Computer Abstractions and Technology
Introduction to RISC-V

UCONN

Caiwen Ding

Department of Computer Science and Engineering

University of Connecticut

CSE3666: Introduction to Computer Architecture

Syllabus Highlights

Syllabus: Check Course Syllabus in HuskyCT
https://huskyct.uconn.edu/ultra/courses/_106660_1/cl/outline

Are you in the right room? There are two independent CSE 3666 sections this semester. Our section ID is SEC**001-**1223.

- **Lecture**
  - Tue&Thu 3:30PM-4:45PM @BOUS A106.
- **Discord:**
  - Join the CSE3666-~~Fall2021~~ Spring 2022 Discord Server!
- **Labs**
  - Lab 001L, ITE 134, Mon 2:30PM - 3:20PM
  - Lab 002L, ITE 134, Mon 3:35PM - 4:25PM
  - Lab 003L, ITE 134, Mon 1:25PM - 2:15PM
- Week 1-2: Lectures (https://uconn-cmr.webex.com/meet/cad15002) and labs (Blackbaord Collaborate) are online synchronously.
- Starting from the third week, Lectures and labs are in person .

Syllabus Highlights

- **Office Hours**
  - Tue: 10:00AM-11:20AM (Prof. Caiwen Ding) at ITE 361 or https://uconn-cmr.webex.com/meet/cad15002 or by appointment.
- TA office hours:
  - TBD

CSE 3666
CSE 3666

NOTE: When emailing TAs/Instructor, please add "**3666**" in the subject line such that we could better track your emails. If you do not receive a reply in 48 hours, please email us again.

- **Grade**
  - Labs: 10%
  - Homework: 25%
  - Quizzes: 5%
  - Exams: 60% (two in-class exams: 10% each, final exam: 25%, the average of two higher exam scores: 15%)
  - Extra credit. Students who participate in lecture by answering iClicker questions can earn up to 2 points. Students earn 2 points if they have answered 60% of the graded questions correctly and 1 point if they have answered 35% of the graded questions correctly. There is no makeup for iClicker questions.
- Submit lab and homework assignments on HuskyCT. In general, students must justify their answers in homework and exam, write concise comments in code, make handwriting legible.

Syllabus Highlights

- Spring 2022. Since lectures and labs are online in the first two weeks, we will **not** use iClicker in these two weeks. Instead, an online quiz will be given in HuskyCT. Each question on the quiz is equivalent to an iClicker question.

Weighted total	**Letter Grade**
$\geq$ 93.00	A
Below A and $\geq$ 90.00	A-
Below A- and $\geq$ 87.00	B+
Below B+ and $\geq$ 83.00	B
Below B and $\geq$ 80.00	B-
Below B- and $\geq$ 77.00	C+
Below C+ and $\geq$ 73.00	C
Below C and $\geq$ 70.00	C-
Below C- and $\geq$ 67.00	D+
Below D+ and $\geq$ 63.00	D
Below D and $\geq$ 60.00	D-

- Caveat: In order to pass the course, you need to score at least **40%** on the final exam.

Syllabus Highlights

- We will make every effort to provide feedback and grades of assignments in one week. For some assignments and exams, we need more time. If you have questions regarding the grading, you MUST contact either the instructor or TAs within ONE WEEK after graded work is returned to you (or to the class). Please check "My Grades" section in HuskyCT frequently.
- **Late Policy**
- All course due dates are specified in the assignments. Deadlines are based on Eastern Time unless otherwise specified. The instructor reserves the right to change dates accordingly as the semester progresses. All changes will be communicated in an appropriate manner.
- If a lab, a homework assignment, or a project cannot be completed by the deadline, you must contact the instructor or TA **before** the deadline to arrange a late submission, and provide valid university accepted reasons and evidence. Otherwise, late submissions are not accepted. There may be a penalty for late submissions.
- Please note that sometimes late submissions cannot be arranged, especially when doing so would slow down the progress of the class.

Syllabus Highlights

- Students in quarantine should inform instructors and TAs as early as possible. We expect those students join lab and lecture remotely. When attending lectures remotely, student can answer iClicker questions remotely if they have iClicker cloud subscrption. **For Labs, the TAs will provide a Discord or Webex meeting room. Students need to turn camera on during the lab section.**
- Students who are not available to attend labs and lectures, in person or remotely, due to medical reasons, need to contact instructors **with justification Per UConn policy** to make arrangement on missing assignments and exams. We do not have resources to make up missing participation.

Outline

- Technology and Moore's law
- Components of a computer
- Layers in a computer system
- Instruction set architecture, the interface between HW/SW
  Hardware
  (ISA)
  Software
  - Abstraction
- Seven great ideas in computer architecture
  - Actually, not only in computer architecture
- RISC-V ISA
  - Register file *****
  - Arithmetic operations *****

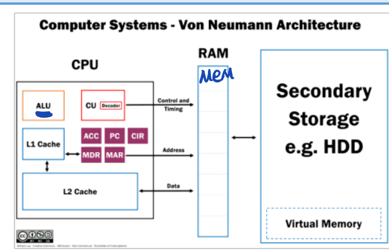
Reading: Sections 1.1 – 1.5, 2.1, 2.2

References: Reference card in textbook

Computer architecture: A Useful Analogy

What is computer architecture?

Building Architecture	Computer Architecture
Form Buildings with the components below	Form computer systems with the components below
Bricks, timber, …	ALUs, flip-flops, bit cells, crossbars, … Arith Logic Unit
Sand, clay, wood, etc	Transistors, logic gates



The Computer Revolution

- Computers are everywhere
- Rapid progress in computer technology
  - Underpinned by Moore's Law
- Makes novel applications feasible
  - Computers in automobiles
  - Cell phones
  - Human genome project
  - World Wide Web
  - Search Engines
  - Machine learning
  - Artificial intelligence

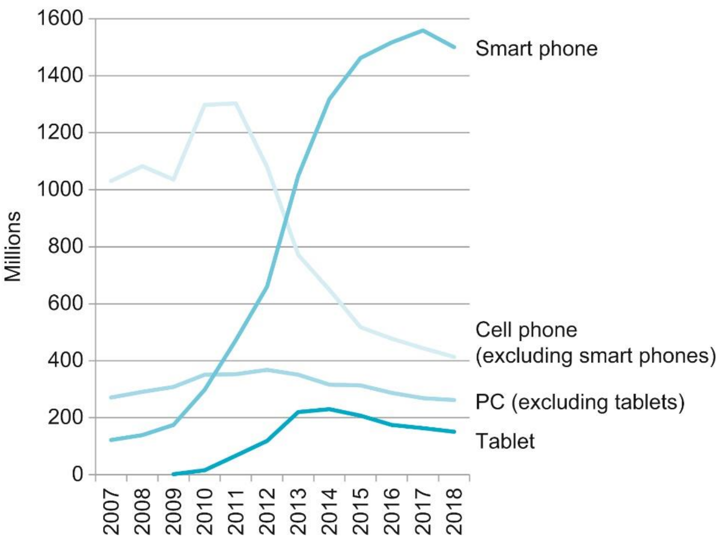
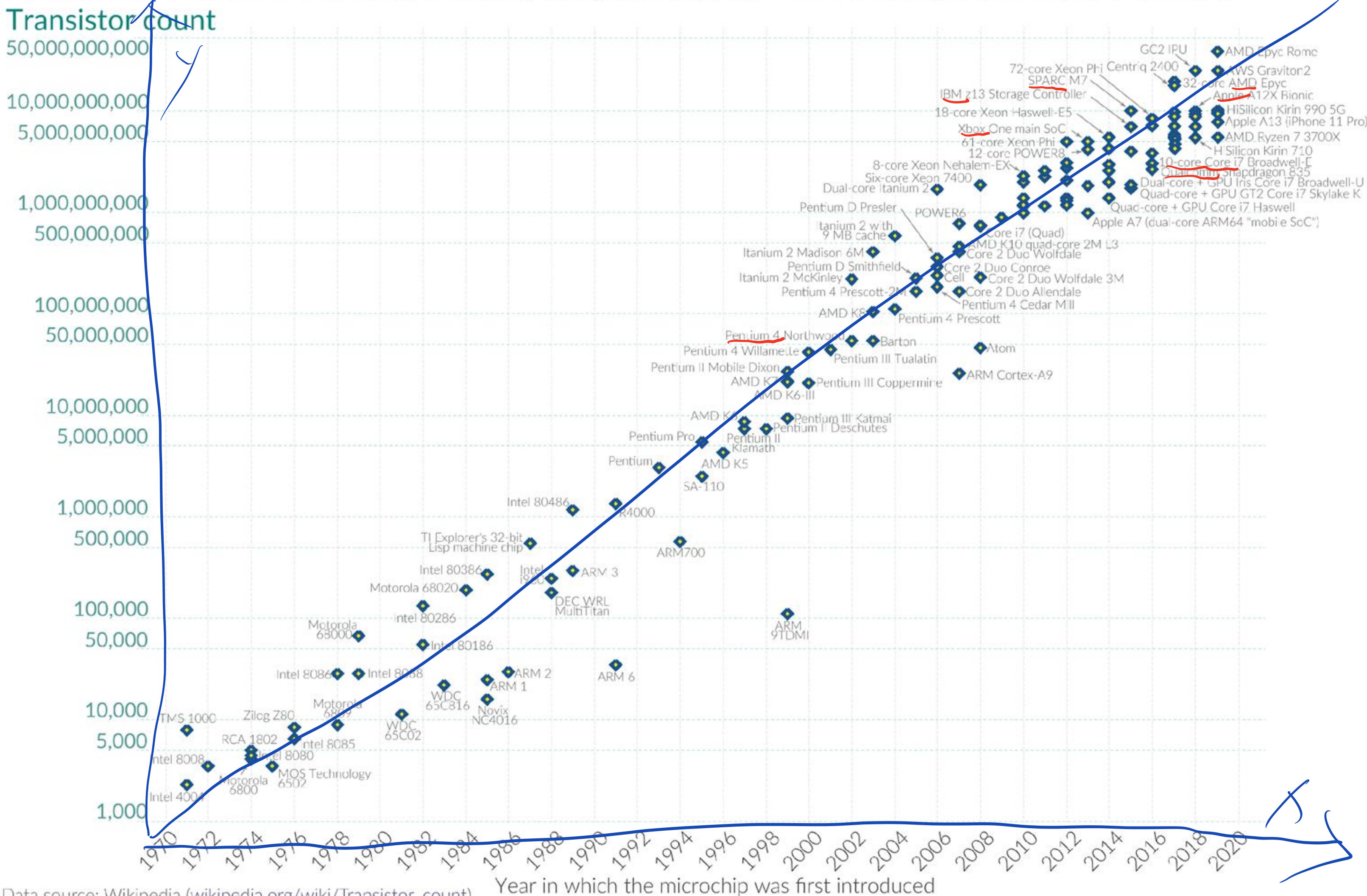


Figure 1.2

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

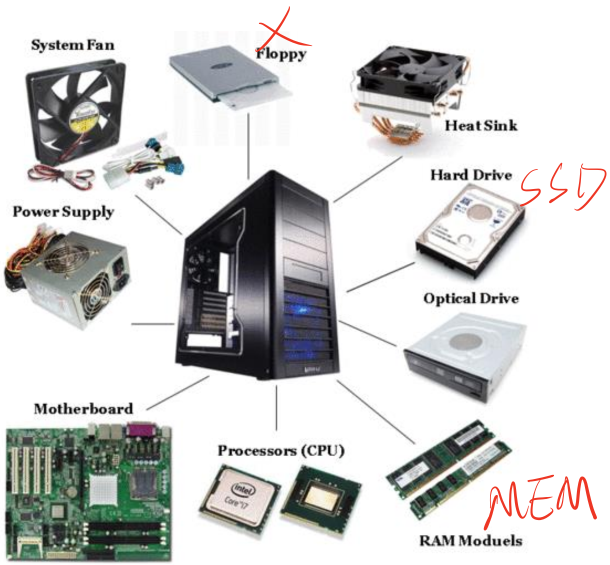
Our World in Data

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



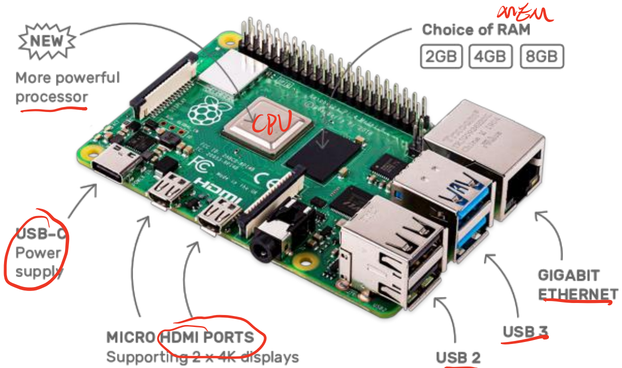
Data source: Wikipedia (wikipedia.org/wiki/Transistor_count)

What are in a computer?



http://hanif360.blogspot.com/2017/11/computer-basic-and-hardware.html

Is this a computer?



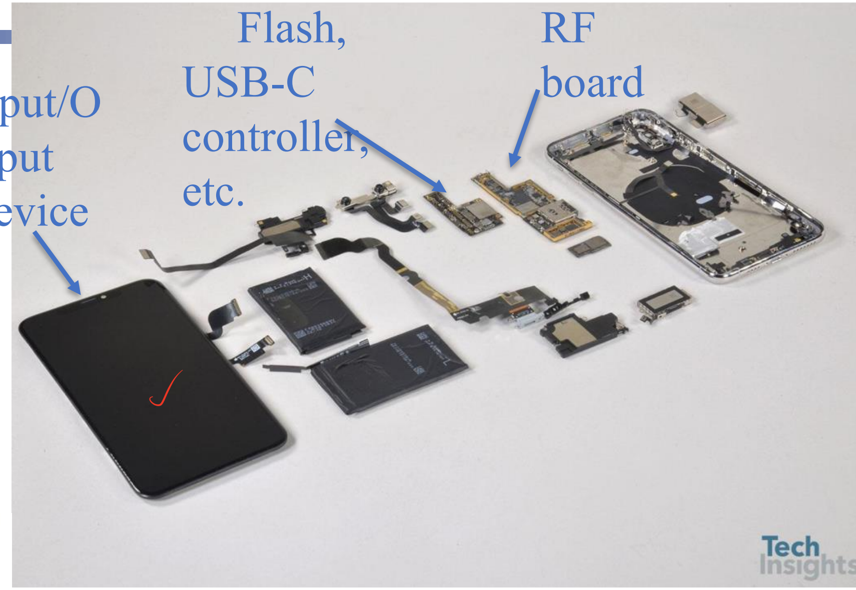
From **$35**

https://www.raspberrypi.com/products/raspberry-pi-4-model-b/

Components of a Computer

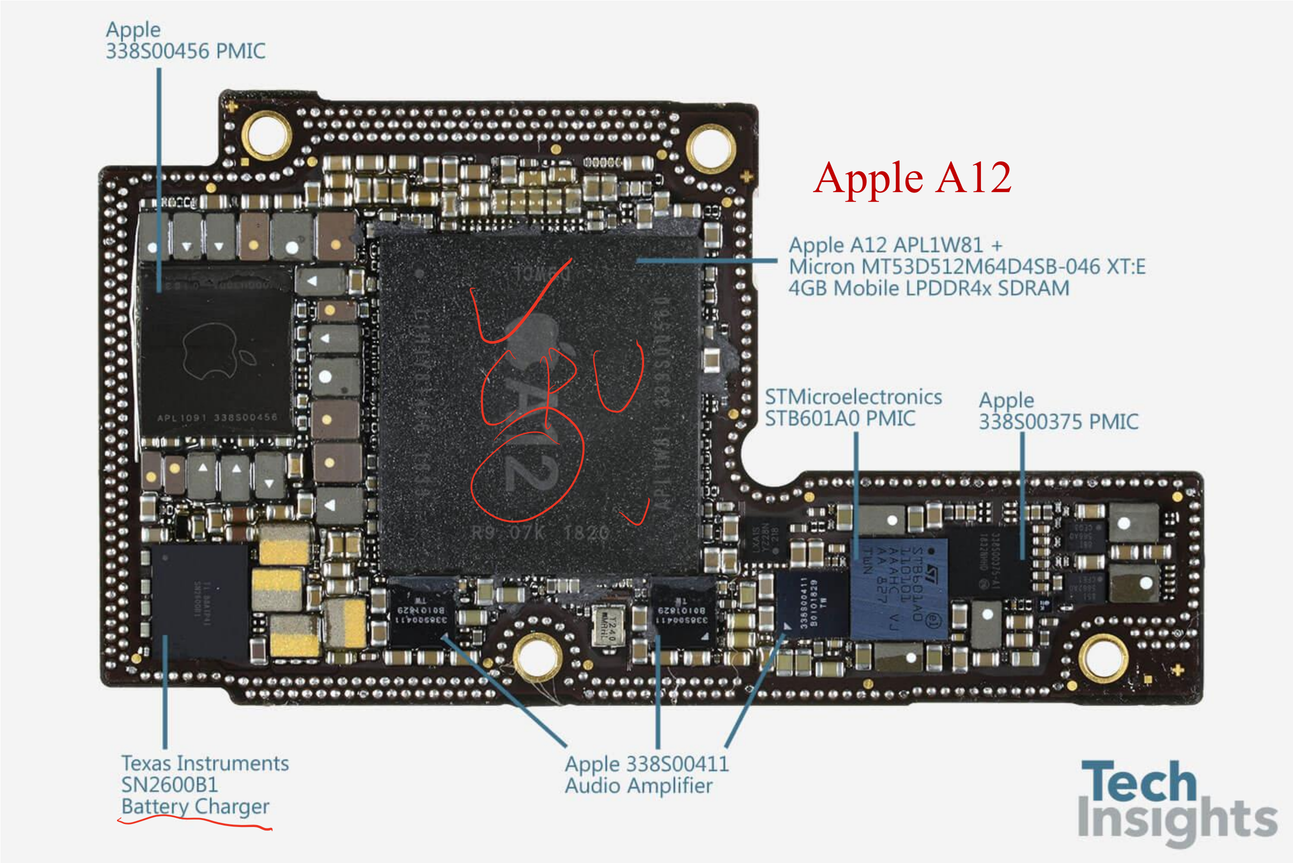
- Input/output
  - User-interface devices
    - Display, keyboard, mouse
  - Storage devices
    - Hard disk, CD/DVD, flash
  - Network adapters
    - For communicating with other computers
- Motherboard
- Memory RAM ✓
- Processor (CPU) ✓ this class
- Misc controller/interface circuit

iPhone XS Max vs. iPhone 4 Teardown



iPhone 4 Teardown- Photo by: Bill Detwiler / TechRepublic

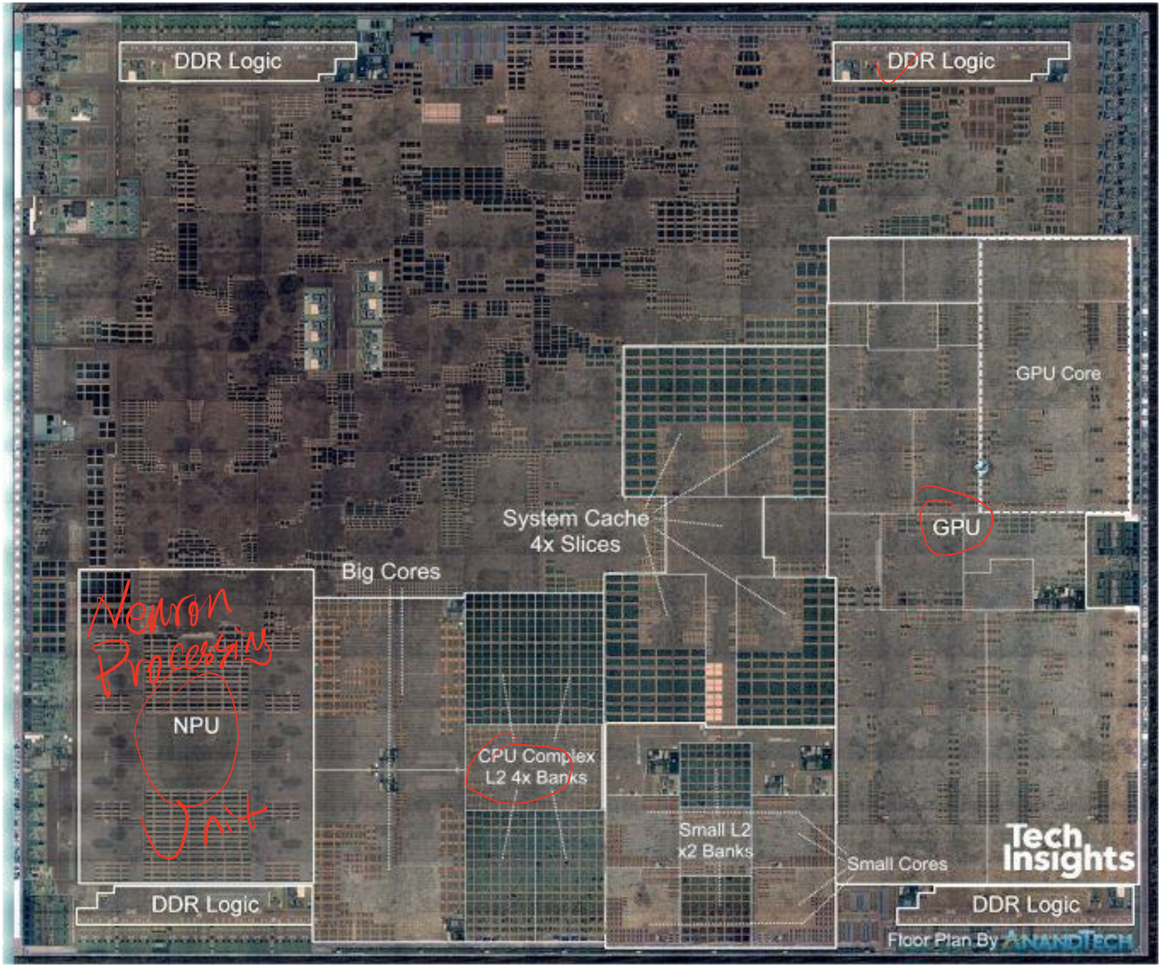
The logic board in iPhone XS MAX



Apple A12

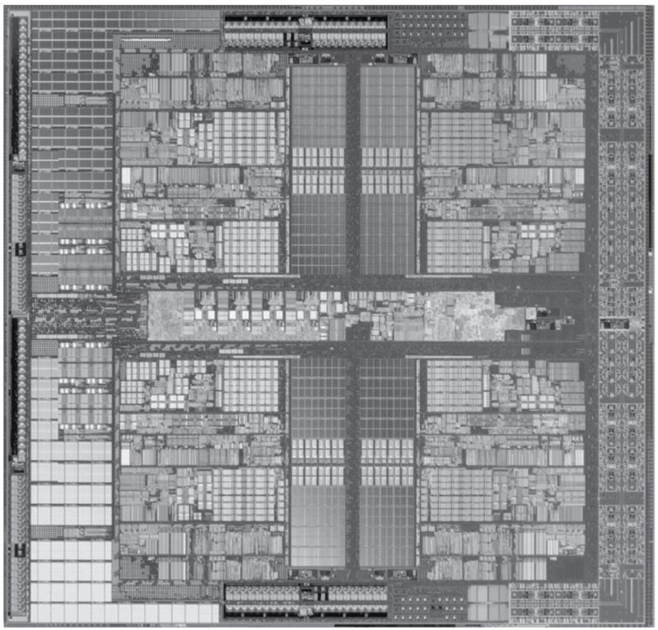
https://www.techinsights.com/blog/apple-iphone-xs-max-teardown

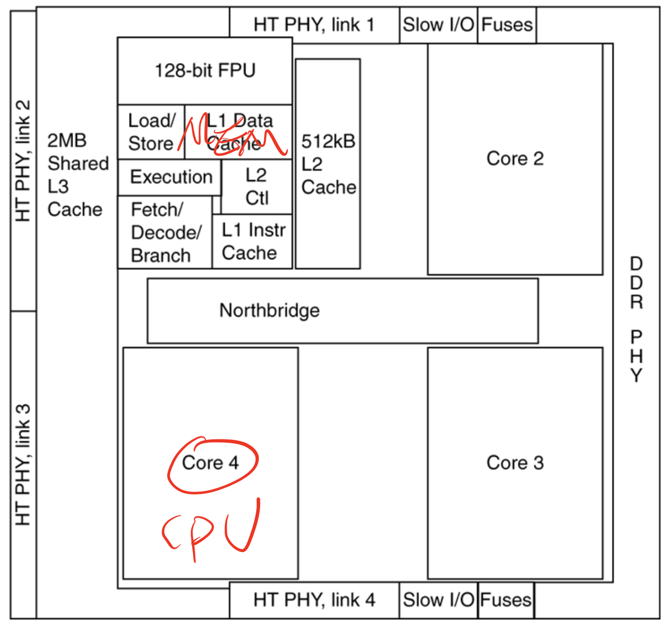
The A12 processor



AMD Barcelona

- 4 processor cores





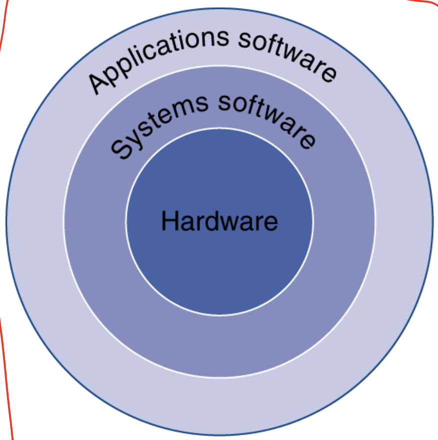
Datapath, Control, Cache, etc.

Question

- What do users see when they use computer?

Below Your Program

- Application software ⟺ users
  - Written in high-level language (HLL)
  - Or scripting language
- System software
  - Compiler: translates HLL code to machine code
  - Operating System: Scheduling tasks & sharing resources
    - Providing services to applications
    - Managing memory and storage
    - Handling input/output
- Hardware
  - Processor, memory, I/O controllers



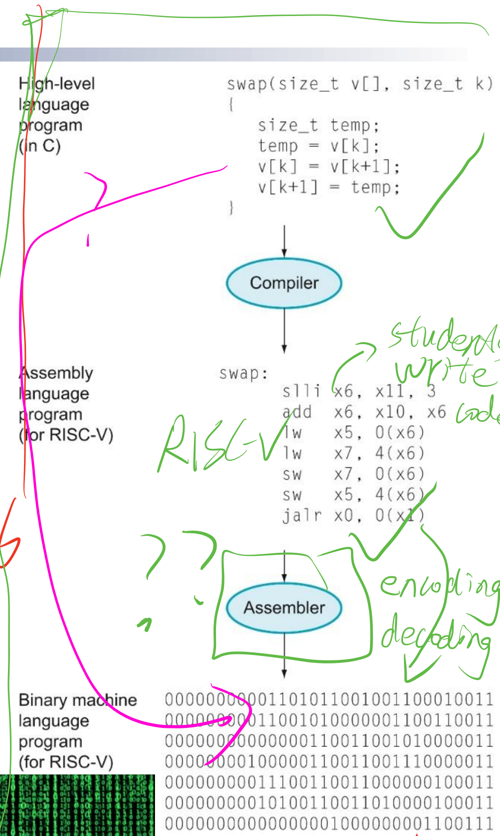
There is abstraction between layers

Levels of Program Code

- High-level language
  - Level of abstraction closer to problem domain
  - Provides for productivity and portability
  - Converted to assembly language by compiler
- Assembly language
  - Textual representation of instructions
  - Converted to machine code by assembler
- Machine code (for computers)
  - Representation stored in hardware
  - Instructions and data are encoded with bits

CSE 3666

bit { 0 1



The matrix

Instruction

- An instruction is a basic operation that software can perform on a processor
  - It is not called command

Roads ahead

VS CISC

(Reduced Instruction Set Computer) (2010)

- RISC-V instructions
  - How do processors run programs?
- Arithmetic
  - \+ − × ÷
  - Digital logic for addition, subtraction, multiplication and division
  - Floating point numbers −∞ +∞
- Computer performance
  - What determines program performance and how it can be improved
  - → Exec Time
- Implementation of processors
  - From single cycle to pipelined
- Cache

computer system?

builder

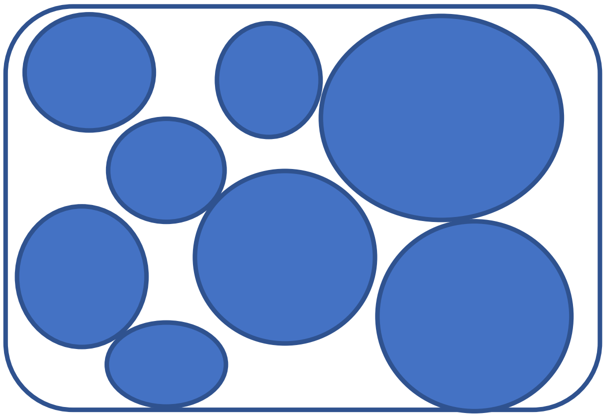
Solving problems and design solutions

How things work and how to make them better

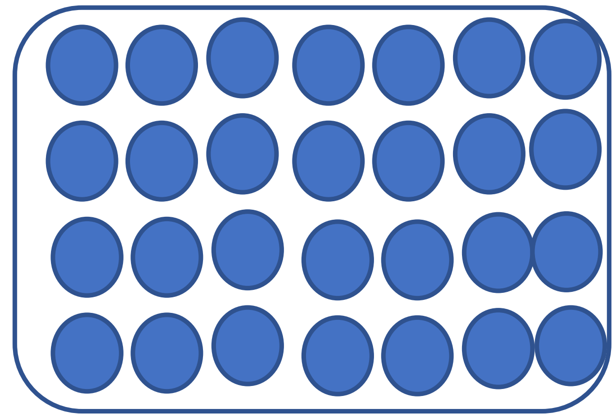
Design and optimization principles

Why is Computer Architecture Hard to Learn?

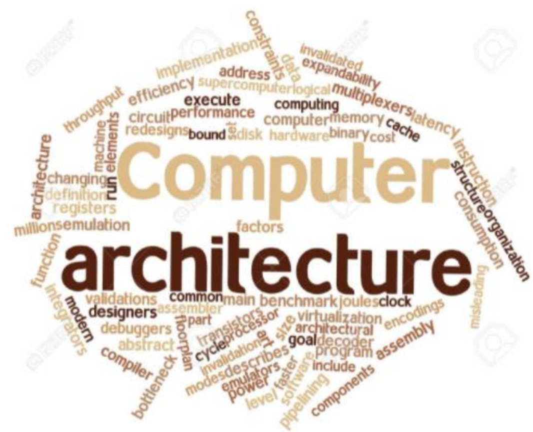
Some other class



Computer Architecture



Many new concepts!!!

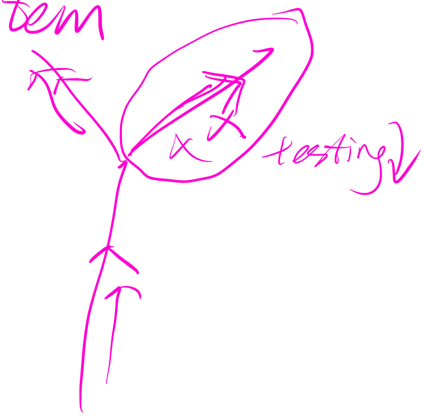


Source: www.123rf.com

Seven Great Ideas/Design Principles

- Use abstraction to simplify design
- Make the common case fast
- Performance via parallelism CPU, core
- Performance via pipelining
- Performance via prediction
- Hierarchy of memories RAM, SSD, cloud
- Dependability via redundancy

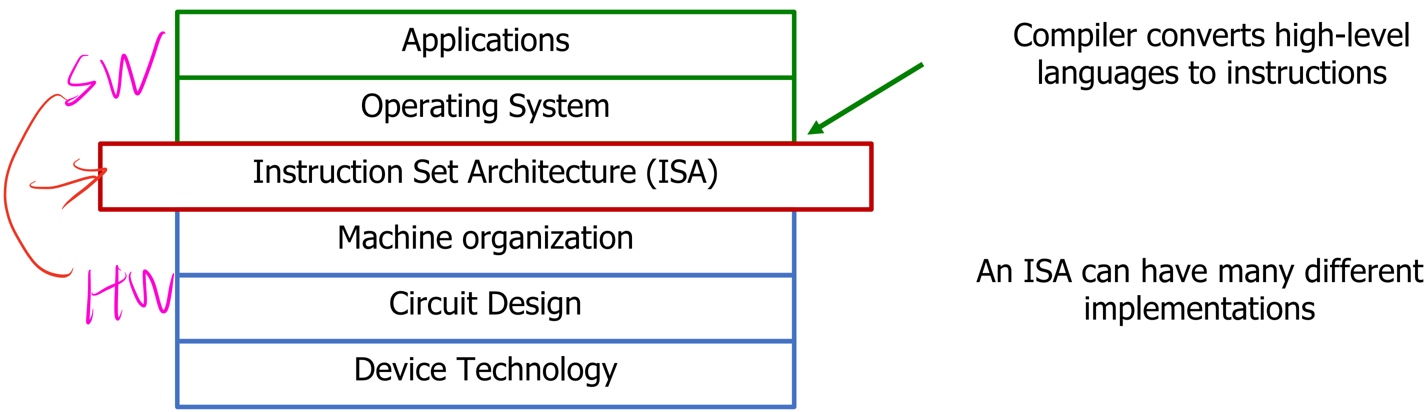
computer system



Read Section 1.2

Instruction Set Architecture (ISA)

- ISA defines the instruction set a processor understands
- ISA provides a level of abstraction for the hardware and software
  - It is the interface between the software that runs on a computer and the hardware that executes it



Family of ISAs

- A "family" of microprocessors understands the same instructions
  - Programs can run on many different processors.
  - We have the same software for i3, i5, and i7. Benefit of abstraction!

- What ISAs have your heard of?

IBM 701	1953
CDC 6600	1963
IBM 360	1964
DEC PDP-8	1965
Intel 8008	1972
Motorola 6800	1974
DEC VAX	1977
Intel 8086	1978
Intel 80386	1985
ARM	1985
MIPS	1985
SPARC	1987
Power	1992
Alpha	1992
HP/Intel IA-64	2001
AMD64 (EMT64)	2003

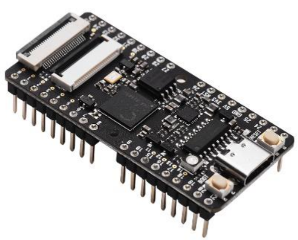
Credit to: hand-to-hand to teach the design of CPU - RISC-V processor

The RISC-V Instruction Set Architecture

2010
2015~2016

- Fifth generation of RISC design from UC Berkeley
- A high-quality, license-free, royalty-free RISC ISA specification
  - Implementors do not pay any royalties
  - Although cost of good-enough processors is dropping
- Experiencing rapid uptake in both industry and academia
- Standard maintained by non-profit RISC-V Foundation
  - 60+ members, including Google, Microsoft, IBM, Nvidia, Sun Technology

SIPEED



RISC-V Dual Core 64-bit.
< $25 at Walmart.com

Edge Devices
computing

More PCs and laptops using RISC-V cores are coming..

AI

Cool kids are building their own cores

RISC-V in this course

- RISC-V be used in many kinds of systems, from micro-controllers to supercomputers
  - 32-bit, 64-bit, and 128-bit variants

$\underbrace{00\ldots101\ldots1111}_{32\text{ bit}}$ XXXXX

- We are going to use RV32I (Base Integer Instruction Set, 32-bit)

- Extensions
  - M, Standard Extension for Integer Multiplication and Division
  - F/D, Standard Extension for floating-point operations

Computer program

Program = Data + Algorithm

You need to know both to write programs!

Always know where your data are and how you can access them

Storing Data in a Computer

- Register file
  - A set of general purposes registers ? future
    - Integers, addresses, characters, etc.
    - Programmers know the type
  - A register is a circuit that can store data
- Memory
  - Another place where data can be stored
  - View it as an array of bytes

KB

3100

MEM

GB

Register File is faster, but much smaller than memory.

Design Principle 2: Smaller is faster

Accessing register file is much faster than accessing memory

RISC-V Register File

- The register file in RISC-V has 32 registers
  - Registers are numbered from 0 to 31

`x0, x1, …, x31`

Register File is small !

x0 is always 0

Register File = 32 Registers

1 Register has 32 bits

- Each register has 32 bits (we are using RV32I)
  - A 32-bit data item called a "word"
- We use registers to keep frequently accessed data
  - Because they are fast!

Register Name and Usage

Register Number	Name	Usage
0	zero	Hardwired 0
1	ra	Return address
2	sp	Stack pointer
3	gp	Global pointer
4	tp	Thread pointer
5 - 7	t0 - t2	Temporary registers
8	s0/fp	Saved register / frame pointer
9	s1	Saved register
10 - 11	a0 - a1	Function arguments/return values
12 - 17	a2 - a7	Function arguments
18 - 27	s2 - s11	Saved registers
28 - 31	t3 - t6	Temporary registers

https://github.com/zhijieshi/cse3666/blob/master/risc-v/rv-registers.md

Arithmetic Operations

- Add and subtract

  c=a+b

  c=a-b
  - Three operands and all operands are registers
  - Two sources and one destination. The 1st operand is the destination

c a b

```
add x1, x2, x3  # x1 = x2 + x3
sub x1, x2, x3  # x1 = x2 - x3
```

- All arithmetic operations have this form
- **Design Principle 1**: Simplicity favors regularity
  - Regularity makes implementation simpler
  - Simplicity enables higher performance at lower cost

Example 1: arithmetic

a=0

```
a = b + c;   a=5
d = a - e;
```

?

Assume

Variable	Register
a	s1
b	s2
c	s3
d	s4
e	s5

Example of arithmetic instructions

$$f = (g \underset{1}{\overset{t0}{+}} h) \overset{3}{-} (i \underset{2}{\overset{t1}{+}} j);$$

Variable	Register
f	s0
g	s1
h	s2
i	s3
j	s4

- Intermediate code
  - Use t0 and t1 to keep the temporary values

```
add t0, g, h    # t0 = g + h
add t1, i, j    # t1 = i + j
sub f, t0, t1   # f = t0 - t1
```

Answer

- RISC-V code:

```
add t0, s1, s2
add t1, s3, s4
sub s0, t0, t1
```

√ → human

- We can also use register numbers, but it is harder to read
  - We will also see learn soon when to use a0, t0, or s0, and so on

```
add x5, x9, x18
add x6, x19, x20
sub x8, x5, x6
```

→ computer

Question

- How many general-purpose registers are in RV32I?

A. 16
B. 32
C. 64
D. It depends on the designer.

Question

- Is the following instruction correct?

Dest   Source , Source

```
add   x10, x9, zero
```

A. Yes

B. No

Question

- How do you set a register, say, t0, to 0?

add t0, zero, zero

Size of data items

	Number of bytes	Number of bits
Byte	1	8
Half-words	2	16
Words	4	32
Double-words	8	64