ENGN2219/COMP6719
Computer Systems & Organization

Convener: Shoaib Akram

shoaib.akram@anu.edu.au

Australian National University

Shoaib Akram
Lecturer, School of Computing, (Jan 2020 –)
Ph.D., 2019
Teaching: Computer Microarchitecure (semester 2)
Interests: Hardware/software interaction and performance analysis

It is an interesting time to learn and research about computer systems. Traditional semiconductor laws are breaking down. But the society needs more compute power and storage capacity: Big Data, AI/ML, communication needs, among others.

My current focus is on enabling fast access to large amounts of information (data) using emerging memory and storage devices.

Logistics

Course webpage: https://comp.anu.edu.au/courses/engn2219/

Lectures (on the website**)**

- Lecture slides
- Lecture videos (I will do my own recording)
- Weekly problem sets (for your practice only, *Not Graded*)
- Key Ideas and summary

Policies (will be up shortly)

- General conduct, assignment groups/submissions, support, management, grading

Resources

- Frequently asked questions
- Writing design documents
- Stuff needed to finish the assignments

Piazza

I will use Piazza for all communication

- If you ignore Piazza, *you will miss key announcements*
    - Drop-in sessions, make-up lectures, problems, exercises, corrections, lecture timing (ENGN2218 conflict)
- Ask questions on Piazza first (most likely you will receive a response quickly)
- Post solutions to weekly problems, ask your classmates if you are on the right track
- *Ask private questions on Piazza to instructors*
- Students are added/dropped automatically

Tutorials/Labs

Labs are a critical component of this course (one every week)

Handout will be posted on the website "Labs" before each lab

First six labs

- In each lab, you will finish a sub-component of design assignment 1
- If you finish the first six labs, you will finish ~50% of the first assignment
- We only mark the week 2 – 3 labs to make sure you are making progress and to give you feedback (due Monday 6 pm week 4)
- Week 3 lab accounts towards 5 points (out of 100) for assignment 1

Week 7 lab (after the teaching break)

- Test your design project

Week 8 – 11 Labs

- C programming (*not really about C, but about learning key computer systems concepts, more on this later*)

Assessments

Two assignments (60%)

- CPU design assignment (30%)
  - 5% due on 6 pm, Monday of week 4
  - Full assignment due on 12 pm, Monday of week 8
- Programming assignment (30%)
  - Due date: Monday 6 pm, week 13

Final Exam (40%)

- I will release problem sets and exercises throughout the course for preparation
- Mock-up exams will be available during/after the break

Assignment Submission

Extensions will be granted on a per-request basis

- Via Email: Shoaib.Akram@anu.edu.au

Assignment submissions are handled via Gitlab

- You will learn about it in the labs
- Make a habit of using Git properly
- Push often, always pull the latest

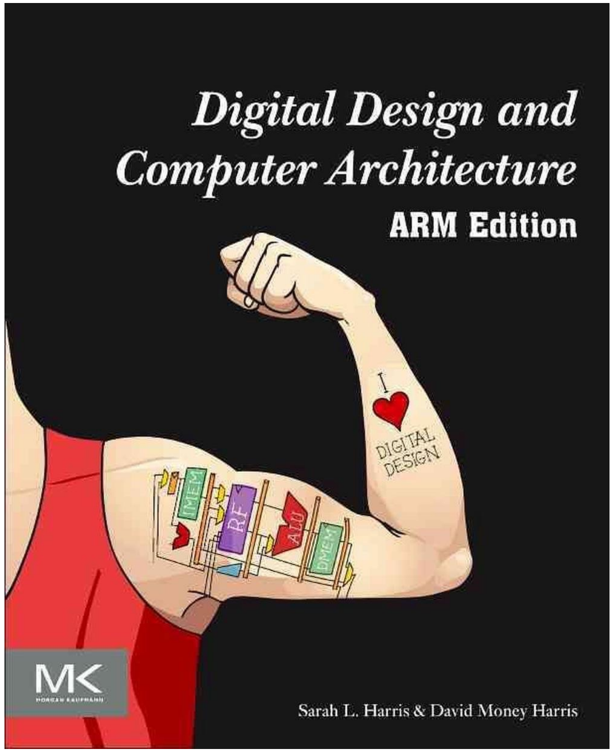
You can form group of up to two students to work on the assignment (one submission per group)

Ink/Whiteboard in Lectures

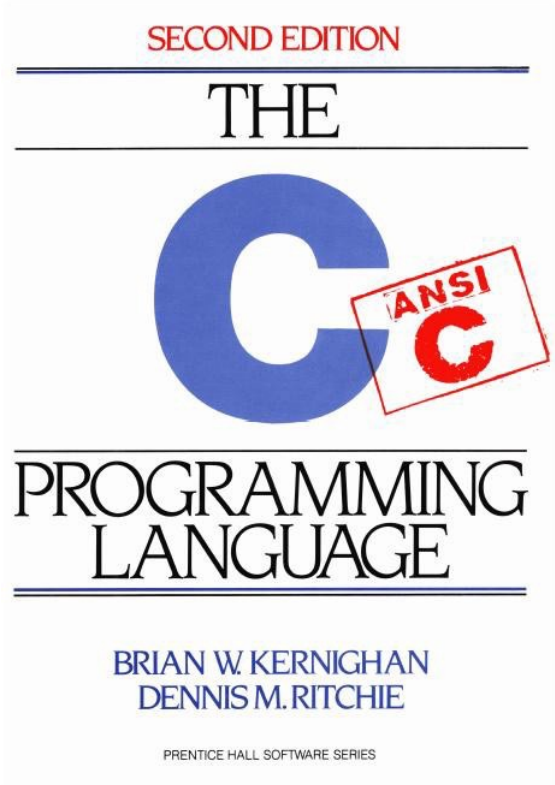
Try to take notes

- Help you think and most of the time I will do this is to solve a problem
- Ink + Wacom (not very stable)

Textbook



Kernighan & Ritchie, The C Programming Language, 2nd Edition
- “ANSI” (old-school) C

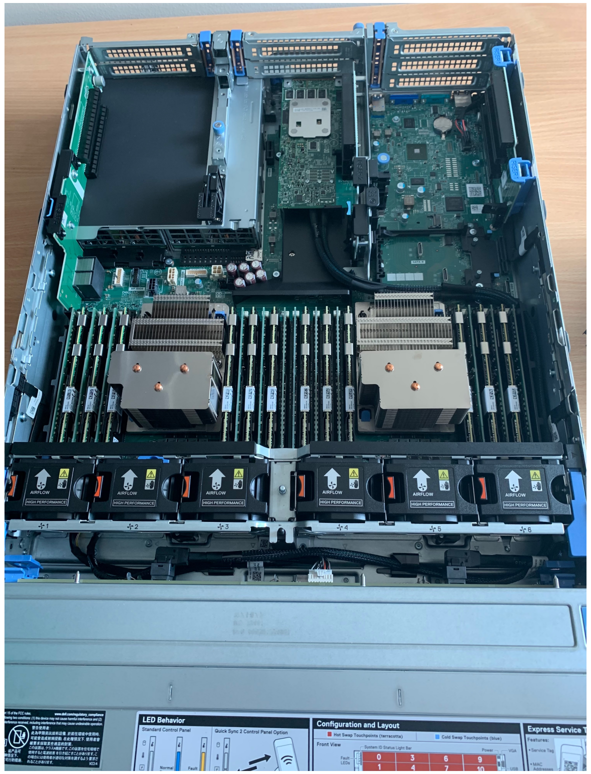


- Freely available online (*check Piazza or course webpage*)
- *I will post the chapters/sections on the Lectures page after the lecture*

Council Bluffs, Iowa data center, Google (115, 000 sq. feet)

Self-flying nano drone
94 milli-watts

Research server for my students with special memory & storage devices



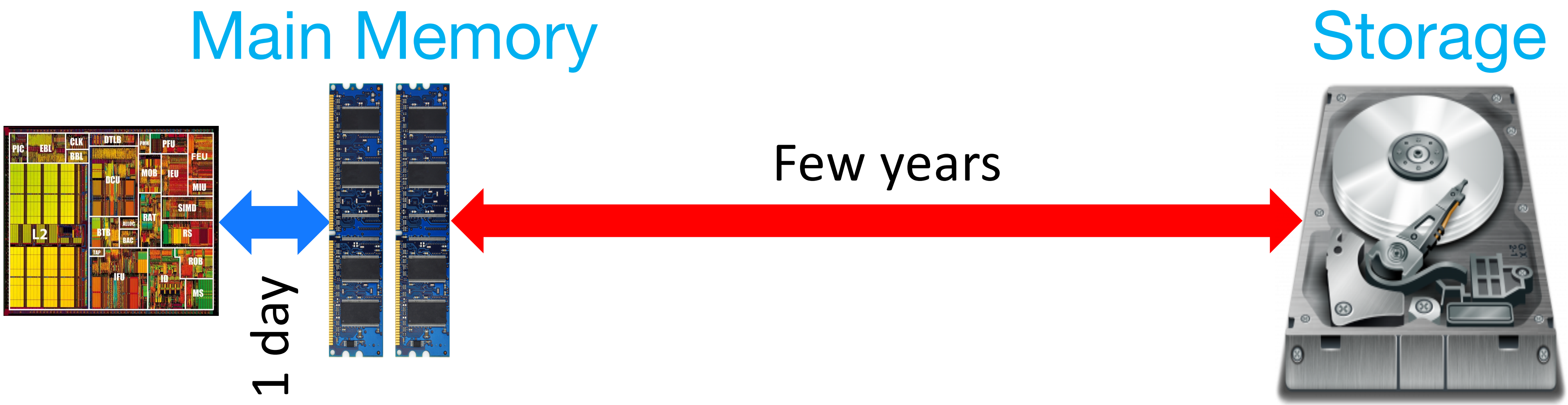
Fundamentals are important

All computer systems, big or small, have a few fundamental components

- **Microprocessor** (processor or central processing unit or CPU) for doing computation
- **Main memory** for storing temporary information and program data close to the processor
- **Storage** devices (e.g., disks or SSDs) for storing long-term or persistent information
- **I/O devices** to communicate with the external environment
  - Sensors
  - Peripherals

Most computer systems can be viewed as below

- Three key resources: CPU, memory, storage
- CPU is the heart of a computer system
- Processor can access memory much faster than storage



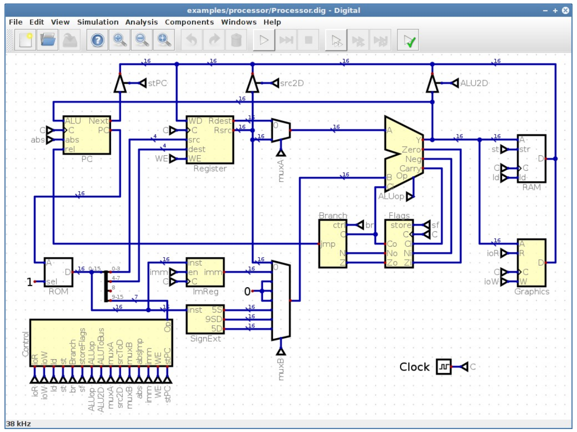
This Course is about ...

How does the general-purpose processor work? How do modern processors perform a wide variety of tasks?

How do processors interact with main memory and storage? How does the memory and storage system work?

The best way to learn how something works is to build one

- You will build a processor in Digital
- URL: https://github.com/hneemann/Digital



This Course is also about ...

A computer system is more than just hardware

- How does hardware and software interact?
- What should programmers know about hardware?
- C is a good vehicle for answering the above questions
- You can talk about hardware resources in high-level terms but still stay close to the hardware
- Key learning outcome of this course: *How can you shoot yourself in the foot when writing C programs?*
- **Remember:** It’s not about C or Java or Python. It’s about gaining a deep insight into computer systems!

A 5-Step Recipe for Failure

And do not learn to use Gitlab

Stay out of the loop

- Do not check Piazza
- Do not ask questions
- Do not care what is going on in lectures

Do not attempt end of week problem sets

- Okay, not every week ....

Do not come to labs OR do not read/attempt the lab handouts

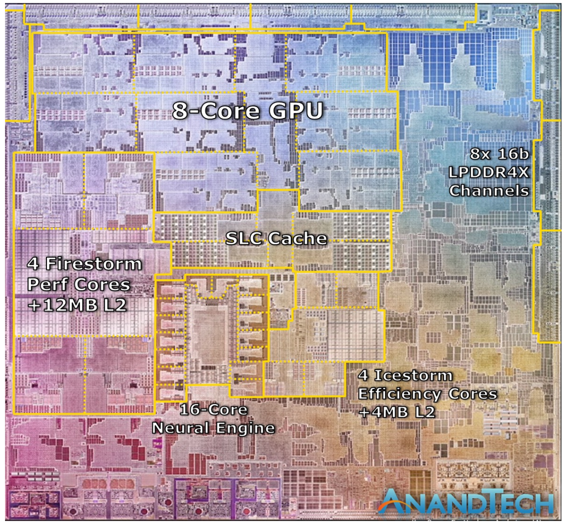
- And start the assignment on your own the weekend before due date (no way!)

Do not seek help from tutors

Do not communicate your problems to me!

How do engineers manage complexity?

- Look at components from a higher level
- Get into detail if necessary

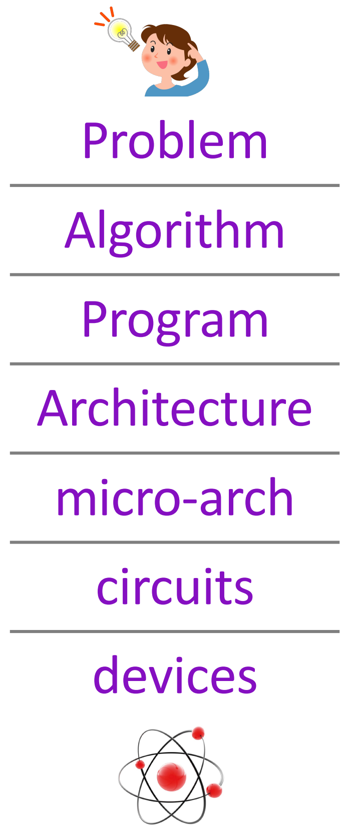


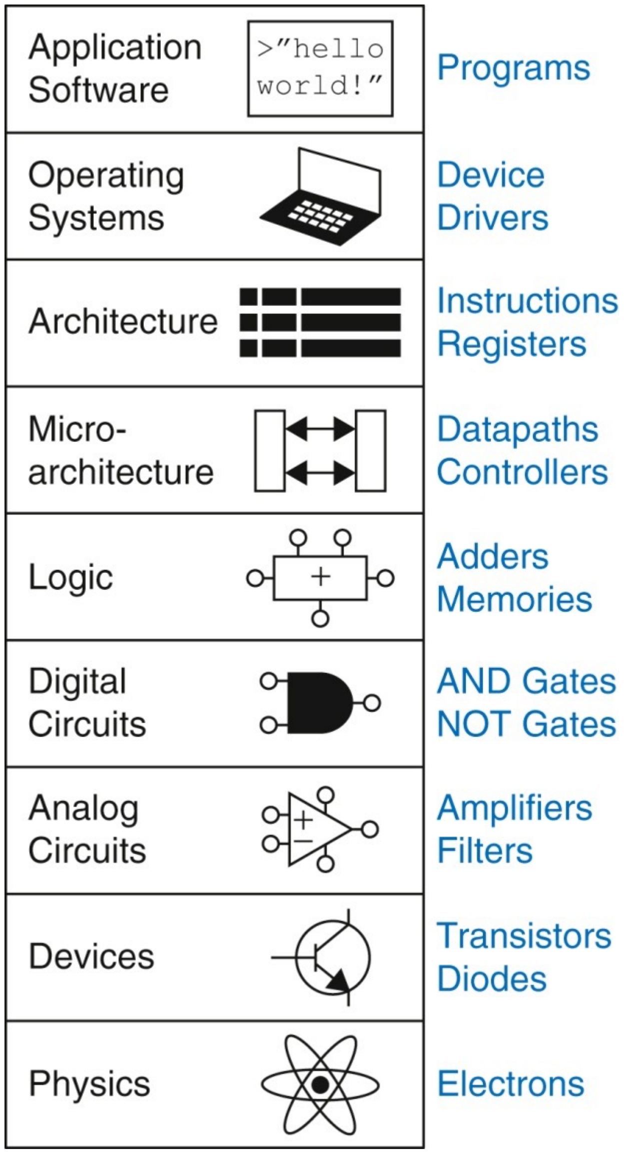
No human (programmer) can track 10 billion elements. **Computer systems work because of abstraction!**

Apple M1 Chip
Billions of transistors
All working in parallel

Transformation Hierarchy

- We think of problems in English
  - Sort students by their UIDs
- The actual work is done by electrons
  - Do electrons speak English?
- How do we make the electrons do the work?
  - *We use a systematic transformation hierarchy to transform the problem in English into electron movement*
  - *This transformation hierarchy is driven by our need to abstract away complexity*





Application Software
>"hello world!"
Programs
Operating Systems
Device Drivers
Architecture
Instructions Registers
Micro-architecture
Datapaths Controllers
Logic
+
Adders Memories
Digital Circuits
AND Gates NOT Gates
Analog Circuits
+
−
Amplifiers Filters
Devices
Transistors Diodes
Physics
Electrons

Definitions

Abstraction: Hiding details to view the system from a high level

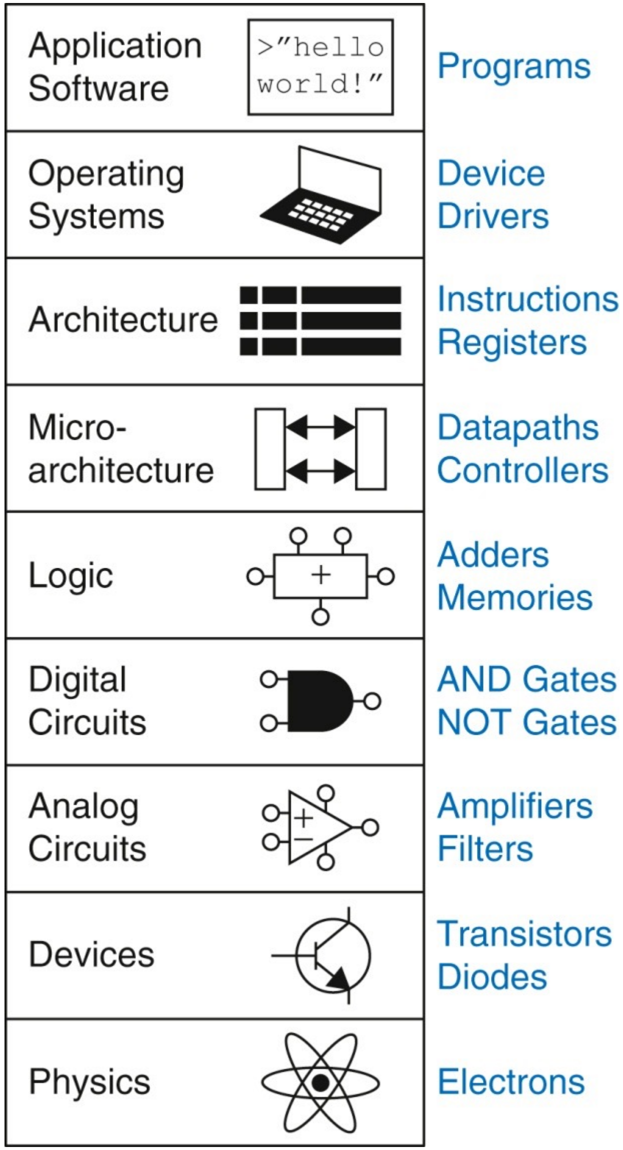
Deconstruction: Going from abstraction back to its component parts (breaking abstraction)

Low-level language: Languages that are tied to the machine architecture. Each architecture supports at least one low-level language called assembly.

High-level language: Programming languages that are at distance higher than the architecture. They are machine-independent. E.g., C, Java, Python, Rust, Ruby, Go

Instruction Set Architecture: A specification of all the instructions a processor can perform. Each instruction is an arithmetic (add, sub) or a data movement (fetch from memory) operation.

Microarchitecture: ISA has no physical significance. Microarchitecture is the physical implementation of an ISA.



Week 4

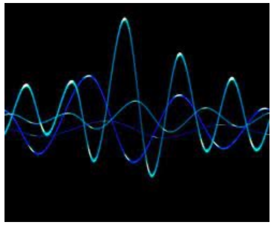
Week 5 – 6

Weeks 2 – 3

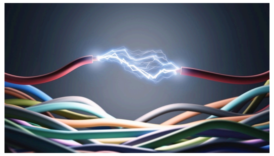
Week 1

Representing Information

Question: How many different values can each of these *physical* variables take?



Frequency of oscillation



Voltage on a wire

Temperature

<u>Answer:</u> <u>Infinite</u>

All these are continuous signals

They contain infinite amount of information

Representing Information

Digital Systems: Represent information with discrete-valued variables, i.e., variables with a finite # distinct values

Modern digital systems use a **binary** (*two-valued*) representation

0 1

0 1

0 1

Binary Representation

Digital systems internally use “voltages” for representing binary variables

→ Low voltage means 0

→ High voltage means 1

B I N A Y   D I G I T

A bit is a unit of information. *A binary variable represents one bit of information. To represent discrete sets with more than two elements, we combine multiple bits into a binary code*

Binary Codes

Suppose we want to represent four colors: {red, blue, green, black}

- How many bits of information do I need?
- (00, 01, 10, 11)
- The assignment of the **2-bit binary code** to colors is *ad-hoc*
- Also legitimate is: (10, 11, 00, 01)

How many bits of information do I need to represent the alphabet set in English?

- For 26 alphabets, we need 5 bits

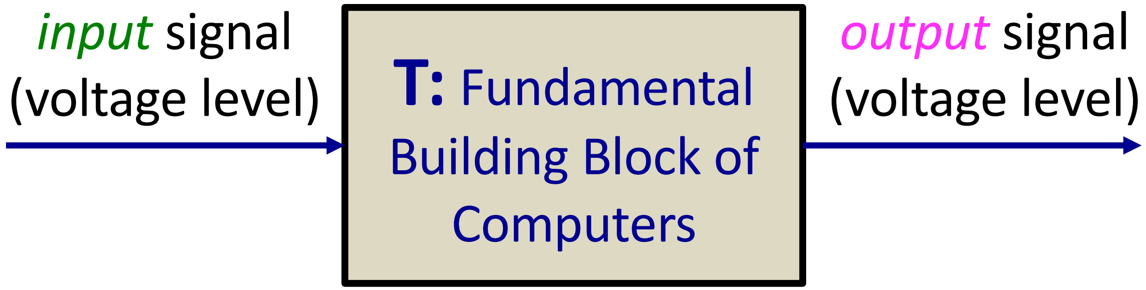
Information Content in a Binary Code

$$\text{D} = Log_2 \, N \; bits$$

The color set has four states: N = 4, # bits = 2
The alphabet set has 26 states: N = 26, # bits = 5

Conversely,
If D is 2, N = 4
If D is 5, N = 32

Why do computers use binary?

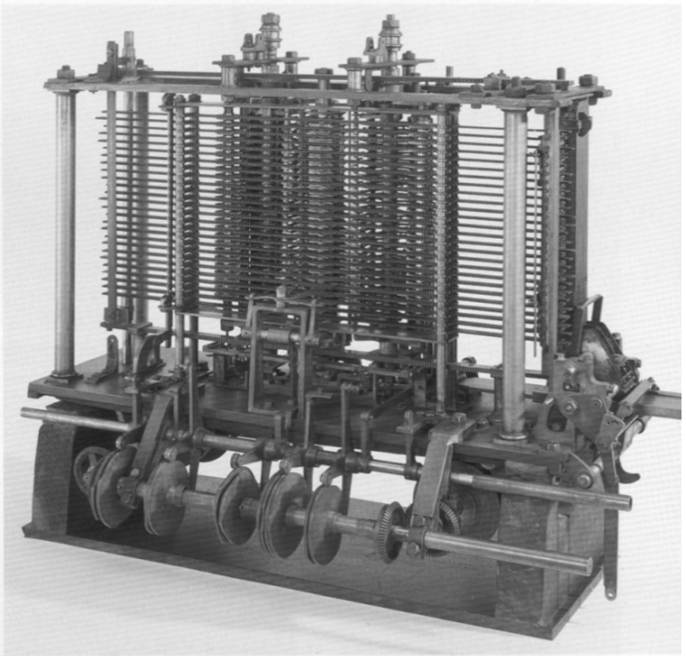


We can divide a continuous voltage range into ten levels to represent 0 – 9, but that would make **T** very complex

The fundamental building block of all computers is a transistor. A transistor can only distinguish two voltage levels. We call these voltage levels 1 and 0

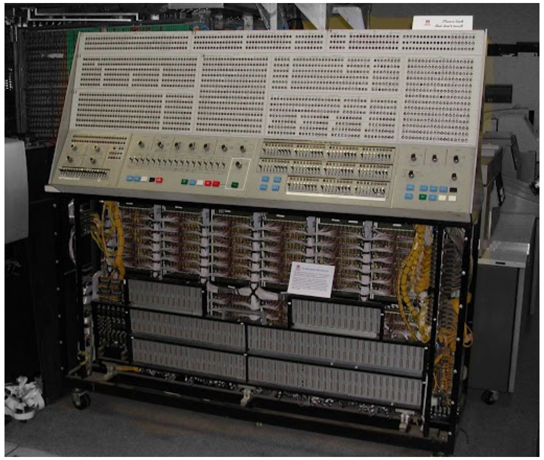
Voltages and Transistors, Why?

Mechanical parts: Not easy to scale to do large computations



The Analytical Engine
Charles Babbage
1834 – 1871

CDC 6600, 1964, $ 2.5 M
Slower than my phone

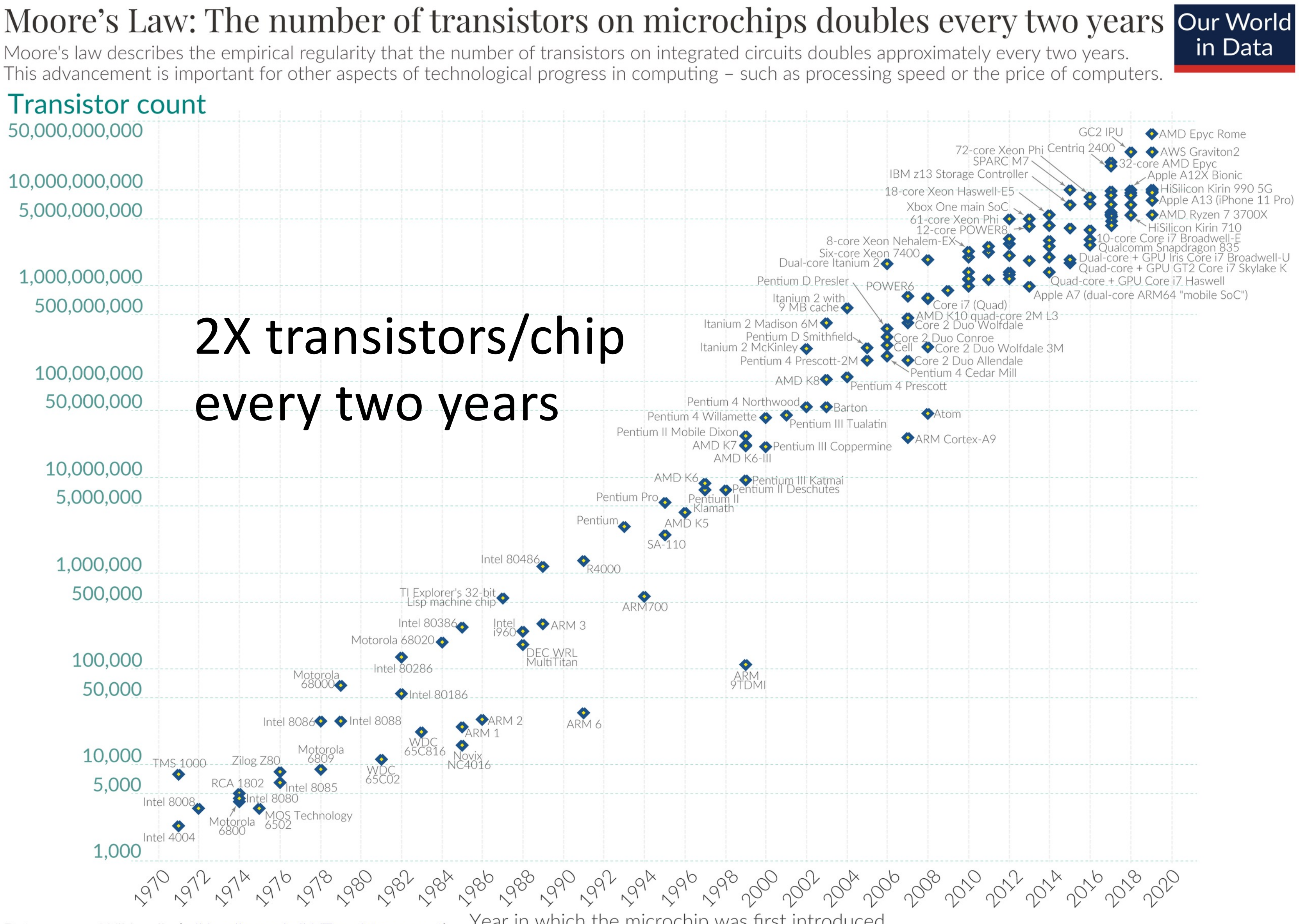


IBM 360, 1964

Apple M1, 2020
400 mm^2
16 billion transistors

Moore's Law: The number of transistors on microchips doubles every two years

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years.
This advancement is important for other aspects of technological progress in computing – such as processing speed or the price of computers.



Data source: Wikipedia (wikipedia.org/wiki/Transistor_count)
OurWorldinData.org – Research and data to make progress against the world's largest problems.

TRUE and FALSE

0 1

F T

False Off

True On

True and False are called logical values

- Logical variable is one that can be 1 or 0 (True or False)
- Boolean logic defines the operations on logic variables

Our Plan

Presenting information to digital circuits

→ *Representing numbers as a string of 1’s and 0’s*

→ *Number systems: to set a foundation for efficient manipulation (add and subtract)*

Operations on binary variables (1’s and 0’s)

→ *Logic gates to perform operations on binary variables*

Breaking the digital abstraction (self study)

→ *1’s and 0’s as continuous physical quantities (voltage)*

Decimal Number System

- Base 10 means 10 digits (0, 1, 2, 3, 4, 5, 6, 7, 8, 9)
- Multiple digits form longer decimal numbers
- Each column of a decimal number has 10 times the weight of the previous column

1000's column
100's column
10's column
1's column

coefficient

power of 10

$9742 = 9 \text{ X } 10^3 + 7 \text{ X } 10^2 + 4 \text{ X } 10^1 + 2 \text{ X } 10^0$

$9 \text{ X } 10^3$	$7 \text{ X } 10^2$	$4 \text{ X } 10^1$	$2 \text{ X } 10^0$
nine thousands	seven hundreds	four tens	two ones

Range of Decimal Numbers

An N-digit decimal number represents one of 10^N possibilities

- 0, 1, 2, 3, ..., $10^N - 1$
- 3 digits: 1000 possibilities in the range 0 – 999

Binary Numbers

- Base 2 means 2 digits (0, 1)
- Multiple bits form longer binary numbers
- Each column of a binary number has **2** times the weight of the previous column

8's column 4's column 2's column 1's column

coefficient

power of 2

$$1001 = 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

one eight	one four	zero two	one one

Range of Binary Numbers

An N-bit binary number represents one of 2^N possibilities

- 0, 1, 2, 3, ..., $2^N - 1$
- 3 bits: 8 (= 2 X 2 X 2) possibilities in the range 0 – 7
- 4 bits: ?
- 5 bits: ?
- 10 bits: ?

Powers of 2

Columns #	Power of 2	Weight
0	2^0	1
1	2^1	2
2	2^2	4
3	2^3	8
4	2^4	16
5	2^5	32
6	2^6	64
7	2^7	128
8	2^8	256
9	2^9	512

Columns #	Power of 2	Weight	
10	2^{10}	1024	Kilo
11	2^{11}	2048	
12	2^{12}	4096	
13	2^{13}	8192	
14	2^{14}	16384	
15	2^{15}	32768	
16	2^{16}	65536	

Powers of 2

Power of 2	Decimal Value	Abbreviation
2^{10}	1024	Kilo (K)
2^{20}	1048576	Mega (M)
2^{30}	1073741824	Giga (G)

What is 2^{24} in decimal?

- 2^{20} X 2^{4} = 1 M X 16 = 16 M

What is 2^{17} in decimal?

- 2^{10} X 2^{7} = 1 K X 128 = 128 K