CPE201
Digital Design

By Benjamin Haas

Class 5: Logic Gates and Boolean

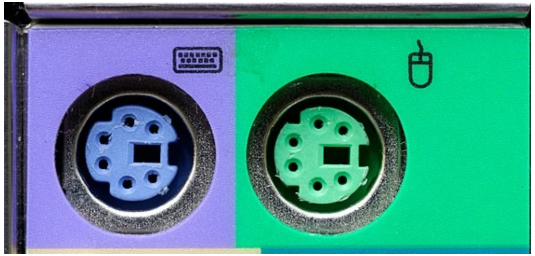
University of Nevada, Reno

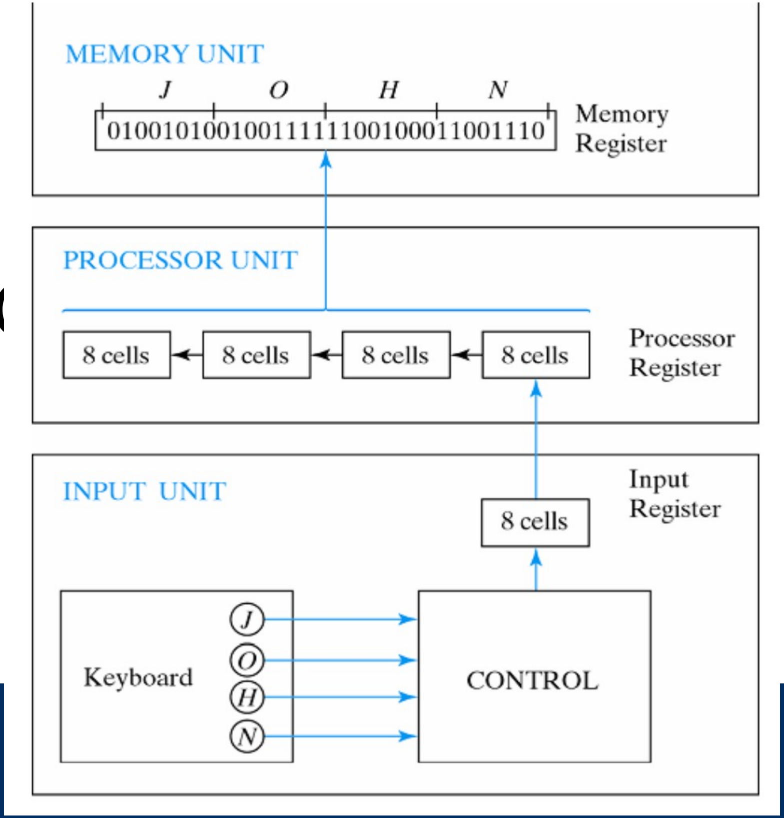
Binary Storage

- Binary cell
  - Stores 1 bit
- Register
  - A group of n cells, stores a value from 0 to $2^n - 1$
  - Encoding scheme matters

Example

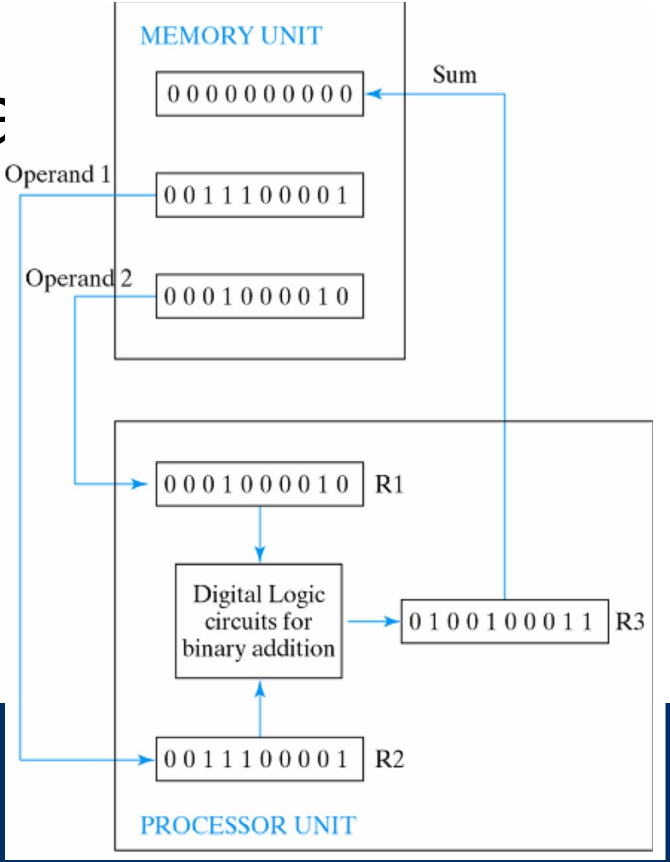
- Keyboard to Memory
  - Not USB here
  - Probably PS/2 using A





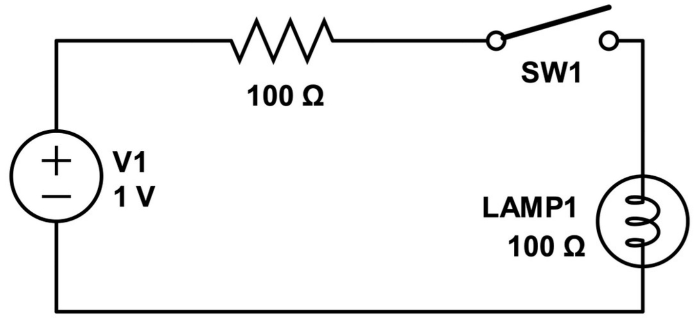
Binary Processing

- There are large blocks in a
  - Specialized functions
- 10 bit adder here
- Need to go deeper



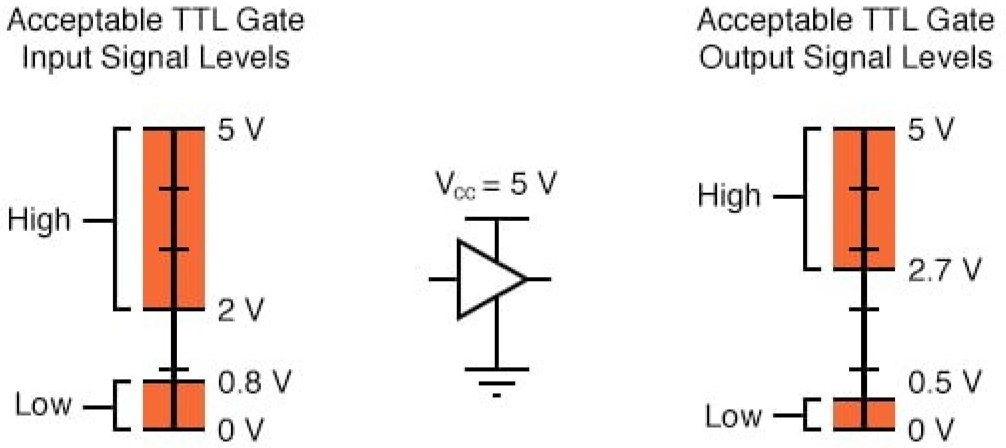
Electrical Review

- Voltage: Difference in electric potential between two points
  - Analogous to water pressure
- Current: Flow of charge
  - Analogous to water flow
- Resistance: Tendency of wire to resist current flow
  - Analogous to water pipe diameter
- V = I x R (Ohm's Law)



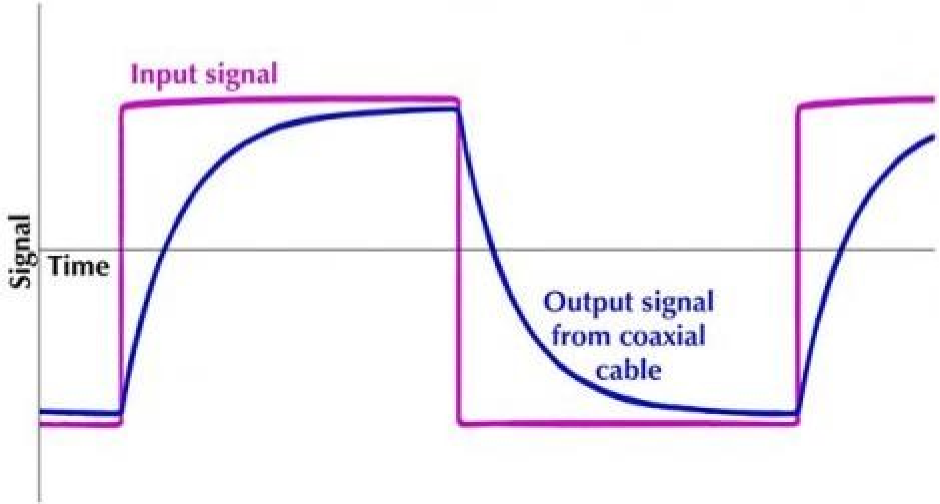
Representing Binary with Voltage

- Logic Levels



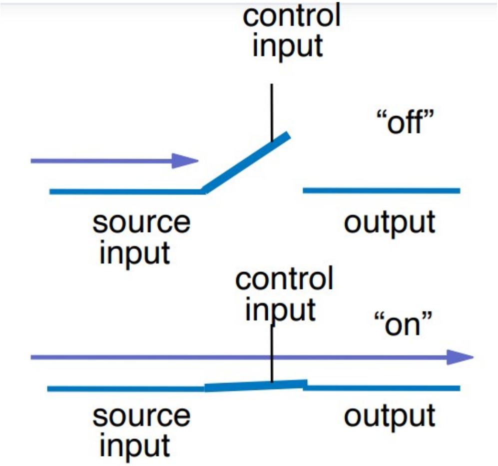
Why the Voltage Ranges?

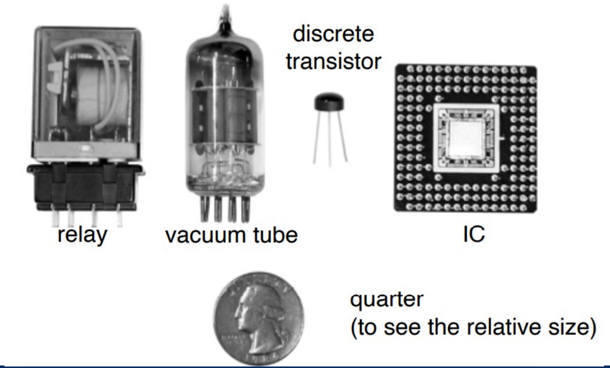
- Resistance
- Capacitance
- Transmission
- Noise
- Floating Ground
- Speed



Switches

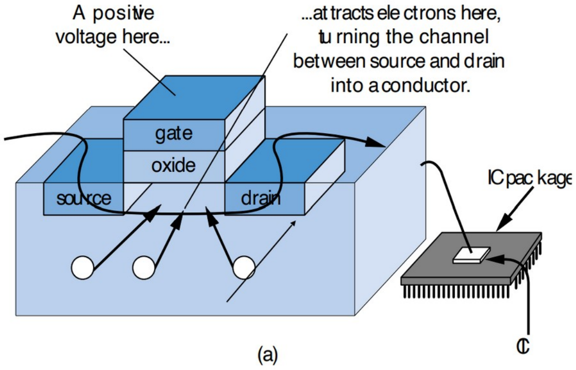
- 3 parts
  - Source input
  - Output
  - Control input
- Manual switches?! No!
- Timeline:
  - 1930s – Relays
  - 1940s – Vacuum Tubes
  - 1950s – Discrete transistors
  - 1960s Integrated Circuits (ICs)
    - From a few, to tens, to hundreds, to billions today



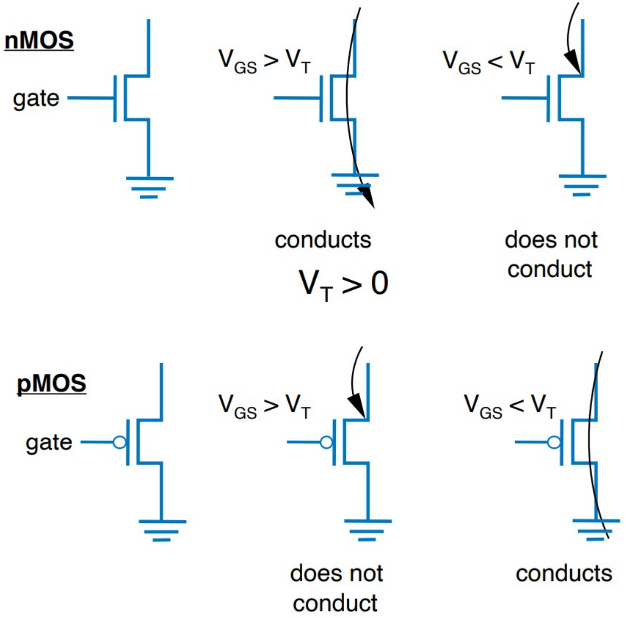


Transistors

- Two Types – BJT and MOSFET
  - Bipolar Junction Transistor
  - Metal-Oxide-Semiconductor Field-Effect Transi

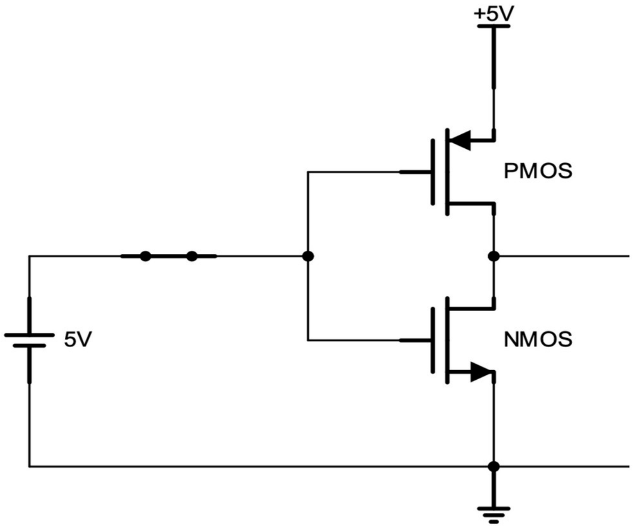


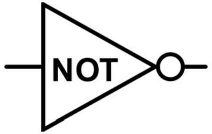
Silicon -- not quite a conductor or insulator: *Semiconductor*



Example

- CMOS Inverter
  - Inverts the input
  - Input 1 makes output 0
  - Input 0 makes output 1





Boolean Logic Gates

Designing systems at the transistor (or tube!) level is possible, but not that human friendly

We would like to think in terms of what we are trying to accomplish and not just voltages and currents

The idea is to go to a *higher level of abstraction*

A *logic gate* is an electronic component that performs binary operations

Gates consist of many transistors, which for the most part we do not need to think about when we design a system

Boolean Algebra

- Variable – symbol to represent action, condition, or data
- Complement – inverse of a variable
- Literal – variable or complement
- Basic Operators – AND, OR, NOT

Truth tables:

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	NOT
0	1
1	0

Boolean in History

- Developed mid-1800's by George Boole to formalize human thought and logic (also in PHIL114)
  - "I'll go to lunch if Mary goes or John goes, and Sally does not go"
    - Let F represent my going to lunch (1 means I go, 0 I don't go)
    - Likewise, m for Mary going, j for John, and s for Sally
    - Then F = (m OR j) AND NOT(s)
- You can formally evaluate the statement
  - m=1, j=0, s=1 then F = (1 OR 0) AND NOT(1) = 1 AND 0 = 0
- You can formally transform the statement (covered later)
  - F = (m AND NOT(s)) OR (j AND NOT(s))
  - Same outputs

Truth tables:

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	NOT
0	1
1	0

Evaluation

- F = (a AND b) OR (c AND d)
  - a=1, b=1, c=1, d=0
    - F = (1 AND 1) OR (1 AND 0) = 1 OR 0 = 1
  - a=0, b=1, c=0, d=1
    - F = (0 AND 1) OR (0 AND 1) = 0 OR 0 = 0
  - a=1, b=1, c=1, d=1
    - F = (1 AND 1) OR (1 AND 1) = 1 OR 1 = 1

Truth tables:

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	NOT
0	1
1	0

English to Boolean

- Pick variables (things that can be represented in 2 different states
- Look at the logic words – both, either, and, etc
- Construct a Boolean statement

Examples

- F is true if
  - a is 1 and b is 1
    - F = a AND b
  - Either a or b is 1
    - F = a OR b
  - a is 1 and b is 0
    - F = a AND NOT(b)

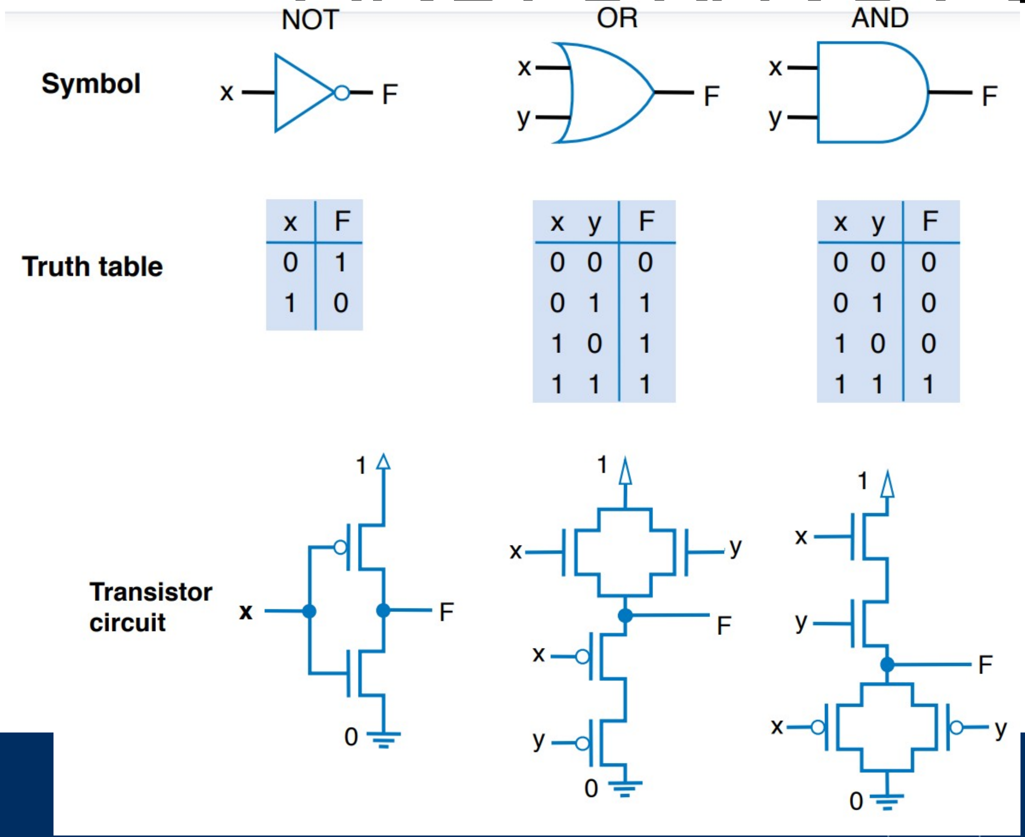
Example

- A fire sprinkler should spray water is high heat is sensed and the system is set to enabled
  - Let h represent "high heat is sensed"
  - Let e represent "enabled"
  - Let F represent "spraying water"
  - Then F = h AND e

Example

- A car alarm should sound if the alarm is enabled, and either the car is shaken or the door is opened
  - Let a represent alarm is enabled, s represent car is shaken, d represent door is opened, and F represent alarm sounds
    - Then F= a AND (s OR d)
  - Alternatively, let d represent that the door is closed (so 1=closed, 0=open)
    - Then F = a AND (s OR NOT(d))

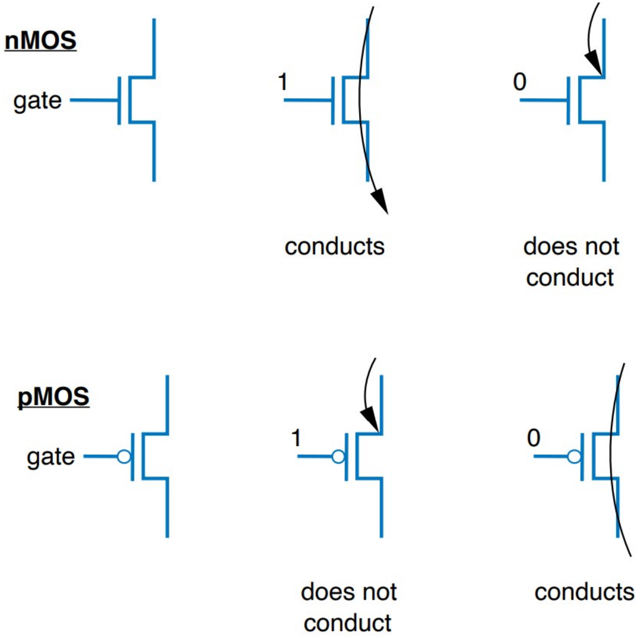
AND/OR/NOT Logic Gates



x	F
0	1
1	0

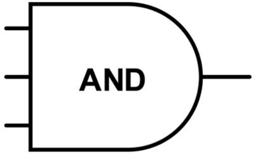
x	y	F
0	0	0
0	1	1
1	0	1
1	1	1

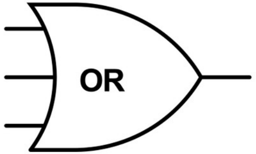
x	y	F
0	0	0
0	1	0
1	0	0
1	1	1



Multiple Inputs

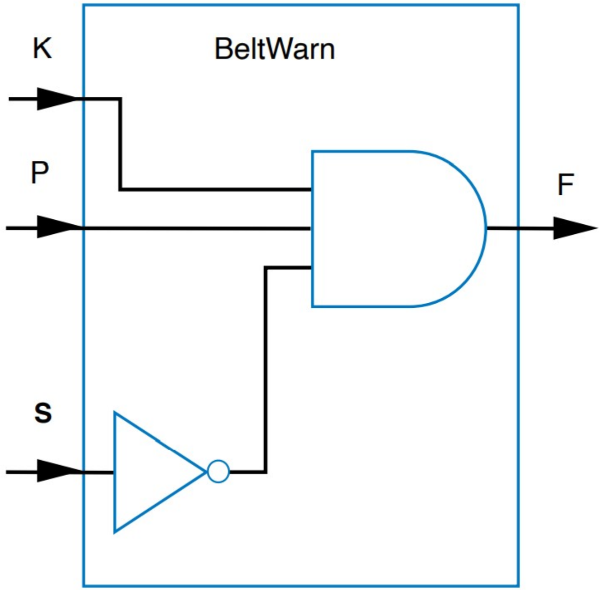
- Multiple input AND
  - Returns 1 when all the inputs are 1
  - Returns 0 when at least one input is 0
- Multiple input OR
  - Returns 1 when at least one input is 1
  - Returns 0 when all the inputs are 0





Example

- Problem: The seatbelt light should turn on if the key is in the ignition, the person is in the seat, and the seat belt has not been fastened
- Variables:
  - S = 1: seatbelt is fastened
  - K = 1: key is in the ignition
  - P = 1: person is in the seat
- Equation
  - F = K AND P AND NOT(S)
- Circuit

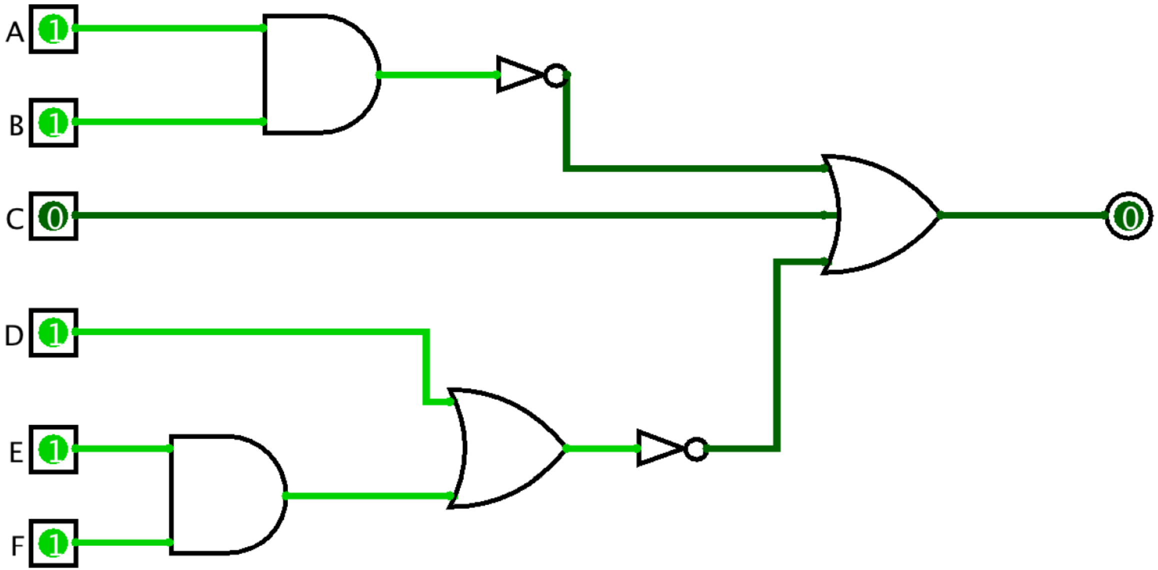


Boolean Notation (Shorthand)

- To not write AND, OR, NOT all the time
  - AND becomes multiplication
  - OR becomes addition
  - NOT becomes apostrophe (X’)or bar ($\overline{X}$)
- F = (A AND B) OR C → F = AB + C
- F = (A AND NOT(B) OR C → F = AB’ + C
- F = NOT(A) AND NOT(B) → F = A’B’

Boolean Equation to Circuit

- F = (AB)′ + C + (D + EF)′



Reading

- This lecture
  - Sections 3.1-3.3
- Next lecture
  - Sections 3.4-3.6, 1.6