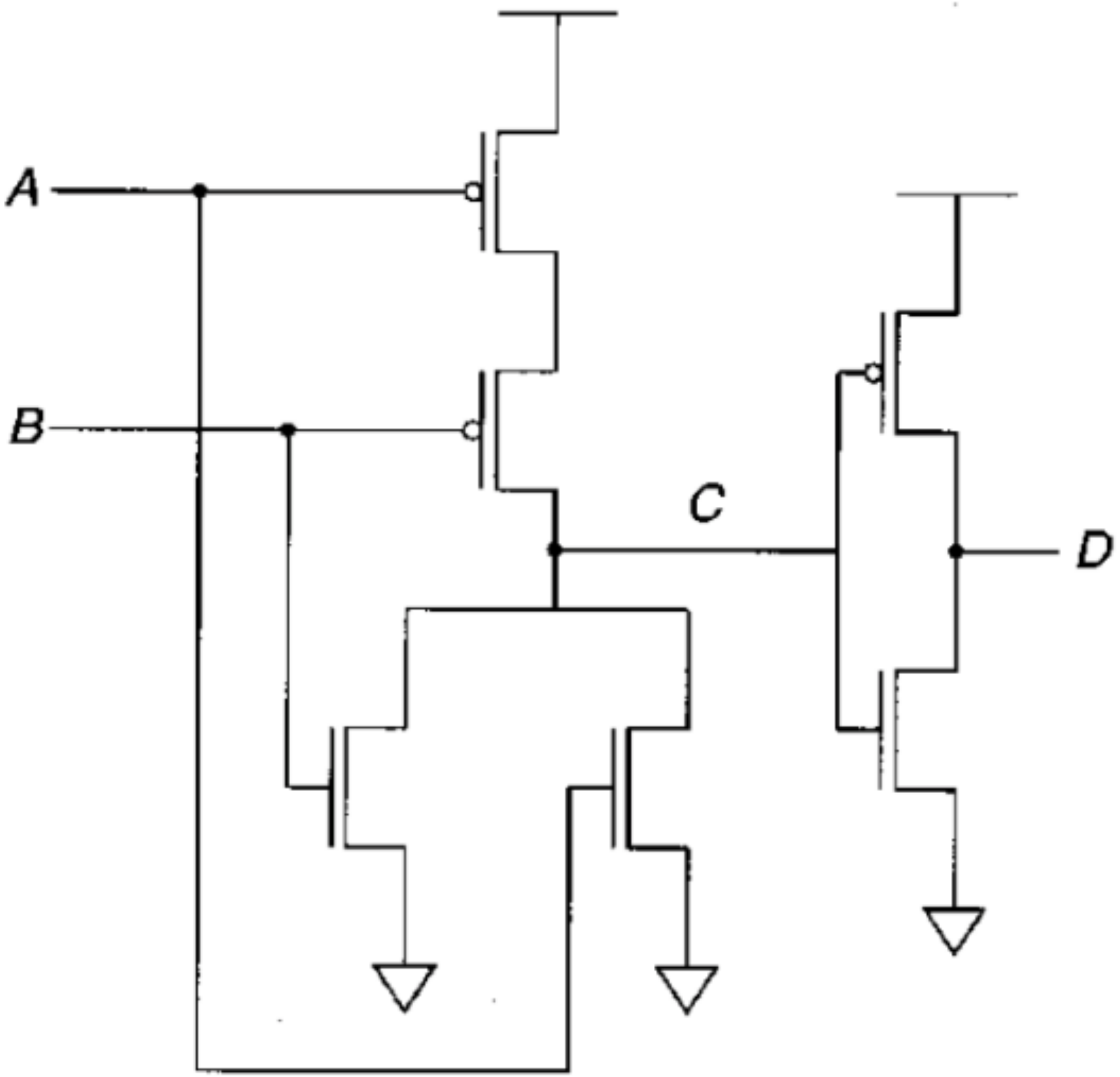
NOT Gate (Inverter)



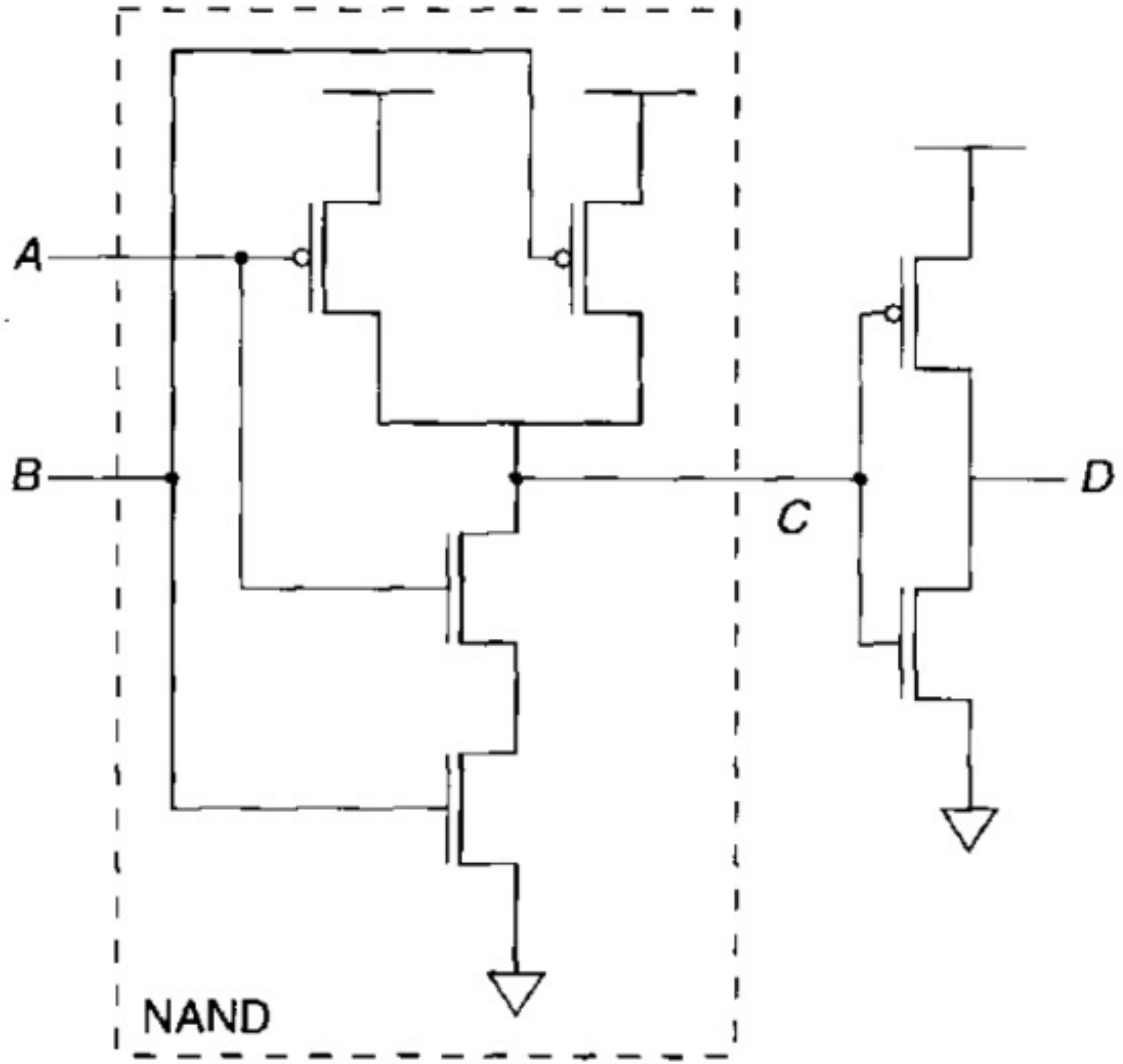
NOR Gates…



...OR Gates



NAND and AND Gates

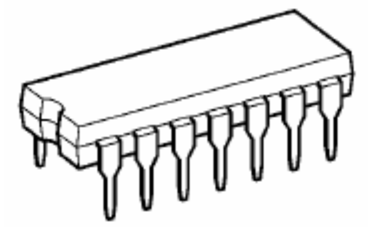


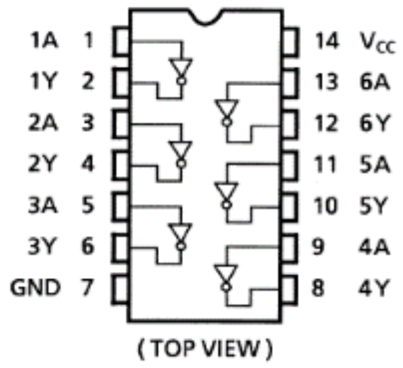
But…

- Discrete transistors
  - Are still big
  - Use a lot of material to build
  - Use a lot of power (and hence generate a lot of heat); however smaller transistors need smaller amounts of power
  - Need a lot of individual connections, all of which are prone to failure
  - Are so useful that we need ***lots*** of them
- So…

Digital Logic: Building Blocks

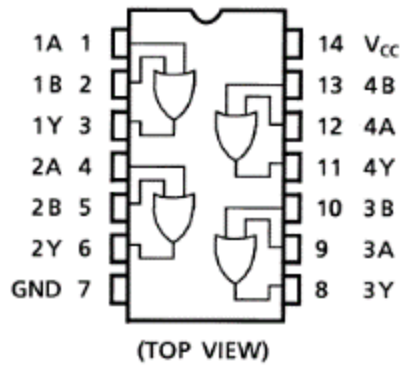
- The basic digital logic building blocks are available as pre-packaged self-contained integrated circuits.
- These chips are fast, inexpensive and easy to work with.
- Each chip contains 1-6 gates. Complicated designs can require thousands of gates and hundreds of chips.





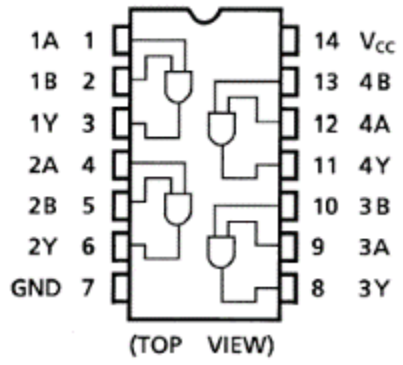
6 NOT Gates

74HC04



4 OR Gates

74HC32

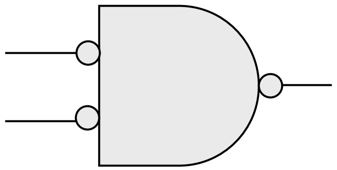


4 AND Gates

74HC08

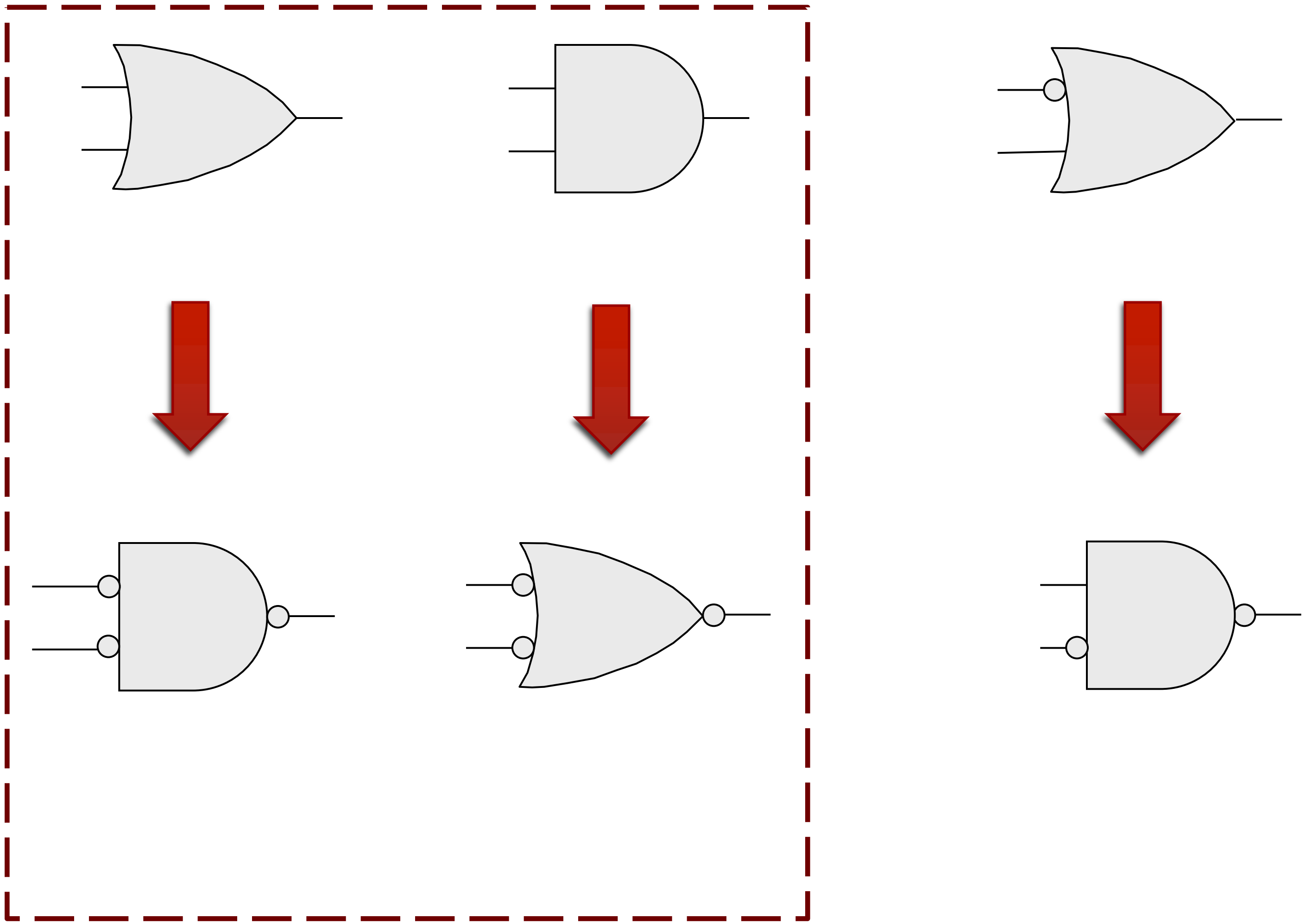
DeMorgan's Law

- $(A'B')' = A+B$
- $(A'+B')' = AB$

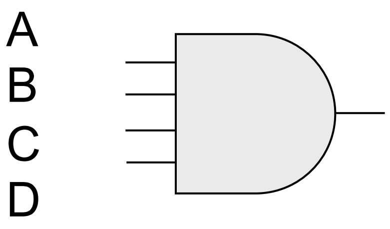


A	B	A'	B'	A'B'	(A'B')'	A'+B'	(A'+B')'	A+B	AB
0	0	1	1	1	0	1	0	0	0
0	1	1	0	0	1	1	0	1	0
1	0	0	1	0	1	1	0	1	0
1	1	0	0	0	1	0	1	1	1

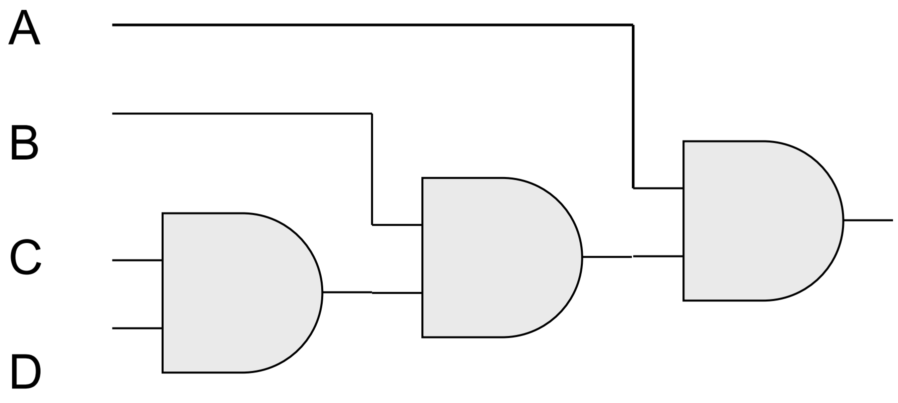
Conte Bubble Theorem



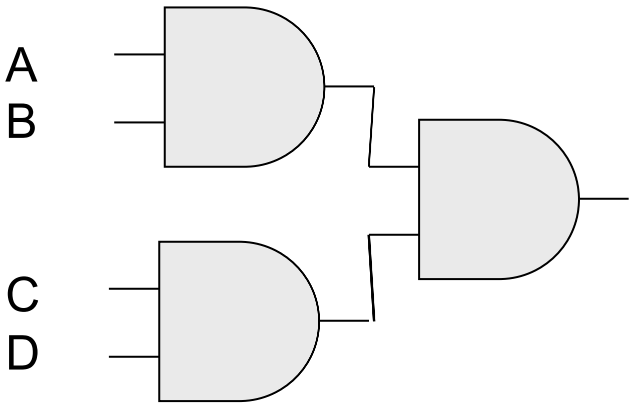
Larger Gates



ABCD

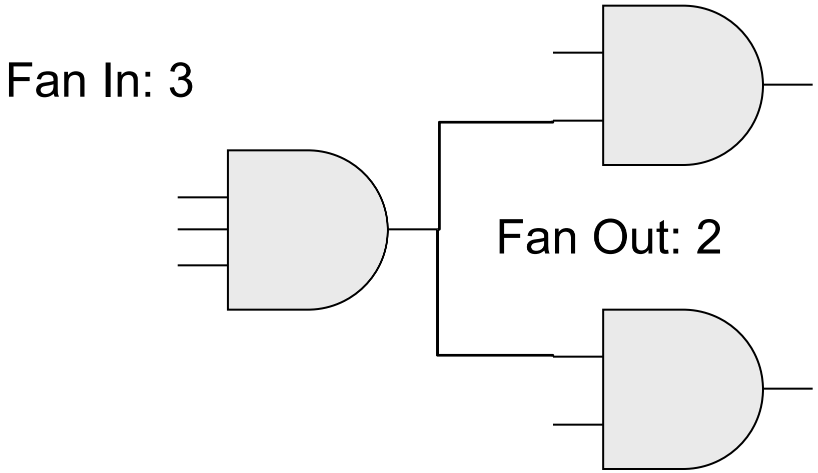


(A(B(CD)))



((AB)(CD))

Fan-In and Fan-Out



Warning

- Important concept approaching!

Combinational Logic

- A combination of AND, OR, NOT (plus NAND & NOR)
- The same inputs always produce same output

Whack-A-Mole!

Fun!

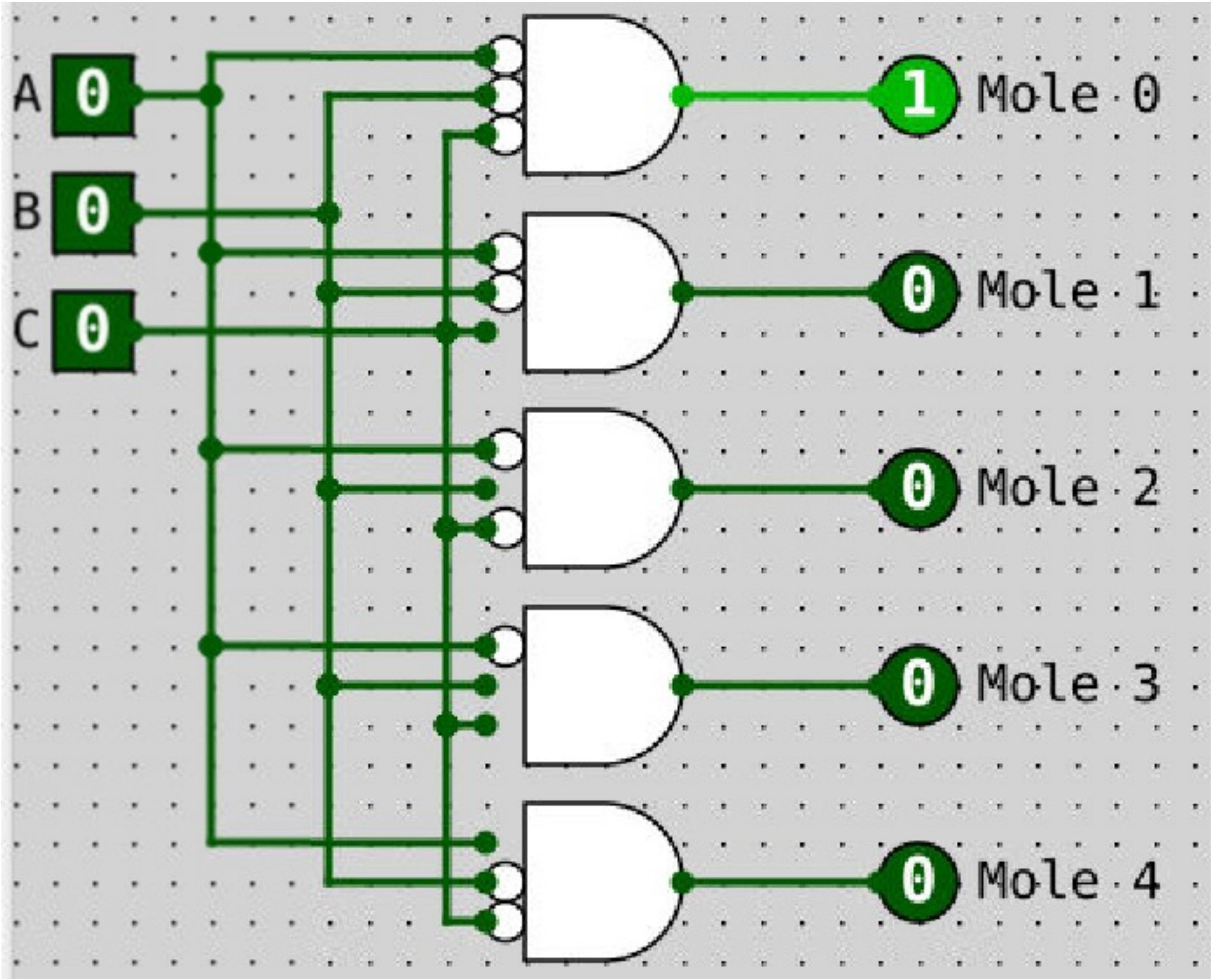

Quality
Cost
Delivery
MANAGER

Problem

- You have an n bit binary number which signifies which mole is selected. How do we electrically actuate the desired mole?

[Csim]

Decoder



Question

If you had a decoder with a 5 bit input (i.e. a 5 bit binary number) how many outputs at most could the decoder have?

2^5 possible input values

So a maximum of $2^5 = 32$ output values

A. 5
B. 10
C. 16
D. 32
E. 64

The winning number is 53,122

Question

If you have a decoder with *n* input bits (i.e. an n bit binary number) what is the most outputs the decoder could have?

A. 2n

B. n^2

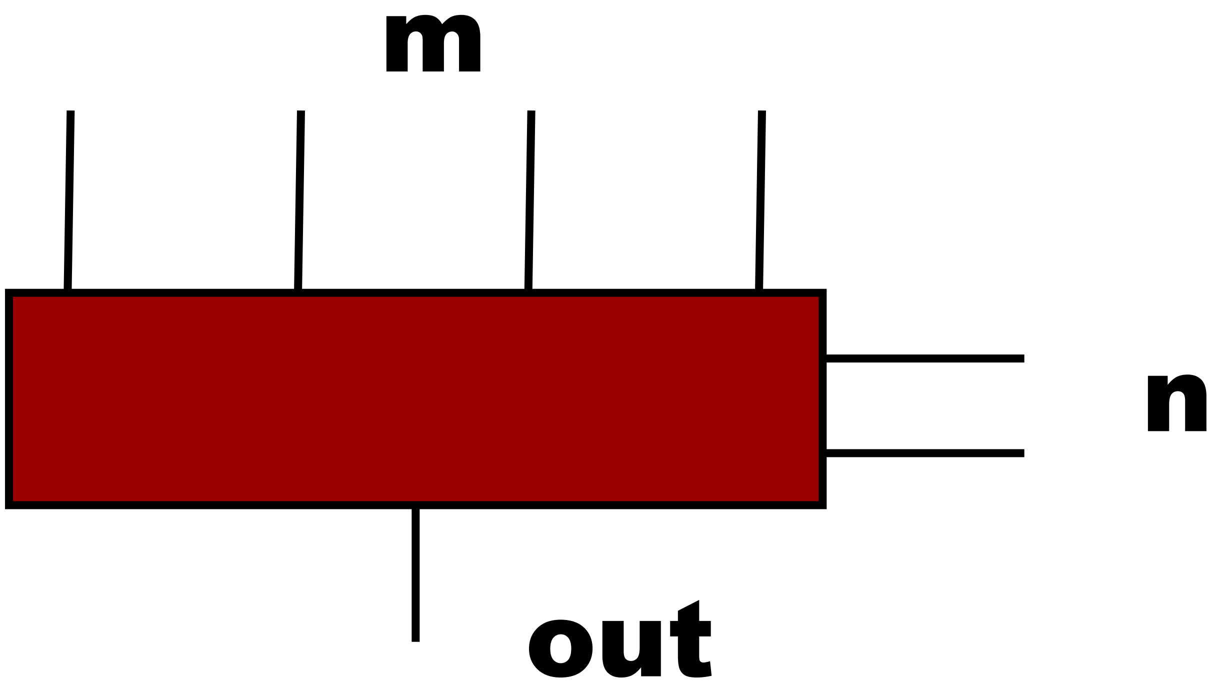
C. 2^n

D. Cannot determine

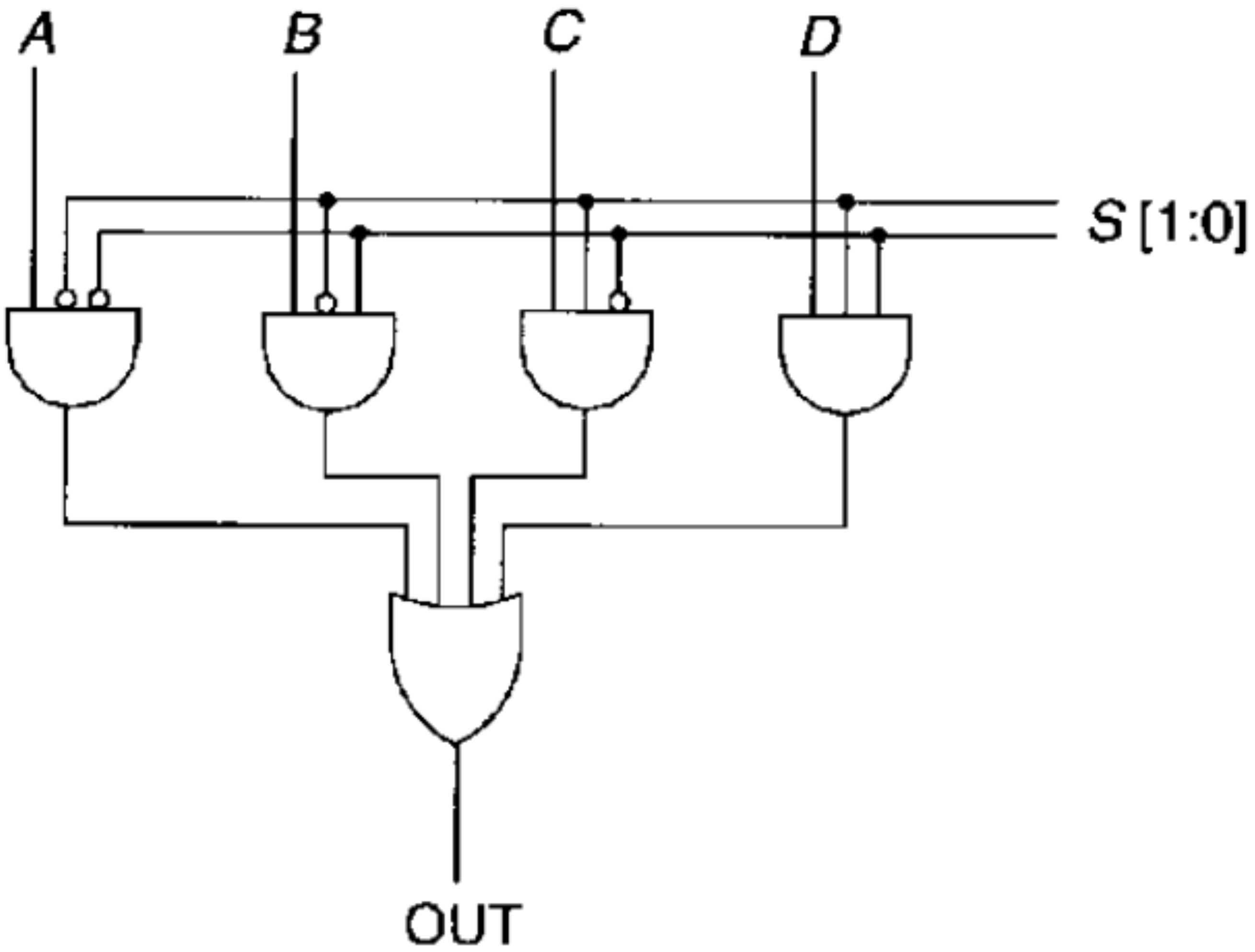
E. 4

Problem

- You have m signals and you want to select the logical value on one of them, determined by a set of n control wires



Multiplexor (MUX)



Question

The basic multiplexor has

A. n outputs, n control lines, 2^n inputs

B. 1 output, 2n control lines, n inputs

C. 1 output, n control lines, 2^n inputs

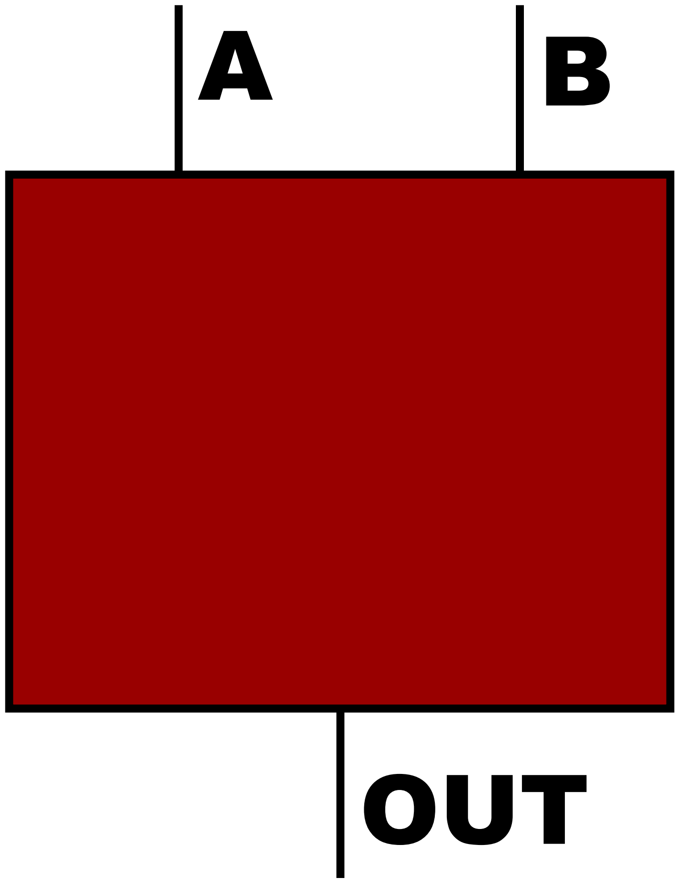
D. 2^n outputs, 2 control lines, n inputs

E. 1 output, 4 control lines, 2 inputs

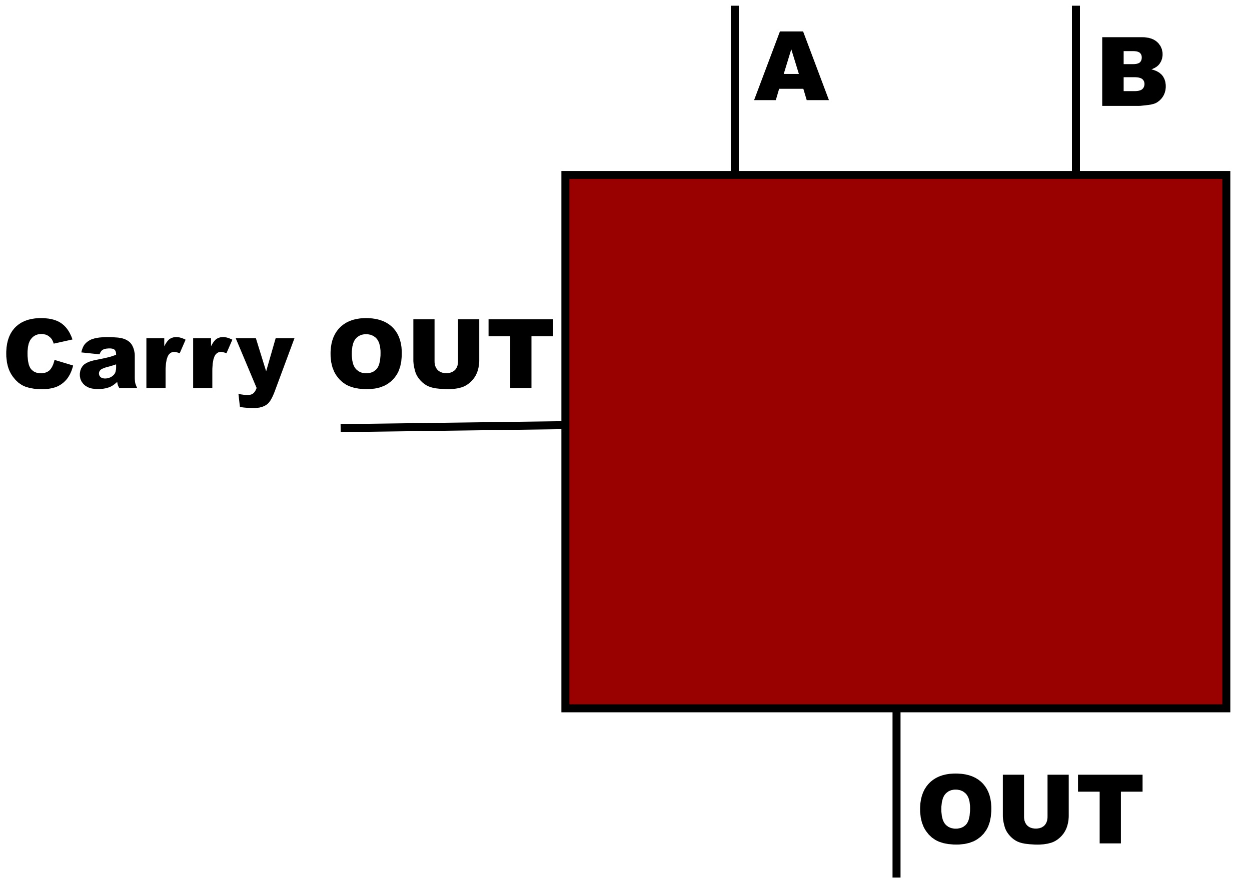
Problem

- No one will buy your new computer design unless it can do at least some math, say, like adding!

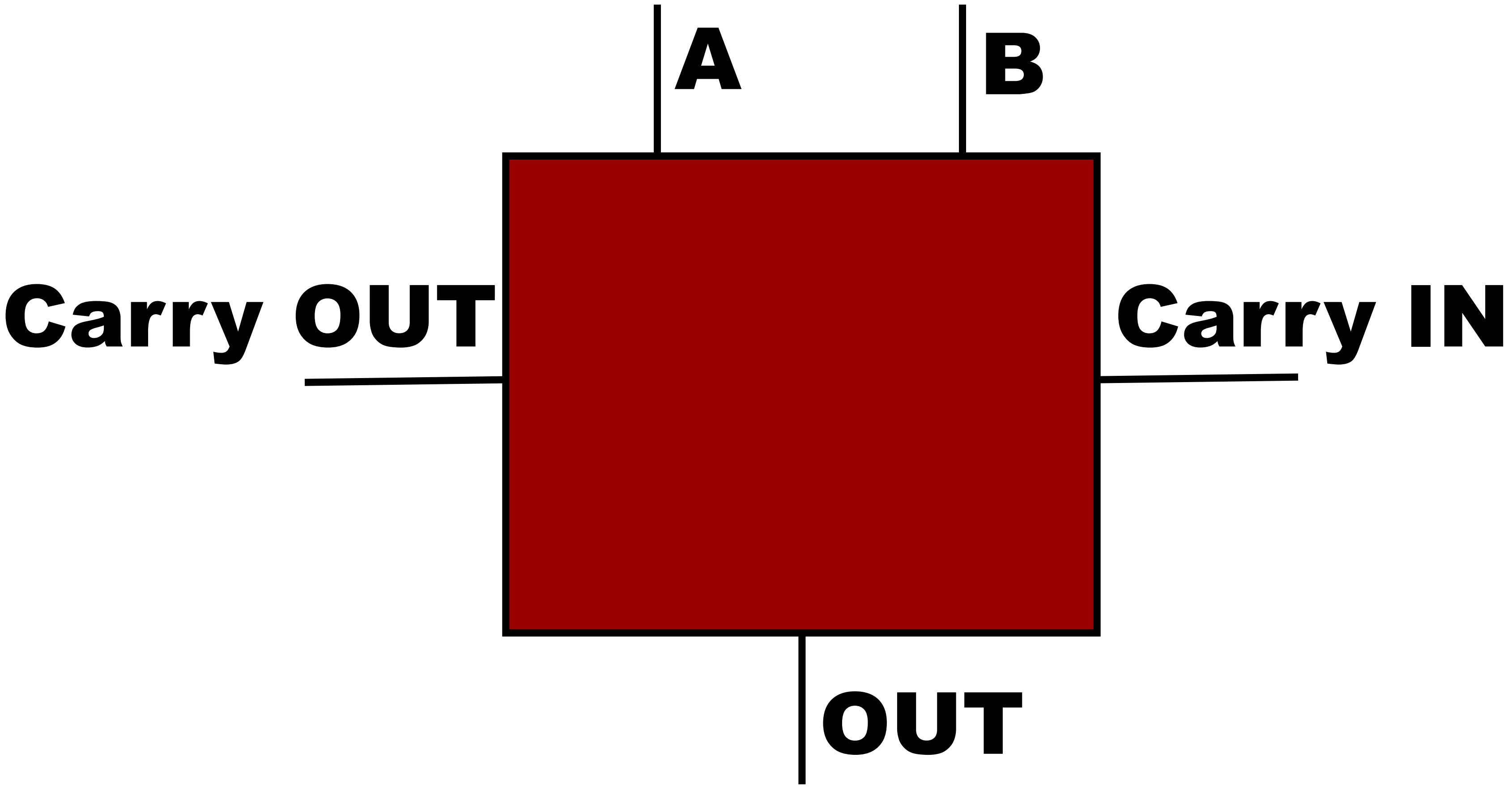
Simple Adder



Half Adder



Full Adder



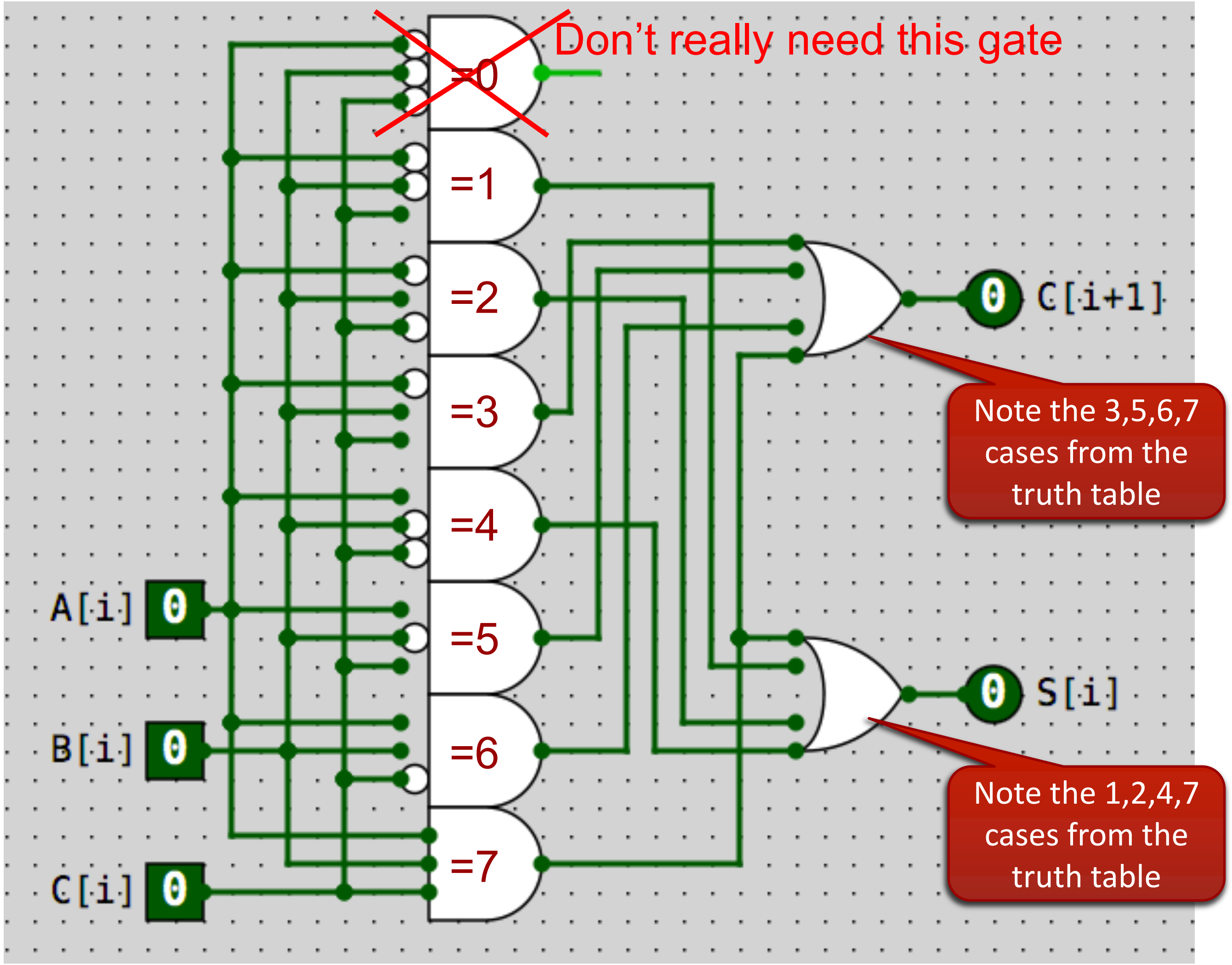
Truth in Adding

A	B	Carry In	Out	Carry 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		

Truth in Adding

A	B	Carry In	Out	Carry 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

Full Adder



Don't really need this gate
=0
=1
=2
=3
=4
=5
=6
=7
A[i]
B[i]
C[i]
0
C[i+1]
S[i]
Note the 3,5,6,7 cases from the truth table
Note the 1,2,4,7 cases from the truth table

The No Thinking Method!

Go buy a full-adder IC chip from Digi-Key.

Boolean Simplification

Boolean Simplification

Basic Boolean algebraic properties

Additive	Multiplicative
$A + B = B + A$	$AB = BA$
$A + (B + C) = (A + B) + C$	$A(BC) = (AB)C$

$$A(B + C) = AB + AC$$

$$\overline{\overline{A}} = A$$

Basic Boolean algebraic identities

Additive	Multiplicative
$A + 0 = A$	$0A = 0$
$A + 1 = 1$	$1A = A$
$A + A = A$	$AA = A$
$A + \overline{A} = 1$	$A\overline{A} = 0$

Useful Boolean rules for simplification

$$A + AB = A$$

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

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

DeMorgan's Laws

$$\overline{AB}$$

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

$$\overline{A + B}$$

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

Simplify the following Boolean expression:

$E = ABCD + BC + A'BC$

(E = A&B&C&D | B&C | ~A&B&C)

A. BC

B. A

C. A'C

D. AD

E. D

E = ABCD + BC + A'BC
[use identity: a + ab = a]

ABCD + BC -> BC

BC + A'BC -> BC

E = BC

AB + A’ → B + A’

A	B	AB	A’	AB+A’	B+A’
0	0	0	1	1	1
0	1	0	1	1	1
1	0	0	0	0	0
1	1	1	0	1	1

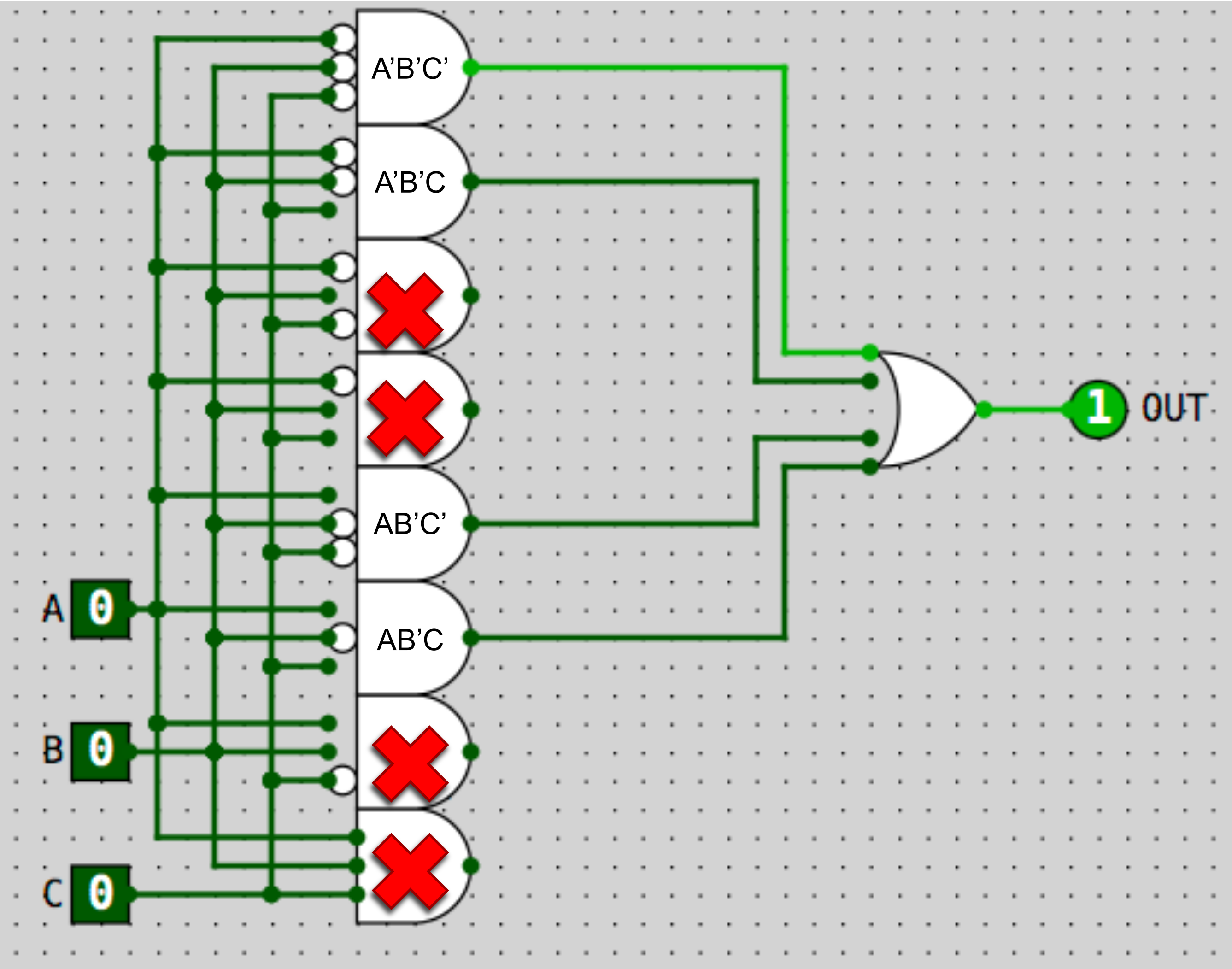
Consider This Circuit

A	B	C	OUT
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	0
1	1	1	0

Truth Table to Circuit

- An AND gate for every line in the truth table (i.e. build a decoder) connected to the inputs
- An OR gate connecting the AND gates for the lines that have 1 in their output column

Implement the Truth Table



F = A'B'C' + A'B'C + AB'C' + AB'C

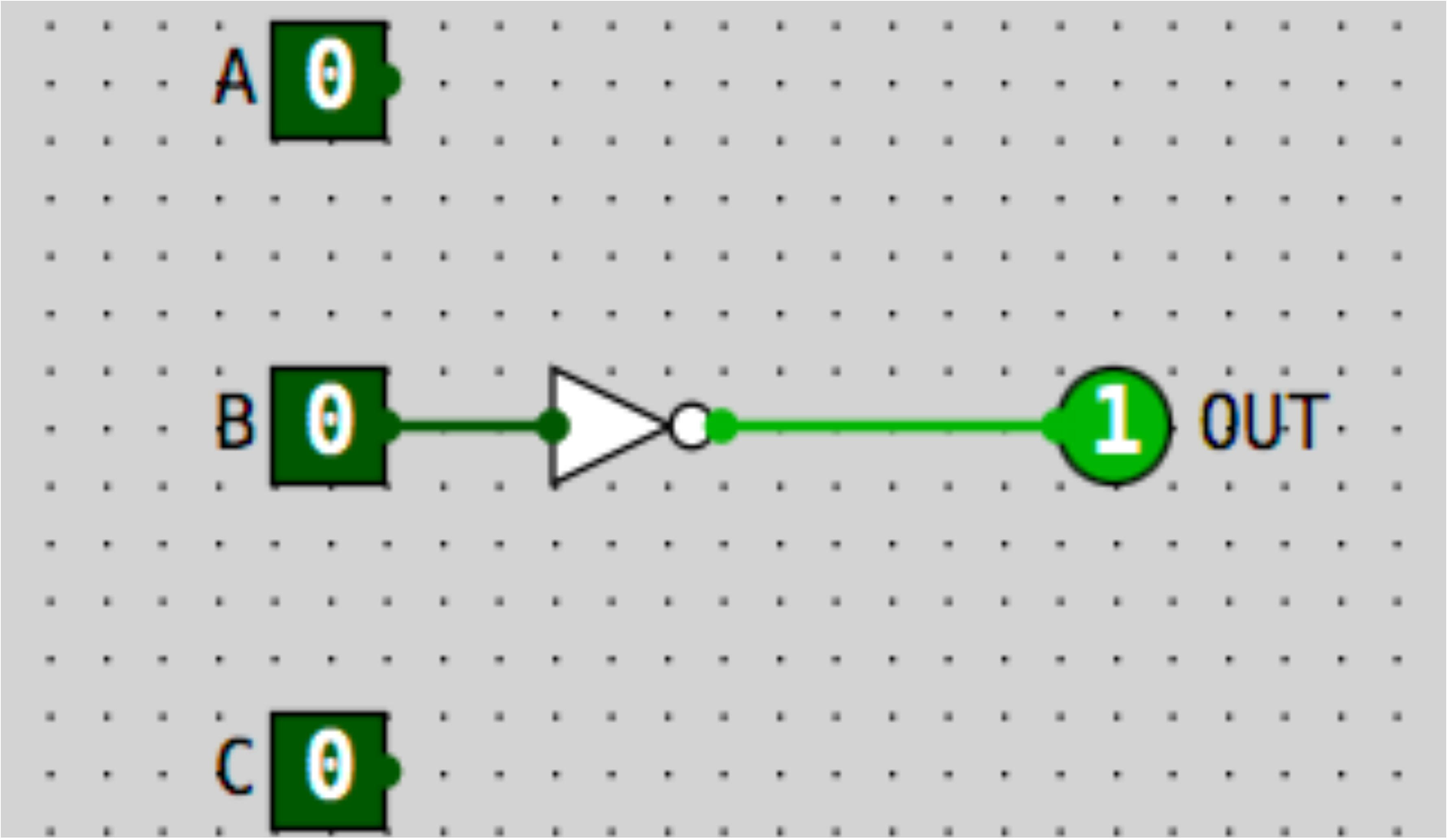
Circuit to Boolean Expression / Truth Table

- Circuit to Boolean Expression
- If it's a sum-of-products circuit then
  - Each AND gate generates a term with the input conditions that yield a 1 from the AND gate (times)
  - All of the terms are ORed together (plus)
- Circuit to Truth Table
  - Each AND gate represents one row of your truth table (inputs)
  - Each OR gate represents one output column of your truth table
    - All of the inputs that produce a 1 output are tied to that OR gate

Classic Simplification

- $F = A'B'C' + A'B'C + AB'C' + AB'C$
- $F = A'B'(C'+C) + AB'(C'+C)$
- $F = A'B' + AB'$
- $F = B'(A'+A)$
- $F = B'$

Quite a Simplification, No?



Something to Note

- We are used to writing sum-of-products with algebraic polynomials, e.g.

  $X^3 + 4X^2 + 2X + 12$

- Turns out that truth tables (and other forms) naturally yield boolean expressions in sum-of-products form, e.g.

  ABC + ABC’ + A’BC + A’B’C’

- This turns out to be very convenient for implementing circuits

Equivalence!

- Boolean Expression
- Truth Table
- Combinational Circuit
- Karnaugh Map

- You can change between ANY of these forms without losing information!

Karnaugh Maps

- Resources:
- http://electronics-course.com/karnaugh-map
- http://www.32x8.com/circuits7

Question

- Can you convert
  - A Karnaugh map to a combinational circuit?
  - A truth table to a Karnaugh map
  - A truth table to a combinational circuit?
  - A Boolean expression into a Karnaugh map?
  - A Boolean expression into a truth table?
  - A truth table into a Boolean expression?

Truth Table

A	OUT
0	x
1	y

1 Variable Karnaugh Map

A'	A
What is the output if A is false?	What is the output if A is true?

What does it mean if both boxes contain 1?

1 Variable Karnaugh Map

A’	A
x	y

What does it mean if both boxes contain 1?

Map: OUT = A’

A’	A

Truth Table: OUT = A’

A	OUT
0	1
1	0

Map: OUT = A’

A’	A
1	0

A	B	OUT
0	0	X_0
0	1	X_1
1	0	X_2
1	1	X_3

2-Variable K-Map

	B’	B
A’	What is the output if both A and B are false?	What is the output if A is false and B is true?
A	What is the output if A is true and B is false?	What is the output if both A and B are true?

2-Variable K-Map

	B'	B
A'	X_0	X_1
A	X_2	X_3

Map: OUT = A'B' + AB

	B'	B
A'		
A		

Map: OUT = A'B' + AB

	B'	B
A'	1	0
A	0	1

Can this expression be simplified?

Map: OUT = A'B' + AB'

	B'	B
A'		
A		

Map: OUT = A'B' + AB'

	B'	B
A'	1	0
A	1	0

Can this expression be simplified?

Question

Which of these conversions between forms of a circuit are allowed because the forms are equivalent?

A. A Karnaugh map to a combinational circuit
B. A truth table to a Karnaugh map
C. A truth table to a combinational circuit
D. A Boolean expression into a Karnaugh map
E. A Boolean expression into a truth table
F. All of the above

Segue

- Suppose you had to give a series of 3 bit codes to disarm the detonation sequence of a nuclear device you were sitting on.
- There would be 8 codes and they had to go in a certain sequence picked by you.
- Any deviation would result in an immediate detonation of the nuclear device
- What codes in what order do you use?

Classic Sequence

000

001

010

011

100

101

110

111

Two switches change at once!
Can you get the timing perfect?
Boom!

Gray Code Sequence

000

001

011

Only one switch per transition!
All the way to the end!
No boom!

010

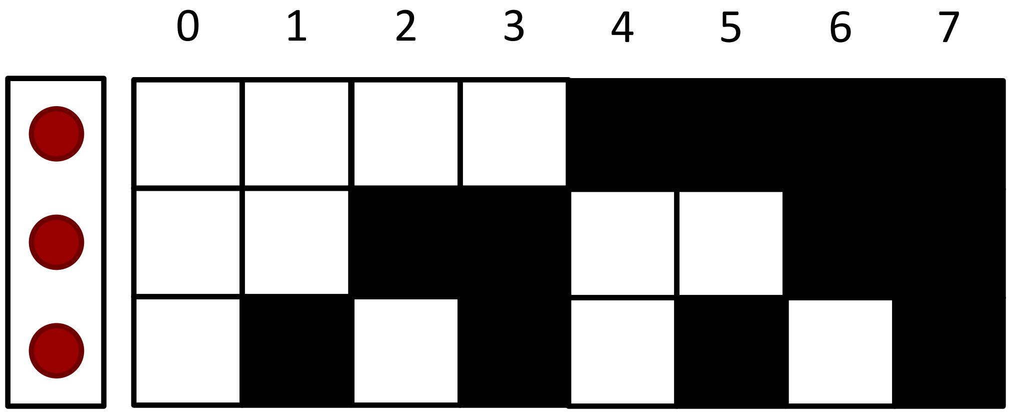
110

111

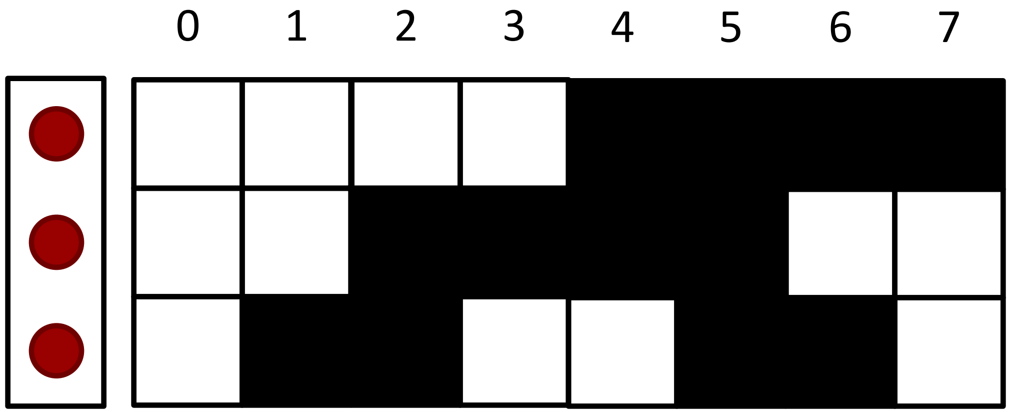
101

100

Optical Encoder

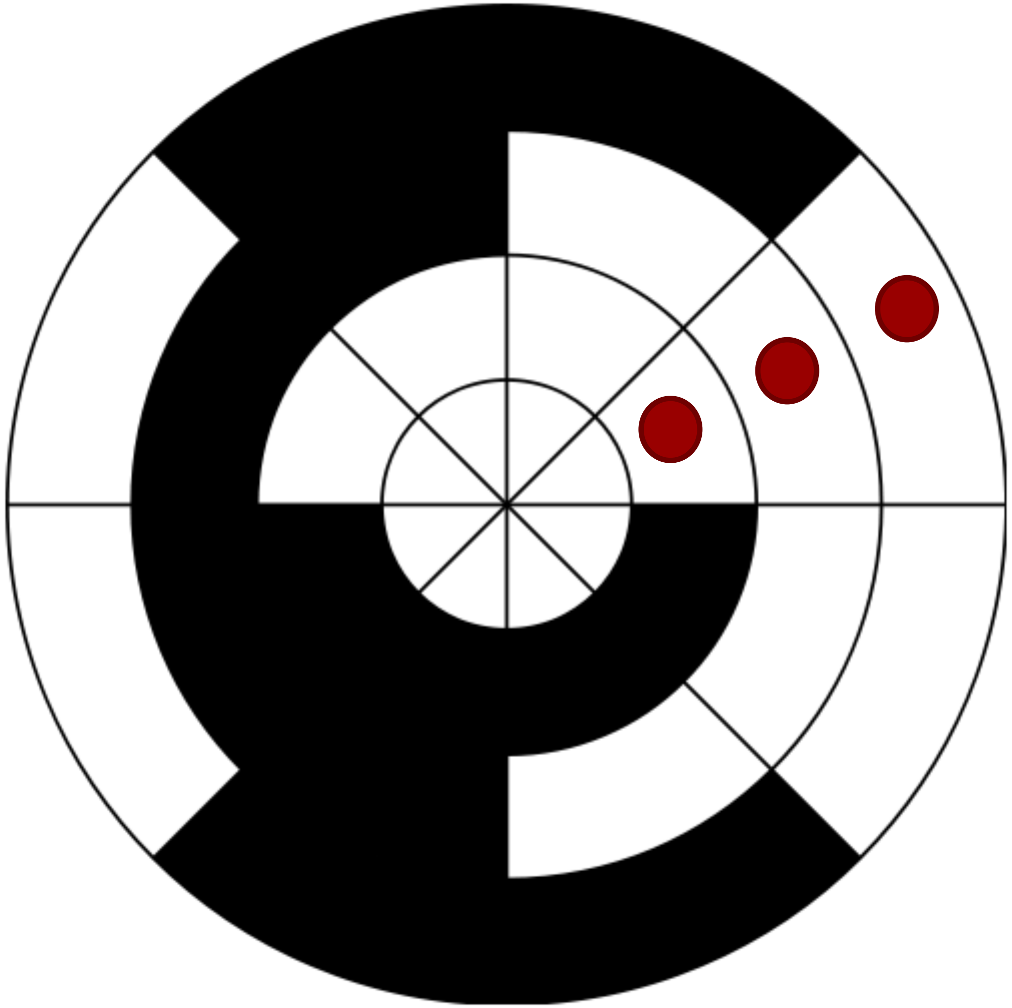


Gray Code Optical Encoder



3-bit Gray Code

000
001
011
010
110
111
101
100



2-bit Gray Code

00
01
11
10

This is the one you need to remember.

4-bit Gray Code

0000
0001
0011
0010
0110
0111
0101
0100
1100
1101
1111
1110
1010
1011
1001
1000

Let's Revisit: OUT = A'B' + AB'

	B'	B
A'	1	0
A	1	0

Each box differs from the adjacent boxes by exactly one bit!

Let’s Revisit: OUT = A’B’ + AB’

	B’	B
A’	1	0
A	1	0

- A’B’ to A’B differs by 1 bit
- A’B to AB differs by 1 bit
- AB to AB’ differs by 1 bit
- AB’ to A’B’ differs by 1 bit

- This means that two adjacent 1 bits tell us that a variable can be removed from two terms

- Since there are one bits in the B’ column, that tells us that both A and A’ are present and the two terms can be consolidated to just the B’ value:

OUT = B’

3-Variable Truth Table

A	B	C	OUT
0	0	0	X_0
0	0	1	X_1
0	1	0	X_2
0	1	1	X_3
1	0	0	X_4
1	0	1	X_5
1	1	0	X_6
1	1	1	X_7

Map: A’B’C + A’BC + ABC

	B’ C’ (00)	B’ C (01)	BC (11)	BC’ (10)
A’ (0)	X_0	X_1	X_3	X_2
A (1)	X_4	X_5	X_7	X_6

Map: A'B'C + A'BC + ABC

	B' C'	B' C	BC	BC'
A'	0	1	1	0
A	0	0	1	0

Can this expression be simplified?

Map: A'B'C + A'BC + ABC

	B' C'	B' C	BC	BC'
A'	0	1	1	0
A	0	0	1	0

Can this expression be simplified?

How Does the Map Remove Variables?

- $A'B'C + A'BC + ABC$
- Two adjacent 1s in the map means there is an $x + x'$ in the formula; the map tells us where
- $A'C(B' + B) + ABC$
- $A'C + ABC$

Map: A'C + ABC

	B' C'	B' C	BC	BC'
A'	0	1	1	0
A	0	0	1	0

Can this expression be simplified: A'C + BC

How Does the Map Remove Variables?

- $A'B'C + A'BC + ABC$
- Two adjacent 1s in the map means there is an $x + x'$ in the formula; the map tells us where
- $A'C(B' + B) + ABC$
- $A'C + ABC$
- $C(A' + AB)$
- $C(A'+B)$
- $A'C + BC$

Map: A’B’C’ + ABC’ + AB’C’ + A’BC’

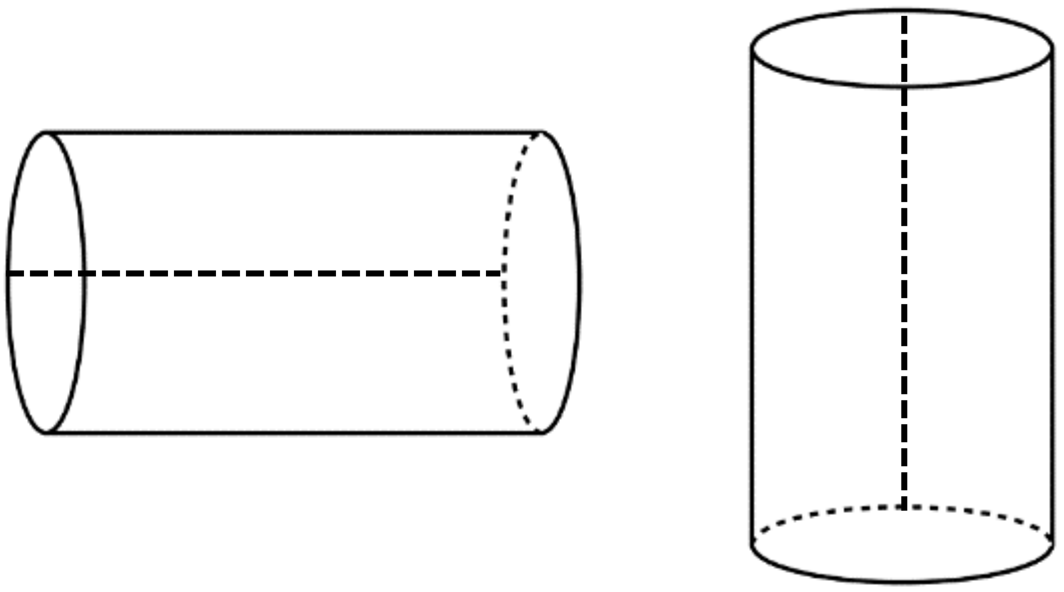
	B’ C’	B’ C	BC	BC’
A’	X_0	X_1	X_3	X_2
A	X_4	X_5	X_7	X_6

Map: A’B’C’ + ABC’ + AB’C’ + A’BC’

	B’ C’	B’ C	BC	BC’
A’	1	0	0	1
A	1	0	0	1

Can this expression be simplified?

Wrap!



Map: A’B’C’ + ABC’ + AB’C’ + A’BC’

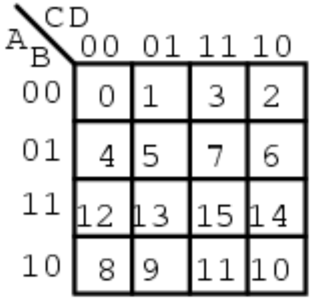
	B’ C’	B’ C	BC	BC’
A’	1	0	0	1
A	1	0	0	1

Expression simplifies to C’

4-Variable Truth Table

A	B	C	D	OUT
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	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	

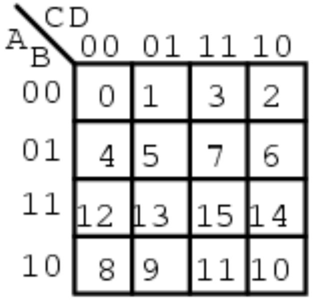
4-Variable K-Map



	C' D'	C' D	CD	CD'
A' B'				
A' B				
AB				
AB'				

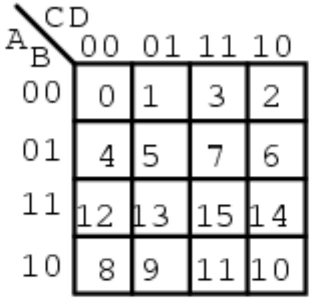
4-Variable Truth Table

A	B	C	D	OUT
0	0	0	0	**1**
0	0	0	1	**0**
0	0	1	0	**1**
0	0	1	1	**0**
0	1	0	0	**0**
0	1	0	1	**0**
0	1	1	0	**0**
0	1	1	1	**0**
1	0	0	0	**1**
1	0	0	1	**0**
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**



Map: A’B’C’D’ + AB’CD’ + AB’C’D’ + A’B’CD’

	C’ D’	C’ D	CD	CD’
A’ B’				
A’ B				
AB				
AB’				



Map: A’B’C’D’ + AB’CD’ + AB’C’D’ + A’B’CD’

	C’ D’	C’ D	CD	CD’
A’ B’	1			1
A’ B				
AB				
AB’	1			1

Can this expression be simplified?

Map: A’B’C’D’ + AB’CD’ + AB’C’D’ + A’B’CD’

AB \ CD	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

	C’ D’	C’ D	CD	CD’
A’ B’	1			1
A’ B				
AB				
AB’	1			1

Simplifies to B’D’

Each Pair Eliminates 1 Variable

AB \ CD	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

	C' D'	C' D	CD	CD'
A' B'				
A' B	1			1
AB		1		
AB'		1		

Simplifies to A'BD' + AC'D

Eliminates 2 Variables

AB \ CD	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

	C’ D’	C’ D	CD	CD’
A’ B’				
A’ B				
AB		1	1	
AB’		1	1	

Simplifies to AD

Eliminates 3 Variables

AB \ CD	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

	C’ D’	C’ D	CD	CD’
A’ B’	1			1
A’ B	1			1
AB	1			1
AB’	1			1

Simplifies to D’

AB \ CD	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

Eliminates 3 Variables

	C' D'	C' D	CD	CD'
A' B'				
A' B	1	1	1	1
AB	1	1	1	1
AB'				

Simplifies to B

Don’t care: Eliminates 3 Variables

AB \ CD	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

	C’ D’	C’ D	CD	CD’
A’ B’				
A’ B	1	1	1	1
AB	X	1	1	1
AB’	X			

Five Variable

E'	C' D'	C' D	CD	CD'
A' B'				
A' B				
AB				
AB'				

E	C' D'	C' D	CD	CD'
A' B'				
A' B				
AB				
AB'				

6 Variable

<table>
<tr><th></th><th>F’</th><th>F</th></tr>
<tr><th>E’</th>
<td><table>
<tr><th></th><th>C’ D’</th><th>C’ D</th><th>CD</th><th>CD’</th></tr>
<tr><th>A’ B’</th><td></td><td></td><td></td><td></td></tr>
<tr><th>A’ B</th><td></td><td></td><td></td><td></td></tr>
<tr><th>AB</th><td></td><td></td><td></td><td></td></tr>
<tr><th>AB’</th><td></td><td></td><td></td><td></td></tr>
</table></td>
<td><table>
<tr><th></th><th>C’ D’</th><th>C’ D</th><th>CD</th><th>CD’</th></tr>
<tr><th>A’ B’</th><td></td><td></td><td></td><td></td></tr>
<tr><th>A’ B</th><td></td><td></td><td></td><td></td></tr>
<tr><th>AB</th><td></td><td></td><td></td><td></td></tr>
<tr><th>AB’</th><td></td><td></td><td></td><td></td></tr>
</table></td></tr>
<tr><th>E</th>
<td><table>
<tr><th></th><th>C’ D’</th><th>C’ D</th><th>CD</th><th>CD’</th></tr>
<tr><th>A’ B’</th><td></td><td></td><td></td><td></td></tr>
<tr><th>A’ B</th><td></td><td></td><td></td><td></td></tr>
<tr><th>AB</th><td></td><td></td><td></td><td></td></tr>
<tr><th>AB’</th><td></td><td></td><td></td><td></td></tr>
</table></td>
<td><table>
<tr><th></th><th>C’ D’</th><th>C’ D</th><th>CD</th><th>CD’</th></tr>
<tr><th>A’ B’</th><td></td><td></td><td></td><td></td></tr>
<tr><th>A’ B</th><td></td><td></td><td></td><td></td></tr>
<tr><th>AB</th><td></td><td></td><td></td><td></td></tr>
<tr><th>AB’</th><td></td><td></td><td></td><td></td></tr>
</table></td></tr>
</table>

Given the following K-map, what is the simplified expression it represents?

	C' D'	C' D	CD	CD'
A' B'	1	1		
A' B	1	1	1	1
AB			1	1
AB'	X	1		

A. AD+BC+CD'

B. A'C'+CB+C'B'

C. A'C'+CB+AB'C'

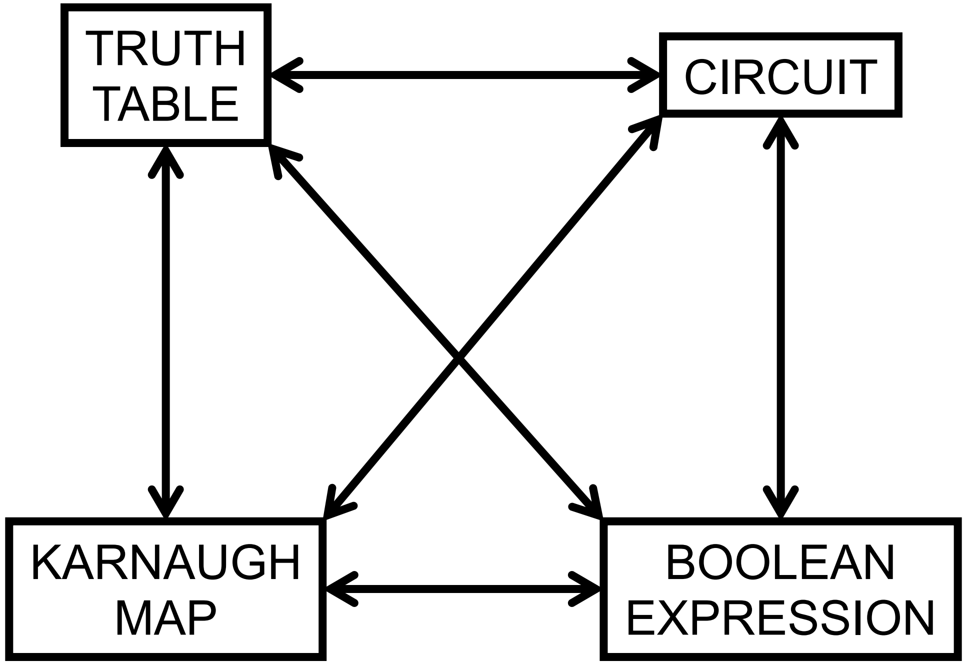
D. A'C'+CB+AB'C'D

E. ABC'+B'C

Remember: Same Data, Different Form

- Combinational Logic Circuit
- Boolean Expression
- Truth Table
- Karnaugh Map

Can You Convert Any of These to Any Other?



Strategy for 4-variable K-Maps

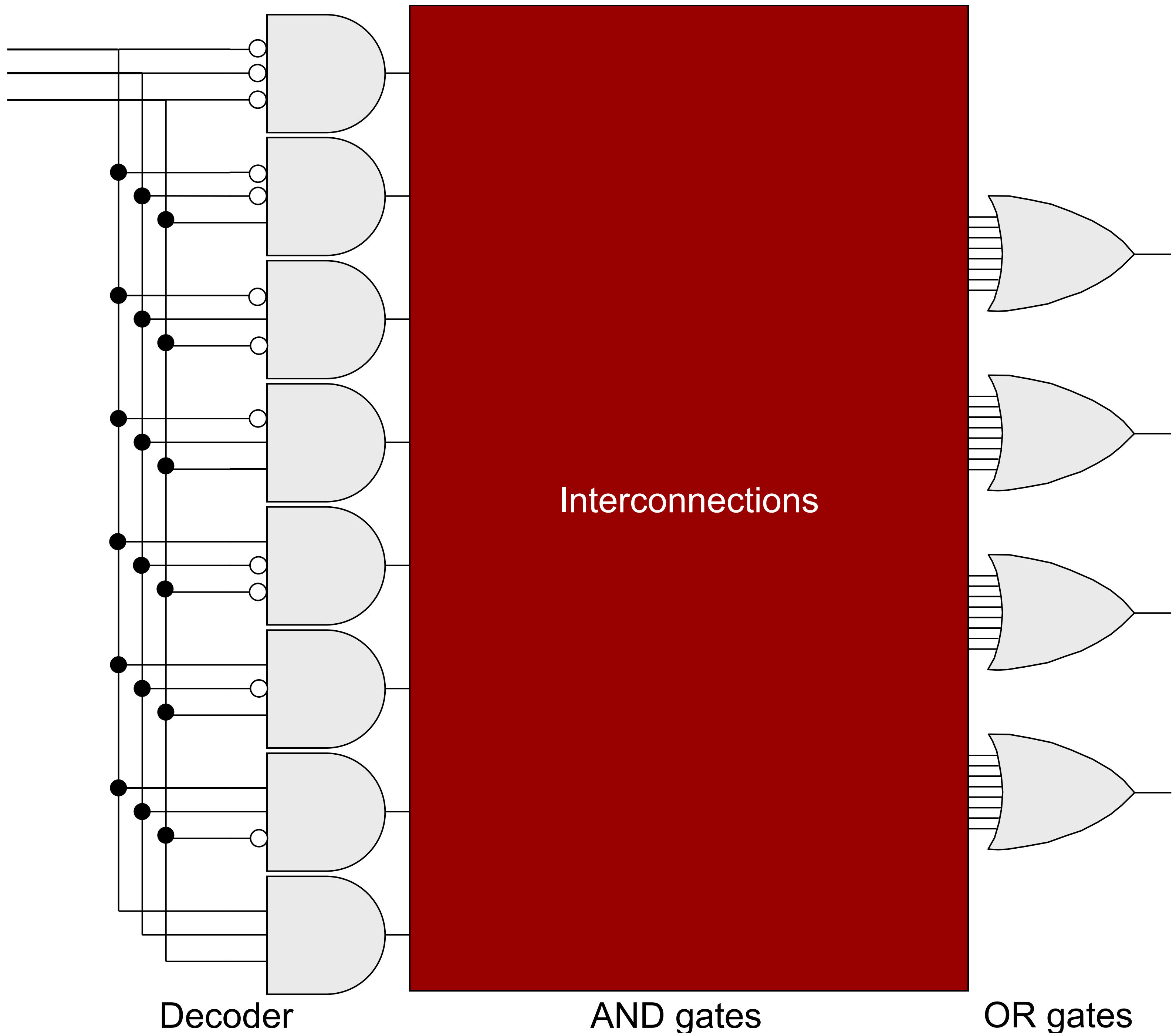
- Find a box around 16 1s.
  - If so, you are done.
- Find boxes around groups of 8 1s.
  - If no 1s are left, you are done.
- Find boxes around groups of 4 1s.
  - If no 1s are left, you are done.
- Find boxes around groups of 2 1s.
  - If no 1s are left, you are done.
- Find boxes around groups of single 1s.
- Done: Now write a term for every box you drew.

Some Automation

- Try http://32x8.com
- Choose “4 variables” and “Kmap with Don’t cares” input option
- Watch it draw the circuits directly from the Kmaps

PLA/PGA/FPGA

- Given a truth table we can implement it using output using a series of NOT gates, AND gates and then OR gates (remember sum-of-products?)
- We need **2^n** AND gates where n is the number of inputs
  - How many rows does the truth table have?
- We need one OR gate for each output
- General purpose ***Programmable Logic Array or (Field) Programmable Gate Array*** devices exist for this purpose



PLA/PGA

Questions?

- Transistors
- Logic Gates
  - NOT, OR, NOR, AND, NAND
  - DeMorgan's Law
  - Larger Gates
- Combinational Logic Circuits
  - Decoder, MUX, Full Adder, PLA,
  - Logical Completeness
- Simplification
  - Boolean
  - Karnaugh Maps
  - PLA/PGA