Unit 8a

Digital Hardware Design Overview

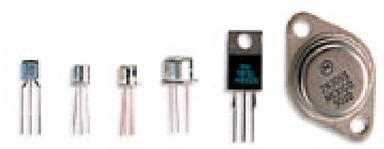
A Brief History

COMPUTERS, SWITCHING TECHNOLOGY, AND TRANSISTORS

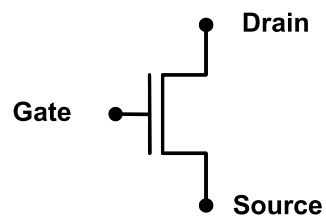
You will not be tested on specifics of transistor operation, but should understand their fundamental role as the building blocks of logic circuits and their abstraction as a switch.

Transistor Overview

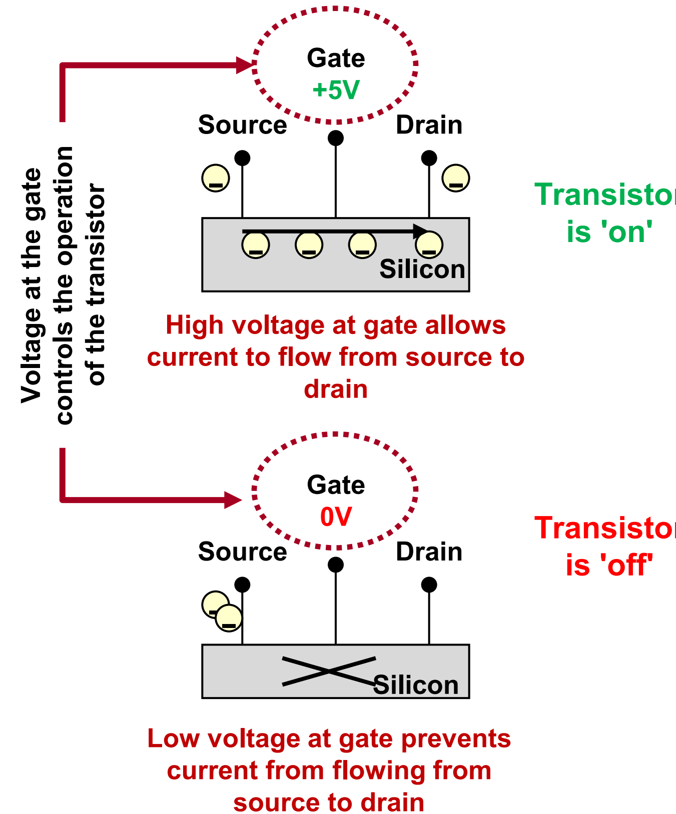
- Electronic computers require some kind of **"switching" (on/off) technology** that allows one signal to turn another signal on or off
  - Initially, that was the vacuum tube but then replaced by the **transistor**
- Invented by Bell Labs in 1948
- Uses semiconductor materials (silicon)
- Much smaller, faster, more reliable (doesn't burn out), and dissipated less power than previous technologies (e.g. vacuum tubes).



Individual Transistors
(About the size of your fingertip)

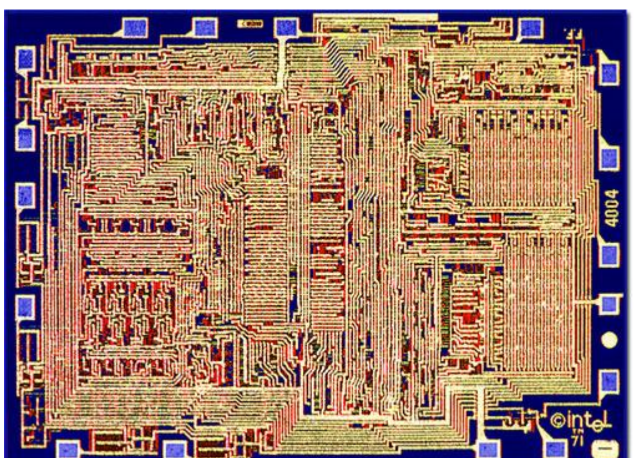


Schematic symbol for MOS transistor

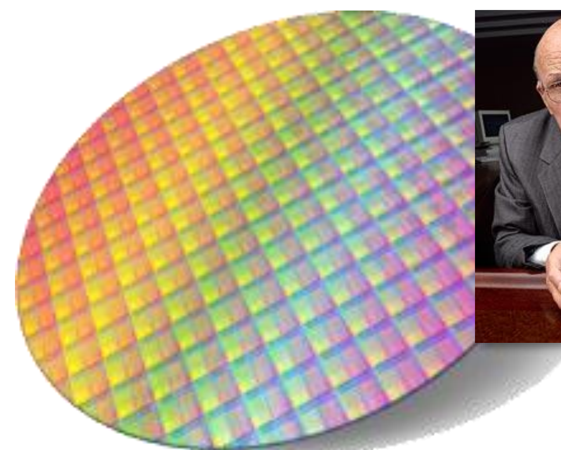


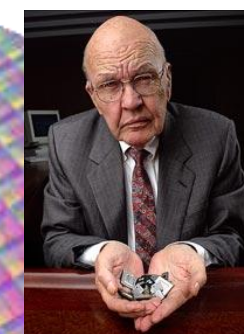
Little Things Can Change the World

- Electronic devices can be built from individual transistors but would be extremely PAINFUL to wire and assemble at any scale
- In 1958, Jack Kilby of Texas Instruments fabricates (and wires) multiple transistors on a single semiconductor substrate and the "integrated circuit" is born (and Kilby gets the Nobel prize.)



https://micro.magnet.fsu.edu/chipshots/intel/images/inteli4004diemedium.jpg





Moore's Law & Transistors

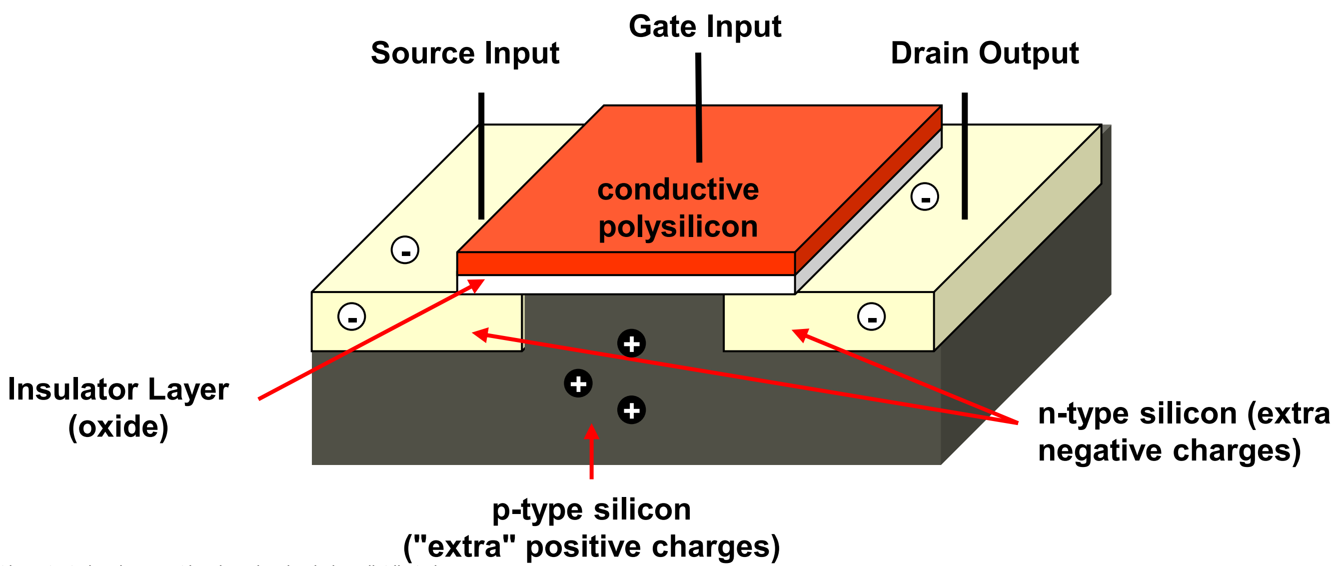
- We continue to shrink the size of the transistor structure
- Moore's Law = Number of transistors able to be fabricated on a chip will double every 1.5-2 years (i.e. exponential growth)
  - We are approaching the physical limitations of this shrinking
  - 53% compound annual growth rate over 50 years
    - No other technology has grown so fast so long
- Transistors are the fundamental building block of computer HW
  - Switching devices: Can conduct [on = 1] or not-conduct [off = 0] based on an input voltage

How Does a Transistor Work

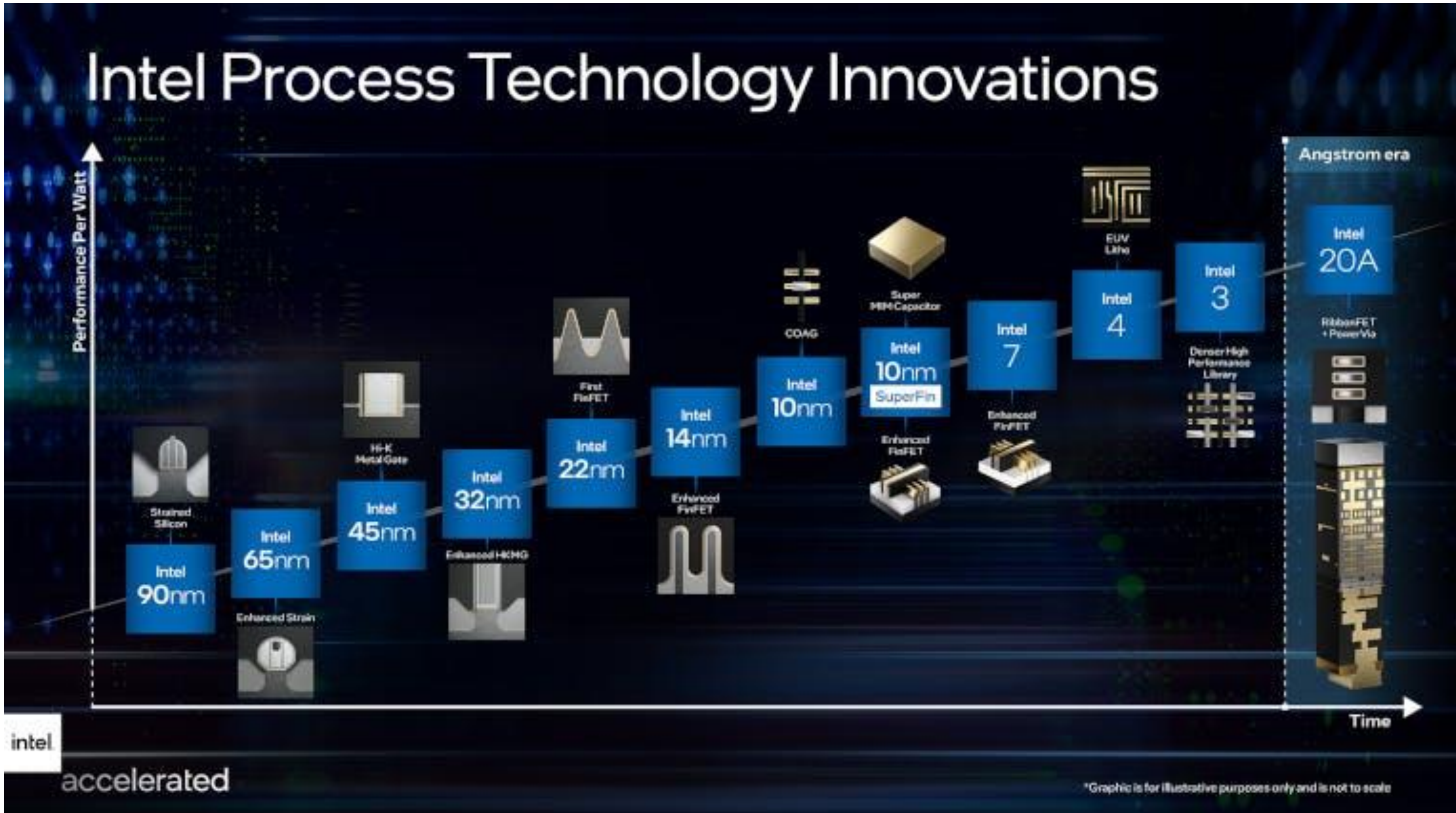
- Transistor inner workings
  - http://www.youtube.com/watch?v=IcrBqCFLHIY

NMOS Transistor Physics

- A thin, insulator layer (silicon dioxide or just "oxide") is placed over the silicon between source and drain
- Conductive polysilicon material is layered over the oxide to form the gate input

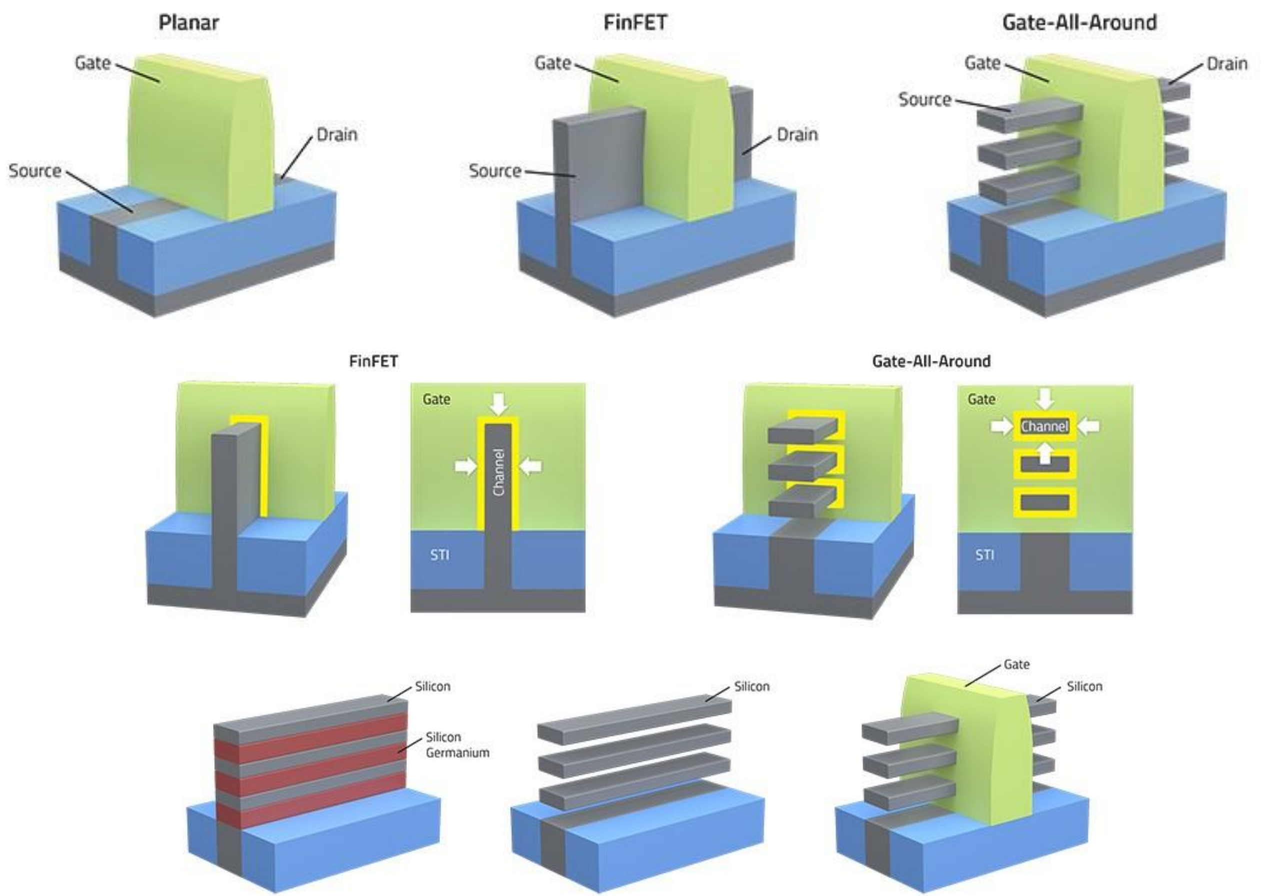


Minimum Feature Size



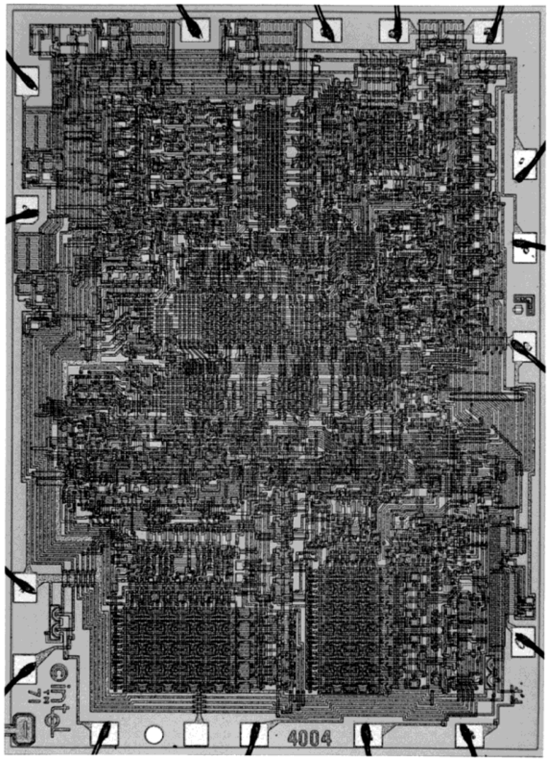
https://img.electronicdesign.com/files/base/ebm/electronicdesign/image/2021/08/Intel_Process_Roadmap_1_edit.610a0f6361e0d.png

New Transistor Technologies

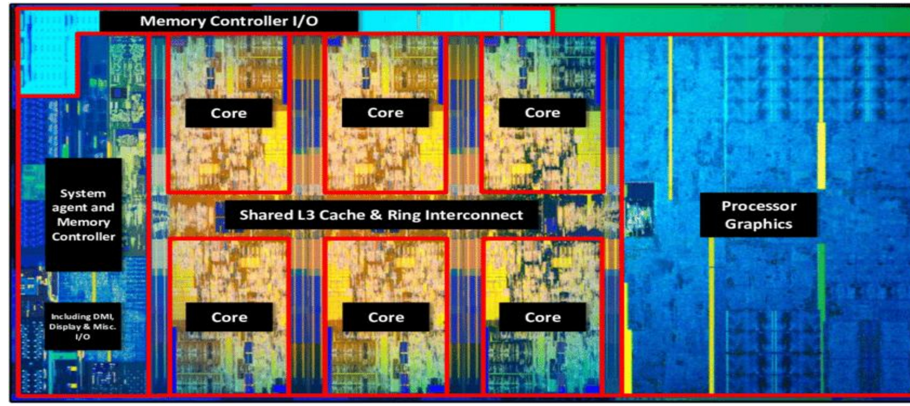


https://blog.lamresearch.com/finfets-give-way-to-gate-all-around/

Processor Trends



1971 – Intel 4004
1000 transistors
10,000 nm wire width
Max 4K-bits addressable memory
1 MHz operation

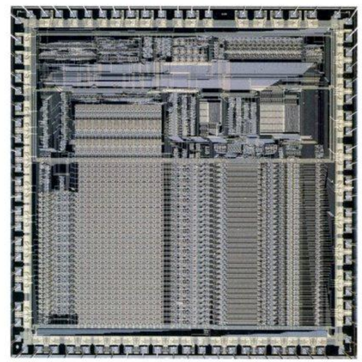


- 8th Gen Coffee-Lake Processor
- #cores/#threads: 6/12
- Technology node: 14nm wire width
- Clock speed: 3.7 GHz
- Transistor count: Over one billion
- Cache: 12MB

Apple's M1 (ARM based) Chip

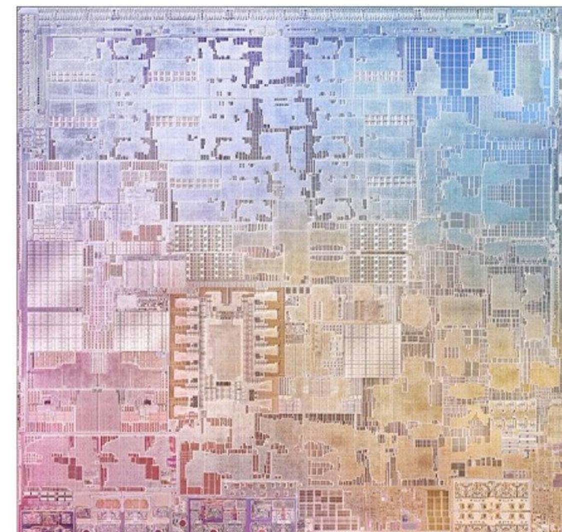
- 8 cores
- Technology node: 5 nm
- Up to 16 billion transistors
- Clock speed: ~3.2 GHz

ARM1 and Apple M1 at the same scale



ARM1 processor
~7mm wide
3 µm process

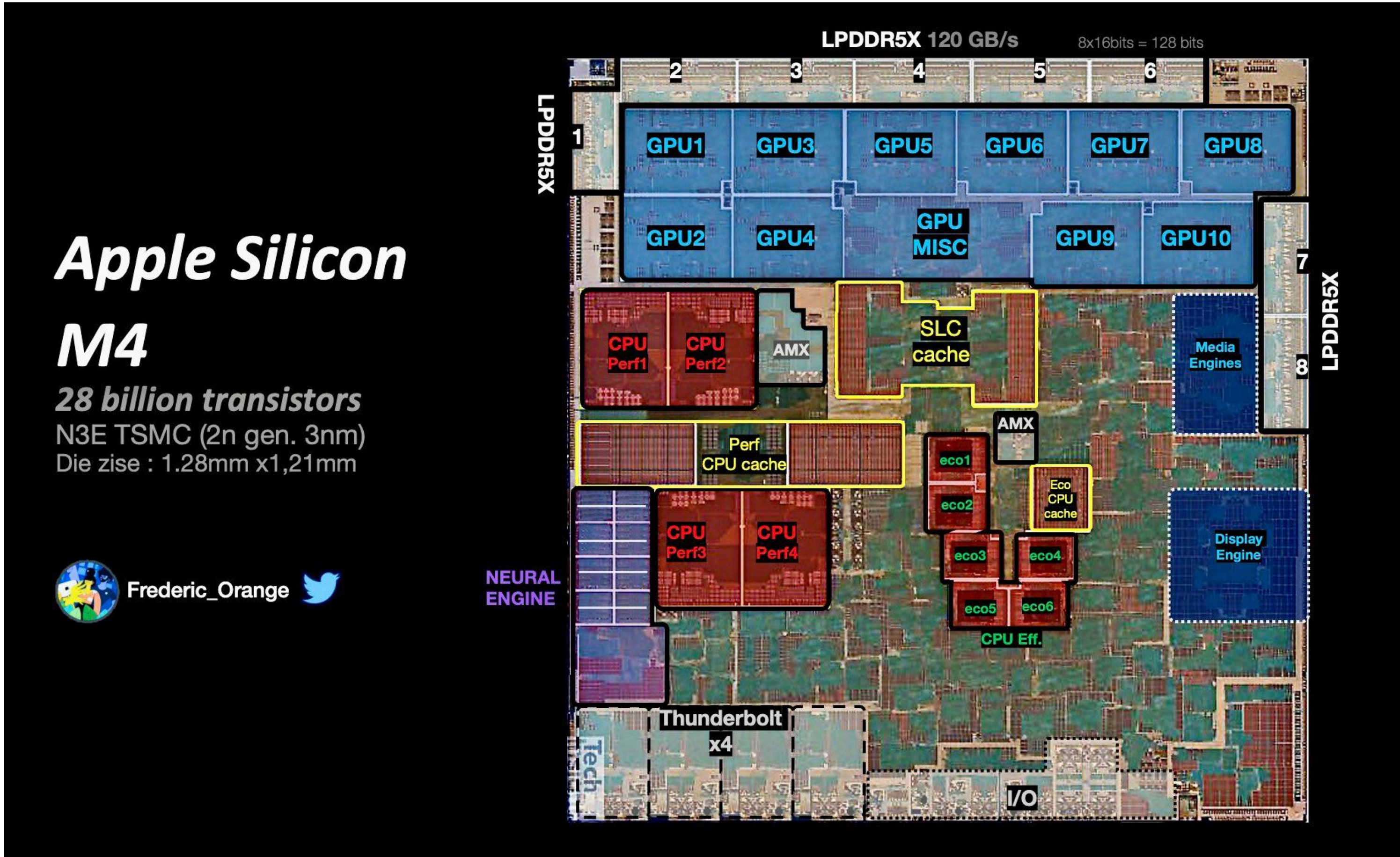
If built with M1 transistors, the entire ARM1 chip would be the tiny dot inside this circle:



Apple M1 processor, ~11mm wide, 5 nm process

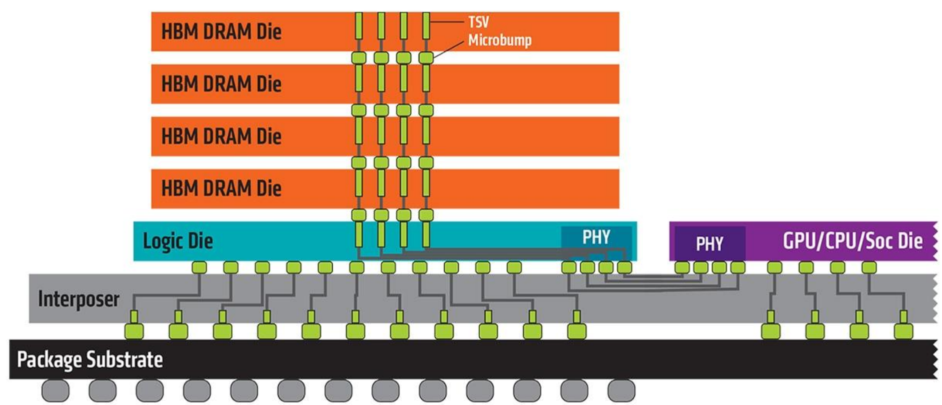
https://pbs.twimg.com/media/EmqIIlsVcAAUlm5.jpg

Apple M4


Apple Silicon
M4
28 billion transistors
N3E TSMC (2n gen. 3nm)
Die zise : 1.28mm x1,21mm
Frederic_Orange
LPDDR5X 120 GB/s
8x16bits = 128 bits
LPDDR5X
1
2
3
4
5
6
7
8
GPU1
GPU3
GPU5
GPU6
GPU7
GPU8
GPU2
GPU4
GPU MISC
GPU9
GPU10
CPU Perf1
CPU Perf2
AMX
SLC cache
Media Engines
Perf CPU cache
AMX
eco1
eco2
Eco CPU cache
CPU Perf3
CPU Perf4
eco3
eco4
eco5
eco6
CPU Eff.
Display Engine
NEURAL ENGINE
Thunderbolt x4
I/O

3D Chips

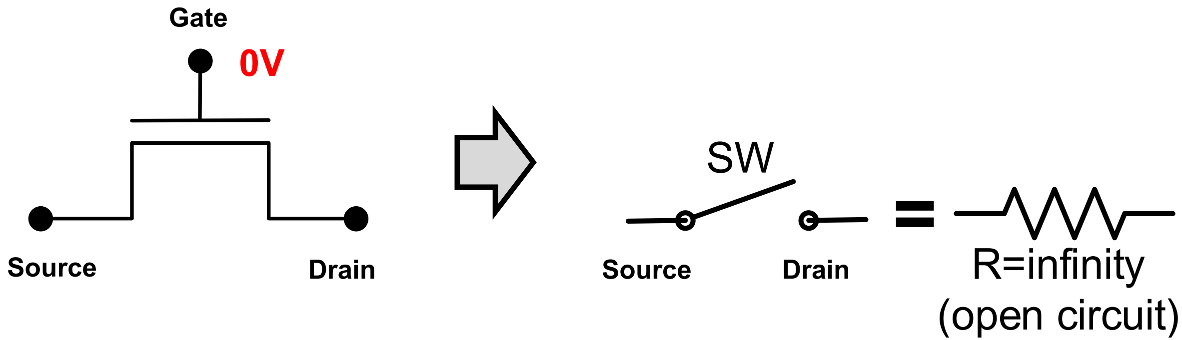
- Stacked chip dies in one package for memory or other processing accelerators rather than separate chips provide greater size, power, and speed advantages



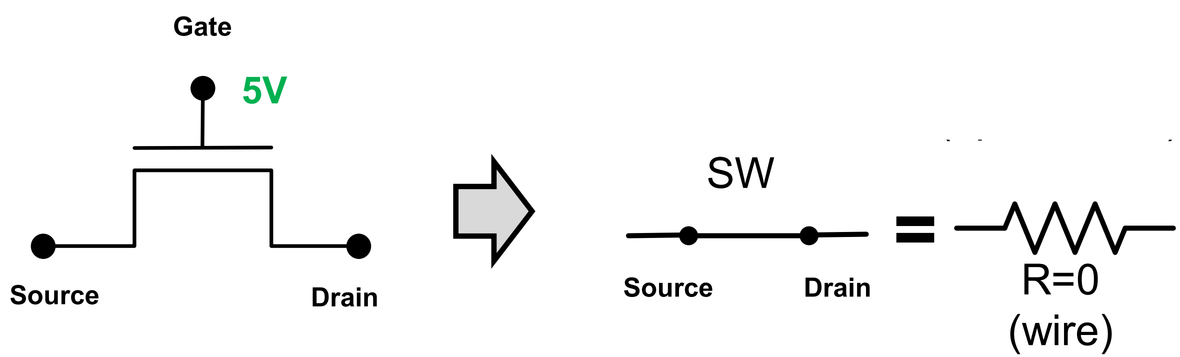
https://www.amd.com/en/technologies/hbm

LOGIC GATES

Transistors as Switches



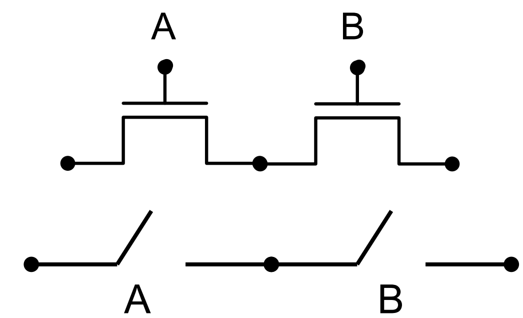
OFF transistor = OPEN SWITCH (infinite resistance)



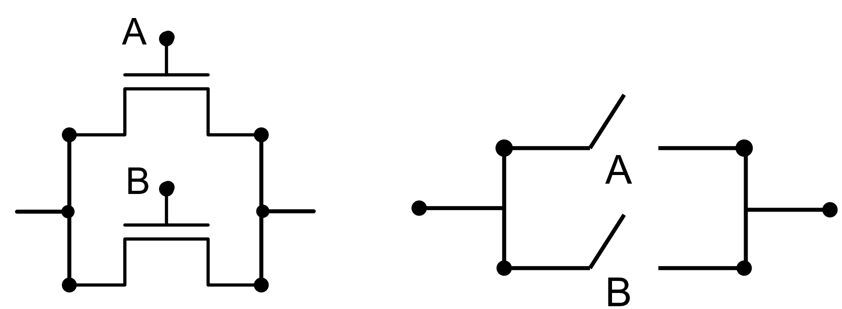
ON transistor = CLOSED SWITCH (wire)

Transistors and Logic

- Transistors act as switches (on or off)
- Logic operations (AND / OR) formed by connecting them in specific patterns
  - Series Connection
  - Parallel Connection



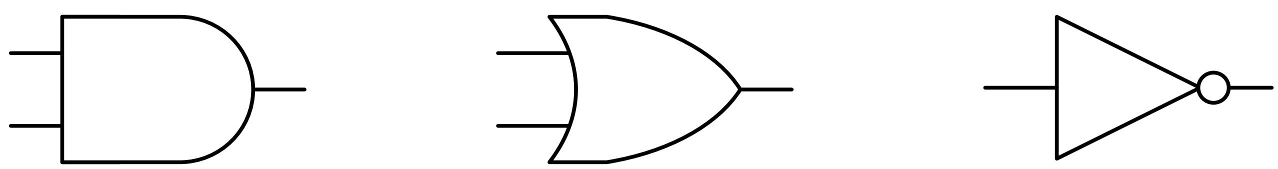
<u>Series Connection</u>
A **AND** B must be on for the input to connect to the output



<u>Parallel Connection</u>
A **OR** B must be on for the input to connect to the output

Gates

- Each logical operation (AND, OR, NOT) can be implemented as a digital device called a "gate"
  - The schematic symbols below are used to represent each logic gate
- Each logic gate can be built by connecting transistors in various configurations

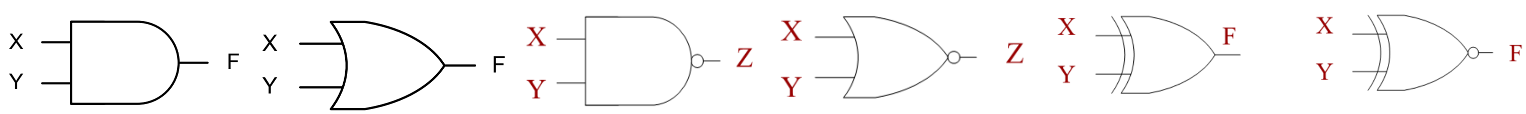


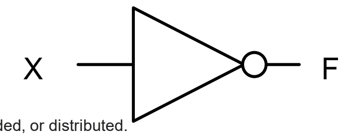
AND Gate | OR Gate | NOT Gate

Gate Summary

- You should know and memorize the truth tables of these basic gates

X	Y	AND	OR	NAND	NOR	XOR	XNOR
0	0	0	0	1	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	1	0	0	0	1

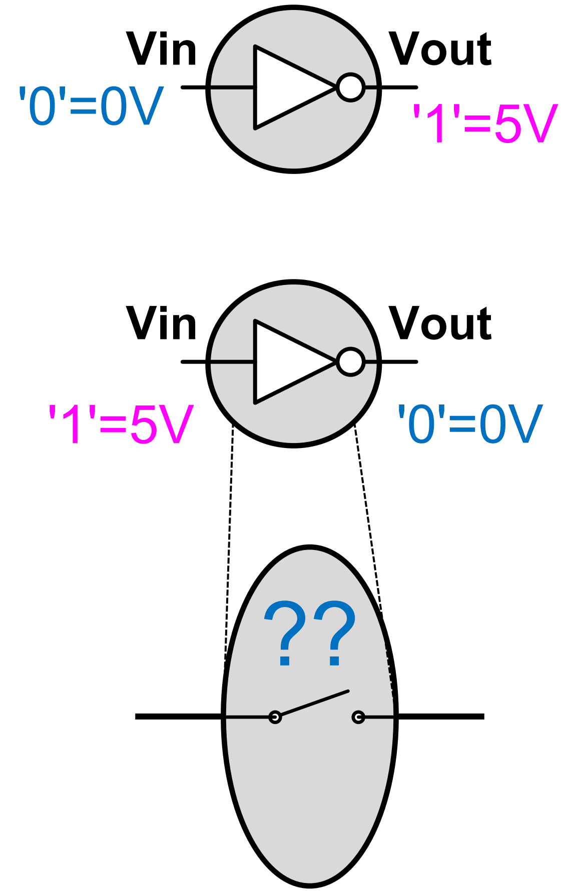




X	F
0	1
1	0

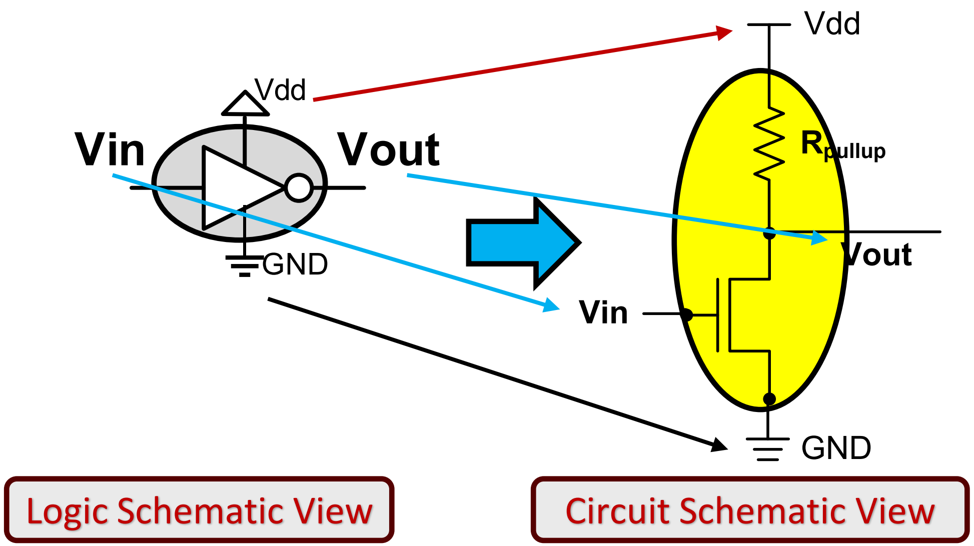
Aside: How Do You Build an Inverter (1)?

- If gates are made from transistors…
- …and transistors act like switches…
- …how can we build an inverter (where the output is the OPPOSITE of the input) from just switches?



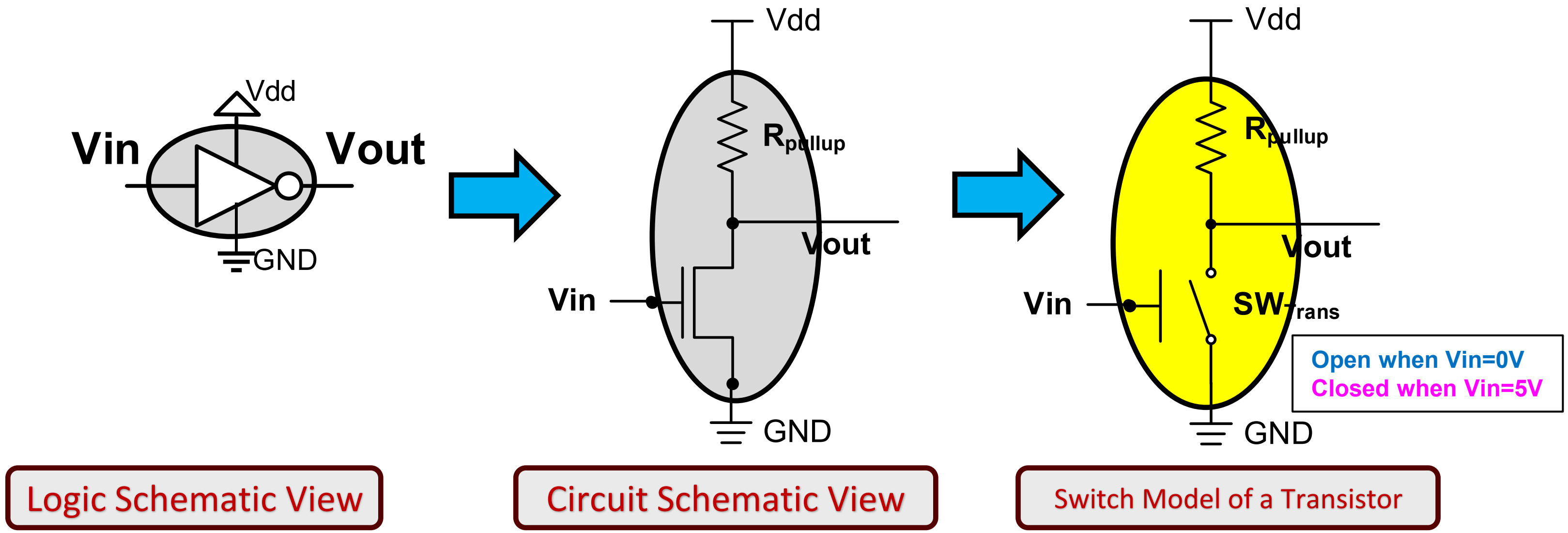
Aside: How Do You Build an Inverter (2)?

- A simple (though maybe not ideal) method to build an inverter is to place a transistor in series with a resistor
  - Input to inverter is input to transistor
  - Output of inverter is node between resistor and transistor



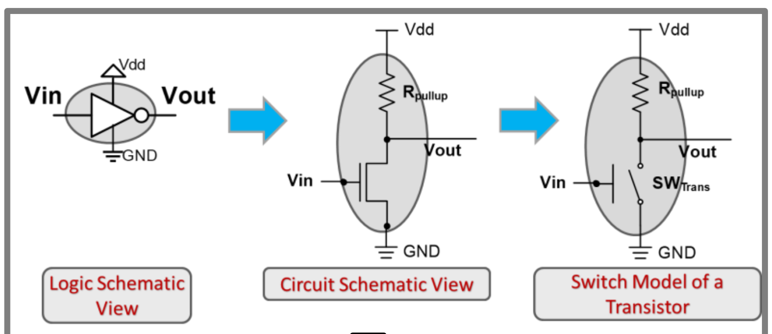
Aside: How Do You Build an Inverter (3)?

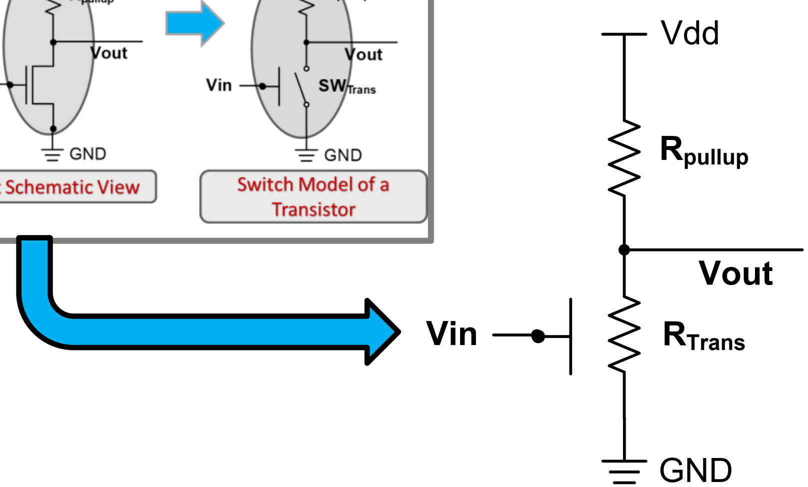
- We can ***model*** the **transistor** as a *switch* which is...
  - Open when Vin='0'=0V (i.e. the transistor is OFF) or
  - Closed when Vin='1'=5V (i.e. the transistor is ON)



Aside: How Do You Build an Inverter (4)?

- But then recall that a ***switch*** can be modeled as a ***resistor*** of:
  - Large (infinite) resistance when the switch is open (Transistor OFF) or
  - Small (0) resistance when the switch is closed (Transistor ON)
- Develop an equation for Vout using the voltage divider eqn.





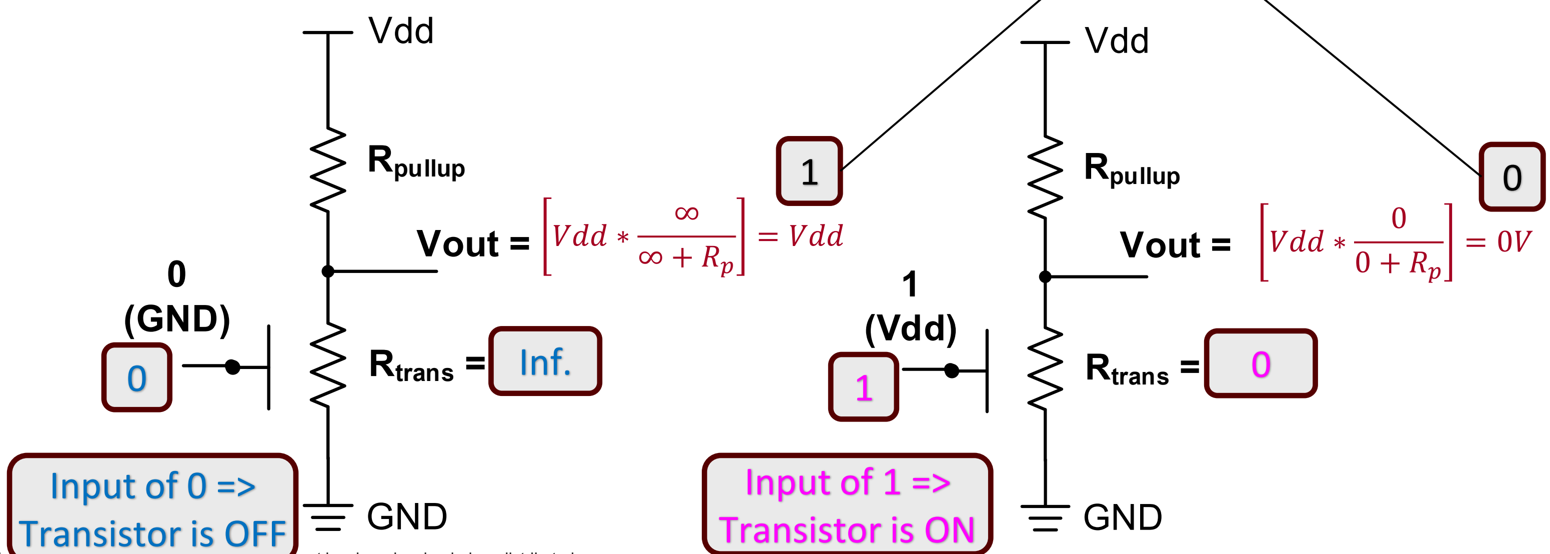
$$Vout = \left[Vdd * \frac{R_{trans}}{R_{trans} + R_{pullup}}\right]$$

Aside: How Do You Build an Inverter (5)?

- Now use your equation to analyze the two scenarios for the inverter input (when the input is 0V = Logic '0' and when the input is 5V = Logic '1')

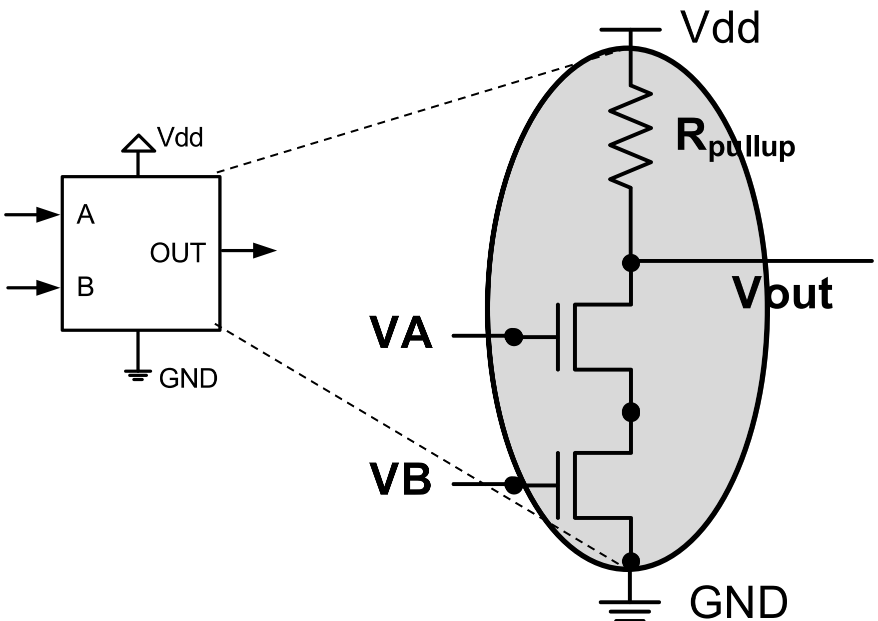
$$Vout = \left[Vdd * \frac{R_{trans}}{R_{trans} + R_{pullup}}\right]$$

In	Out
0	1
1	0



What Is This Mystery Gate?

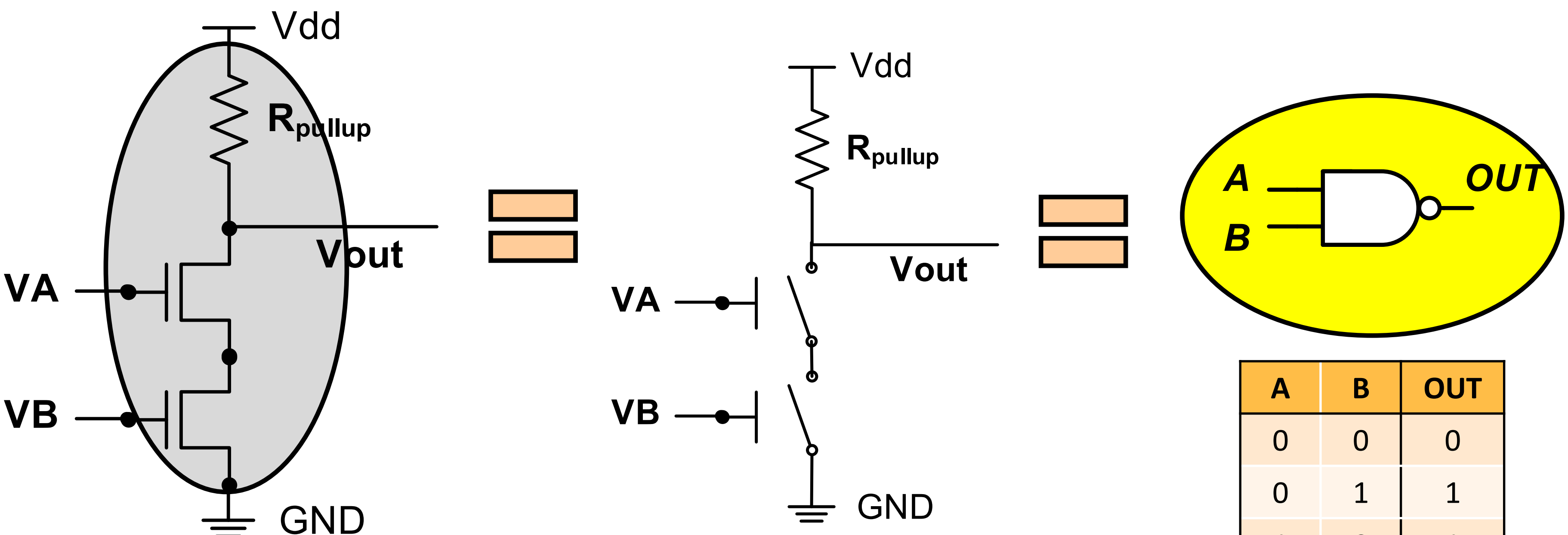
- Try to derive the truth table of this circuit



A	B	OUT
0	0	
0	1	
1	0	
1	1	

What Is This Mystery Gate?

- This circuit implements a NAND logic function.



A	B	OUT
0	0	0
0	1	1
1	0	1
1	1	0

Big Idea

- There are digital circuits (aka logic "gates") that process and manipulate bits
  - Any **operator** in software (+, -, *, /, ==, <, > , <=, >=, !=) can be implemented with some digital circuits
- Other digital circuits (aka "registers" or RAM) can "**store**" bits, just as a variable does in software
- Starting with this unit and for the rest of the semester we will learn about how these **digital circuits (i.e. hardware) can be built to do everything software can**.

Processing and Storing bits

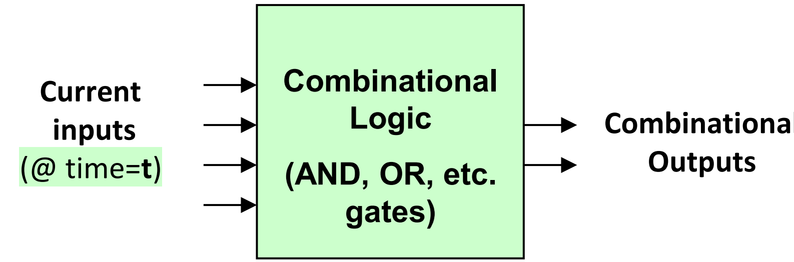
COMBINATIONAL VS. SEQUENTIAL LOGIC

Combinational vs. Sequential Logic

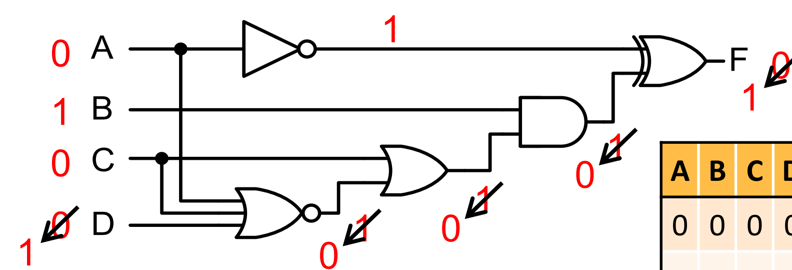
- All logic is categorized into 2 groups
  - Combinational logic:
    - Outputs = f(current inputs)
    - **MEMORYLESS** circuits – Outputs only depend on inputs now
  - Sequential Logic
    - Outputs = f(current + past inputs)
    - **Stateful** (having "**memory" or storage**) - Remembering inputs or events that happened in the past

Combinational Logic

- **Combinational Logic Examples**: All the gates (AND, OR, XOR, NOT, NAND, NOR etc.) that we've already seen.
- Outputs **MAY** CHANGE when the inputs change
- Can always be described with a **truth table**!
  - Though that table might be quite large (n-inputs => 2^n outputs)



Outputs depend only on current inputs (time=t)

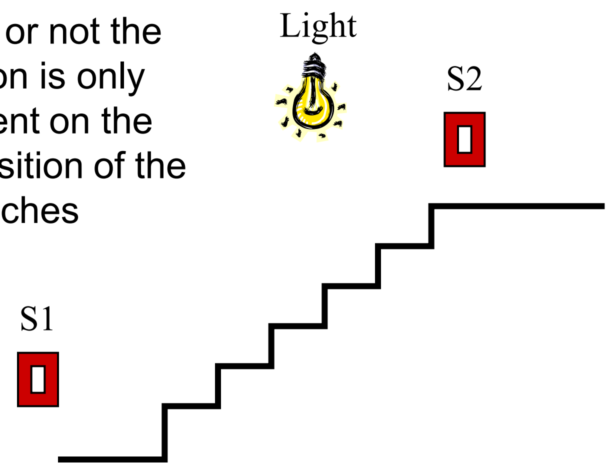


A	B	C	D	F
0	0	0	0	1
	.	.	.	
0	1	0	0	0
	.	.	.	
1	1	1	1	1

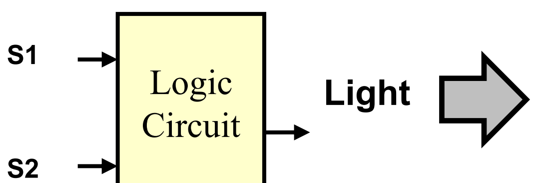
When the inputs change, the outputs MAY change.

Combinational Example: Staircase Light Switch

Whether or not the light is on is only dependent on the current position of the switches



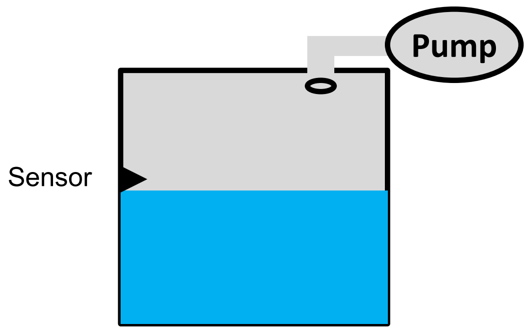
The light, **L**, should be on when…



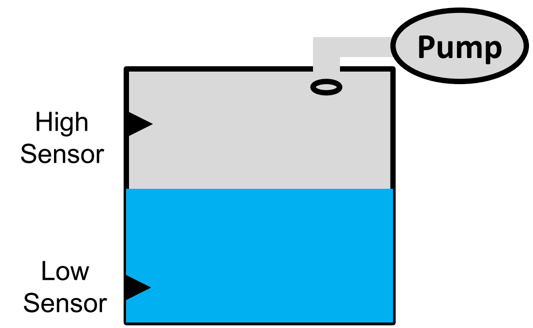
S1	S2	Light
0	0	0
0	1	1
1	0	1
1	1	0

Water Tank Problem

- Build a control system for a pump to keep the tank from going empty



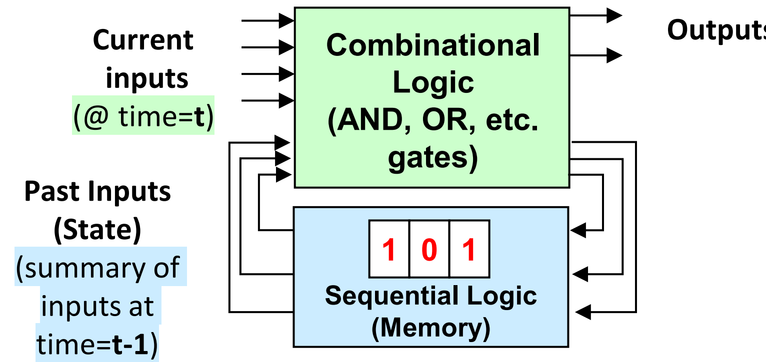
S	P
0	1
1	0



H	L	P
0	0	1
0	1	?
1	0	?
1	1	0

Sequential Logic

- **Sequential Logic** can store and **remember** a signal's value at a specific time so that it can be used in the future.
  - Often, we store the previous input or output signals to be used in processing future inputs.



Previous inputs summarized via state and fed back to the combinational logic.

Speed, area, and power

DIGITAL DESIGN GOALS

Digital Design Goals

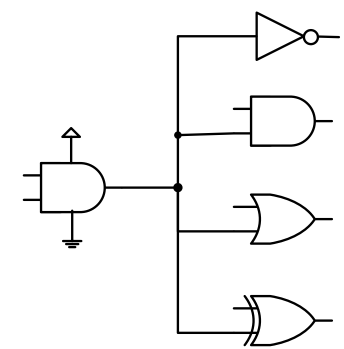
- When designing a circuit, we want to optimize for the following three things:
  - Area or Circuit Size (minimize)
  - Speed (maximize) or Delay (minimize)
  - Power (minimize)
- Can usually only optimize one or two of the 3
  - There is a huge trade space! This is what engineering is all about!

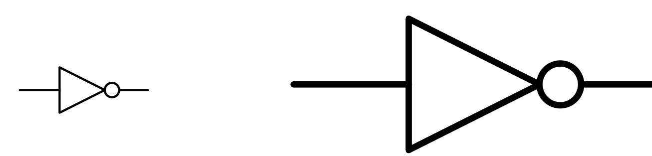
Minimizing Circuit Area

- Approaches:
  - Reduce the number of gates used to implement a circuit
  - Reduce the number of inputs to each gate
    - In general, a gate with n inputs requires $2n$ transistors to implement
- Simplify logic expressions (usually by factoring and then canceling terms) to reduce the number of gates
  - We'll learn more about this soon

Maximizing Speed

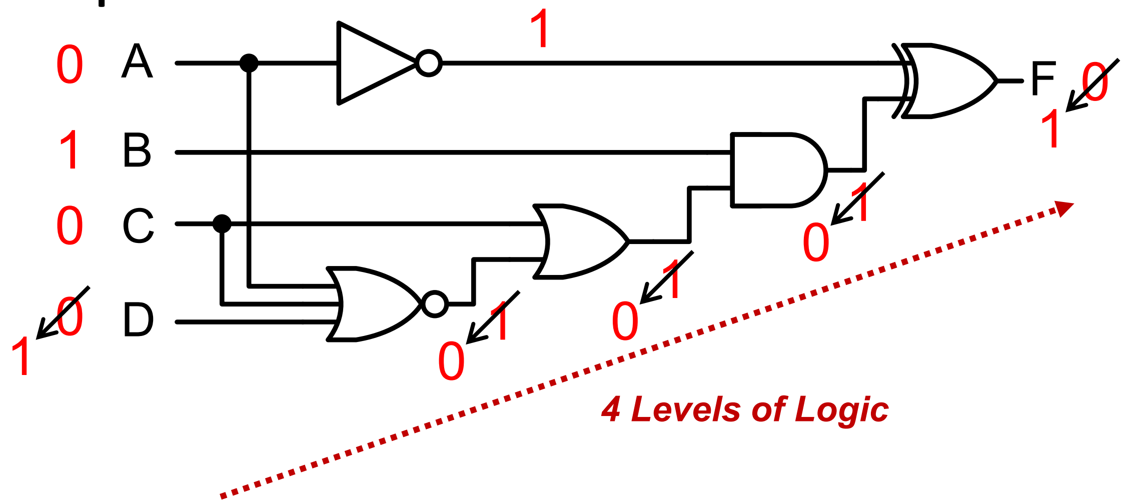
- Speed is affected by:
  - Levels of logic (path length)
  - Gate type
  - Number of inputs (fan-in) to the gate
    - Usually limited to 4-5 input gates
  - Number of outputs a gate connects to (fan-out)
  - Feature size and implementation technology





Delay Example

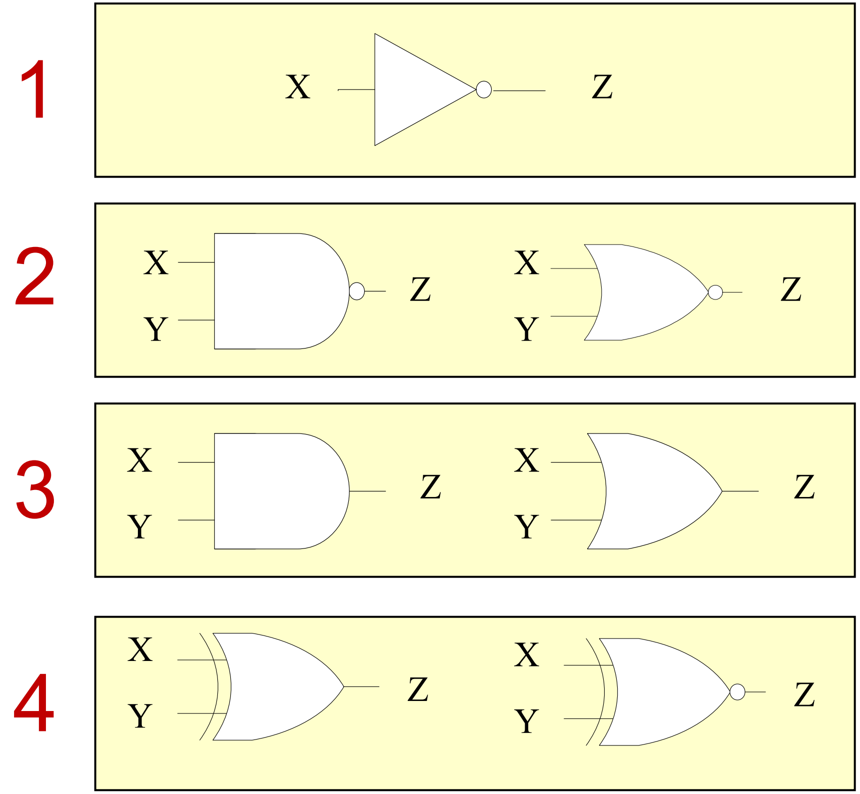
- Gates take time for the output to change once the inputs change (aka propagation delay)
- Levels of Logic = Maximum # of gates on any path from input to output



Change in D, C, or A must propagate through 4 levels of gates!

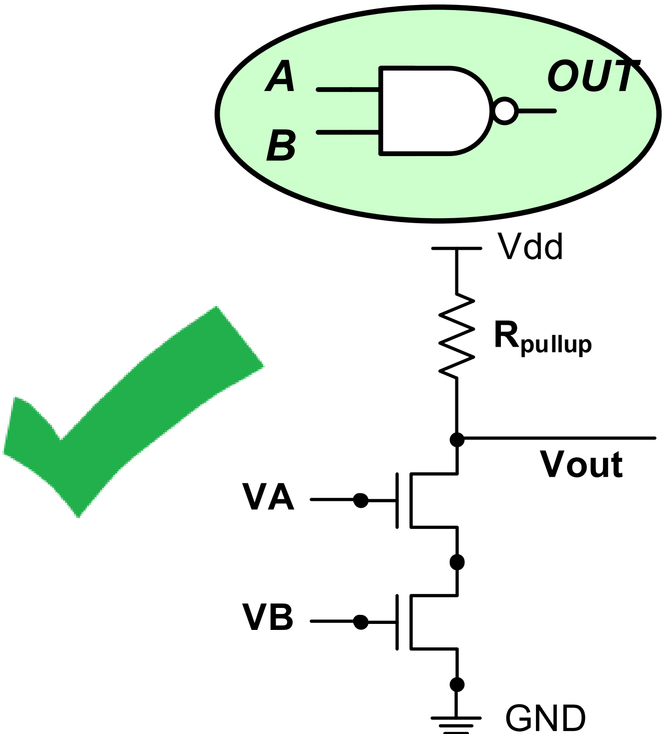
Gate Delays

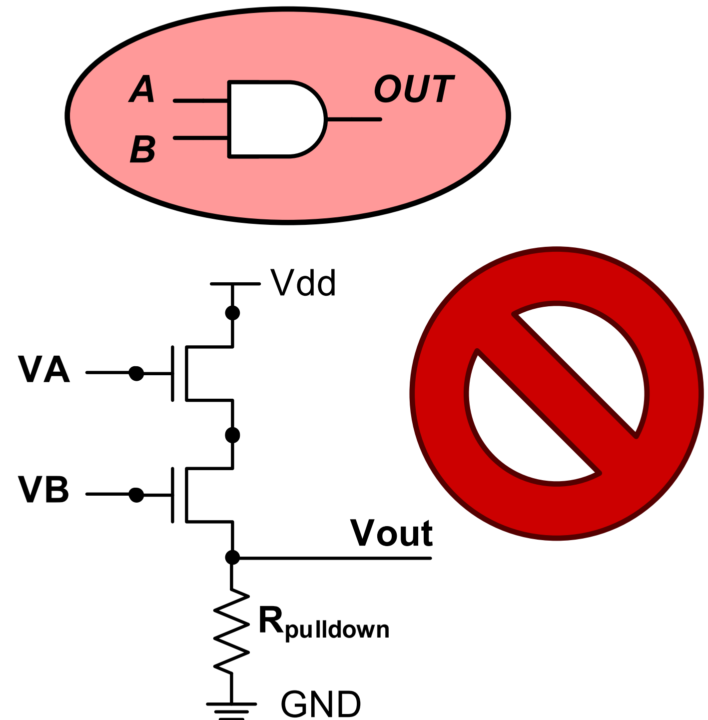
- Order the gate types in terms of fastest to slowest?
- Typical gate delay for a 2-input NAND or NOR is under a 100 ps



Self-Exploration: Why NANDs not ANDs

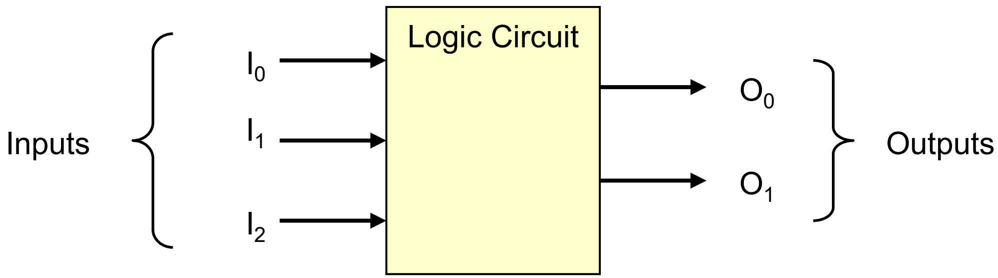
- Hint: There are different kinds of transistors with different "strengths" and "weaknesses"...





Logical Operations Summary

- All digital circuits can be described using AND, OR, and NOT
  - Note: You'll learn in future courses that digital circuits can be described with any of the following sets:
    - {AND, NOT}, {OR, NOT}, {NAND only}, or {NOR only}
- Normal convention: 1 = true / 0 = false
- A logic circuit takes some digital inputs and transforms each possible input combination to a desired output values



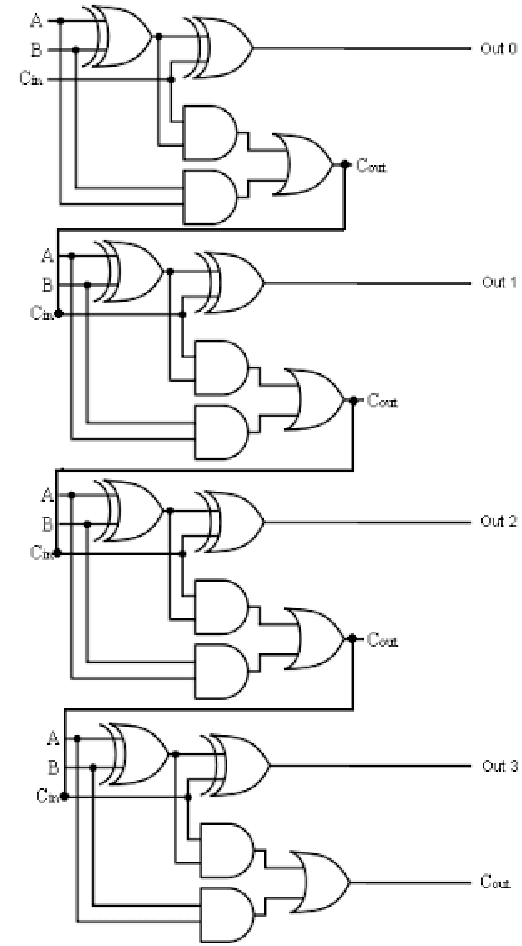
Trivia-of-the-day: The Apollo Guidance Computer that controlled the lunar spacecraft in 1969 was built out of 8,400 3-input NOR gates.

Looking ahead to our next few lectures…

READY TO LAUNCH

From Gates to Operators...

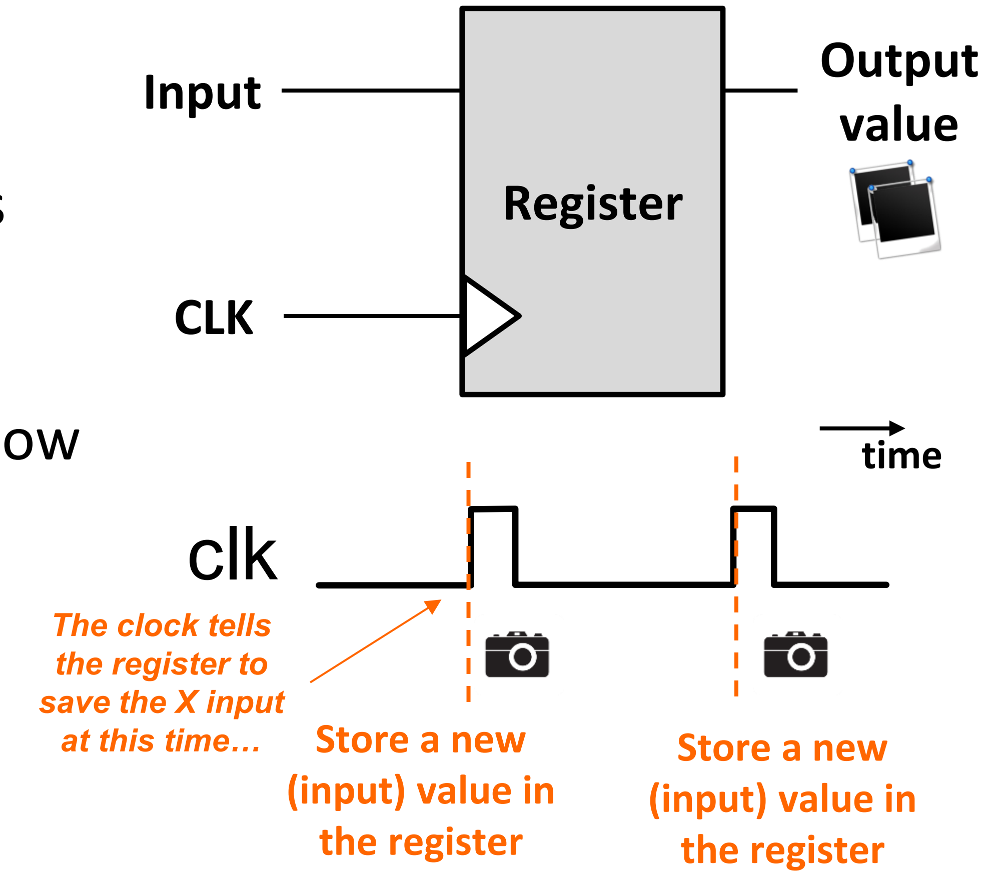
- By putting many gates together into networks like this we can implement any operation we can perform in software:
  - +, -, *, /,
  - <, >, ==,
  - pow(), log(),
  - etc.
- Circuit to the right adds 4-bit numbers.
- What are the design techniques we can use to not only create this design, but optimize it?



https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcTVEw3EBrthY0fCPUQmCIXvcK0efZep5vaE1MEhzWYERQ&s=10

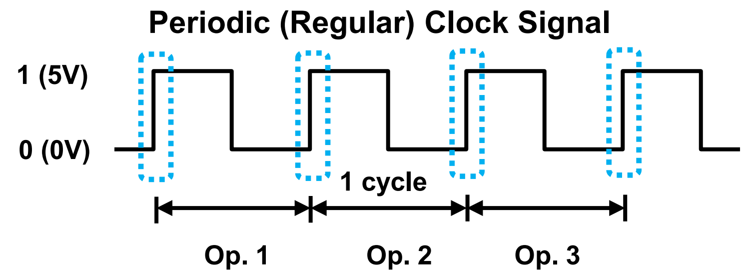
Sequential Logic: The Register

- Sequential logic "remembers" bits.
- Recall that a ***REGISTER*** (like **PORTD**, **DDRB**, etc.) stores bits with the help of a **clock** signal.
- The register is a key building block of sequential logic, but how is it built?
  - We will learn how soon!

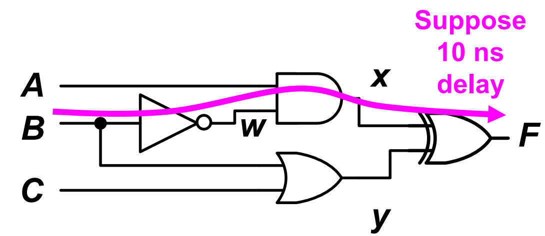


Clock Signals (Period and Frequency)

- Registers need a clock pulse (or edge) to trigger
- Usually, the clock pulses occur at a regular interval (i.e. period, **T**)
  - Alternating **high/low voltage pulse train**
  - 1 cycle is usually measured from rising/positive edge to rising/positive edge
- Clock frequency (**F**) = # of cycles per second
  - **F = 1/T**
- The clock period of a digital circuit is set based on the **slowest** delay path from the output of a register to the input of the next



2 GHz = $2*10^9$ cycles per second = 0.5 ns/cycle



Clock Period (T) must be > 10 ns (i.e. F < 100 MHz)

Summary

- **Combinational logic** corresponds to operations that manipulate number
  - similar to software operators +, -, *, /, <, >, etc.

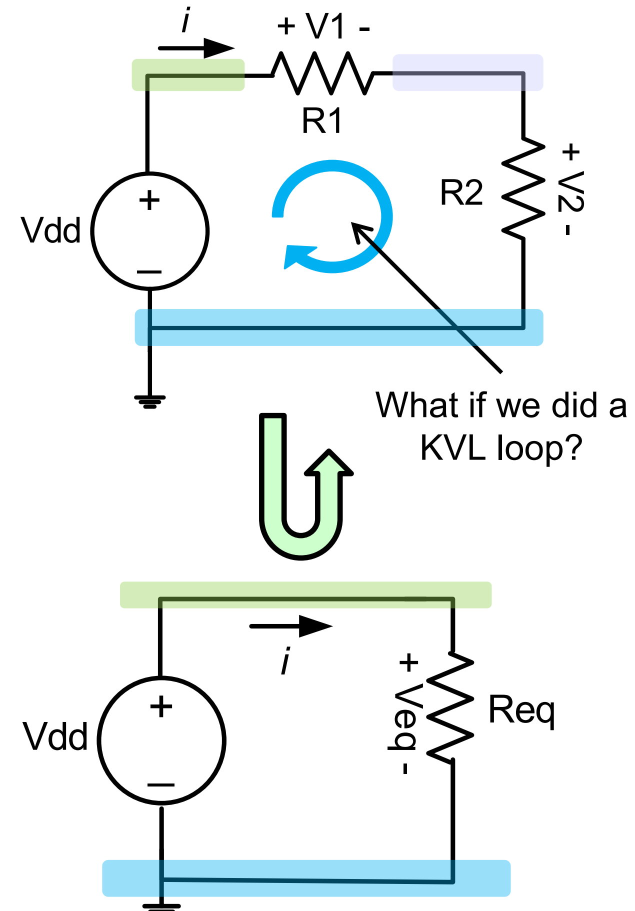
- **Sequential logic** corresponds to variables that can store values for further processing

ANALOG CIRCUIT ANALYSIS

Ex 1: Solving Voltage & Current (1)

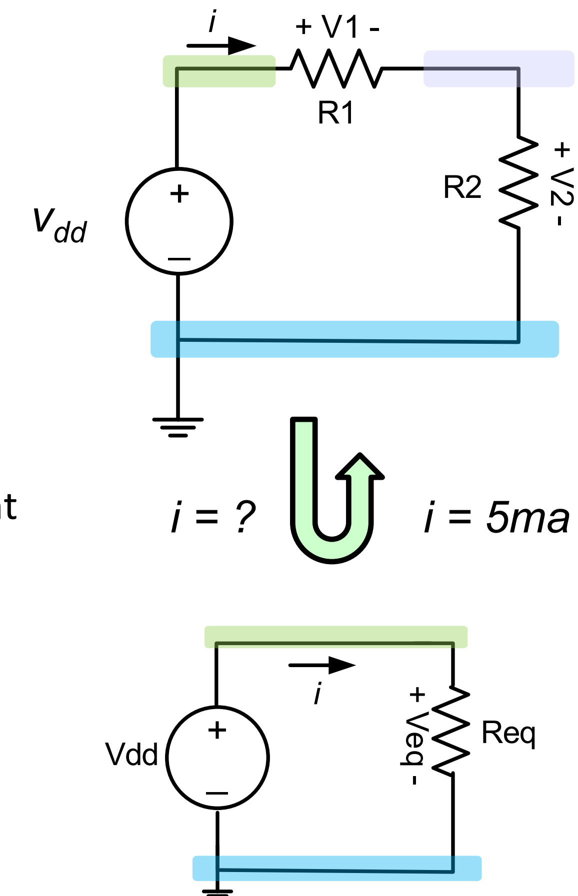
- Given the circuit to the right, let…
  - Vdd = +5V, R1 = 400 ohms, R2 = 600 ohms
- Solve for the current through the circuit and voltages across each resistor (i.e. V1 and V2)

- Consider the 3 laws we have learned (KCL, KVL, and Ohms). Given that we want to solve for the voltages V1 and V2, which seem most applicable? KVL and Ohms
  - **KVL Loop**: Only 1 loop and and TWO unknowns!
  - Ohm's law: **V1 = i*R1** and **V2 = i*R2**, but we don't know the current, i
- **Common approach**: ***Collapse*** resistors to a single value where Ohm's law can be used (usually to solve for, i) which can then be **substituted** back in the actual (re-expanded) circuit.



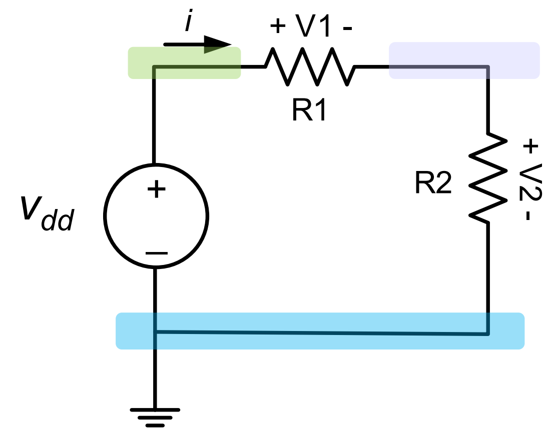
Ex 1: Solving Voltage & Current (2)

- Given the circuit to the right, let…
  - Vdd = +5V, R1 = 400 ohms, R2 = 600 ohms
- Solve for the current through the circuit and voltages across each resistors (i.e. V1 and V2)
  - Identify nodes to help identify series and parallel
  - Collapse the circuit to a single **series** equivalent, Req.
  - By KVL we know Vdd = Veq
  - Since everything is in series, KCL teaches us that the current through each component must be the same, let's call it *i*
    - *i = Veq / (Req) = Vdd / (R1+R2) = 5/1000 = 5 mA*
  - This alone lets us compute V1 and V2 since Ohm's law says
    - V1 = i*R1 and V2 = i*R2
    - V1 = 2V and V2 = 3V
  - Though unneeded, KVL teaches us that
    - Vdd-V1-V2=0 or that Vdd = V1 + V2



Ex 1: Using the Voltage Divider Eqn.

- Reconsidering the circuit to the right with…
  - Vdd = +5V, R1 = 400 ohms, R2 = 600 ohms
- Solve for the current through the circuit and voltages across each resistors (i.e. V1 and V2)
  - We can use the voltage divider concept to immediately arrive at the value of V2
  - $V2 = V_{dd}\left[\frac{R2}{R1+R2}\right]$

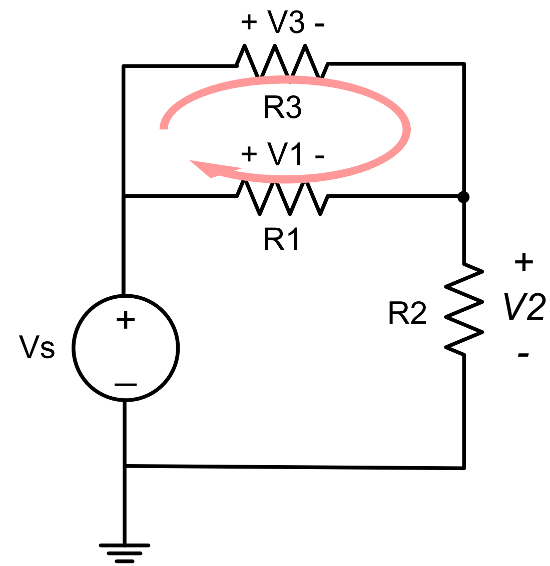


Be sure to:
- Confirm resistors are in series
- Find the total voltage dropped across the entire series

Ex 2: Solving Voltage & Current (1)

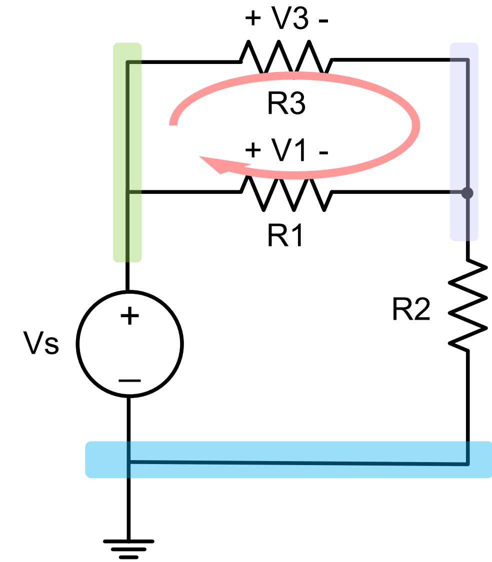
- Consider the circuit on the right…

- Can we solve for V2 directly using the voltage divider equation:
  - $V2 = \left[\frac{R2}{R1+R2}\right] \cdot Vs$
  - **No!!! (Resistors aren't in series)**
- What is the relationship between V1 and V3?
  - Equal. Use a KVL Loop to prove (-V3+V1=0) => V3 = V1



Ex 2: Solving Voltage & Current (2)

- Consider the circuit on the right…
- Can you solve for the voltage V2 (in terms of Vs, R1, R2, R3)?
  - Combine R1 and R3 using parallel resistor relationship
  - R1 and R3 can be combined to R_{eq} = (R1R3)/(R1+R3)
  - Now use voltage divider since "R_{eq}" and R2 are in series…
  - $V2 = \left[\frac{R2}{Req+R2}\right] \cdot Vs = \left[\frac{R2}{\frac{R1 \cdot R3}{R1+R3}+R2}\right] \cdot Vs$

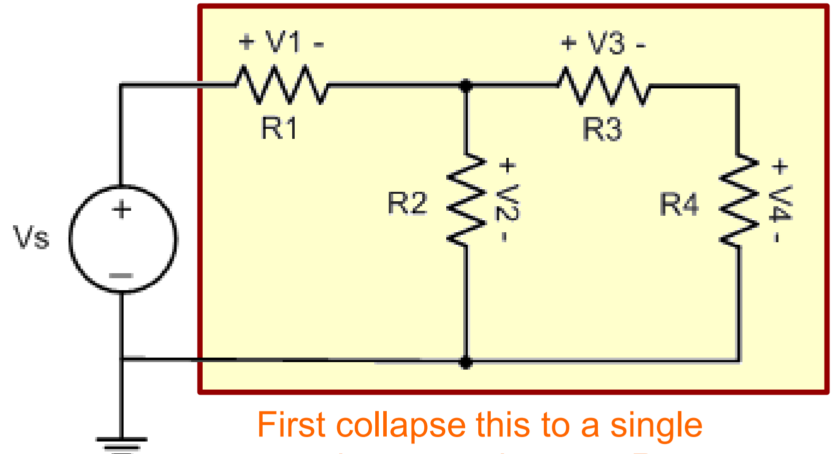


- Can you solve for the voltage V1 (in terms of Vs, R1, R2, R3)?
  - KVL says Vs – V1 – V2 = 0. We know Vs and just solved for V2 so we can plug into: V1 = Vs – V2
  - Or, just use the voltage divider equation for "Req"
  - $Veq = \mathrm{V1} = \mathrm{V3} = \left[\frac{Req}{Req+R2}\right] \cdot Vs = \left[\frac{\frac{R1 \cdot R3}{R1+R3}}{\frac{R1 \cdot R3}{R1+R3}+R2}\right] \cdot Vs$

Ex 3: Cumulative Example (1)

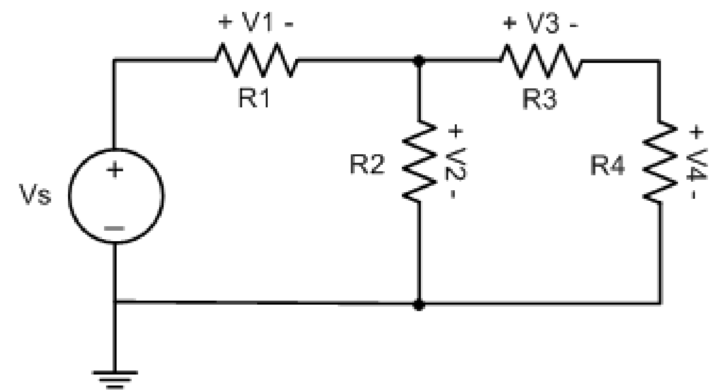
- Given the following parameters…
  - Vs=5V, R1=4, R2 = 12, R3 = 2 and R4 = 10 ohms.
- Can we use the voltage divider concept to immediately solve the voltage across R2 or do we need to first do some manipulation? What about R4?
- First, find the total equivalent resistance (R_{eq}) seen by Vs and then solve for the voltage across each resistor

Req = R1+[R2*(R3+R4)]/[R2+R3+R4]
= 10 ohms
i1 = Vs / Req = 5/10 = 0.5A
V1 = i1*R1 = 2V
V2 = 5V – 2V = 3V (using KVL)
V3 = 3 * 2/(2+10) (volt. divider)
= 3V * 1/6
= 0.5 V
V4 = V2 * R4/(R3 + R4) (volt. divider)
= 3V * 5/6
= 2.5V



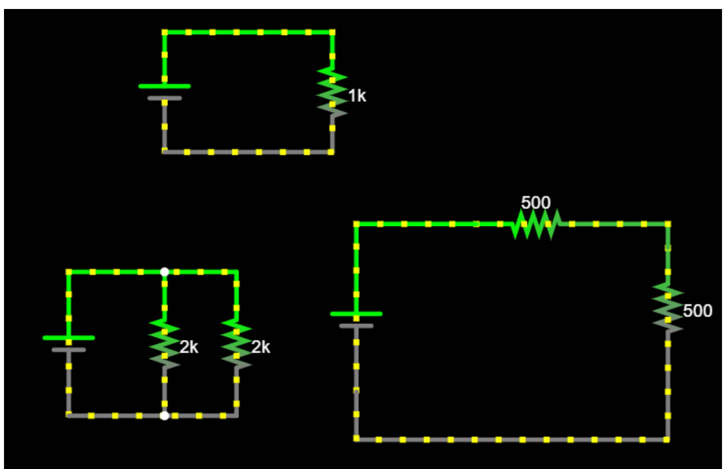
First collapse this to a single equivalent resistance, R_{eq}

Ex 3: Cumulative Example (2)



Optional: Interactive Resistor Demo

- Go to this website (Falstad.com)
  - Choose File..Import from Text
  - Paste in the text shown to the right
- Mouse over the resistors and wires to see the voltage and current values.
- Change the resistor values to explore



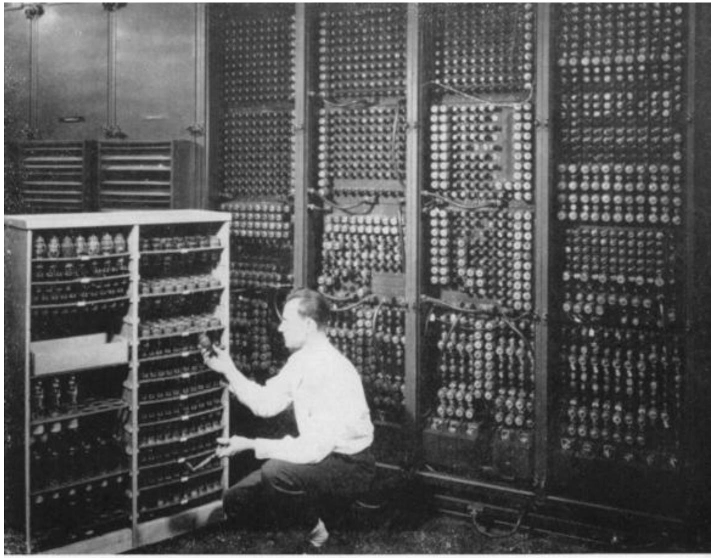

```
$ 1 0.000005 10.20027730826997 50 5
43 5e-11
v 352 400 352 304 0 0 40 5 0 0 0.5
r 432 288 512 288 0 500
r 560 304 560 384 0 500
w 560 384 560 432 0
w 560 432 352 432 0
w 352 432 352 400 0
w 352 304 352 288 0
w 352 288 432 288 0
w 512 288 560 288 0
w 560 288 560 304 0
v 160 416 160 320 0 0 40 5 0 0 0.5
r 224 320 224 416 0 2000
r 272 320 272 416 0 2000
w 160 320 224 320 0
w 160 416 224 416 0
w 224 416 272 416 0
w 224 320 272 320 0
v 224 240 224 160 0 0 40 5 0 0 0.5
w 224 160 336 160 0
r 336 160 336 240 0 1000
w 336 240 224 240 0
```

(Background material – NOT TESTED)

HISTORY OF DIGITAL COMPUTING

Electronic Computers

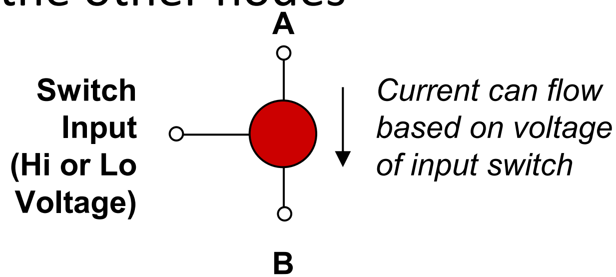
- Replaced mechanical computation engines
- Electronic computers require some kind of "switching" (on/off) technology
  - Initially that was the vacuum tube
- Example: ENIAC (1945)
  - One of the first, fully electronic computers
  - Used 19,000 vacuum tubes
  - Weighed 30 tons
  - 15,000 square feet
  - Largest number range: 10 decimal digits (±9,999,999,999)
- Many early computers were programmed with patch panels (wire plugs)



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

Vacuum Tube Technology

- Digital, electronic computers use some sort of voltage controlled switch (on/off)
- Looks like a light bulb
- Usually 3 nodes
  - 1 node serves as the switch value allowing current to flow between the other 2 nodes (on) or preventing current flow between the other 2 nodes (off)
  - Example: if the switch input voltage is 5V, then current is allowed to flow between the other nodes



Vacuum Tube

Vacuum Tube Disadvantages

- Relatively large
  - Especially when you need 19,000 to make 1 computer
- Unreliable
  - Can burn out just like a light bulb
- Dissipate a lot of heat (power)