ECE 550
Fundamentals of Computer Systems and Engineering

Transistors -> Gates

Last time….

(Almost) every class will start with the same question:

- Who can remind us what we talked about last time?

  (besides course policies)

Last time....

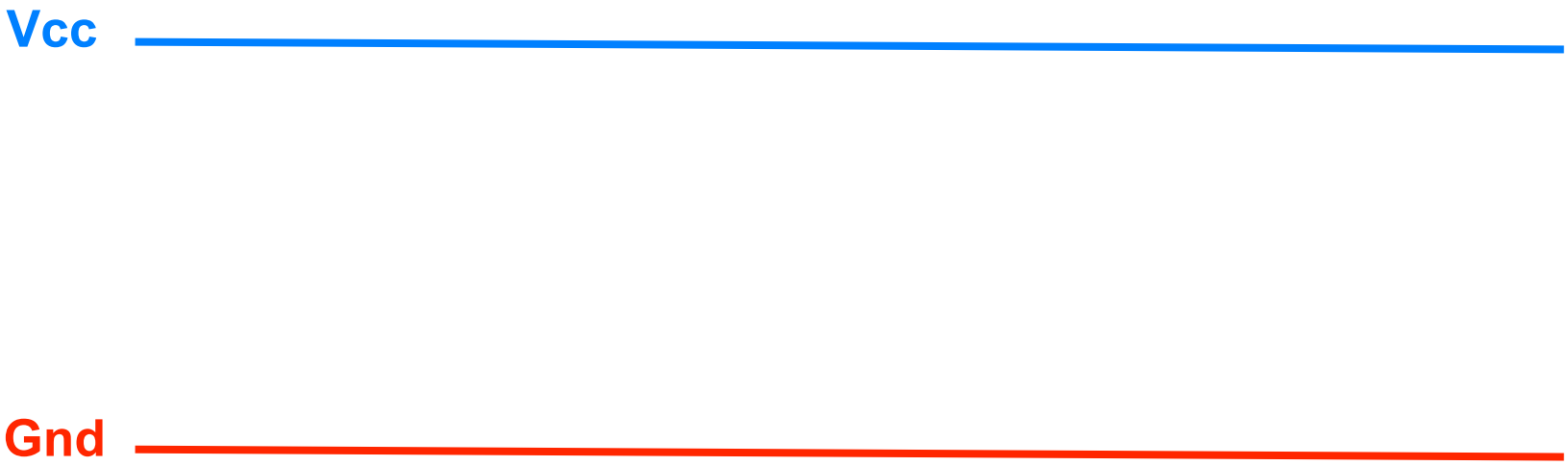
(Almost) every class will start with the same question:

- Who can remind us what we talked about last time?

  (besides course policies)

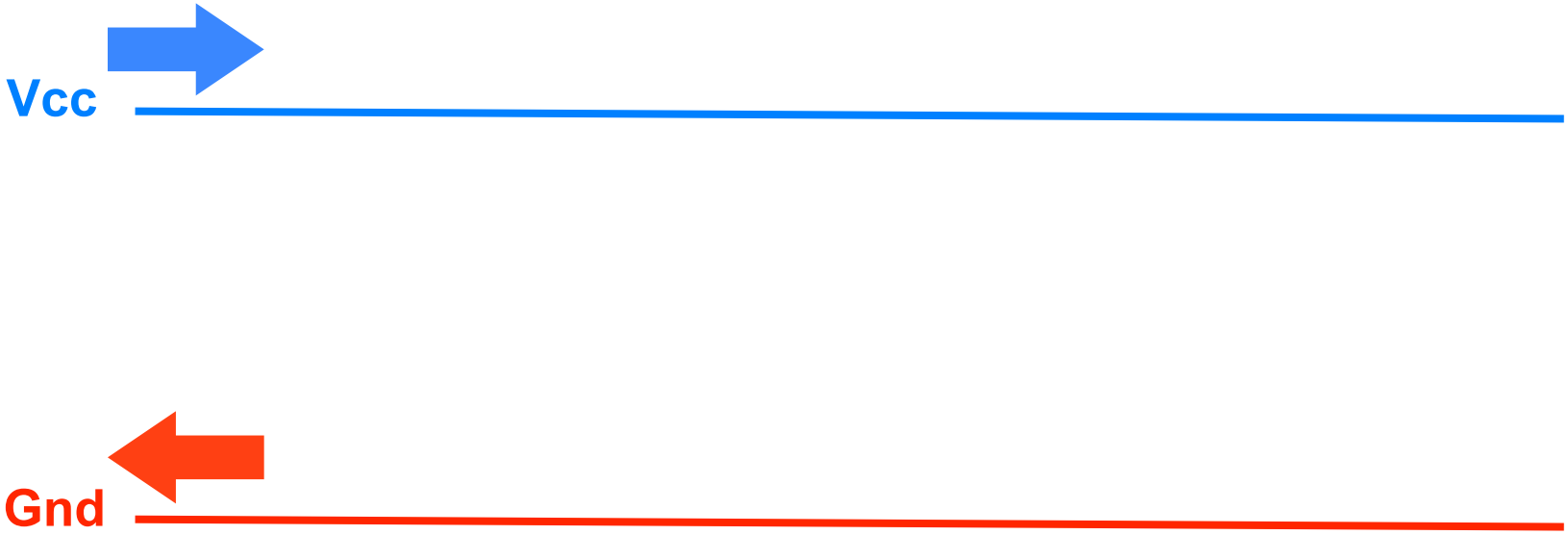
- Abstraction
  - Interface vs Implementation
- Tools
  - VHDL, Quartus
- Transistors and Gates
  - More on this today

Power (Vcc) and Ground (Gnd)



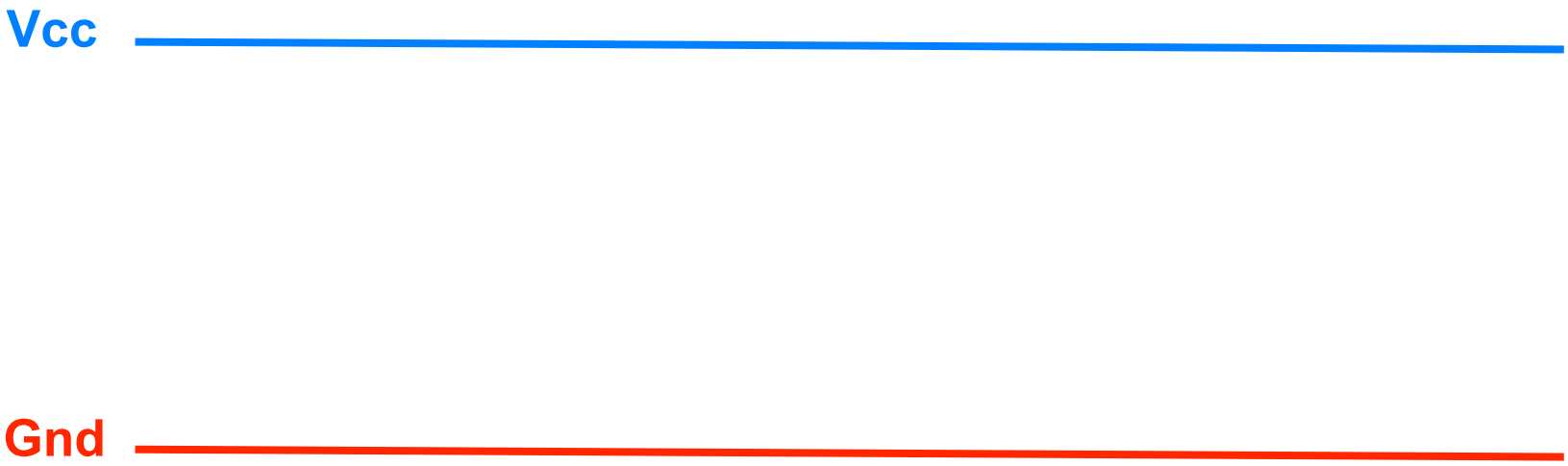
- Two supply rails:
  - Power (aka Vcc, sometimes called Vdd) , e.g., +1.0 V
    - Logically, 1
  - Ground (Gnd, or Vss), e.g., 0 V
    - Logically, 0
  - I'm going to use Vcc/Gnd because that's what Quartus uses

Power (Vcc) and Ground (Gnd)



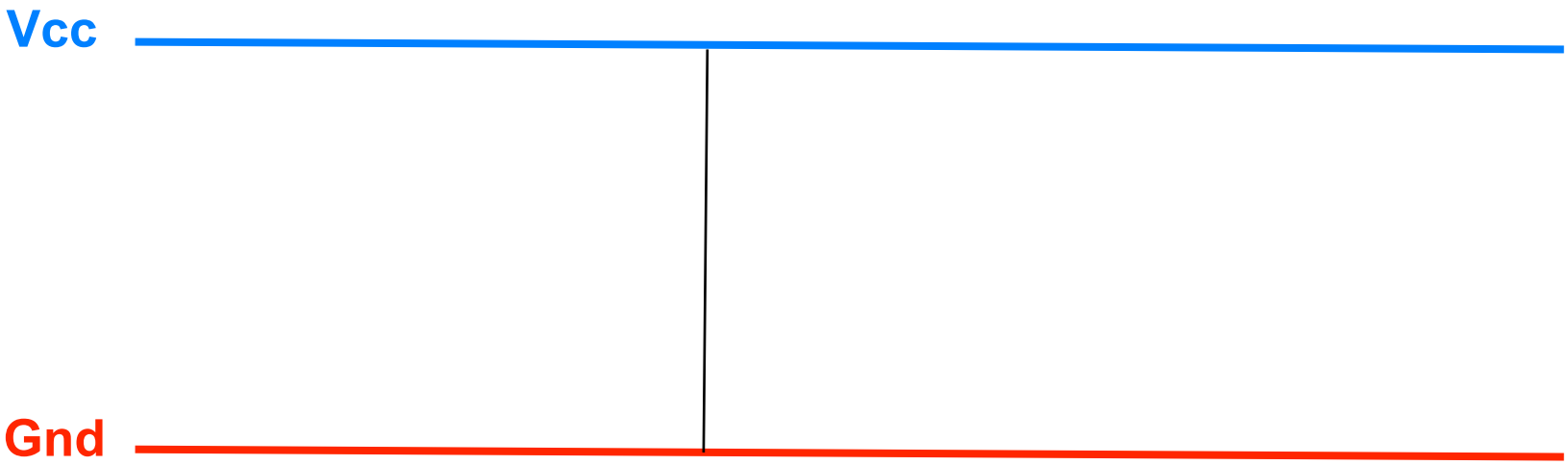
- Water analogy
  - Power: think of this as a pump, pushing water in
  - Ground: think of this as a pump sucking water out

Wires



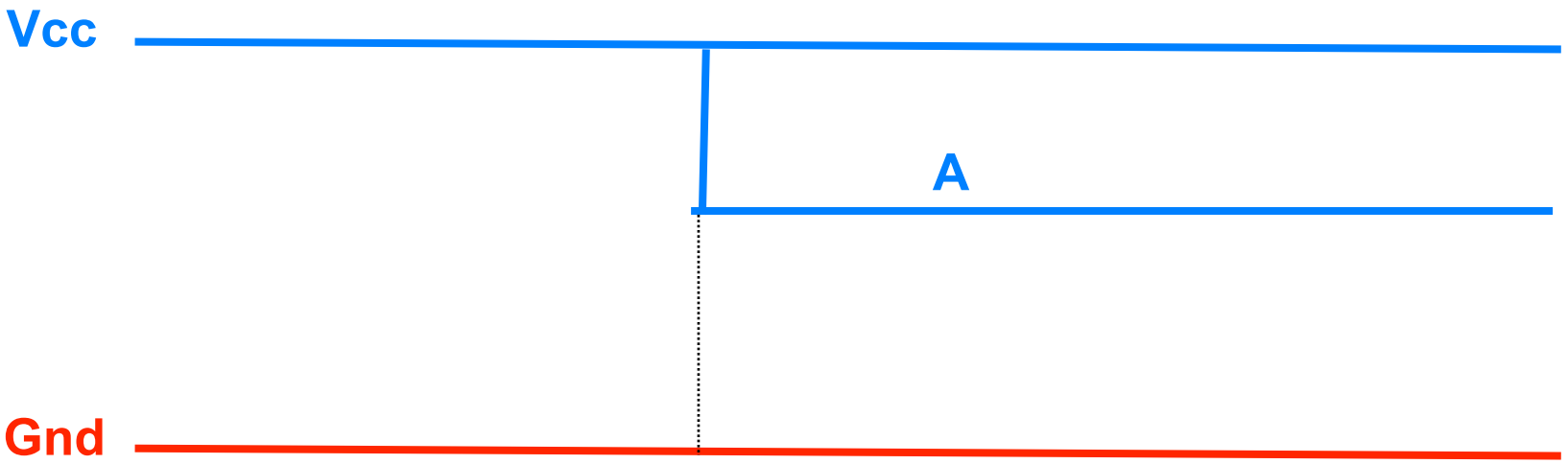
- A wire (or other conductor) causes current to flow
  - Attempts to equalize voltage
  - Water analogy: think of a pipe

Short circuit



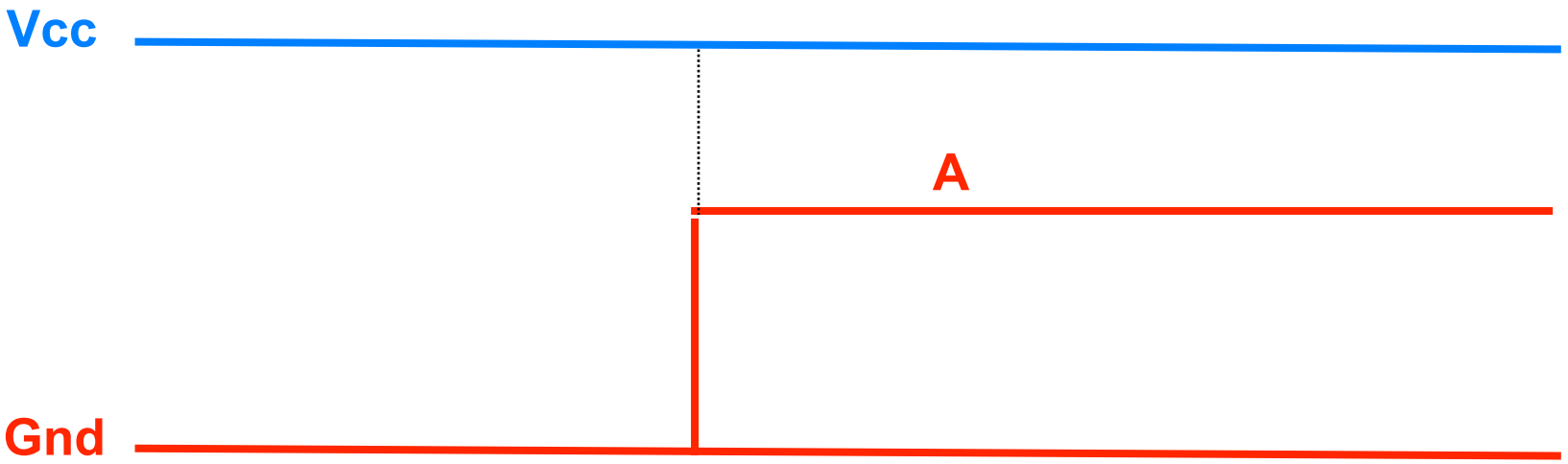
- Short circuit: direct connection from power to ground
  - Very high current (think of fast, continuous flow of water)
  - Generates a lot of heat
  - Destroys your chip
  - Very bad!

Switching



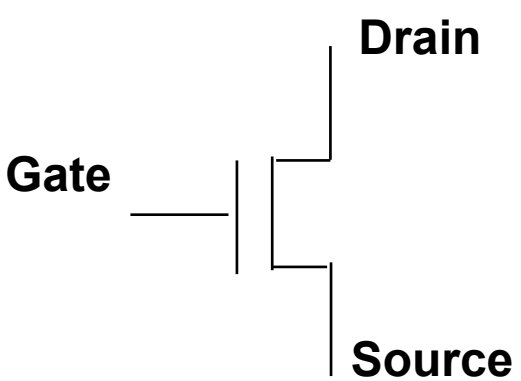
- Suppose instead we had some sort of switch
  - Think "valve"
  - Here, top connection conducts, connecting A to Vcc
    - Think of A as a pipe, pumped full of water
  - The bottom half resists (think closed value) insulating A from Gnd
    - The water in A cannot get sucked down

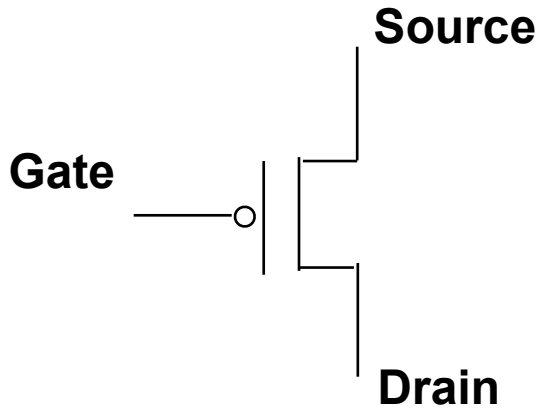
Switching



- If we switch our connection…
  - Current flows as A changes voltage levels
    - Think of a pipe draining out as its connected to suction
  - Connection to power is closed, so no short circuit

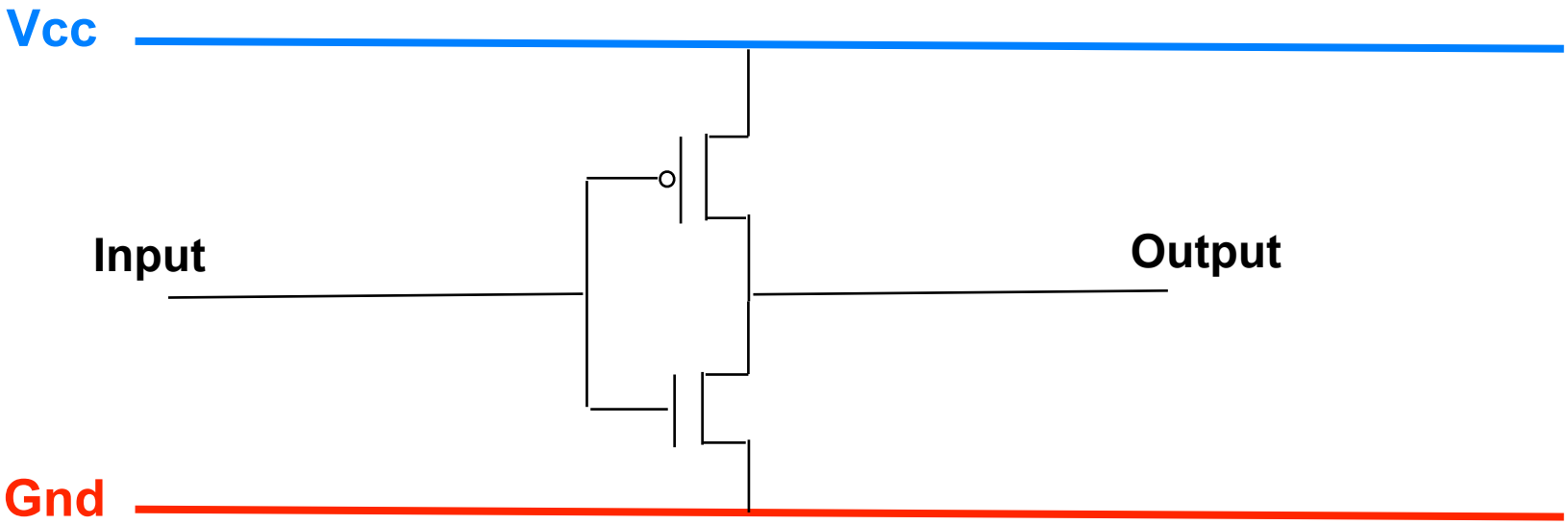
Transistors: Electrically controlled





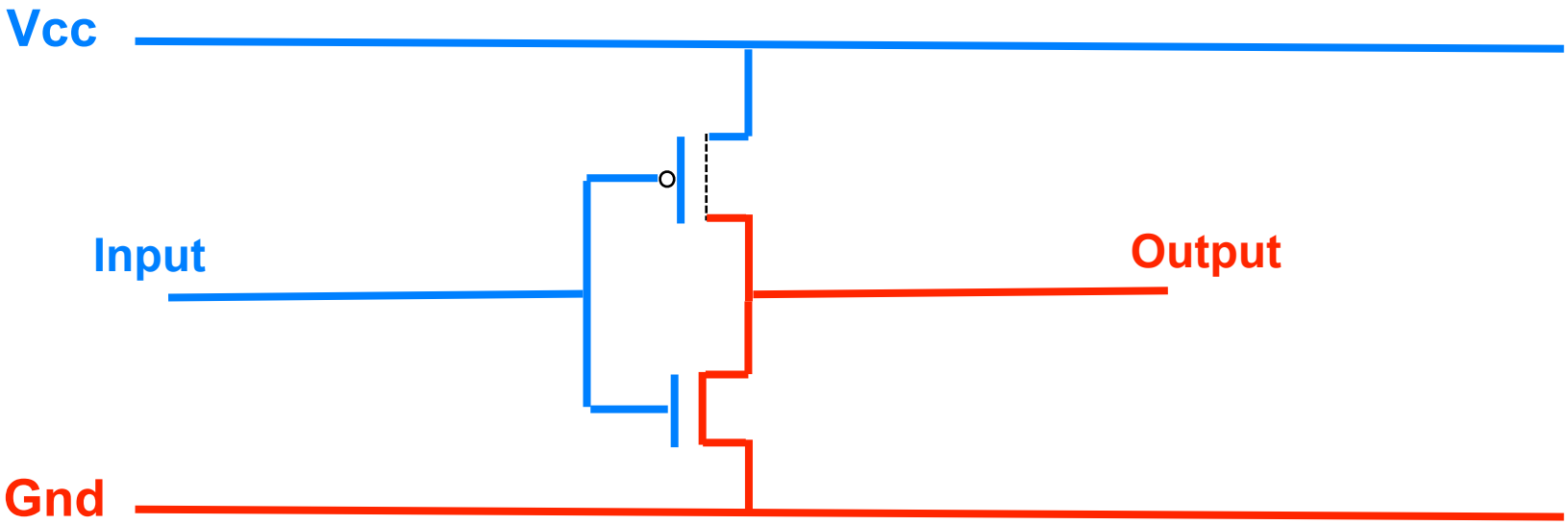
- Two types:
  - NMOS (left: no circle):
    - Conducts when gate is 1, resists when gate is 0
    - Connect source to either Ground or (Drain of another NMOS)
  - PMOS (right: circle):
    - Conducts when gate is 0, resists when gate is 1
    - Connect source to either Vcc or (Drain of another PMOS)

CMOS: Complementary MOS



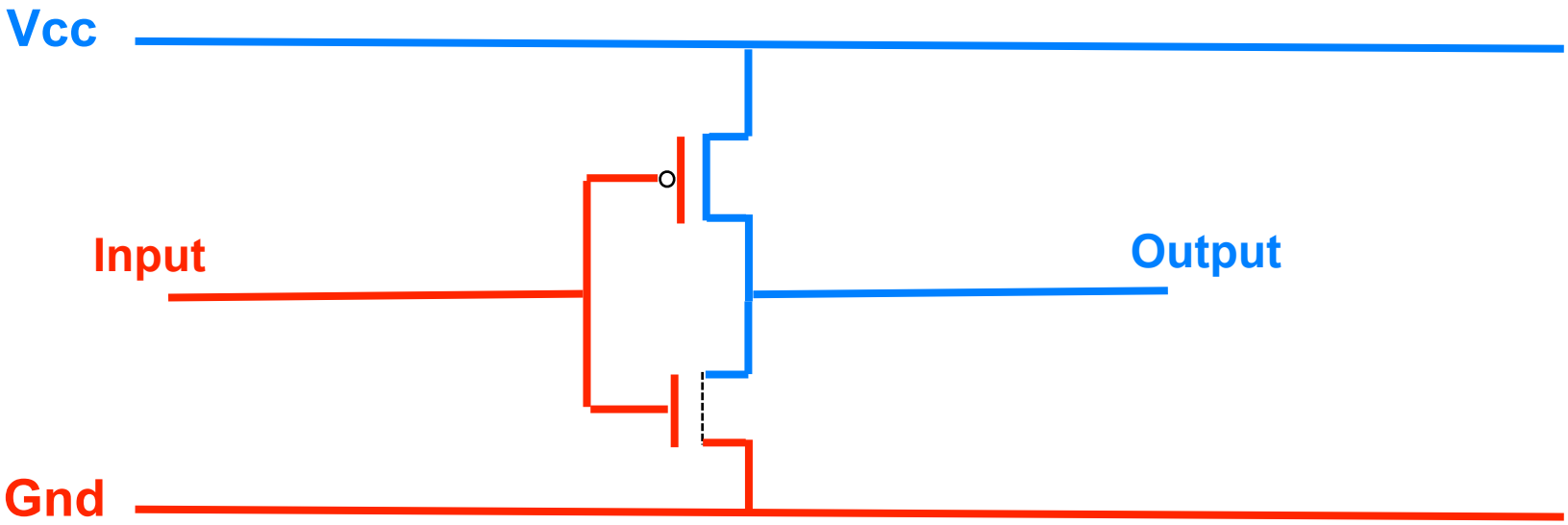
- CMOS (most common, all we care about):
  - Put PMOS and NMOS in complementary fashion
    - Either PMOS conducts or NMOS conducts, but not both
- Form a logic gate
  - Input (s): connected to gates of transistors
  - Output: connected to drains

CMOS: Complementary MOS



- Let's see how this works.
  - Suppose Input = 1 (circuitry to control input, not shown)
  - PMOS transistor resists
    - No connection between Output and Vcc
  - NMOS transistor conducts
    - Connection between Output and Gnd
  - Output is 0

CMOS: Complementary MOS

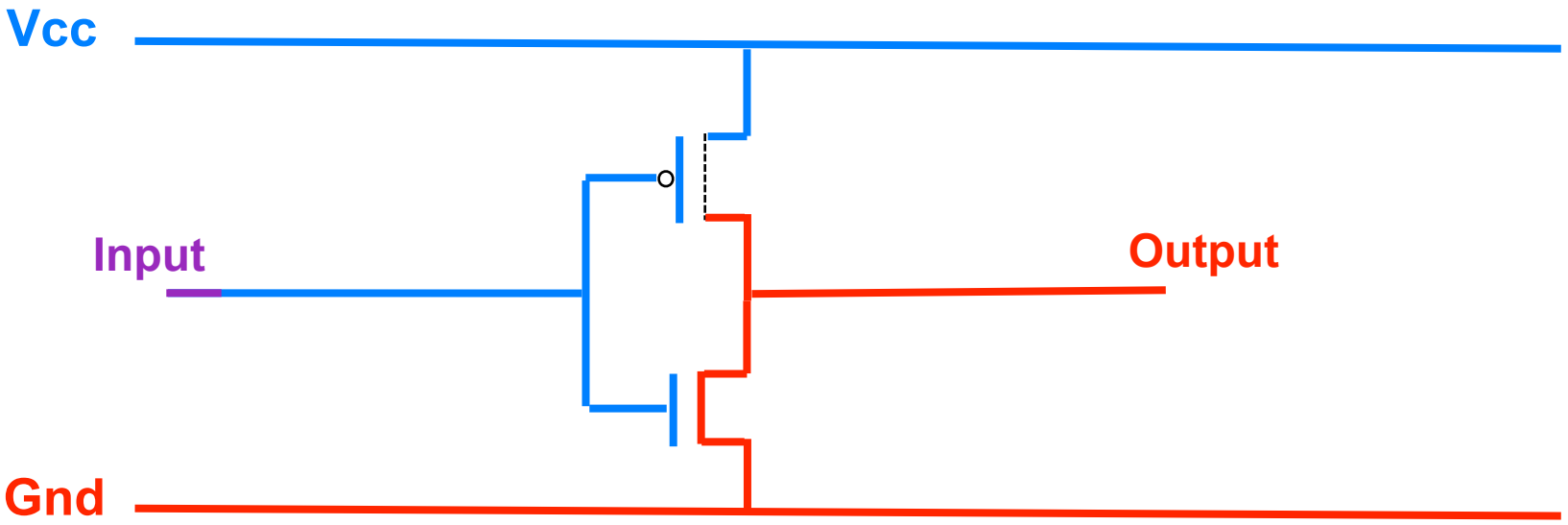


- Now suppose Input changes to 0
  - PMOS transistor conducts
    - Connection between Output and Vcc
  - NMOS transistor conducts
    - No Connection between Output and Gnd
  - Output is 1

Switching delays

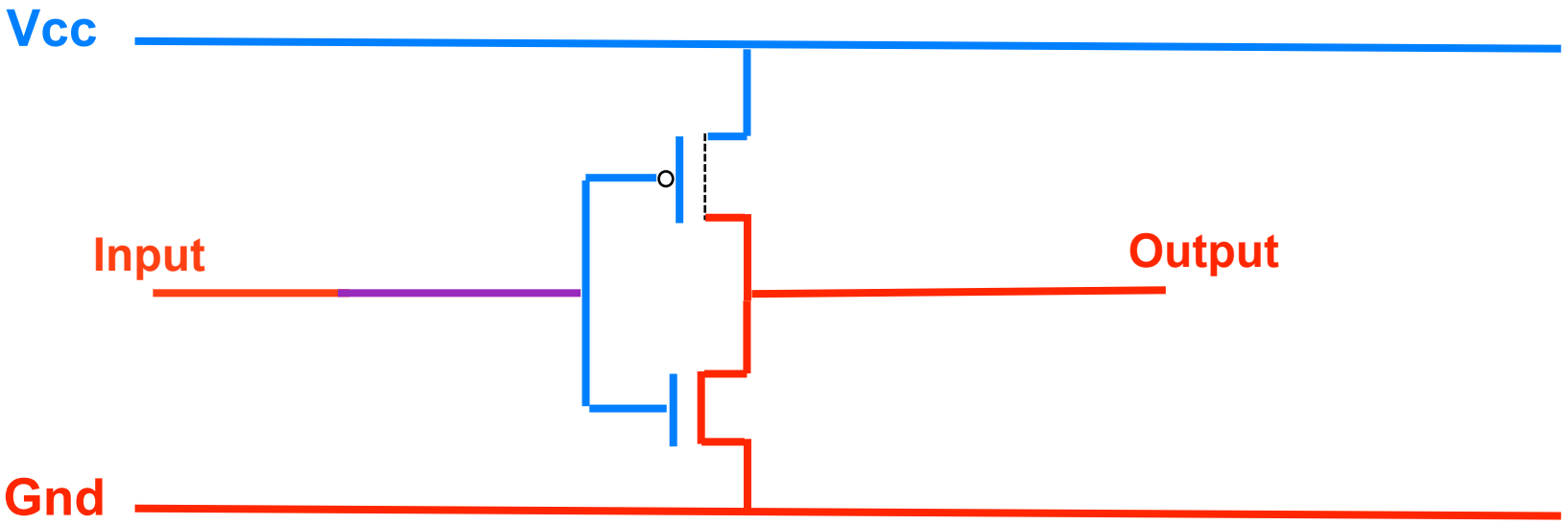
- Note: this doesn't happen instantly
  - There is some delay as these change
  - Imagine again, pipes full of water
    - Draining out input pipe takes time…
    - Once its drained enough the valves start to change…
    - Filling the output pipe takes time
  - Factors the affect the delay
    - Voltage: analogous to water pressure
      - Higher voltage = faster switching, but more power/energy
    - Resistance: analogous to pipe narrowness
      - Lower resistance = faster switching
    - Capacitance: analogous to pipe volume (how much to fill)
      - Lower capacitance = faster switching
  - Calculating delay = hard, so we let our tools do it

CMOS: Complementary MOS



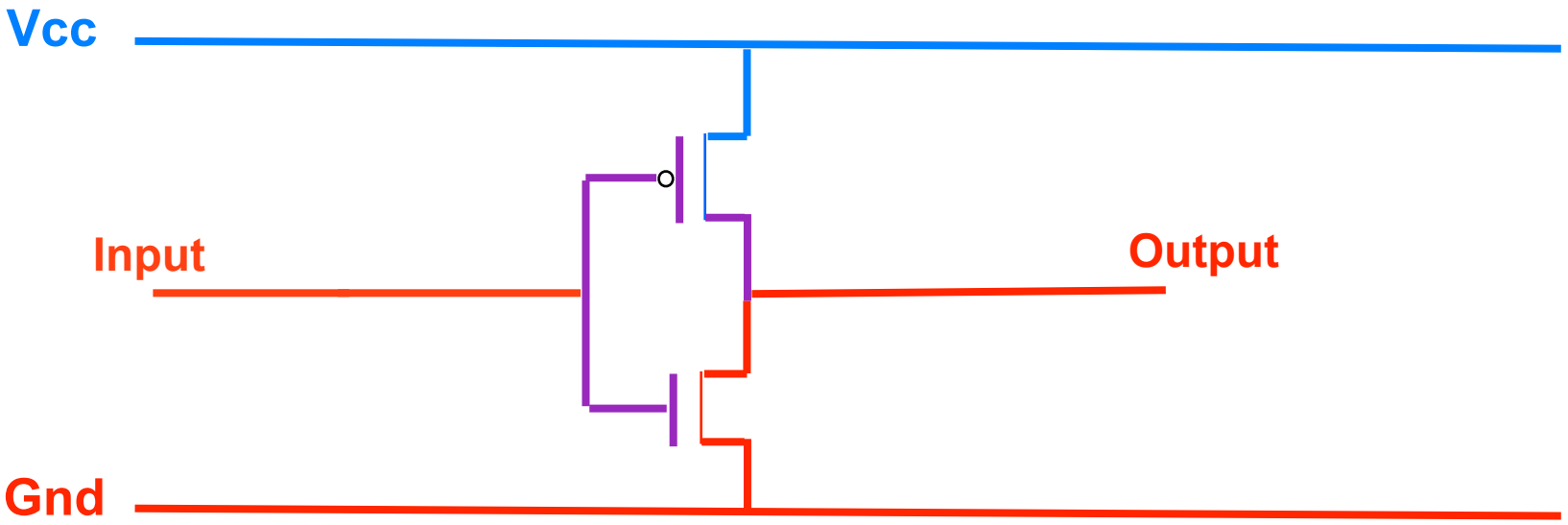
- Slightly more accurate with respect to time
  - Input starts to swing from 1 to 0 (not instant)

CMOS: Complementary MOS



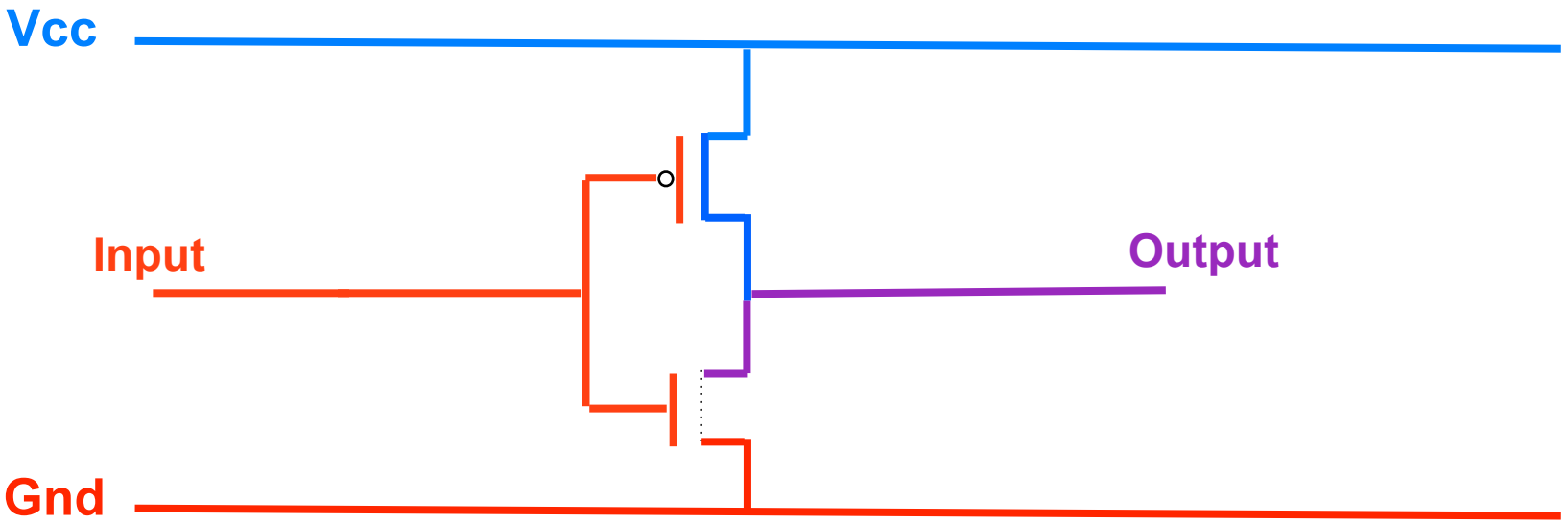
- Slightly more accurate with respect to time
  - Input starts to swing from 1 to 0 (not instant)
  - Change propagates along wires (also takes time

CMOS: Complementary MOS



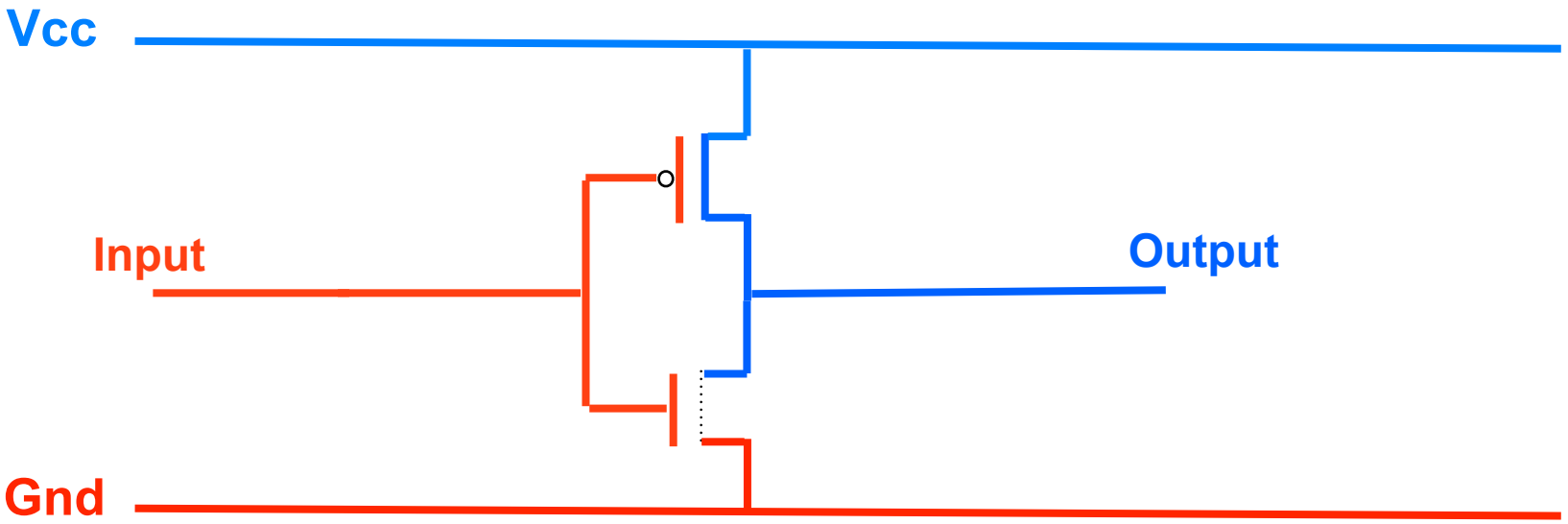
- Slightly more accurate with respect to time
  - Input starts to swing from 1 to 0 (not instant)
  - Change propagates along wires (also takes time)
  - Transistors start to switch (partially conductive)

CMOS: Complementary MOS



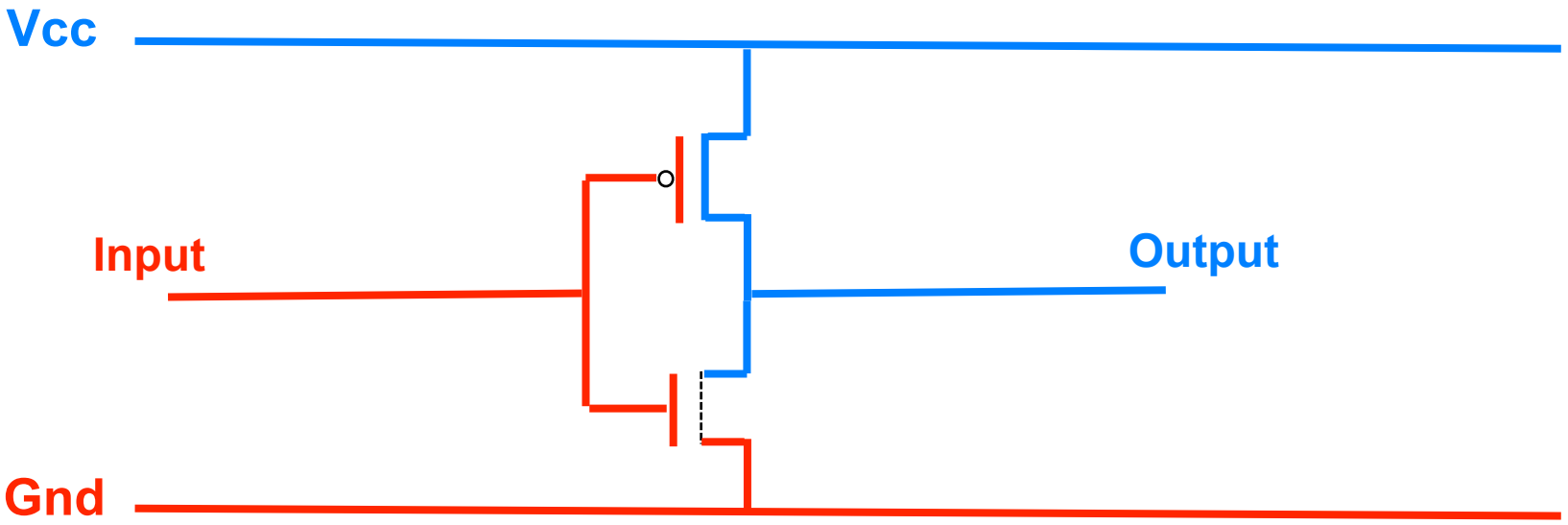
- Slightly more accurate with respect to time
  - Input starts to swing from 1 to 0 (not instant)
  - Change propagates along wires (also takes time)
  - Transistors start to switch (partially conductive)
  - Inputs reach 0 (sometime)
    - Transistors fully open/closed
    - Output may take time to transition

CMOS: Complementary MOS

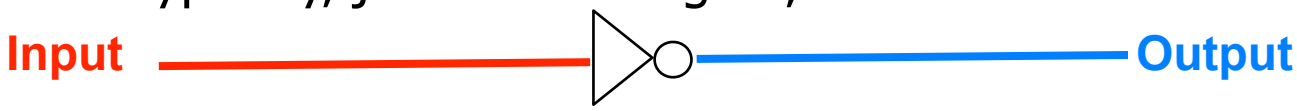


- Slightly more accurate with respect to time
  - Input starts to swing from 1 to 0 (not instant)
  - Change propagates along wires (also takes time)
  - Transistors start to switch (partially conductive)
  - Inputs reach 0 (sometime)
    - Transistors fully open/closed
    - Output may take time to transition

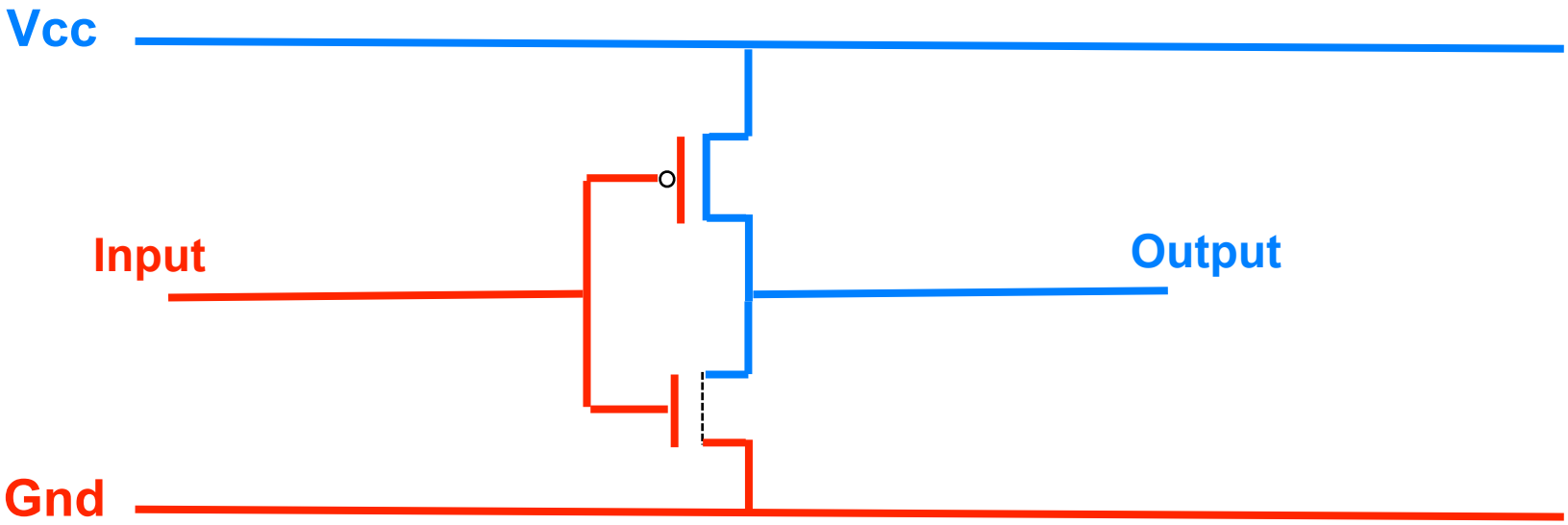
Our first logic gate: The inverter



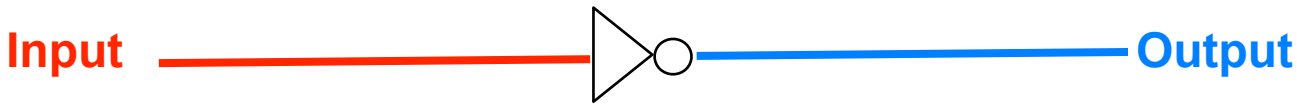
- This circuit is a logic gate: inverter or "NOT gate"
  - Gives logical negation of its input
    - Input = 0, Output = 1
    - Input = 1, Output = 0
  - Typically, just draw the gate, instead of the transistors:



Our first logic gate: The inverter



- (Small) Example of abstraction
  - Interface: "do logical negation"
  - Implementation: how to hook up the transistors



Let's build a more interesting gate

- Next, let us build a 2-input NOR gate
  - Here is a **truth table** for NOR
    - Shows output values for all possible inputs
  - Output =1 when A and B = 0
    - Connect PMOS in **series**
    - (Not A) and (Not B)
  - Output = 0 when A or B = 1
    - Connect NMOS in **parallel**
    - Not (A or B)

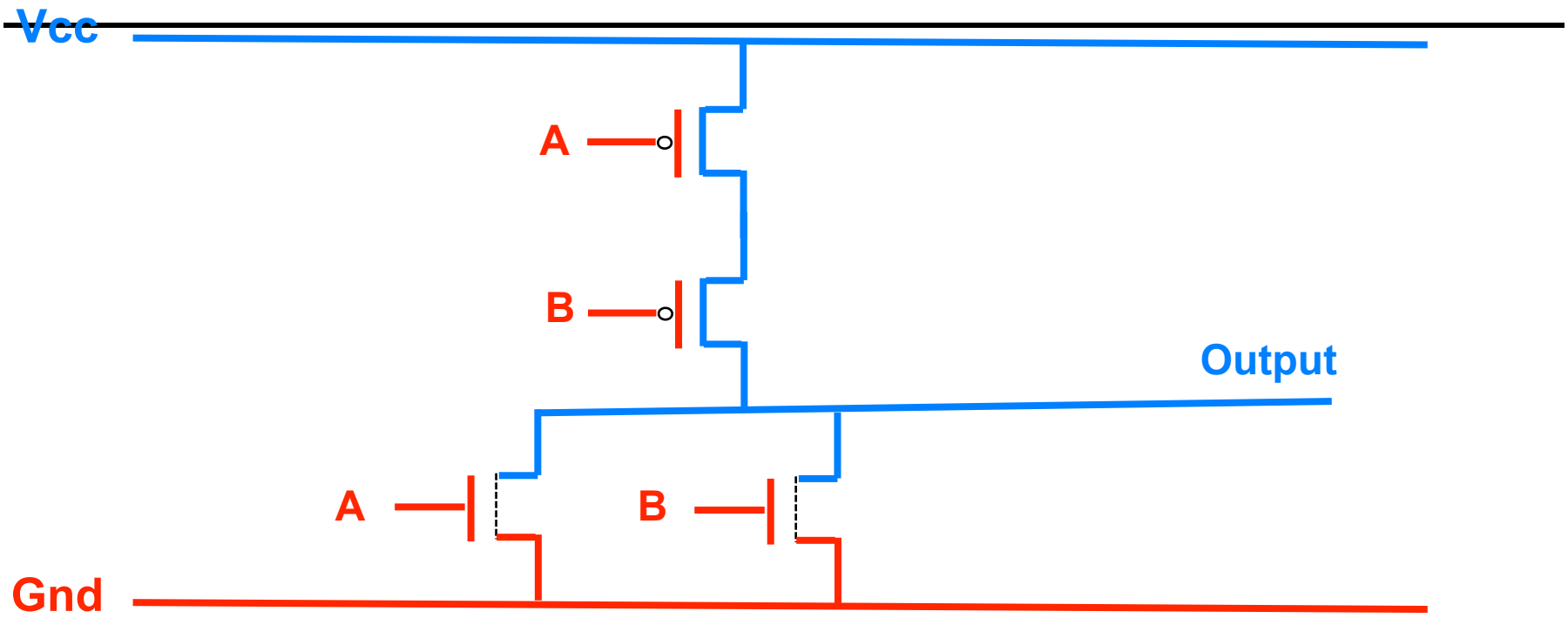
A	B	Output
0	**0**	**1**
0	1	0
1	0	0
1	1	0

Note: two formulas are logically equivalent (DeMorgan's Laws)

- PMOS formula has NOTs on inputs
- NMOS formula has NOT on output

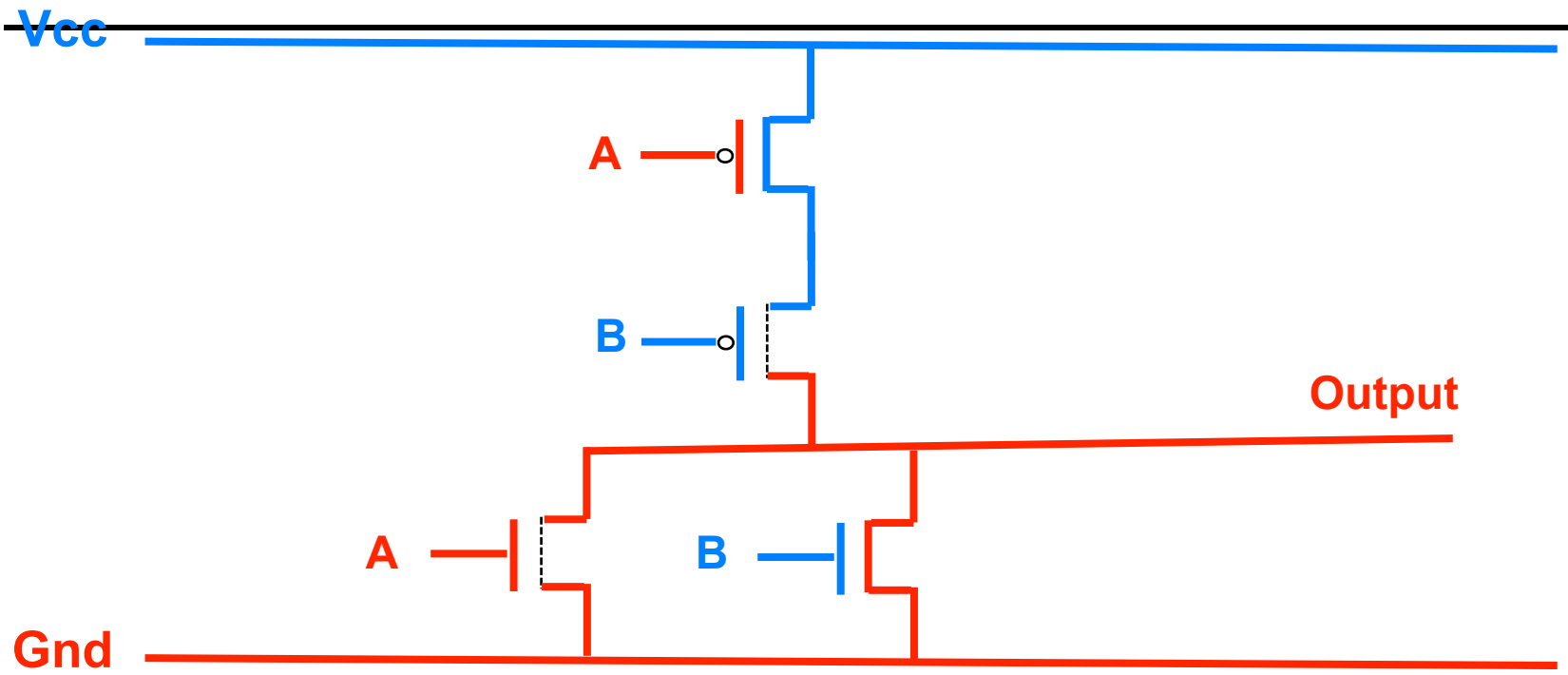
(why?)

The NOR Gate



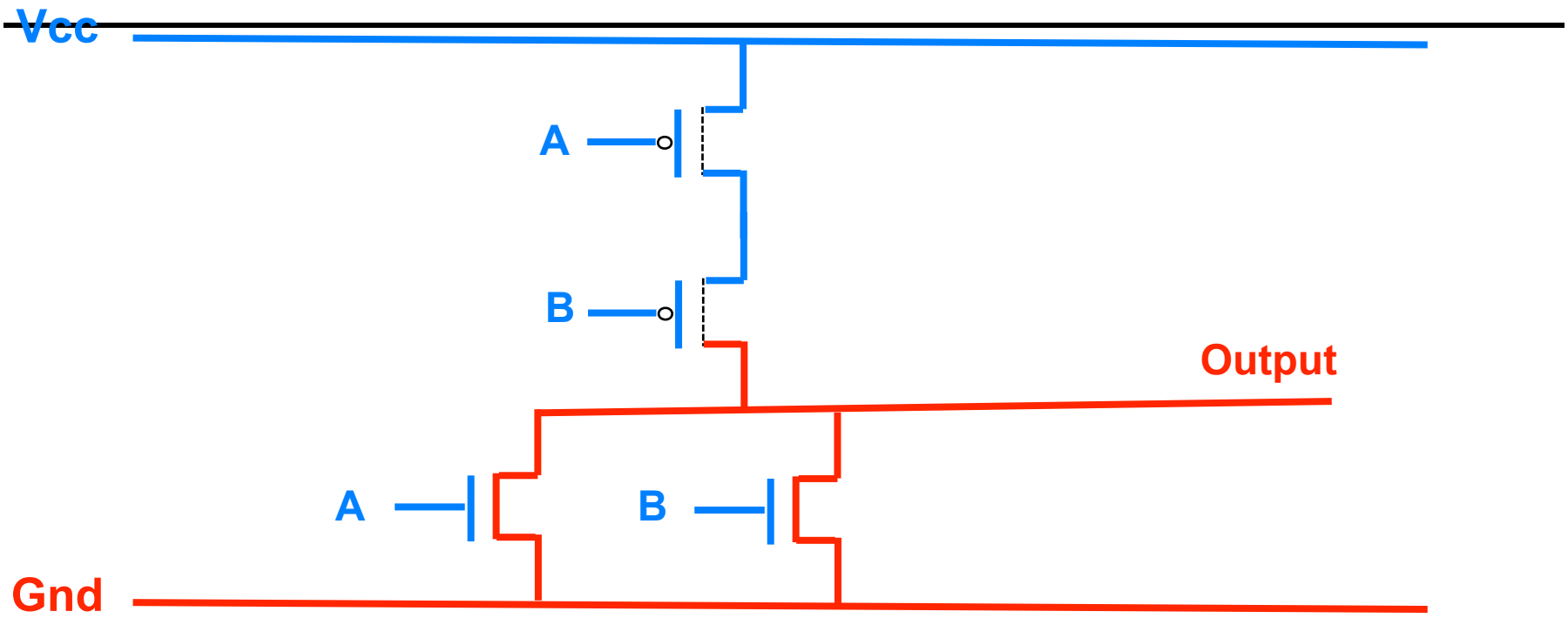
- NOR Gate
  - PMOS in series (both A and B must be 0 to get 1)
  - NMOS in parallel (either A or B at 1 results in 0)
- Side note: real chips have several layers to route wires
  - 3D drawing is hard, just label inputs

The NOR gate

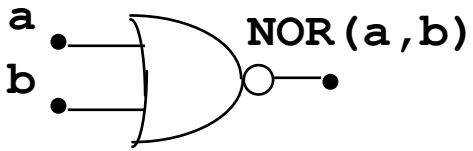


- NOR Gate
  - Same gate, just changed B's value to 1
  - Now output = 0
  - PMOS connected to B resists, blocking connection to Vcc
  - NMOS connected to B conducts, forming connection go Gnd

The NOR gate



- NOR Gate
  - Same gate, now change A to 1
  - Output stays at 0
  - Two connections to Gnd, but that's fine

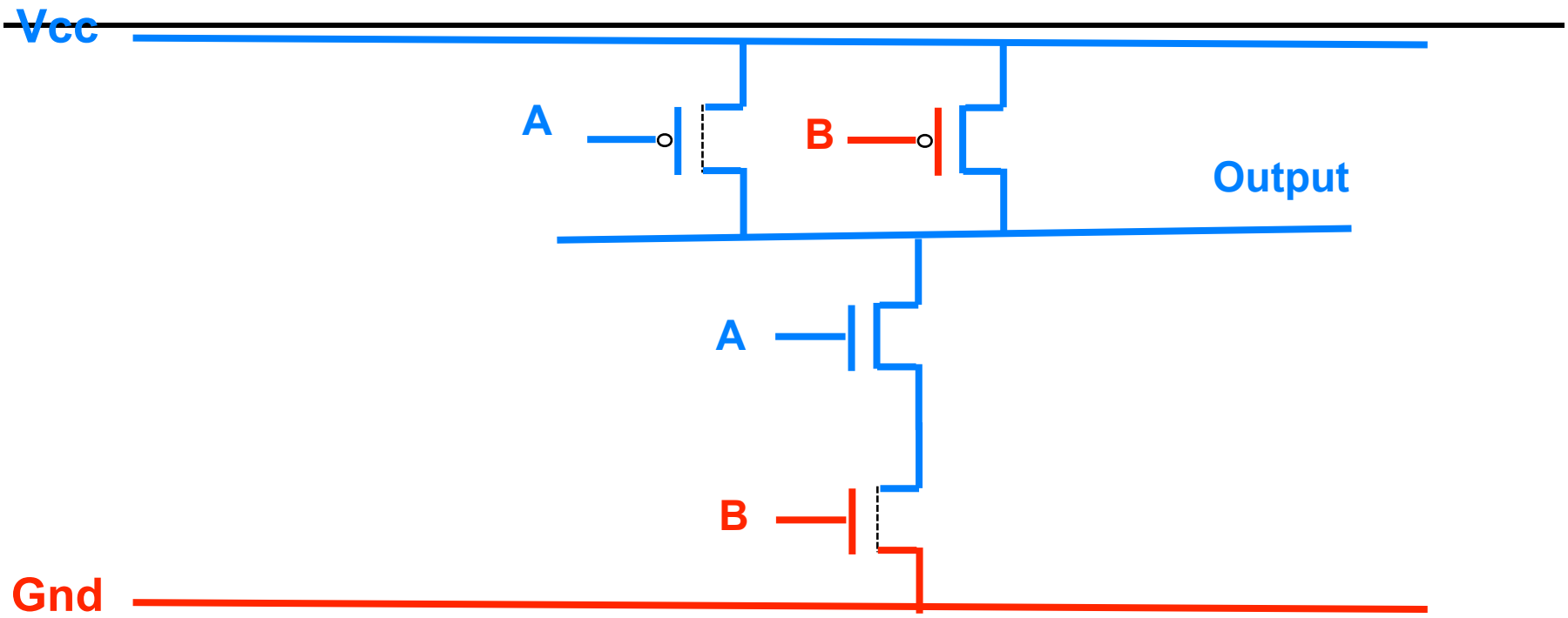


Let's build a more interesting gate

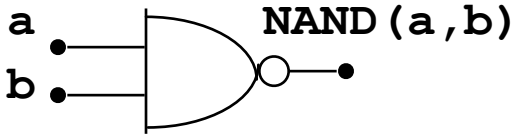
- I'll let you all try a 2-input NAND gate
  - Here is a **truth table** for NAND

A	B	Output
0	0	1
0	1	1
1	0	1
1	1	0

The NAND Gate

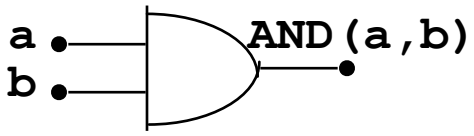


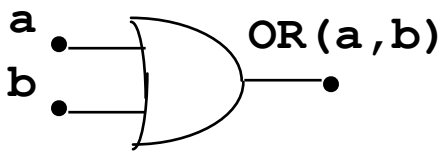
- The NAND Gate
  - PMOS in parallel (either at 0 results in a 1)
  - NMOS in series (both at 1 results in 0)

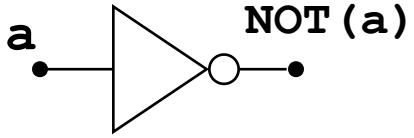


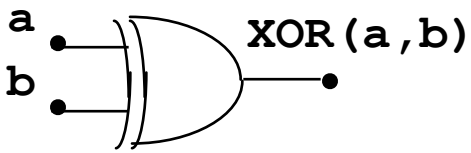
Boolean Gates

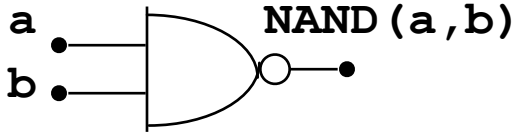
- Actually a bunch of standard logic gates:

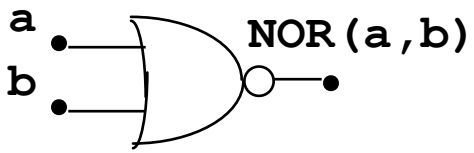


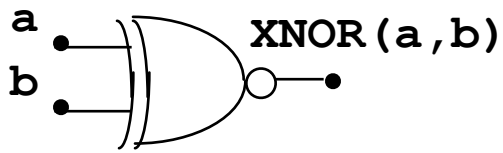








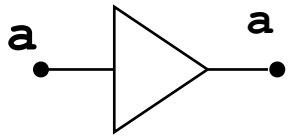




How to keep them all straight?

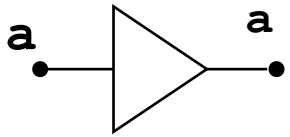
Drew's Guide to Remembering your

- This one looks like it just points its input where to go
  - It just produces its input as its output
  - Called a buffer
  - (won't really worry much about these)

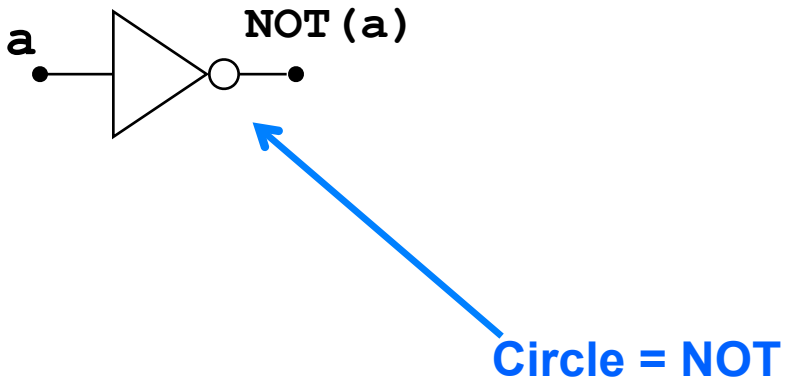


Drew's Guide to Remembering your

- This one looks like it just points its input where to go
  - It just produces its input as its output
  - Called a buffer

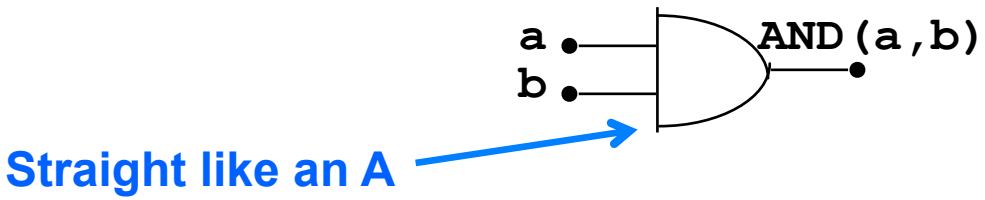


- A circle always means negate

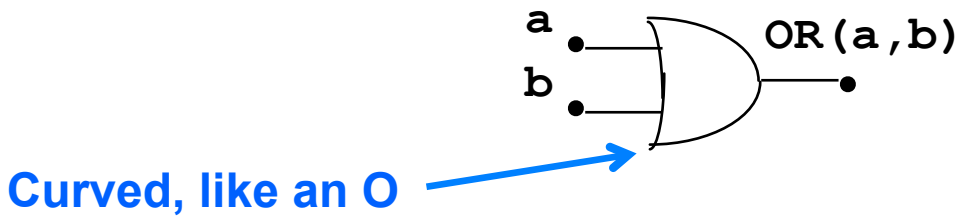


Drew's Guide to Remembering Your

- And Gates have a straight edge, like an A (in AND)

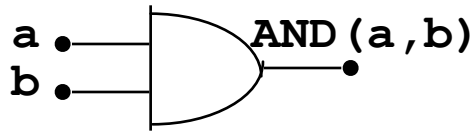


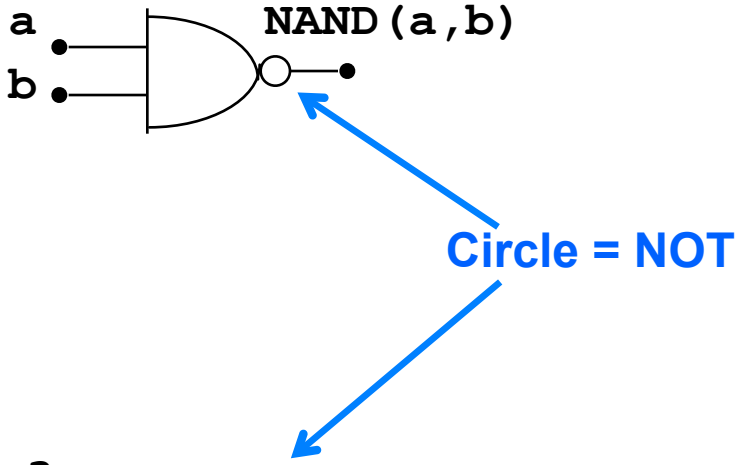
OR Gates have a curved edge, like an O (in OR)

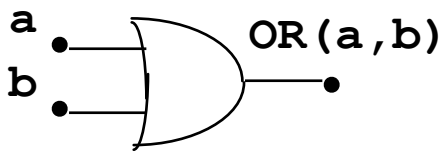


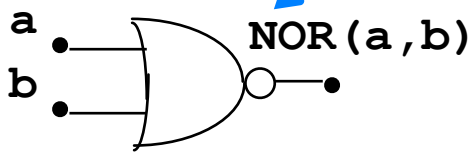
Drew's Guide to Remembering Your

- If we stick a circle on them…





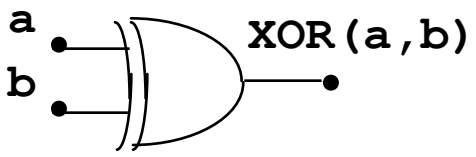




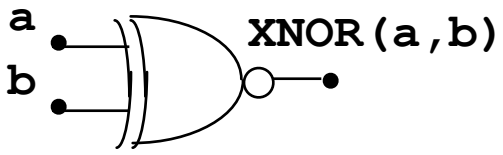
- We get NAND (NOT-AND) and NOR (NOT-OR)
  - NAND(a,b) = NOT(AND(a,b))

Drew's Guide to Remembering Your

- XOR looks like OR (curved line)
  - But has two lines (like an X does)

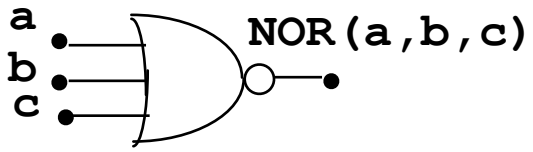


- Can put a dot for XNOR
  - XNOR is 1-bit "equals" by the way



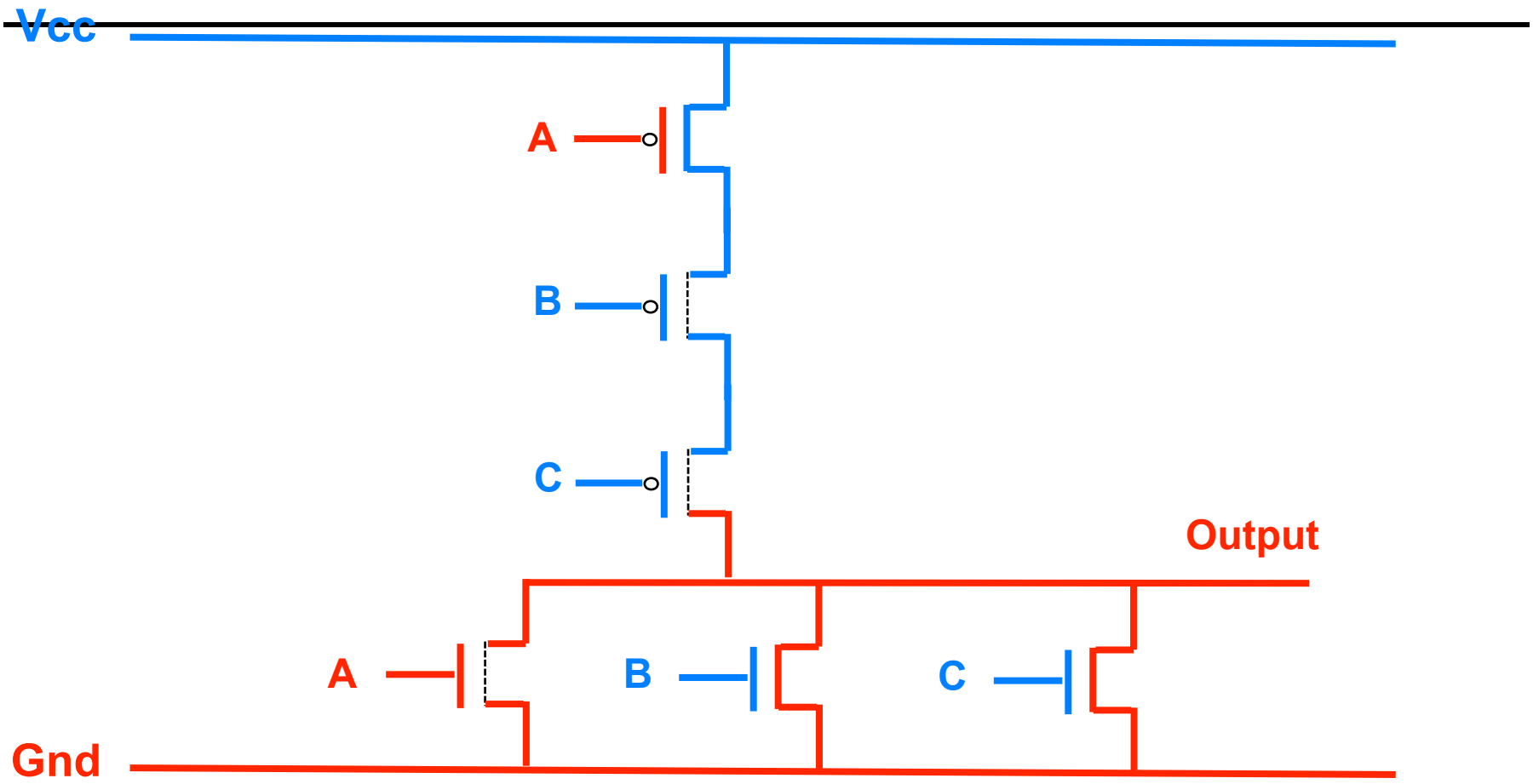
Multi-input gates

- So far gates have had 1 or 2 inputs
  - Can have more, though typically stop at 3 or 4
  - Symbols stay the same, just have more input lines



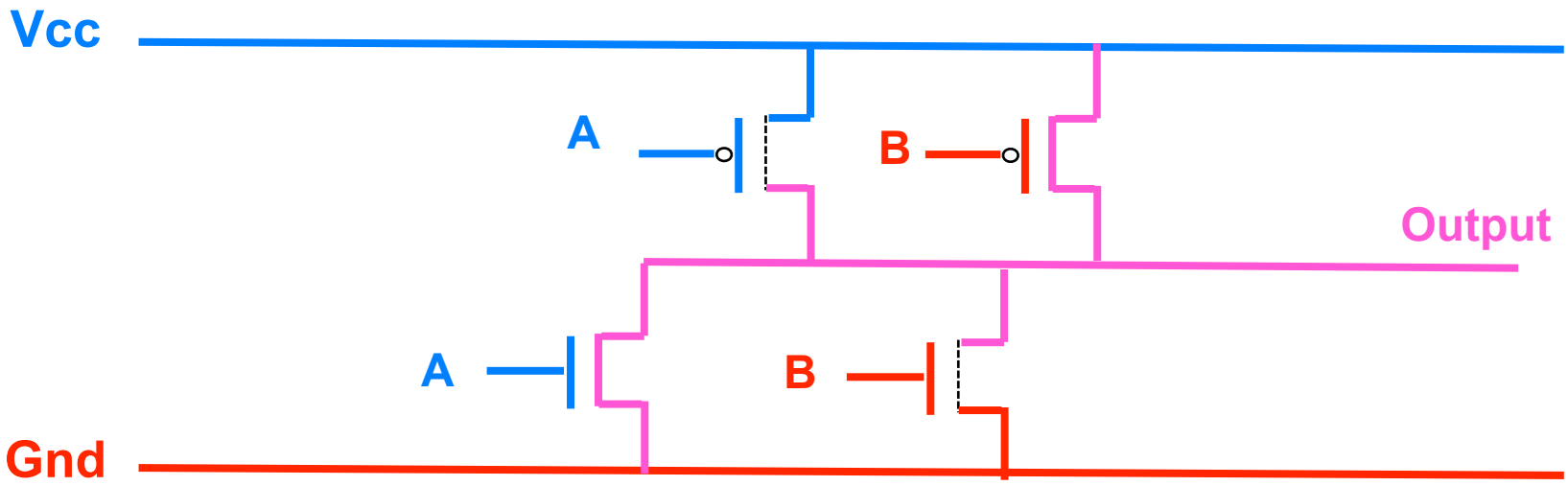
A	B	C	Ouput
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Three input NOR Gate



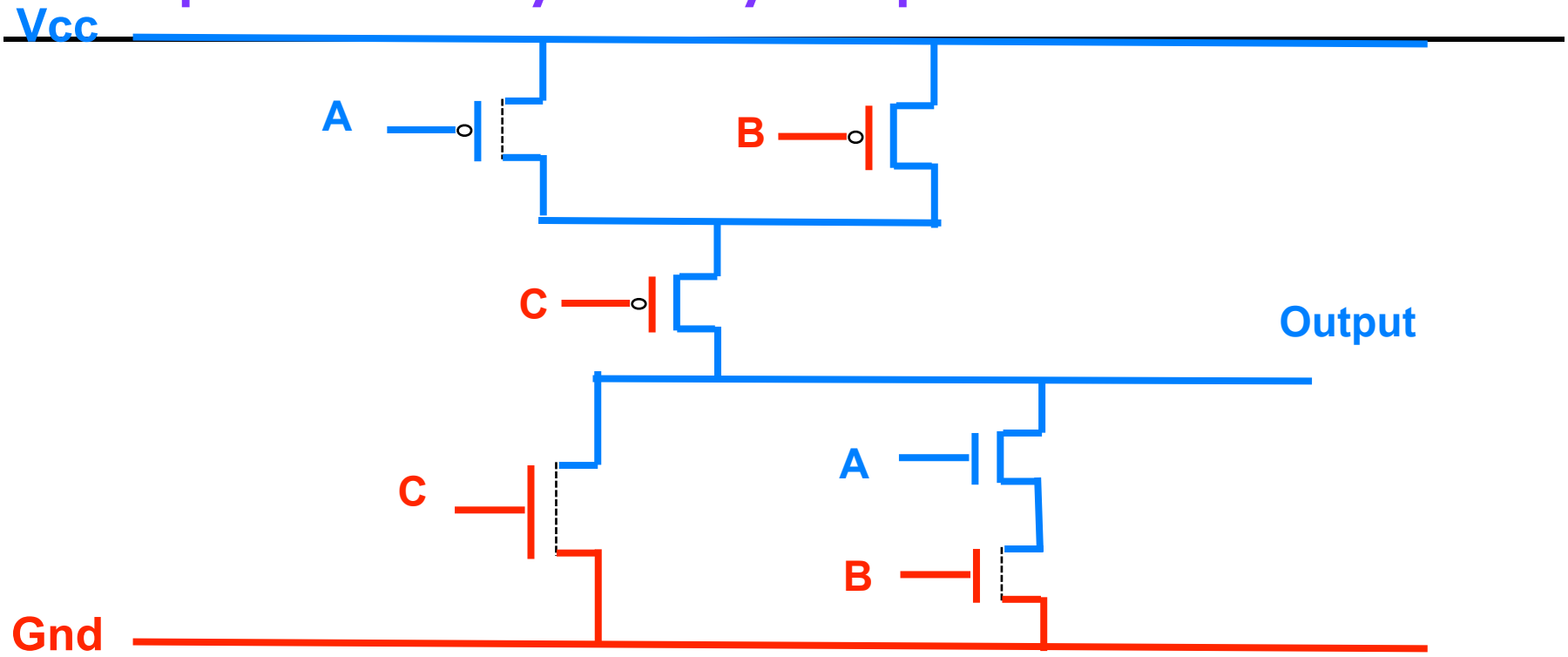
- Similar to two input, more transistors
  - Slightly slower

Complimentary: very important



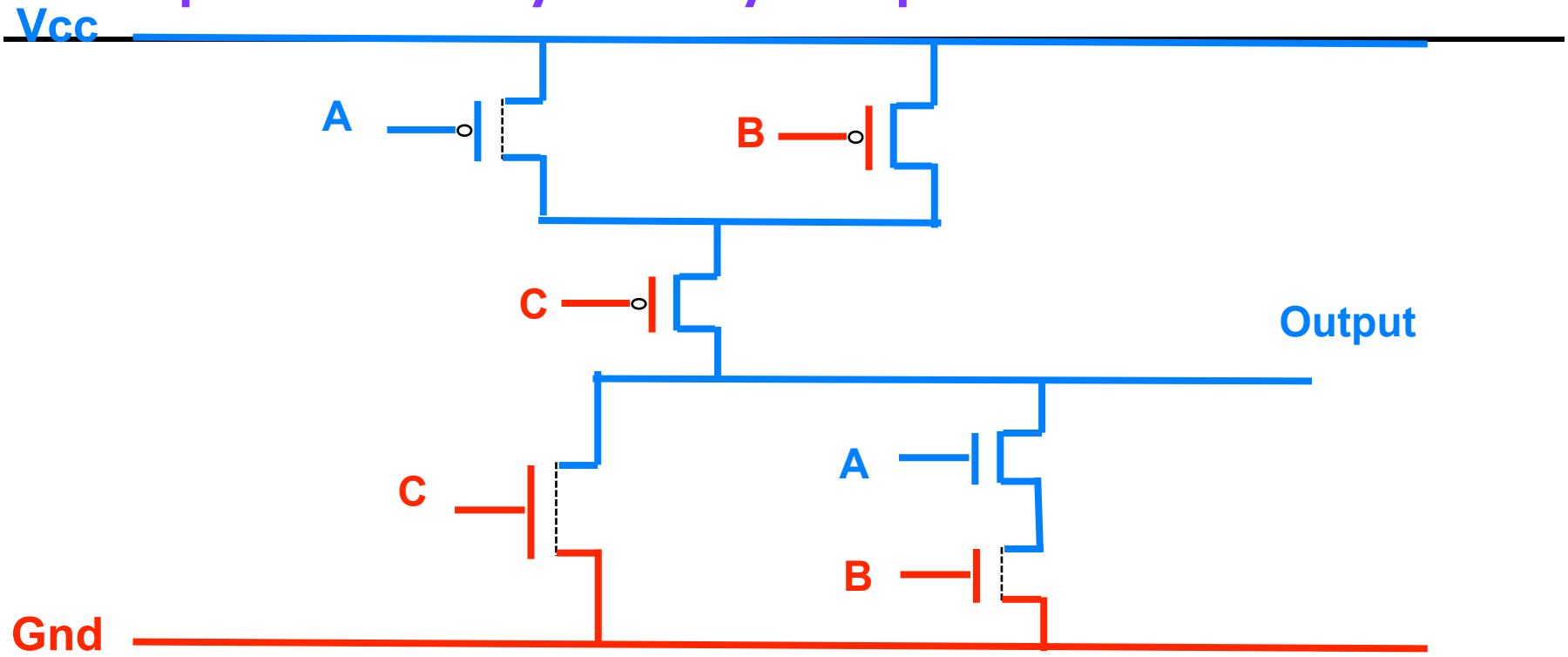
- Complementary nature: very important
  - Without it, we have a problem
  - Here: both PMOS and NMOS in parallel…
  - A=1, B= 0 (or A =0, B=1) forms short-circuit
    - Chip catches fire ☹

Complimentary: very important



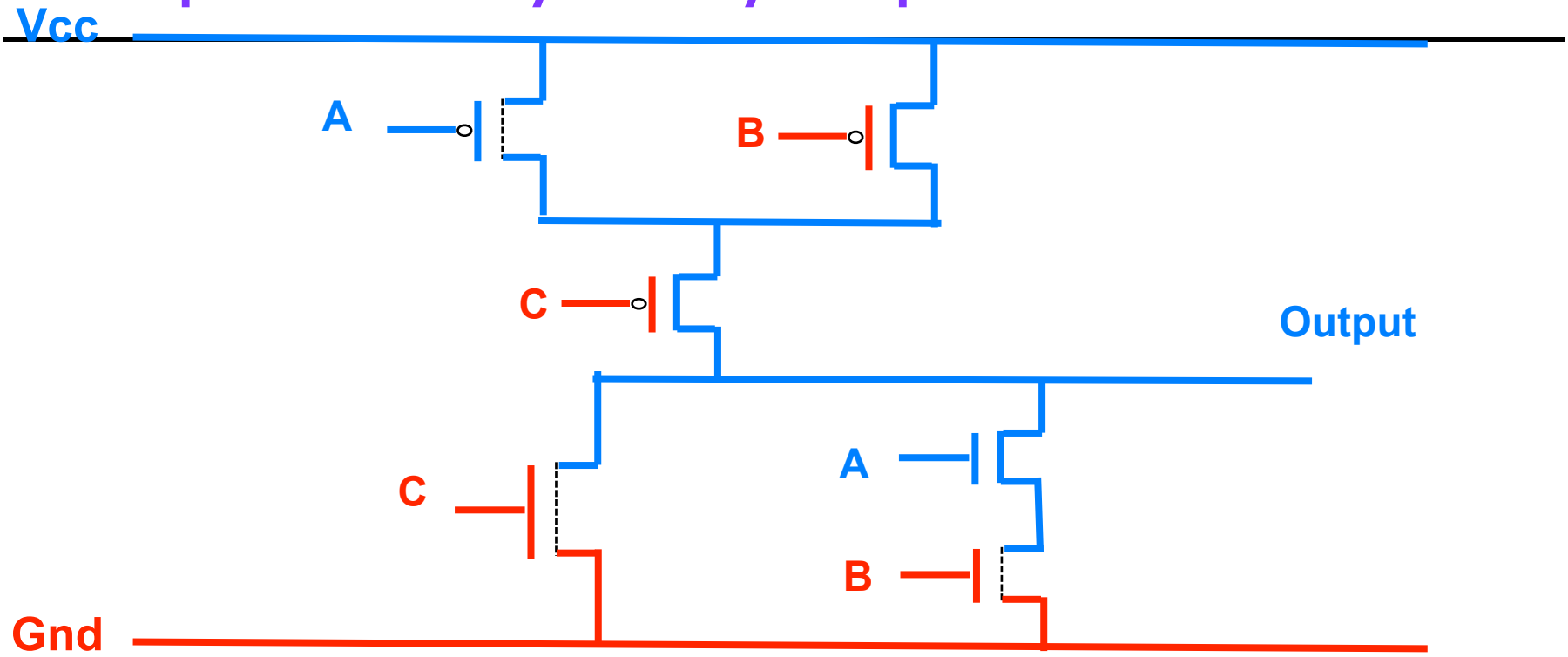
- With more than 2 inputs, can get very complicated
  - Are the PMOS and NMOS complimentary here?
  - We can go the other way: transistors -> formulas
  - Check if formulas logically equivalent

Complimentary: very important



- Everyone take a second to write down the formulas
  - PMOS: NOTs on inputs
  - NMOS: NOT around the outside

Complimentary: very important



- Everyone take a second to write down the formulas
  - PMOS: ((Not A) or (Not B)) and (Not C)
  - NMOS: Not (C or (A and B))

    = (Not C) and (Not (A and B))

    = (Not C) and ((Not A) or (Not B))

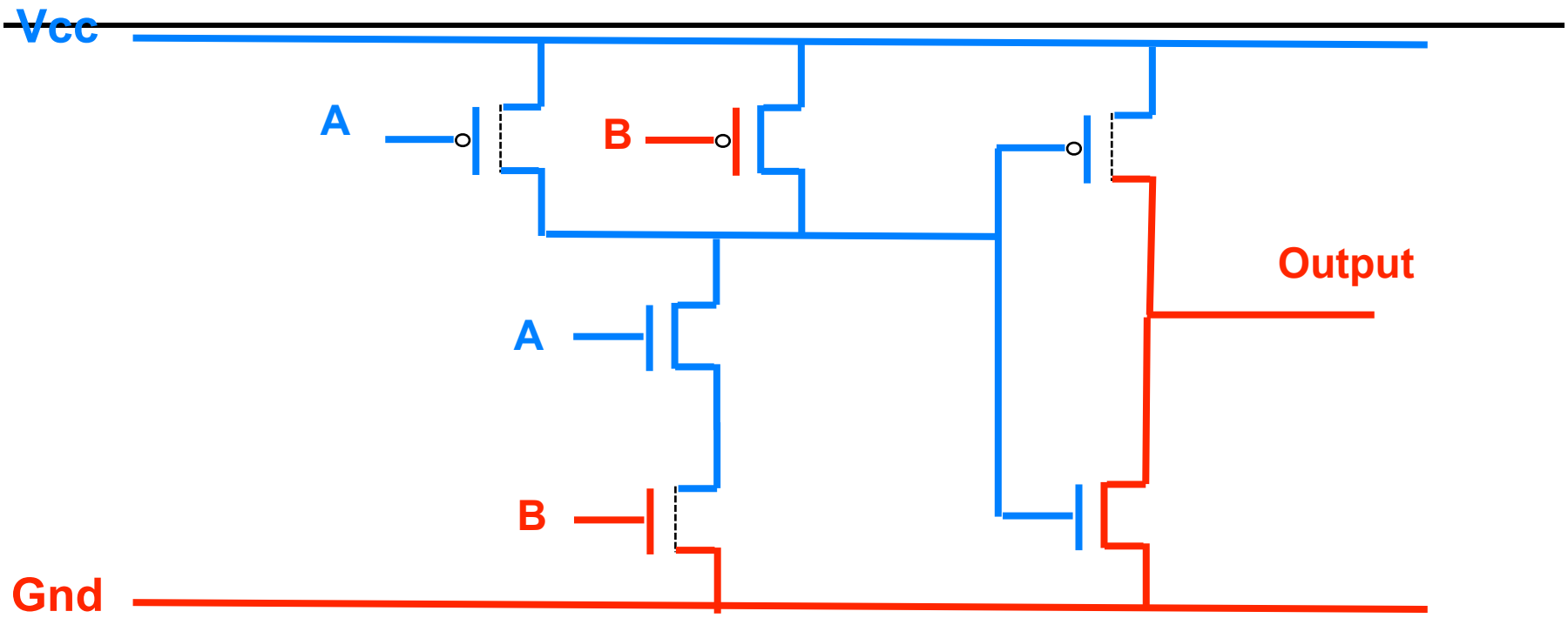
What about… AND?

- Saw and did NAND, but what about AND?
  - Truth table on the right…

A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1

  - Trying to do this causes problems
    - PMOS formula: NOTs on inputs
    - NMOS formula: NOT around outside
    - … can't seem to find a formula which works (need more NOTs):
    - (**Not** (**Not** A)) and (Not (**Not** B))
    - **Not** ((**Not** A) or (**Not** B))
  - AND gate is really a couple gates squished together
    - Not (Nand (A,B))
    - Nor(Not A, Not B)

The AND Gate



- The AND gate
  - A NAND gate followed by a NOT gate
  - Also a good example of how gates connect together
    - Output of one gate goes to input of another

Speaking of transistors…

- Moore's law:
  - Transistor density doubles roughly every 18 months
  - Has been going on for many years (~1970)
    - Self-fulfilling prophecy?
    - Ending soon? …. We've heard that before, but….
  - Commonly stated as "computers get faster"… why?

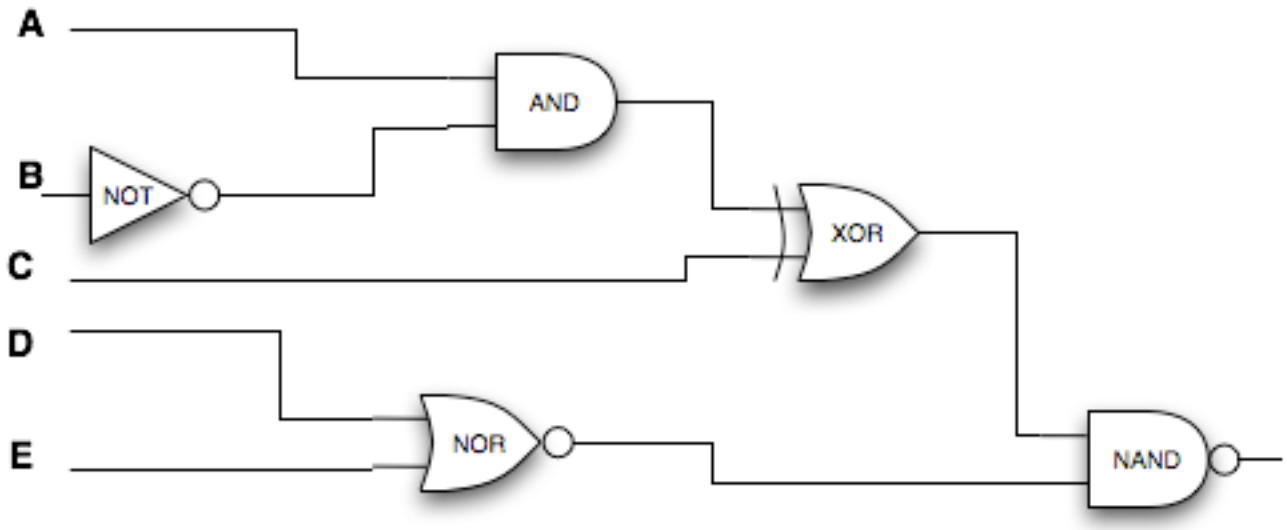
Moore's law and speed

- How are size and speed related?
  - Historically: clock frequency ("How many MHz/GHz")
    - How fast do the transistors switch (more on this later)
    - Has leveled off: Diminishing returns, power in-efficient,…
  - Now: put more in same chip area
    - Larger caches
    - Better (bigger) predictors
    - …
  - Future: ??????
    - Significant concern among micro-architects
    - Reason: power density

Power and Energy

- We won't focus too much on power and energy, but…
  - Very important concerns these days, so at least some mention
  - Energy costs money (power bills + cooling)
    - Water analogy: think total water pushed in/sucked out of system
  - Power: energy per time
    - Water analogy: How fast are we pumping in water
    - Power => Heat. Heat must be cooled, physical limits on cooling
- Previously:
  - Smaller transistors => lower voltages => less power/transistor
  - More transistors / area
  - Power density (power/area) held roughly constant

Going forwards: Logic Gates



- Going forwards, will mostly design from gates
  - Abstract away transistor level implementation
  - More on this next time…

Next Time…

- Next time, we'll delve into Combinatorial Logic
  - Putting gates together to do useful things
- Reading:
  - Continue to read Chapter 1 and Appendix C
- Homework:
  - Homework 1 will be out soon
  - Keep an eye on Piazza
  - I'll also post the code for the lab door on Piazza soon
- Recitation: Friday (here)
  - Bring laptops