SIGARCH Meeting

Introduction

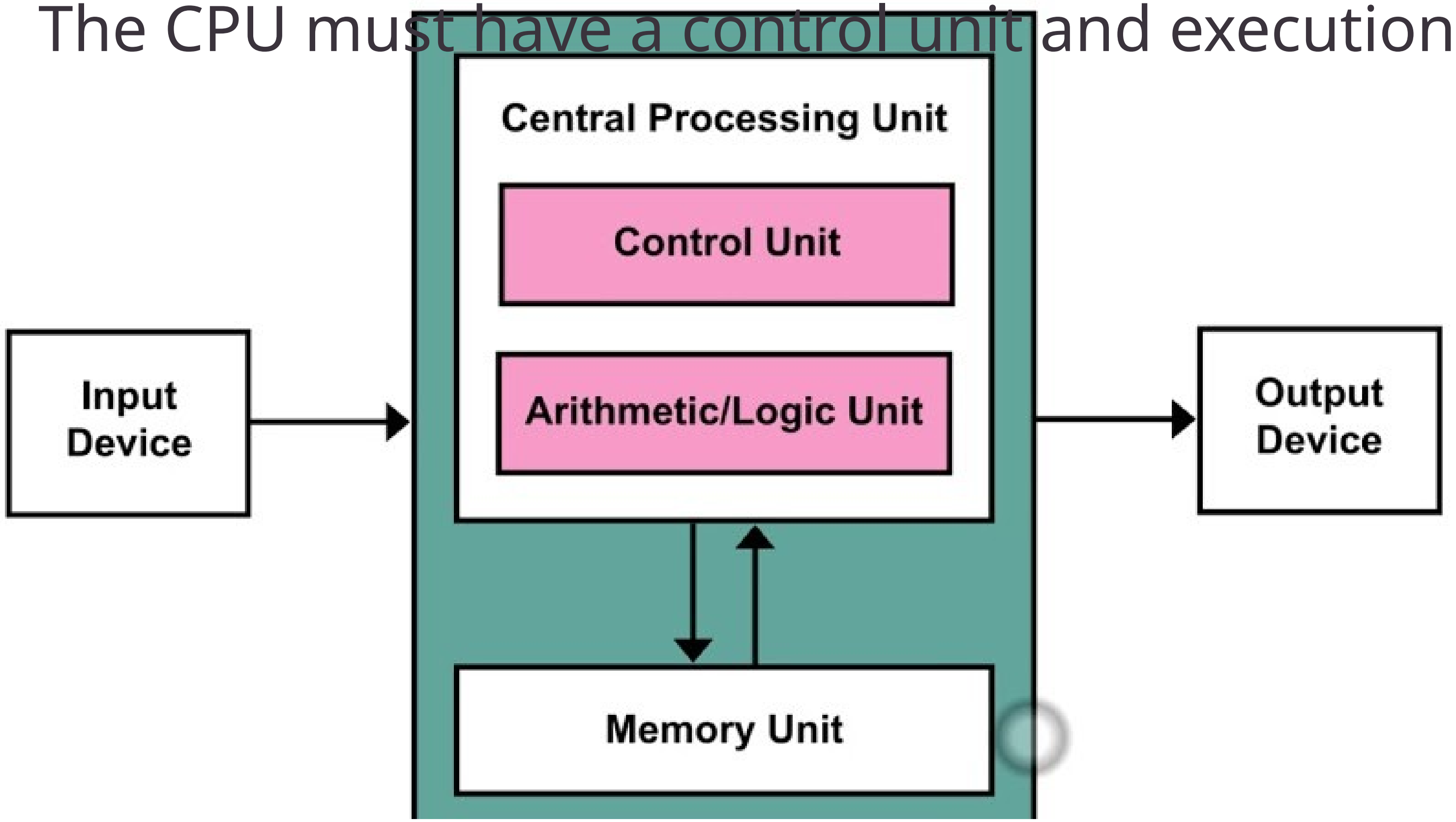
September 10, 2025

What is Computer Architecture?

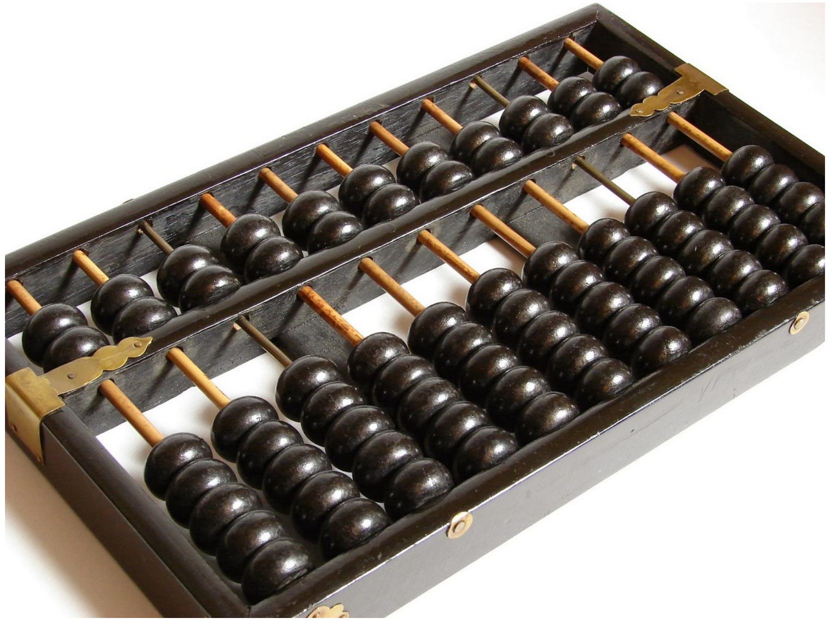
- Computer Architecture is the science and art of designing the structure of a computer
- Distinction:
  - Architecture: the highest-level design decisions
    - The memory it uses
    - The type of processor
    - The choice of interface and instructions it will use (ISA)
  - Microarchitecture: the implementation of a particular computer chip and the feature choices it makes
    - How the datapath is laid out
    - What the cache size is
    - How many execution units it has

First, a High-Level View

- The Von-Neumann Computer
  - Consists of input devices, output devices, memory, and a CPU
  - The CPU must have a control unit and execution units

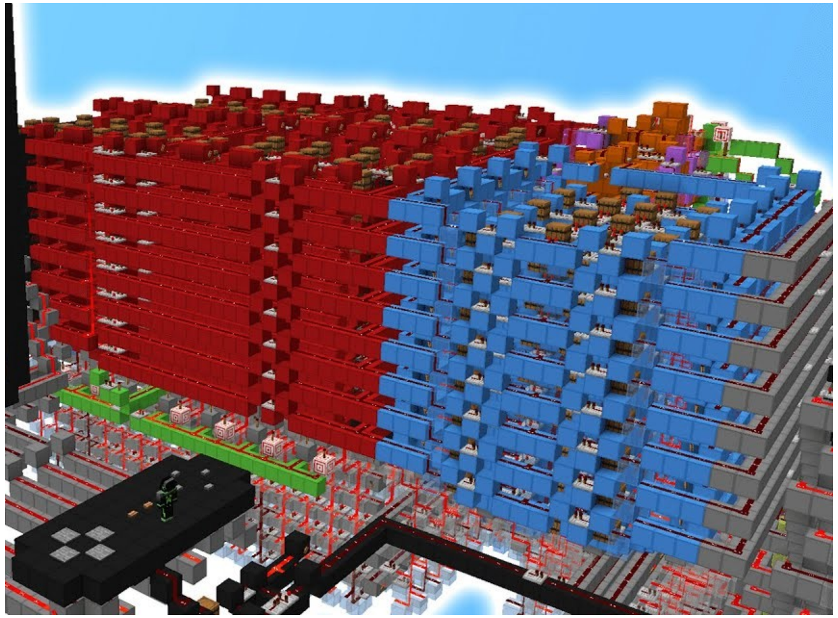


How are computers implemented?



Abacus

Analog Water Computer



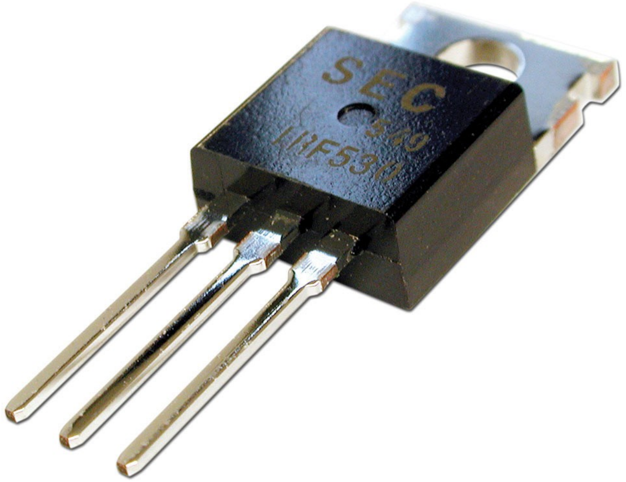
Minecraft

Analog Vs. Digital Computers

- Analog Computers
  - Use continuous signals (voltage, current, fluid, etc)
  - Example: water computers, AI voltage ASICs
  - Strength: very fast for specific, particular problems
  - Weakness: Hard to scale, imprecise, noise-sensitive
- Digital Computers:
  - Use discrete signals (almost always binary)
  - Uses arbitrary limit between what is a 1 and what is a 0
  - Examples: Abacus, all CPUs, GPUs, etc,
  - Strengths: High precision, easy to replicate and scale, reliable with error correction
  - Require more parts per computation

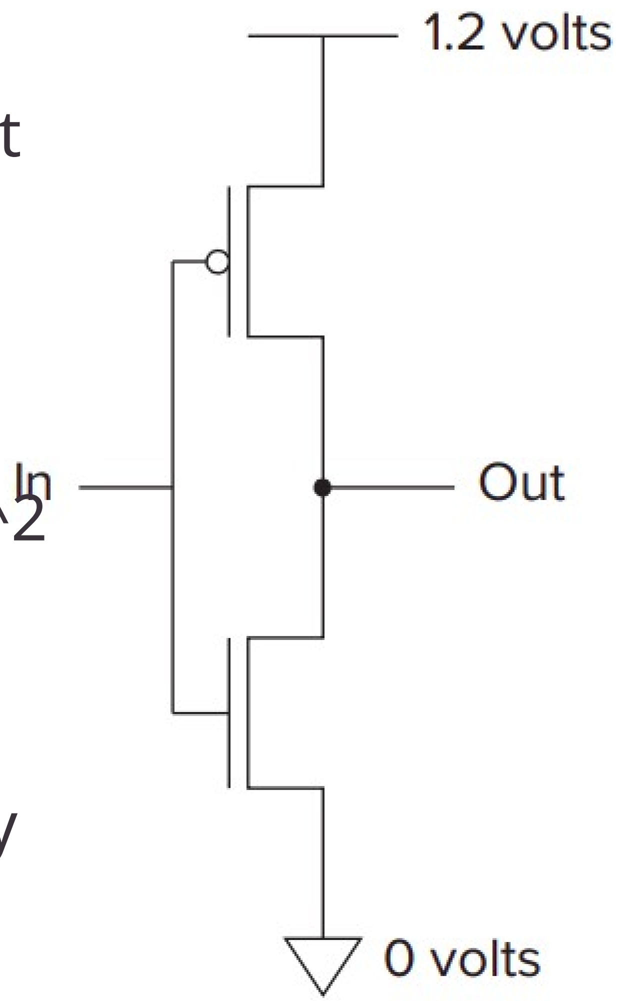
Today, We Use Transistors

- The transistor is the backbone of modern computing
  - It can be tiny, fast, and require little power, all key to making a great computer
- The transistor has 3 nodes
  - The source, or where the electrical current flows from
  - The drain, or where the electrical current flows to
  - The gate, or the switch for allowing electrical current to flow



Modern Silicon

- In 1954, the first silicon computer chip was fabricated at Bell Labs
- Ever since, we have used silicon due to its ability to be imbued with transistors using a process called “Photolithography”
- Today, we can fit about 224 million transistors in a mm^2 of silicon (3P TSMC)
- They can also switch states from passing current to stopping current very quickly
  - Switching speeds are held a lot closer to the chest by foundries
- Two types of transistors: N-Type and P-Type
  - N-Type allow current to pass when gate voltage is high
  - P-Type allow current to pass when gate voltage is low

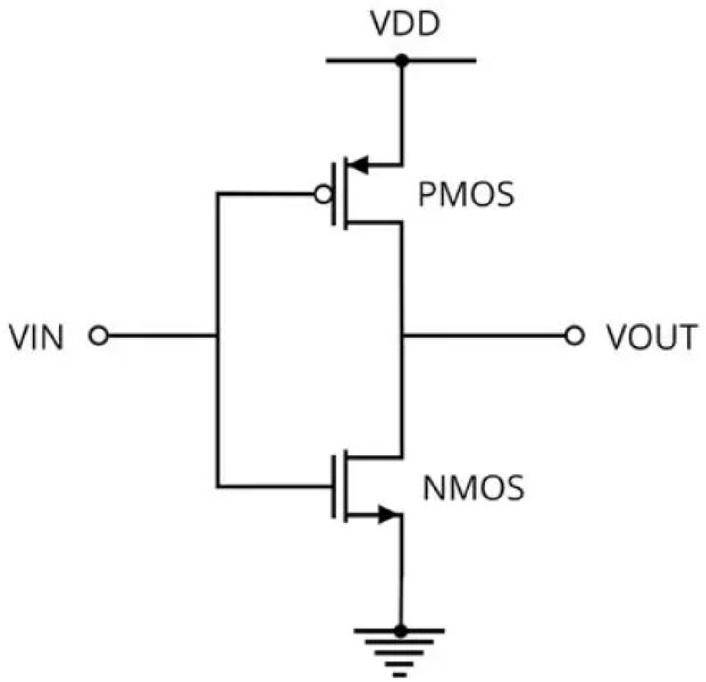


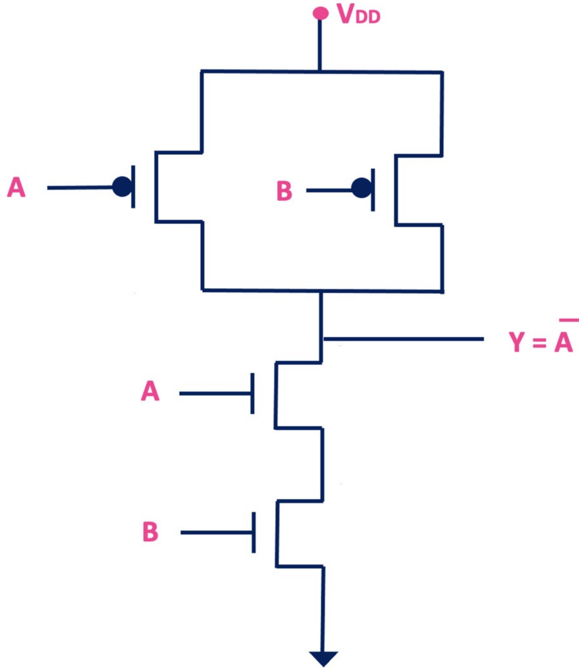
How can we combine transistors?

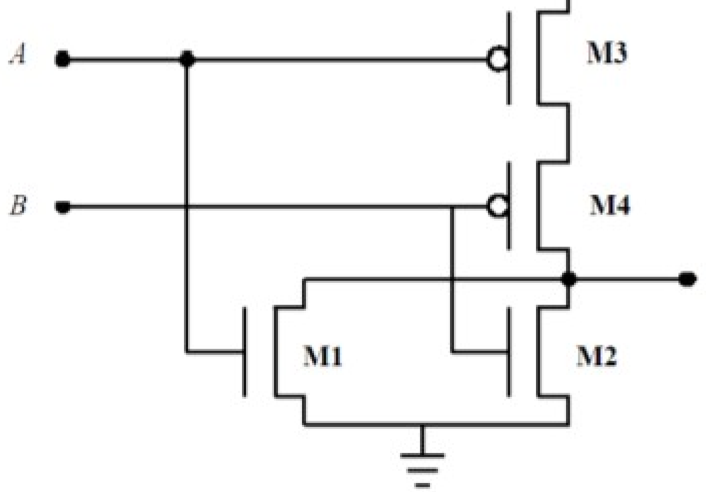
- We pair N-Type and P-type together through a *Complimentary* system
  - Why? Complicated voltage reasons.
- In most gates, there are equal numbers of N-Type and P-type

e

ther



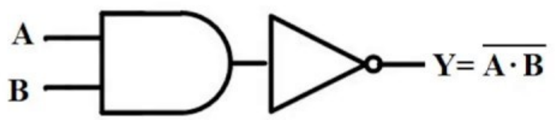


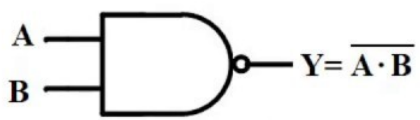


Common Basic Gates

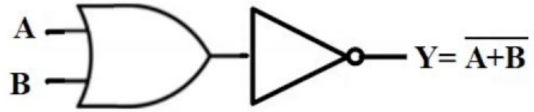
- The three most common gates are NAND, NOR, and NOT
- NAND and NOR require 4 transistors, while NOT requires 2
- NOT inverts the input A, from 0 -> 1 or 1 -> 0
- AND goes high (to logical 1) when its inputs are all high
- OR goes high when any of its inputs are high
- NAND is the inversion of the output of an AND gate
- NOR is the inversion of the output of an OR gate

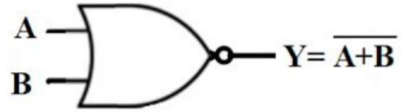
NAND → NOT of AND





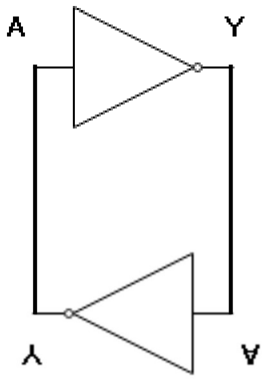
NOR → NOT of OR

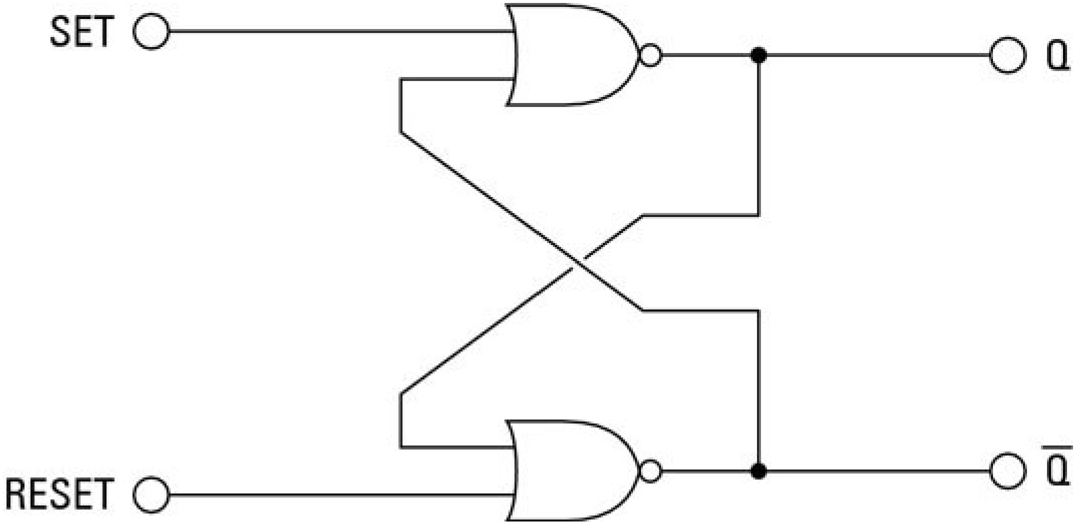




Silicon Data Storage

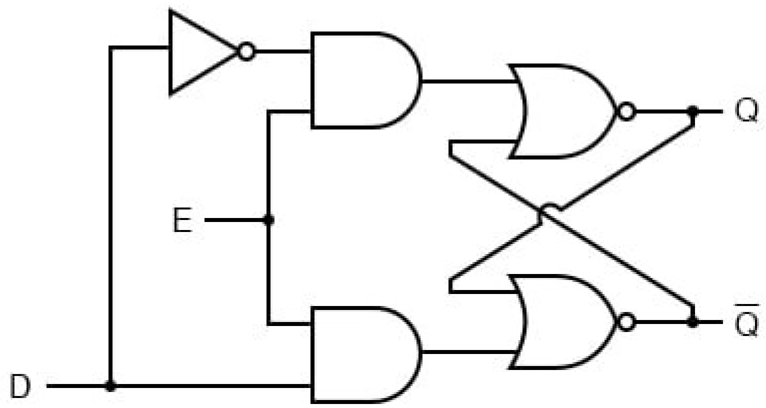
- We've established how we can create Boolean functions with gates
  - But gates are always *combinational* – their outputs depend only on their previous inputs
- But we need to store data
- How can we do that?
- NOT Loop





Silicon Data Storage

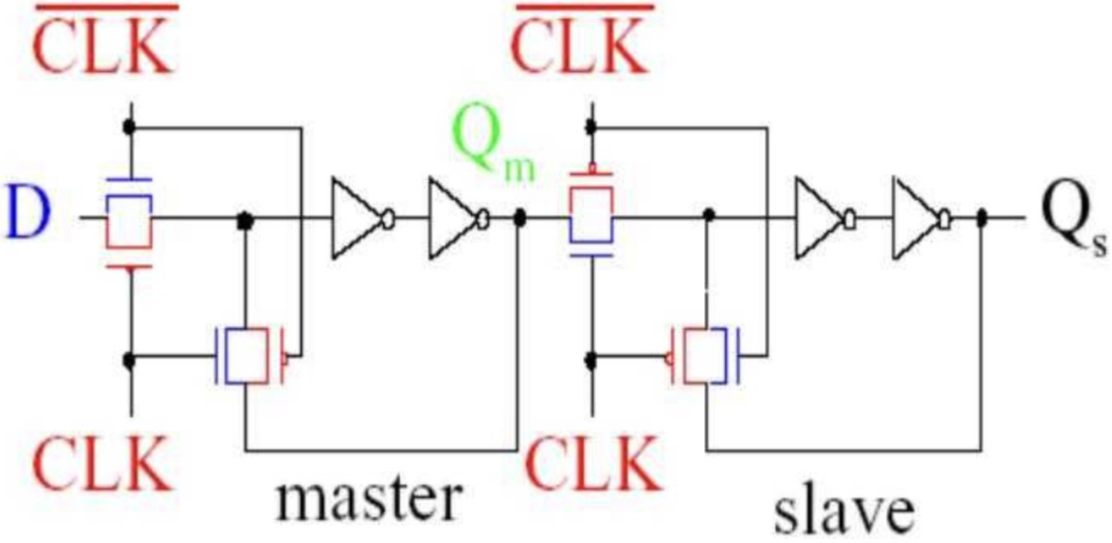
- For data storage, we need to be able to write a value **D**, and decide when to write that value with an enable signal **E**
- For this, we use what we call a D latch – a latch which has an input and an enable signal to take in a new value



E	D	Q	$\overline{Q}$
0	0	latch	latch
0	1	latch	latch
1	0	0	1
1	1	1	0

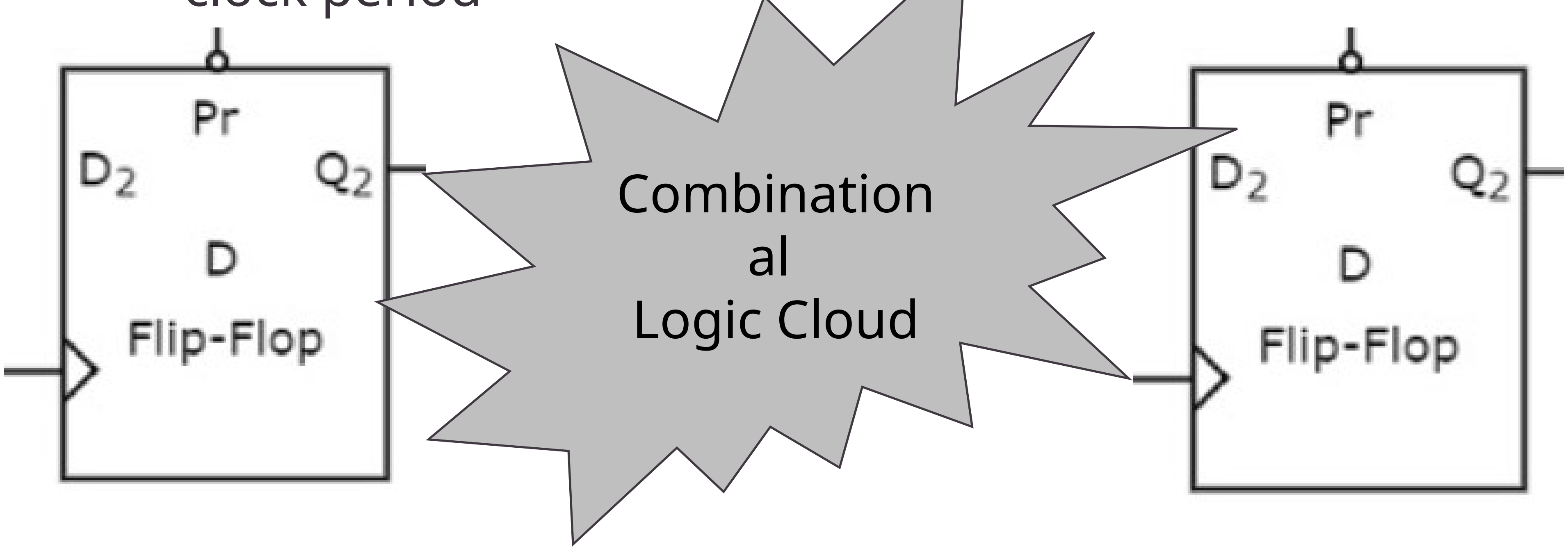
Registers

- Is there a way we can hold data steady for a certain period of time, and then have a way to shift *new* data in?
  - Yes!
- We can use two latches to *trap* data between two latches
- By using an oscillating signal, we can keep data steady and shift new data in at a fixed
- These are called regis



Sequential vs. Combinational Circuits

- The combination of registers and basic gates form (for the most part) create the circuits in modern computer chips!
- We put registers in between logic clouds (basic gate combinations) and shift data between the two registers at every clock period



Clock speeds

- The clock speed is the rate at which the registers take in new data
- It is usually limited by how fast the transistors are at switching states
- The faster the clock speed, the faster the data moves around your chip, and thus the faster your chip
- The bigger your combinational cloud, the slower the clock speed must be
- Your computer chip's clock speed is limited to *the longest path which is governed by that clock*

Review

- Computers are built from electronic switches called transistors
- Transistors combine to form logic gates, which implement Boolean functions
- By feeding gates back into themselves, we can store information with latches and flip flops
- A clock signal controls when registers capture new data and keeps the system in sync
- Logic and registers together form the synchronous circuits of a processor
- At the next lecture meeting on 9/24, we will discuss how we can use these building blocks to implement a Von Neumann computer
- At our first workshop meeting on 9/17, we will discuss how we can write circuits with these building blocks using RTL and program an FPGA