# CPE 221: Computer Organization

THE UNIVERSITY OF
ALABAMA IN HUNTSVILLE

## 01 Introduction

rahul.bhadani@uah.edu

Rahul Bhadani

# Course Logistics

**Lecture:** Mon/Tue 04:20 PM – 05:40 PM SST

**Location:** SST 105

**Prerequisites:** EE 213 - Electrical Circuit Analysis I, and MA 238 – Applied Differential Equations

**Office Hour:** Mon/Tue/Wed 2:00 PM - 4:00 PM
ECE 217-H, and by appointment

# Grading Details

| | | | | |
|---|---|---|---|---|
| 90-100% | A | | Homework | 30% |
| 80-89% | B | | Attendance/Quiz/In-Class Participation | 10% |
| 70-79% | C | | Exam 1 | 15% |
| 60-69% | D | | Exam 2 | 15% |
| 0-59% | F | | Final Exam | 30% |

The percent score will be rounded to the nearest integer before assigning the final grade.

# Homework Submission Policy

- Naming convention:
  - rahul.bhadani_CPE221_hw01.zip or rahul.bhadani_CPE221_hw01.zip
- Only .pdf or .zip will be accepted. No .docx, .rar, .7z
  - Wrong format will result in zero score.
- Empty file will result in zero score.
- Late submission:
  - Each day late submission will be penalized by 10% regardless of the reasons, and after a week of late submission, 0 score will be given.
- No Bonus assignment.
- Final grading will not be curved.

# Exam Dates

Midterm 1: February 19, 2025, Wednesday

Midterm 2: April 2, 2025, Wednesday

Final Exam: May 2, 2025, Friday (3:00 pm to 5:30 pm)

# Plagiarism and LLM Use Policy

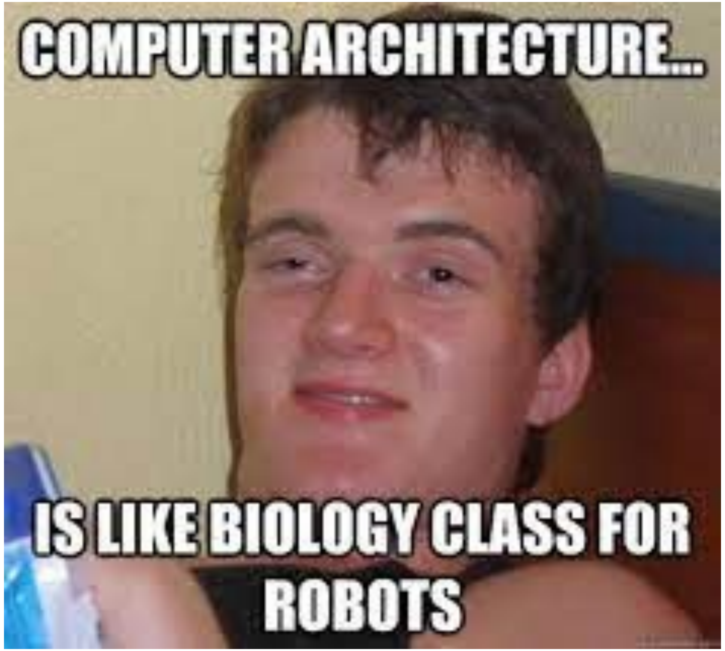
- Cheating, copying, or plagiarizing any work submi, ed for grading is strictly prohibited.
- Students found engaging in such activities may receive a failing grade and face disciplinary action, including possible expulsion.
- Use of Large Language Models (LLMs) for homework assistance is permi, ed, but students must submit:
  - Screenshots of the LLM conversation, showing prompts and responses.
  - A reflection on how the tool was used and its impact on their understanding
- Failure to comply with these requirements will result in penalties as per the university's academic integrity policy.

# Get to know.

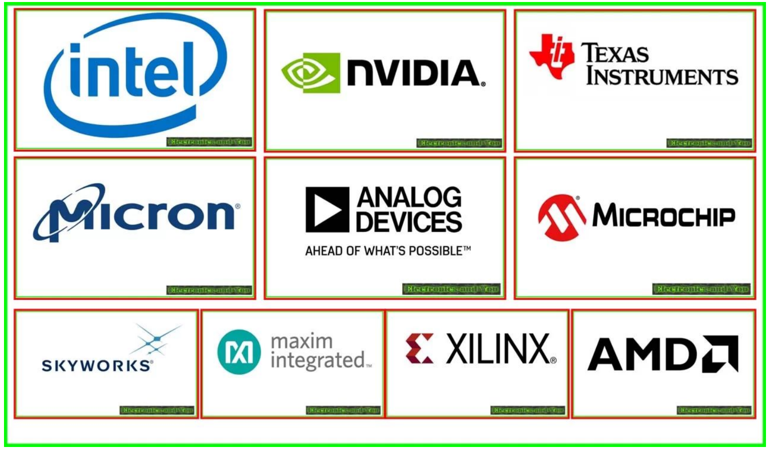
In-class activity

- Introduce yourself
- Why computer engineering
- Why do you want to take this course
- One fun fact about you that’s not on social media

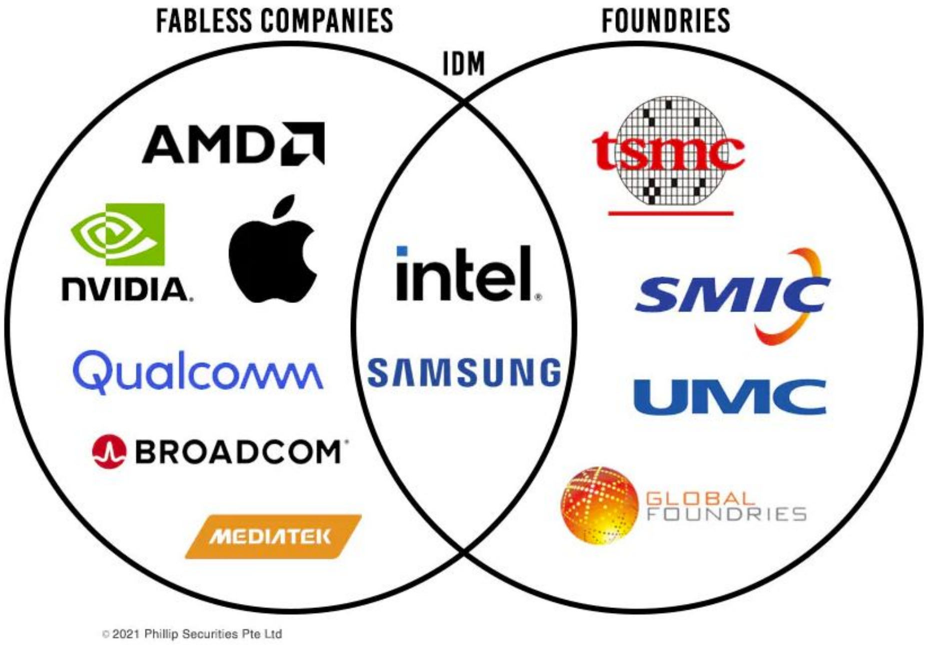
# Why study Computer Organization?



# Where can you work with skills you will learn?



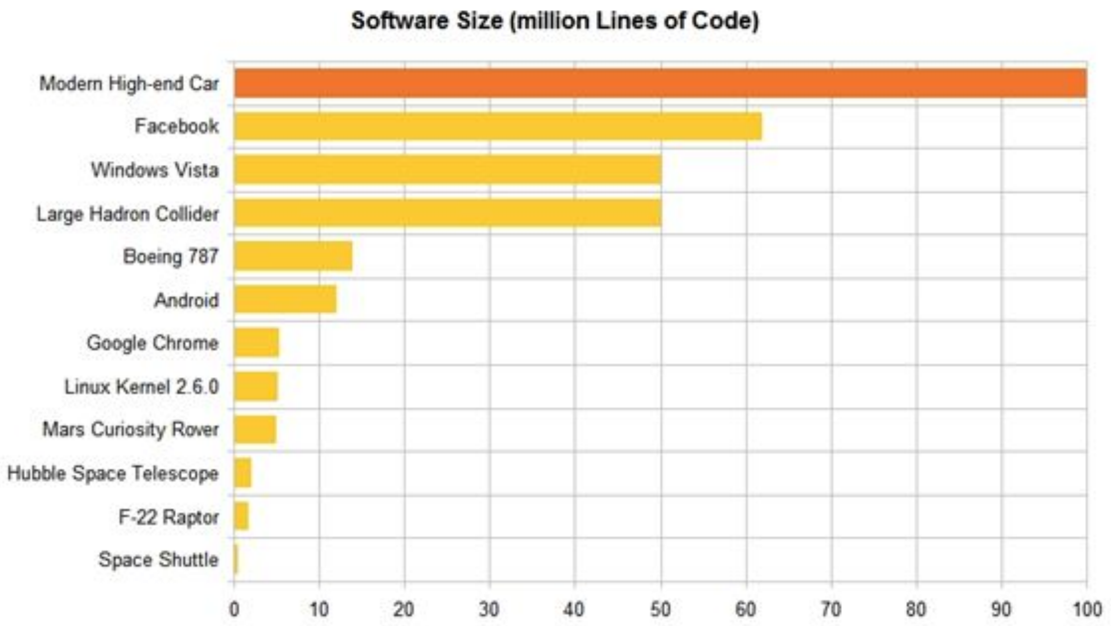

Top 20 Most Valuable Semiconductor Brands
© Brand Finance Plc 2022
1 intel $25.6bn -20%
2 tsmc $20.5bn +66%
3 NVIDIA $16.1bn +100%
4 SK hynix $9.4bn +22%
5 QUALCOMM $7.8bn +34%
6 BROADCOM $6.4bn -2.0%
7 Micron $6.3bn +34%
8 AMD $6.1bn +122%
9 ASML $5.9bn +45%
10 Texas Instruments $4.4bn -5%
11 infineon $4.1bn +32%
12 MEDIATEK $4.0bn +74%
13 Lam Research $2.9bn +17%
14 ASE GROUP $2.8bn New
15 TEL $2.5bn +16%
16 muRata Innovator in Electronics $2.2bn -8%
17 ST life.augmented $2.1bn -3%
18 NXP $2.0bn +19%
19 KLA $2.0bn +29%
20 ARROW $2.0bn +30%

FABLESS COMPANIES
IDM
FOUNDRIES
AMD
NVIDIA
Qualcomm
BROADCOM
MEDIATEK
intel
SAMSUNG
tsmc
SMIC
UMC
GLOBAL FOUNDRIES
© 2021 Phillip Securities Pte Ltd

# Many Computing Platforms


# Course Context – Software Comparisons



# Embedded Systems Everywhere


# Goal of the Course

- Multiple levels of computer operation
  - Application level
  - High Level Language(s), HLL, level(s) — CPE 211, 212
  - Assembly/machine language level: instruction set — CPE 221
  - System architecture level: subsystems & connections — CPE 221
  - Digital logic level: gates, memory elements, buses — EE 202
  - Electronic design level — EE315
  - Semiconductor physics level — EE310
- Interactions and relations between levels
  - View of machine at each level
  - Tasks and tools at each level
- Historical perspective
- Trends and research activities

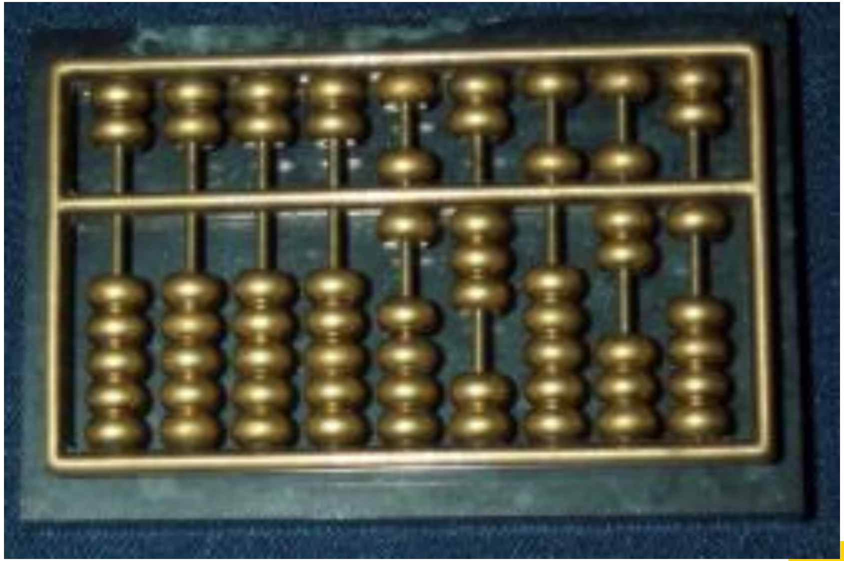
# What's computer?

Anything that conducts mathematical calculations.

# Mechanical Computers

Pascal's adding machine

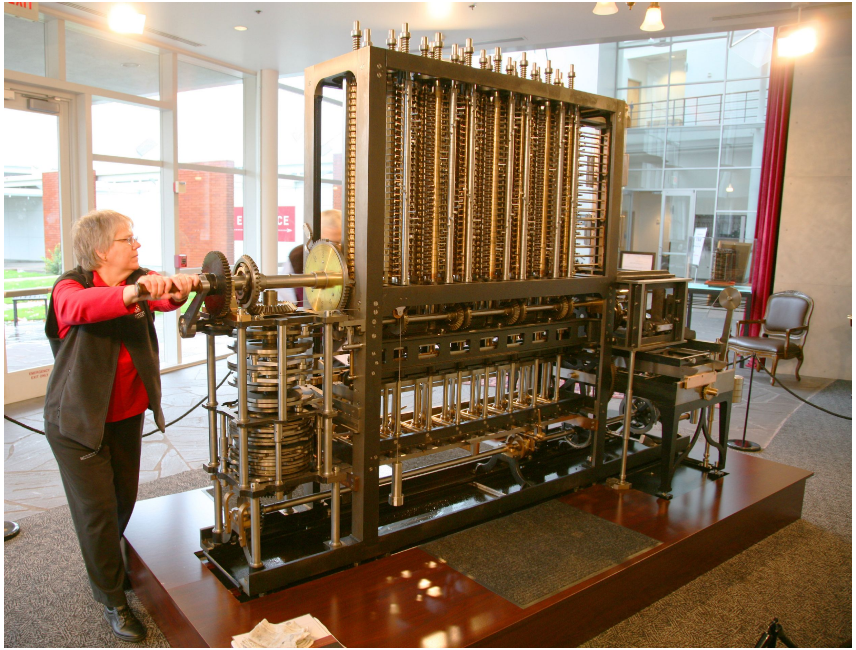
Abacus





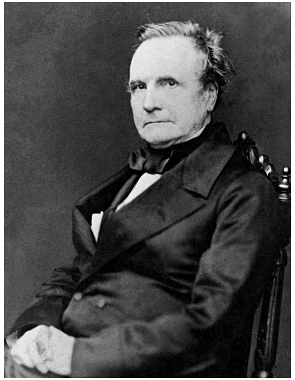
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

# Difference and Analytical Engine



# Origin of Modern Computers

- Charles Babbage
  - Known as the first computer engineer
  - Designed the Difference and Analytical Engine
  - Couldn’t get it done by himself and died before finishing it
  - Queen Victoria funded his project
  - Goal was to help in selection of winning horses in the races.



# Origin of Modern Computers

Augusta Ada King, Countess of Lovelace

Considered the first computer scientist

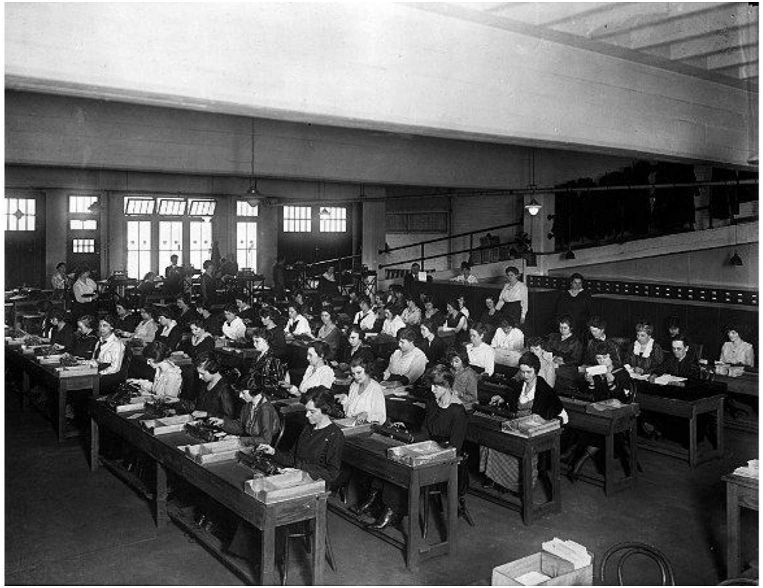
Born over 100 years before the ENIAC was entering design

Translated, documented, and analyzed the algorithm that the machine used.

# Origin of Modern Computers

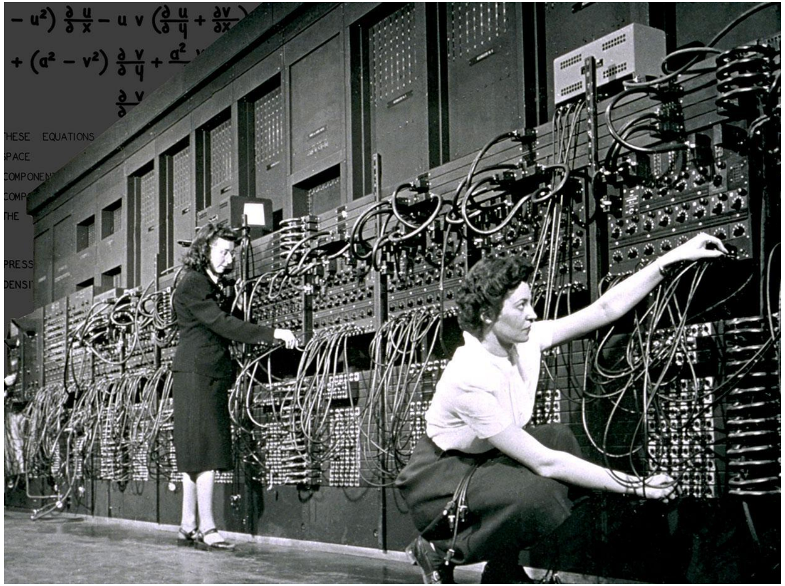
"Kilo Girls": Computing Power Used to Be Measured in 'Kilo-Girls' (https://www.theatlantic.com/technology/archive/2013/10/computing-power-used-to-be-measured-in-kilo-girls/280633/)

Responsible for gigabytes of data crunching throughout WW2

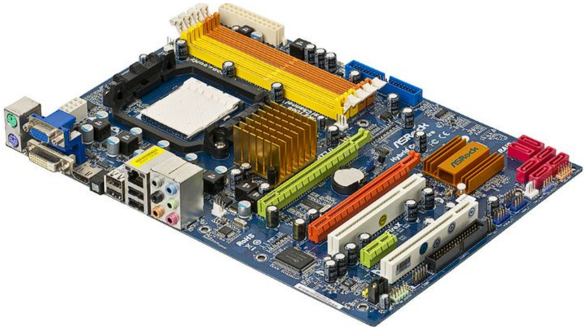


# Origin of Modern Computers

## Electronic Numerical Integrator and Computer (ENIAC)



# Modern Computers

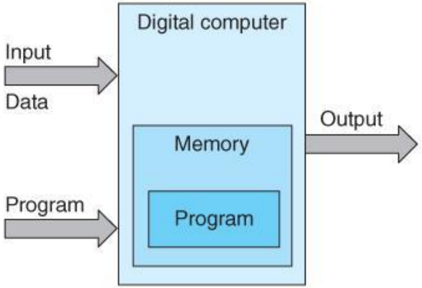



# What *should* a (modern) computer have?

- Required
  - Processing Unit (CPU)
  - Memory
- Optional
  - Graphics Processing Unit (GPU)
  - Audio Processing Unit (Sound card)
  - Mouse
  - Keyboard
  - Monitor
  - etc

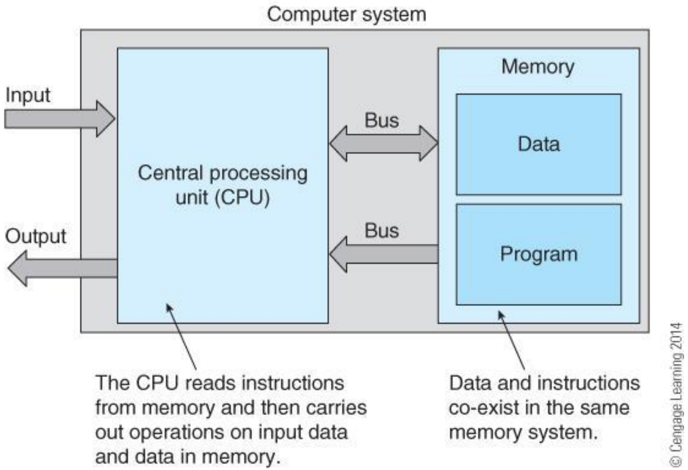
# What's a Computer?

- Computers are dedicated or general-purpose
- Dedicated (Embedded) – CPE 323, also specialized hardware (CPE 322/324)
- General Purpose (CPE 221)
  - It can be programmed to solve any problem (within its Limitations)
  - A key feature of almost all general-purpose computers is that the program and its data are held in the same memory
  - This kind of computer is called Von Neumann Architecture (alternate is called Harvard architecture, but this means something different)



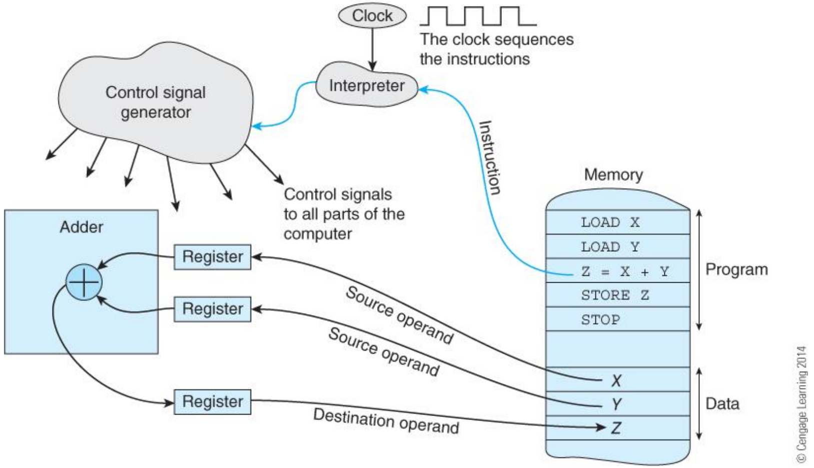
# What is a Computer? (More Details)

- We take data and instructions out of Memory and put them in Registers on the CPU for faster access
- A Register is specified in terms of the number of bits it holds, typically 32 or 64-bit for modern PCs, smaller for embedded systems.



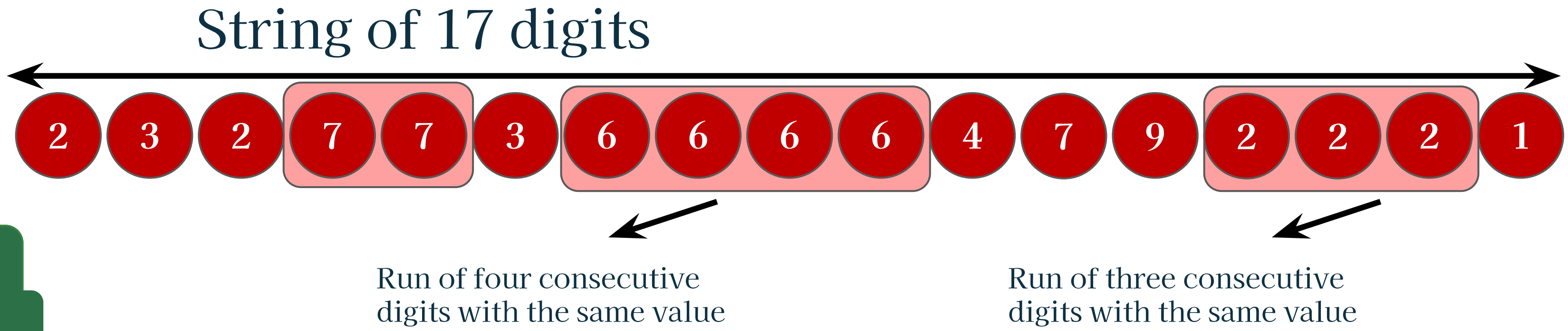
# Program Execution

- Everything happens at the direction of the Clock.
- The Actions of the hardware support the Operations of a High level language such as A = B, or C = A + B.
- The operation shown below is Z = X + Y.



# The Stored Program Computer

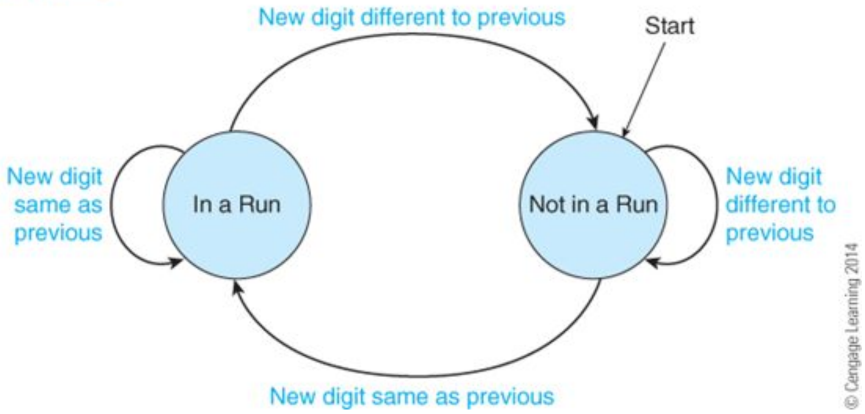
- A computer is a tool to solve problems
- Dumb, but very powerful and can follow instructions
- Consider the problem of finding the longest sequence of repeated digits in a stream of digits.
- In the figure shown, the longest run of repeated digits is four consecutive sixes.



# The Stored Program Computer

State diagram shows the states the computer can take, and the conditions needed to move from one state to another. A modern computer would have a state diagram too complex to read comfortably.

FIGURE 1.8 A state diagram for a run-length counter



# The Problem Solution (Pseudocode)

```
i          The current position in the string
New_Digit     The value of the current digit just read
Current_Run_Value  The value of the elements in the current run
Current_Run_length The length of the current run
Max_Run       The length of the longest run we've found so far

Read the first digit in the string and call it N
Set the Current_Run_Value to New_Digit
Set the Current_Run_Length to 1
Set the Max_Run to 1
REPEAT
  Read New_Digit
  IF New_Digit is the same as Current_Run_Valu
  THEN Current_Run_Length = Current_Run_Length
  ELSE {Current_Run_Length = 1
    Current_Run_Value = New_Digit}
  IF Current_Run_Length > Max_Run
  THEN Max_Run = Current_Run_Length
UNTIL no more digits to read
```

Identifies the **length of the longest contiguous run of the same digit** in a string of digits.

# The Problem Solution (Language)

| | |
|---|---|
| 0 | i = 21 |
| 1 | New_Digit = Memory(i) |
| 2 | Set Current_Run_Value to New_Digit |
| 3 | Set the Current_Run_Length to 1 |
| 4 | Set the Max_Run to 1 |
| 5 | REPEAT |
| 6 | i = i + 1 |
| 7 | New_Digit = Memory(i) |
| 8 | IF New_Digit = Current_Run_Value |
| 9 | THEN Current_Run_Length = Current_Run_Length + 1 |
| 10 | JUMP to 13 |
| 11 | ELSE Current_Run_Length = 1; |
| 12 | Current_Run_Value = New_Digit |
| 13 | IF Current_Run_Length > Max_Run |
| 14 | THEN Max_Run = Current_Run_Length |
| 15 | UNTIL i = 37 |
| 16 | Stop |
| 17 | **New_Digit** |
| 18 | **Current_Run_Value** |
| 19 | **Current_Run_Length** |
| 20 | **Max_Run** |
| 21 | **2 (the first digit in the string)** |
| 22 | **3** |
| 23 | **2** |
| 23 | **7** |
| … | … |
| 37 | **1 (the last digit in the string)** |



Instructions

Data

# What does a computer need to solve a problem?

1. Needs to be able to start running a program at a specific location (line) or memory address
2. Once finished with current instruction by default it has to go to the next location (line)
3. Read or get data from memory. Use symbolic names line X, location to reference the memory locations.
4. Set memory to a specific value
5. Update something from memory (add 1)
6. Store a value back out to memory
7. Loop or jump to a specific instruction
8. Conditional behavior. Check a value against another value and branch or not branch depending on the result.

# The Stored Program Concept

Stored_program_machine

Point to the first instruction in memory

REPEAT

Read instruction at the memory location pointed at

Point to the next instruction

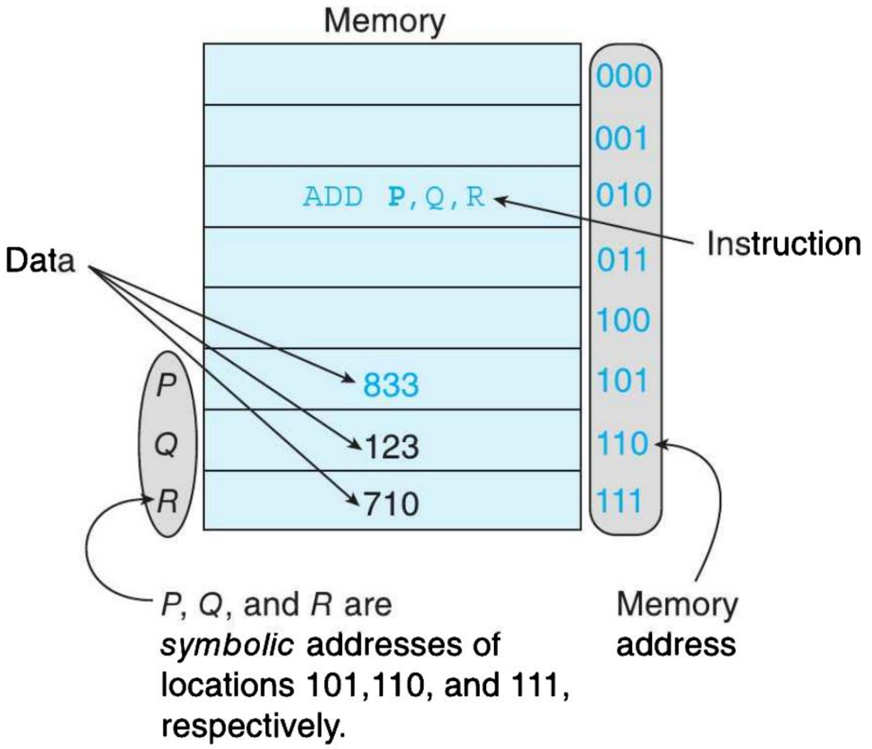
Decode the instruction read from memory

Execute the instruction

FOREVER

End

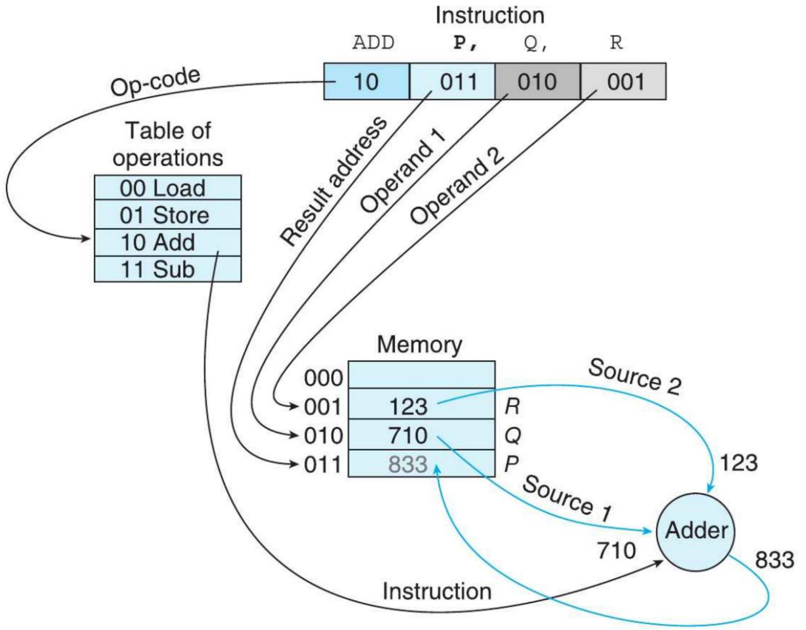
# An Instruction and its Operands



# The Three-Address Instruction

Interpreting the instruction

**ADD P, Q, R**



# The Two-Address Instruction

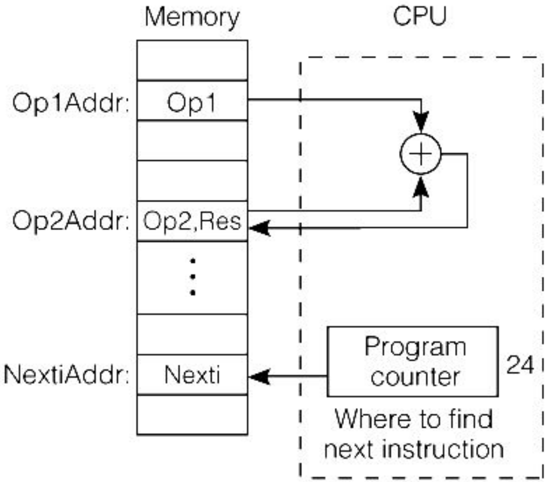
Two-address instruction of the format:

add Op2, Op1 (Op2← Op2 + Op1)

Operation Address1 Address2

Address2 is source operand
Address1 is both source and destination operand.

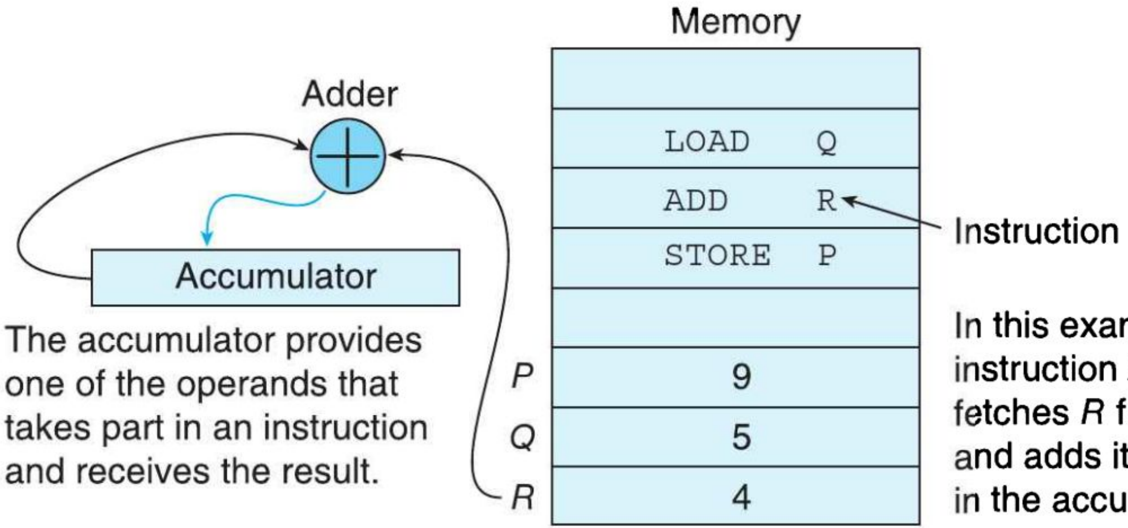
[P] ← [P] + [Q]




| Bits: | 8 | 24 | 24 |
|---|---|---|---|
| | add | Op2Addr | Op1Addr |
| | which operation | where to find operands | |

where to put the result

# The One-Address Instruction




The accumulator provides one of the operands that takes part in an instruction and receives the result.

In this example, the instruction ADD R fetches *R* from memory and adds it to the number in the accumulator.

# Introducing Register Transfer Level (RTL) Notation

RTL is a *notation* used to define operations. Square brackets indicate the *contents* of a memory location. The expression $[15] = \text{Max\_Run}$ means "the contents of memory location 15 contains the value of Max_Run".

The backward arrow symbol, $\leftarrow$, indicates a *Data Transfer*.

# RTL Example

For example, $[15] \leftarrow [15] + 1$ is interpreted as “the content of memory location 15 is increased by 1 and the result put in memory location 15”.

**Consider:**

A. $R1 \leftarrow 25$: Put 25 in Register R1

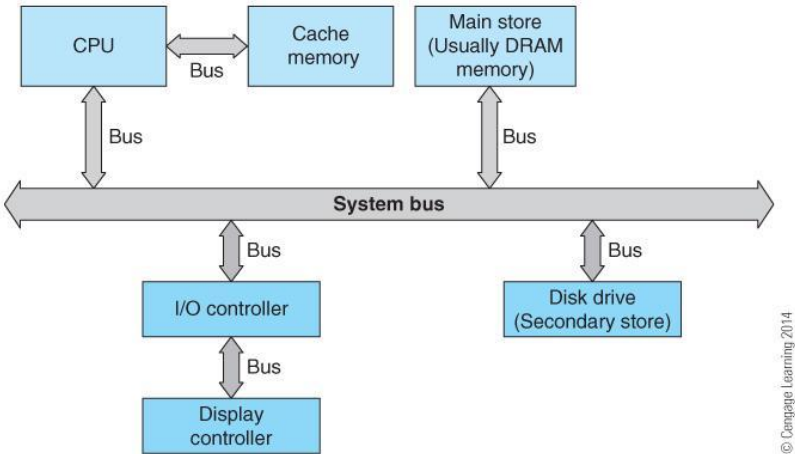
B. $M[20] \leftarrow 6$: Put 6 in memory location 20

C. $M[20] \leftarrow M[6]$: Copy the content of memory location 6 into memory location 20

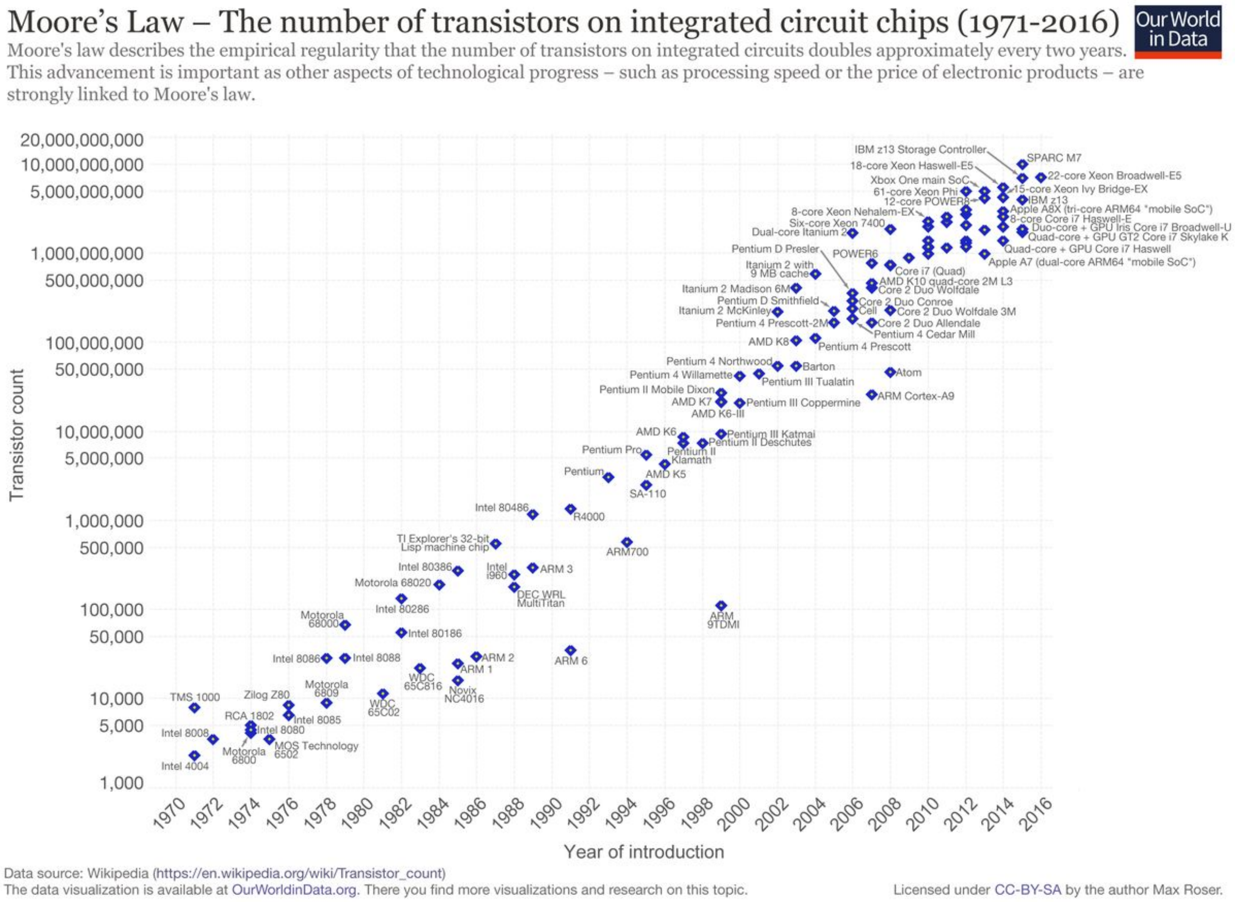
D. $R1 \leftarrow R6$: Copy the content of register R6 into R1

# Structure of a Computer System

One challenge is that subsystems improve at different rates. For decades, processors improved faster than hard drives. As solid-state drives become more prevalent, this problem is alleviated for now. At the same time, processors are now power-limited and increases in clock speeds are no longer possible so we now must make multi-cores work.



# Moore's Law

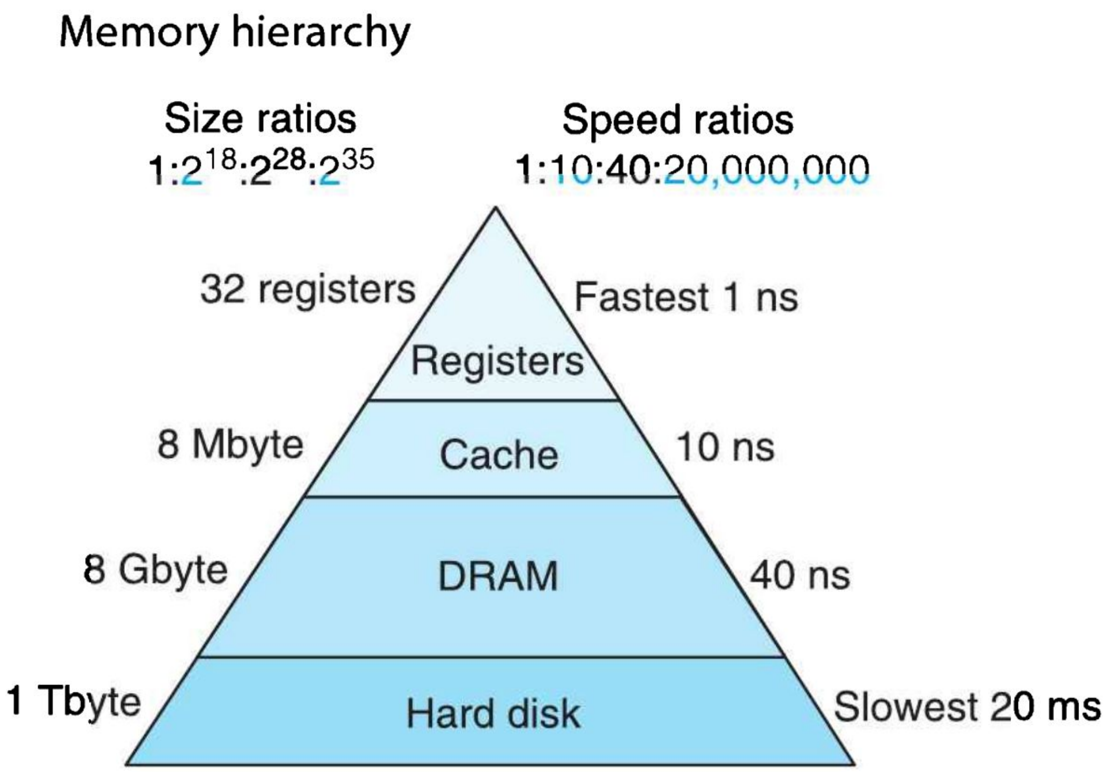


Moore's Law – The number of transistors on integrated circuit chips (1971-2016)
Our World in Data
Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress – such as processing speed or the price of electronic products – are strongly linked to Moore's law.
Transistor count
20,000,000,000
10,000,000,000
5,000,000,000
1,000,000,000
500,000,000
100,000,000
50,000,000
10,000,000
5,000,000
1,000,000
500,000
100,000
50,000
10,000
5,000
1,000
Year of introduction
1970 1972 1974 1976 1978 1980 1982 1984 1986 1988 1990 1992 1994 1996 1998 2000 2002 2004 2006 2008 2010 2012 2014 2016
IBM z13 Storage Controller
SPARC M7
18-core Xeon Haswell-E5
22-core Xeon Broadwell-E5
Xbox One main SoC
15-core Xeon Ivy Bridge-EX
61-core Xeon Phi
IBM z13
12-core POWER8
Apple A8X (tri-core ARM64 "mobile SoC")
8-core Xeon Nehalem-EX
8-core Core i7 Haswell-E
Six-core Xeon 7400
Duo-core + GPU Iris Core i7 Broadwell-U
Dual-core Itanium 2
Quad-core + GPU GT2 Core i7 Skylake K
Pentium D Presler
POWER6
Quad-core + GPU Core i7 Haswell
Itanium 2 with 9 MB cache
Apple A7 (dual-core ARM64 "mobile SoC")
Core i7 (Quad)
AMD K10 quad-core 2M L3
Itanium 2 Madison 6M
Core 2 Duo Wolfdale
Pentium D Smithfield
Core 2 Duo Conroe
Itanium 2 McKinley
Cell
Core 2 Duo Wolfdale 3M
Pentium 4 Prescott-2M
Core 2 Duo Allendale
Pentium 4 Cedar Mill
AMD K8
Pentium 4 Prescott
Pentium 4 Northwood
Barton
Pentium 4 Willamette
Atom
Pentium III Tualatin
Pentium II Mobile Dixon
AMD K7
ARM Cortex-A9
Pentium III Coppermine
AMD K6-III
AMD K6
Pentium III Katmai
Pentium II Deschutes
Pentium Pro
Pentium II
Klamath
Pentium
AMD K5
SA-110
Intel 80486
R4000
TI Explorer's 32-bit Lisp machine chip
ARM700
Intel 80386
Intel i960
ARM 3
Motorola 68020
DEC WRL MultiTitan
Intel 80286
ARM 9TDMI
Motorola 68000
Intel 80186
Intel 8086
Intel 8088
ARM 2
WDC 65C816
ARM 1
ARM 6
Novix NC4016
TMS 1000
Zilog Z80
Motorola 6809
WDC 65C02
RCA 1802
Intel 8085
Intel 8080
Intel 8008
Motorola 6800
MOS Technology 6502
Intel 4004
Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)
The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.
Licensed under CC-BY-SA by the author Max Roser.

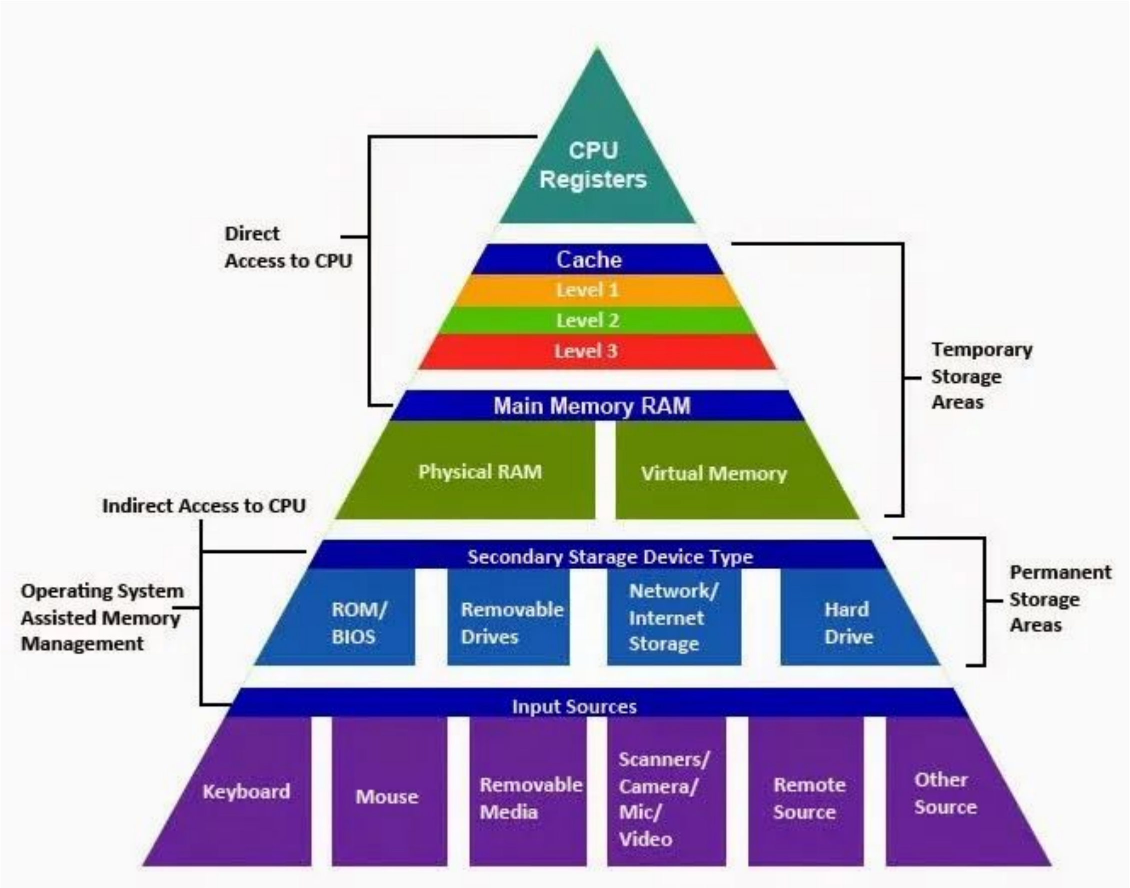
# Technological Obstacles

- Power Wall
  – Energy isn’t infinite, so energy consumption cannot continue to expand forever
- Heat Wall
  – Wires heat up when conducting. Add enough wires, eventually enough heat accumulates to damage the material
- C Wall
  – Electricity cannot move faster than the speed of light
- Atomic Wall
  – Wires can be placed only so close together before the current will arc

# Overview of the Computer System – Memory Hierarchy



# Overview of Data Storage

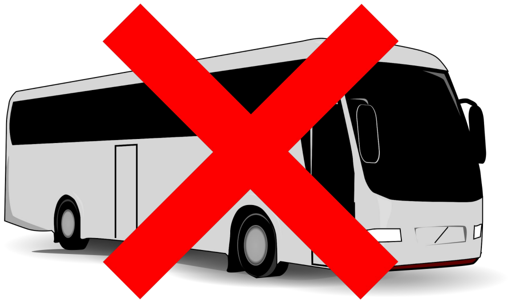


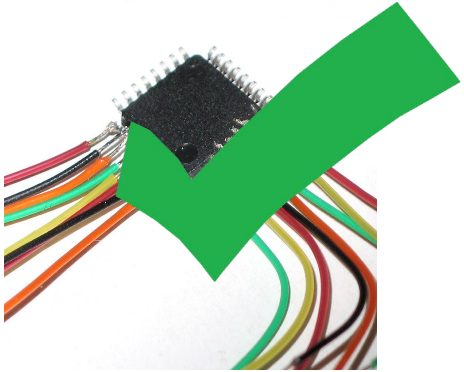
# Old Secondary Storage

Then came CDs and other Optical disk drives, like DVDs

# What's a Bus?

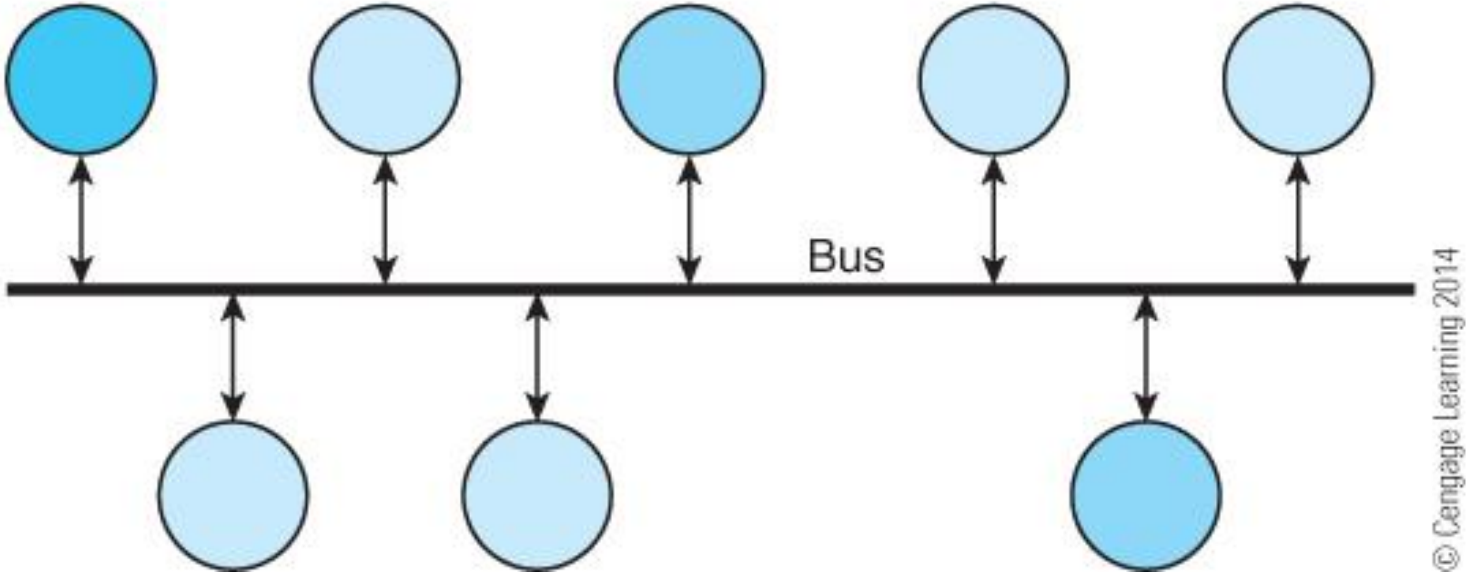
- Strictly speaking, it is only a wire that connects two devices together. That's all a bus is at its base level.
- However, the protocols used to govern the behavior of signals across a bus can be quite complicated





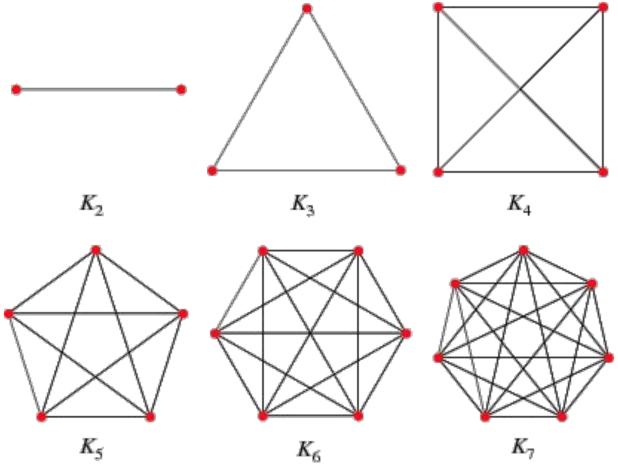
# Overview of the Computer System – Simple Bus

Having a single bus "highway" means all the nodes must wait their turn to use the wire.



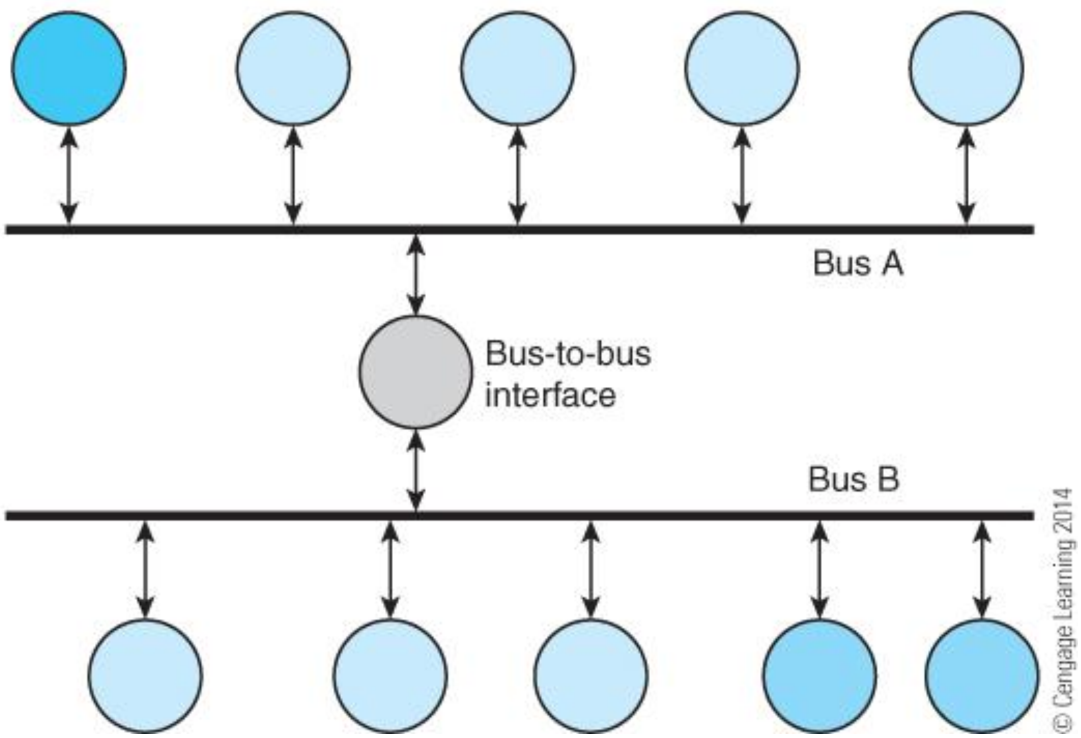
# Overview of the Computer System – Simple Bus

Point to point buses would be faster and simpler to program, but manufacturing a chip would be impractical as the amount of wires required would make the logistics become prohibitively complicated.




| Nodes | Edges |
|---|---|
| 2 | (2*1)/2 = 1 |
| 3 | (3*2)/2 = 3 |
| 4 | (4*3)/2 = 6 |
| 5 | (5*4)/2 = 10 |
| 6 | (6*5)/2 = 15 |
| 7 | (7*6)/2 = 21 |
| 16 | (16*15)/2 = 120 |
| 32 | (32 * 31)/2 = 496 |
| 64 | (64 * 63)/2 = 2016 |

# Overview of the Computer System – Buses



# Some Important Terms

- Instruction Set Architecture (ISA)
- Computer Organization
- MicroArchitecture
- Machine Code
- Assembly language code
- Reduced Instruction Set Computer (RISC)
- Complex Instruction Set Computer (CISC)
- Moore's Law
- Applications
- Clock
- Central Processing Unit (CPU)
- RAM
- Algorithm
- Program
- Pseudocode

# Video of the Day

https://www.youtube.com/watch?v=-M61ANfzFsM