History, Processors, ISAs

Intro to Computer Systems, Fall 2022

Instructor: Travis McGaha

TAs:

Ali Krema	Andrew Rigas	Anisha Bhatia
Audrey Yang	Craig Lee	Daniel Duan
David LuoZhang	Eddy Yang	Ernest Ng
Heyi Liu	Janavi Chadha	Jason Hom
Katherine Wang	Kyrie Dowling	Mohamed Abaker
Noam Elul	Patricia Agnes	Patrick Kehinde Jr.
Ria Sharma	Sarah Luthra	Sofia Mouchtaris

Upcoming Due Dates

- HW10/11 (J compiler) to be released soon
  - HW10 & 11 make up a 2-part assignment that take a while to complete.
  - Recitation for this assignment has been VERY helpful
  - Can grant extensions on this, but there will be reduced office hours and Ed activity after a bit
  - **Took some students a long time in Fall 2021**

Lecture Outline

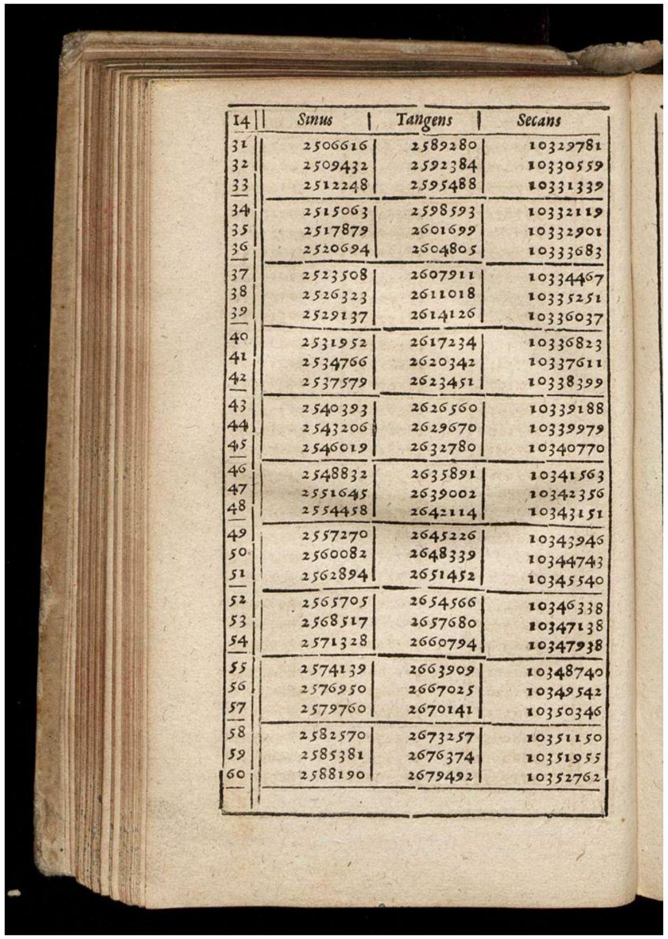
None of this is on the final exam or HW10/HW11

- Some History on Computer Systems
- Modern Processors & ISAs

Mathematical Tables

- Before non-human calculators, you'd look up the result of a computation in a table

- These were calculated by "Human-Computers"
  - Some tables (such as logarithm tables) were prone to errors. Calculations were complicated…



14	*Sinus*	*Tangens*	*Secans*
31	2506616	2589280	10329781
32	2509432	2592384	10330559
33	2512248	2595488	10331339
34	2515063	2598593	10332119
35	2517879	2601699	10332901
36	2520694	2604805	10333683
37	2523508	2607911	10334467
38	2526323	2611018	10335251
39	2529137	2614126	10336037
40	2531952	2617234	10336823
41	2534766	2620342	10337611
42	2537579	2623451	10338399
43	2540393	2626560	10339188
44	2543206	2629670	10339979
45	2546019	2632780	10340770
46	2548832	2635891	10341563
47	2551645	2639002	10342356
48	2554458	2642114	10343151
49	2557270	2645226	10343946
50	2560082	2648339	10344743
51	2562894	2651452	10345540
52	2565705	2654566	10346338
53	2568517	2657680	10347138
54	2571328	2660794	10347938
55	2574139	2663909	10348740
56	2576950	2667025	10349542
57	2579760	2670141	10350346
58	2582570	2673257	10351150
59	2585381	2676374	10351955
60	2588190	2679492	10352762

Difference Engine

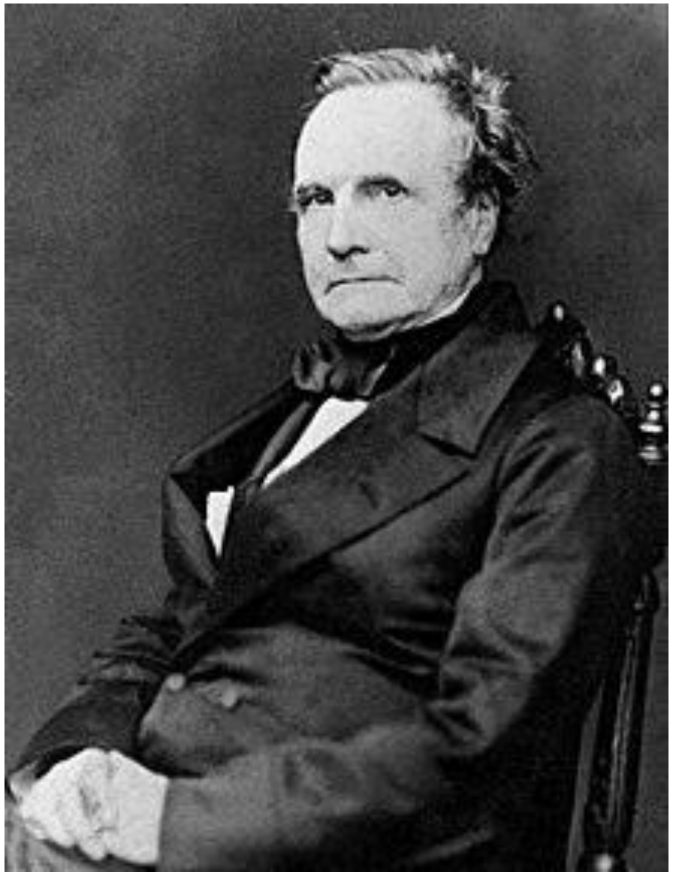
- Instead of calculating mathematical tables by hand, calculate them with an automatic mechanical calculator!
  - The Difference Engine!
- This was the inspiration for Charles Babbage
- Funding was received by British government. Human Computers were time consuming and expensive
  - This sounds a lot like the argument for automation today

First “Computer”?

- Was Charles Babbage the first person to design mechanical “computers”?
- Antikythera mechanism 205-87 BC
- Adding machines
- Johann Helfrich von Müller
  - First to come up with the idea of calculating mathematical tables by machine. (1786)
    - Difference Engine
  - Plan was never funded

Analytical Engine

- The first proposed general-purpose computer
  - First described in 1837
  - First Turing Complete computer!
  - Never built ☹
- Babbage turned interest away from the Difference Engine
- British government stopped funding
  - Was more interested in seeing the output of the engine, not the development of such machines



Ada Lovelace

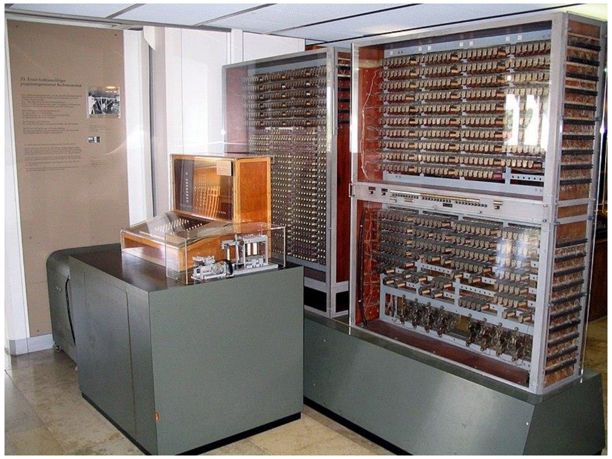
- Worked with Charles Babbage on the analytical engine
- Described an algorithm to computer Bernoulli numbers
- “First” programmer is disputed.
  - Babbage already had some “programs”
  - She is the first to *publish* a program
- Was first to see the potential of computers. As something that would act on more than numbers

The Analytical Engine Influence

- Had basic architecture very similar to modern machines
  - Instruction based
  - Had an I/O unit
  - Turing complete

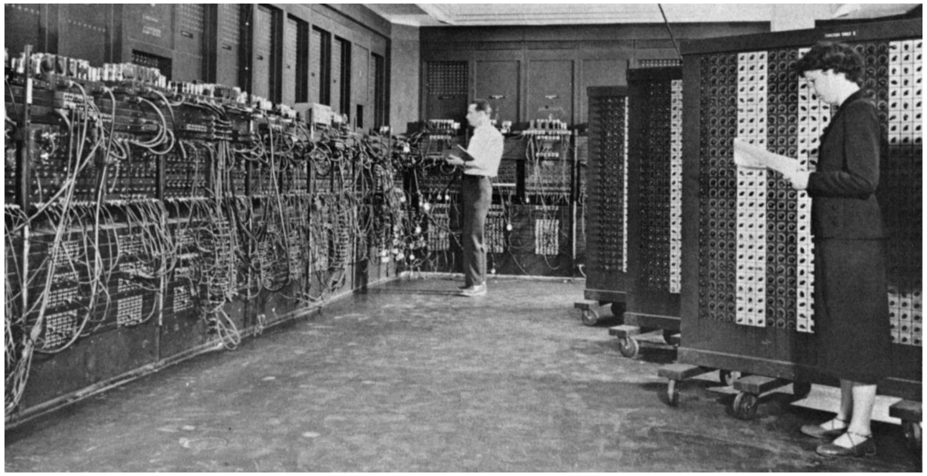
- Would be unknown to the builders of computers in the 1930’s and 1940’s

- First built general purpose computer The Z3 in 1941
  - Over 100 years since the analytical engine was first described



ENIAC

- Electronic Numerical Integrator and Computer
  - First programmable, electronic, general-purpose digital computer
  - Being entirely electronic instead of electro-mechanical made it ~1000 times faster than other machines
- First used to calculate feasibility of thermonuclear weapons and artillery trajectories
- Programmed "manually" with switches.
- Original programmers were 6 women. Did not get recognition until mid-1980s.

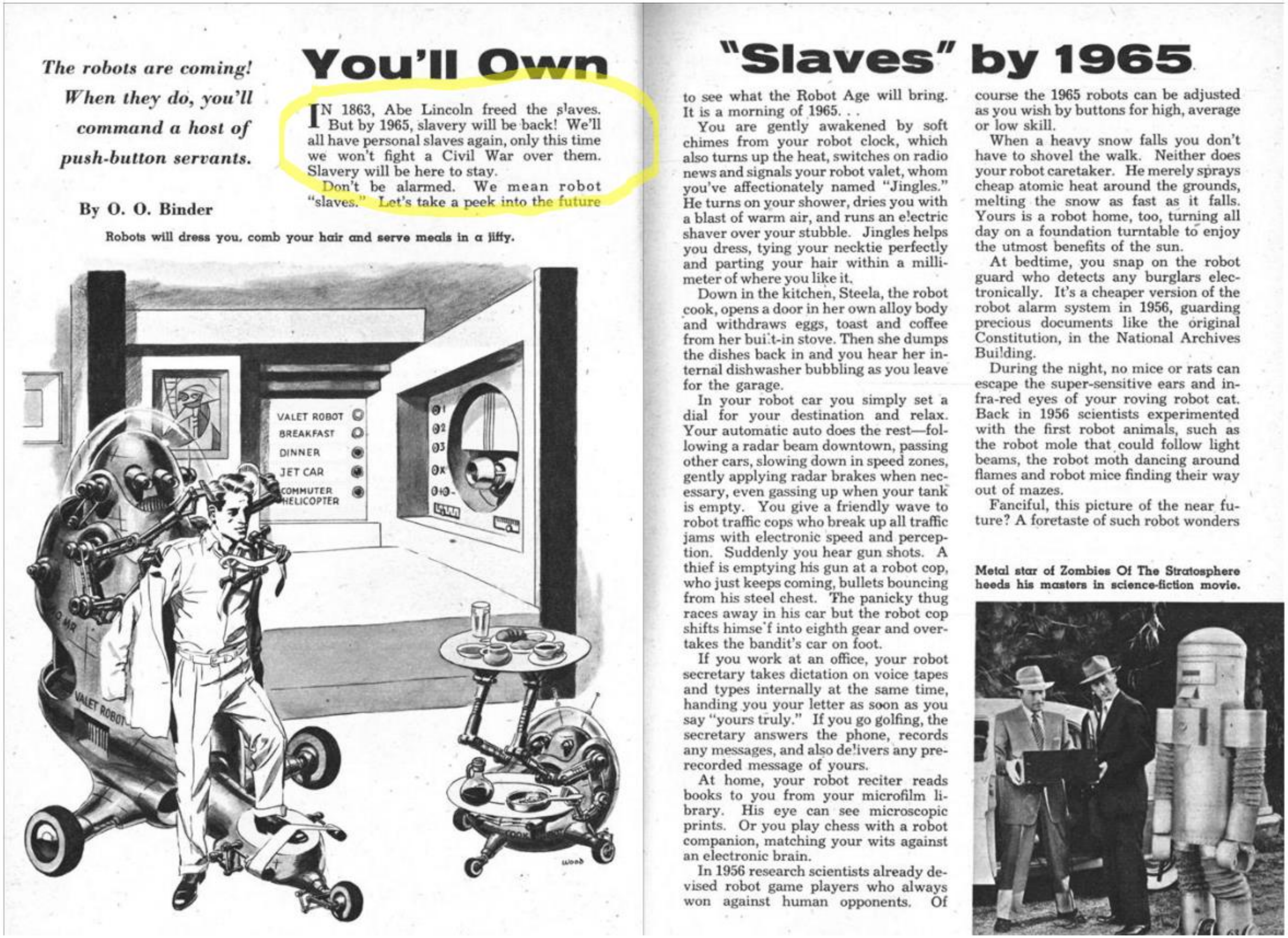


Tangent: Terminology

- Another term for a D-flip-flop is "*D master-slave flip-flop*"
  - Unsurprisingly, this term raises ethical concerns

- This term is/was used in multiple technological applications
  - Typically used to convey the concept of a "free master that did no work, and a slave that followed the mater's orders"

- Master-slave has been used as a term in technology since 1904



The robots are coming! When they do, you'll command a host of push-button servants.

You'll Own "Slaves" by 1965

By O. O. Binder

IN 1863, Abe Lincoln freed the slaves. But by 1965, slavery will be back! We'll all have personal slaves again, only this time we won't fight a Civil War over them. Slavery will be here to stay.

Don't be alarmed. We mean robot "slaves." Let's take a peek into the future to see what the Robot Age will bring. It is a morning of 1965. . .

You are gently awakened by soft chimes from your robot clock, which also turns up the heat, switches on radio news and signals your robot valet, whom you've affectionately named "Jingles." He turns on your shower, dries you with a blast of warm air, and runs an electric shaver over your stubble. Jingles helps you dress, tying your necktie perfectly and parting your hair within a millimeter of where you like it.

Down in the kitchen, Steela, the robot cook, opens a door in her own alloy body and withdraws eggs, toast and coffee from her built-in stove. Then she dumps the dishes back in and you hear her internal dishwasher bubbling as you leave for the garage.

In your robot car you simply set a dial for your destination and relax. Your automatic auto does the rest—following a radar beam downtown, passing other cars, slowing down in speed zones, gently applying radar brakes when necessary, even gassing up when your tank is empty. You give a friendly wave to robot traffic cops who break up all traffic jams with electronic speed and perception. Suddenly you hear gun shots. A thief is emptying his gun at a robot cop, who just keeps coming, bullets bouncing from his steel chest. The panicky thug races away in his car but the robot cop shifts himself into eighth gear and overtakes the bandit's car on foot.

If you work at an office, your robot secretary takes dictation on voice tapes and types internally at the same time, handing you your letter as soon as you say "yours truly." If you go golfing, the secretary answers the phone, records any messages, and also delivers any prerecorded message of yours.

At home, your robot reciter reads books to you from your microfilm library. His eye can see microscopic prints. Or you play chess with a robot companion, matching your wits against an electronic brain.

In 1956 research scientists already devised robot game players who always won against human opponents. Of course the 1965 robots can be adjusted as you wish by buttons for high, average or low skill.

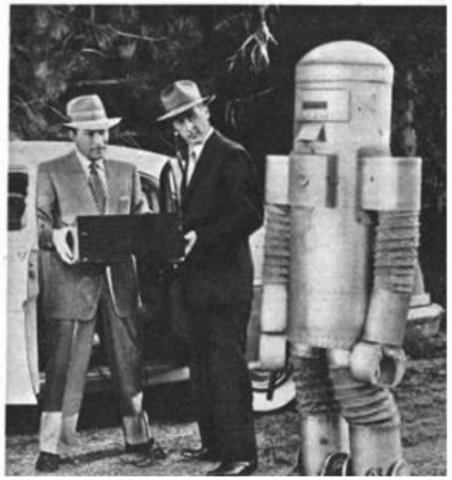
When a heavy snow falls you don't have to shovel the walk. Neither does your robot caretaker. He merely sprays cheap atomic heat around the grounds, melting the snow as fast as it falls. Yours is a robot home, too, turning all day on a foundation turntable to enjoy the utmost benefits of the sun.

At bedtime, you snap on the robot guard who detects any burglars electronically. It's a cheaper version of the robot alarm system in 1956, guarding precious documents like the original Constitution, in the National Archives Building.

During the night, no mice or rats can escape the super-sensitive ears and infra-red eyes of your roving robot cat. Back in 1956 scientists experimented with the first robot animals, such as the robot mole that could follow light beams, the robot moth dancing around flames and robot mice finding their way out of mazes.

Fanciful, this picture of the near future? A foretaste of such robot wonders

Robots will dress you, comb your hair and serve meals in a jiffy.


Metal star of Zombies Of The Stratosphere heeds his masters in science-fiction movie.

Tangent: Terminology

- Many new terms have been proposed (usage varies on context)
  - Primary – Replica, Writer – Reader, Producer – Consumer, …

- Is it a problem in contexts where only “master” is used?
  - E.g. a “Master” branch in git?
    - In this context, Master could mean “Master Copy”

- Similar issues with “Whitelist” and “Blacklist”?

- Changing these terms is good… but we shouldn’t forget about other systemic issues

Early Programming

- The first programs were not stored "in the computer"
  - Often, programs were manually "set" with switches, wires, etc.
- Kathleen Booth is credited with inventing assembly languages in 1947 while working on the ARC2
- Higher Level programming languages were developed around the 1950s
  - FORTRAN in 1954 was the first widely used high-level general-purpose programming language
- ASM and programming language developments were made to "lower the cost of programming"

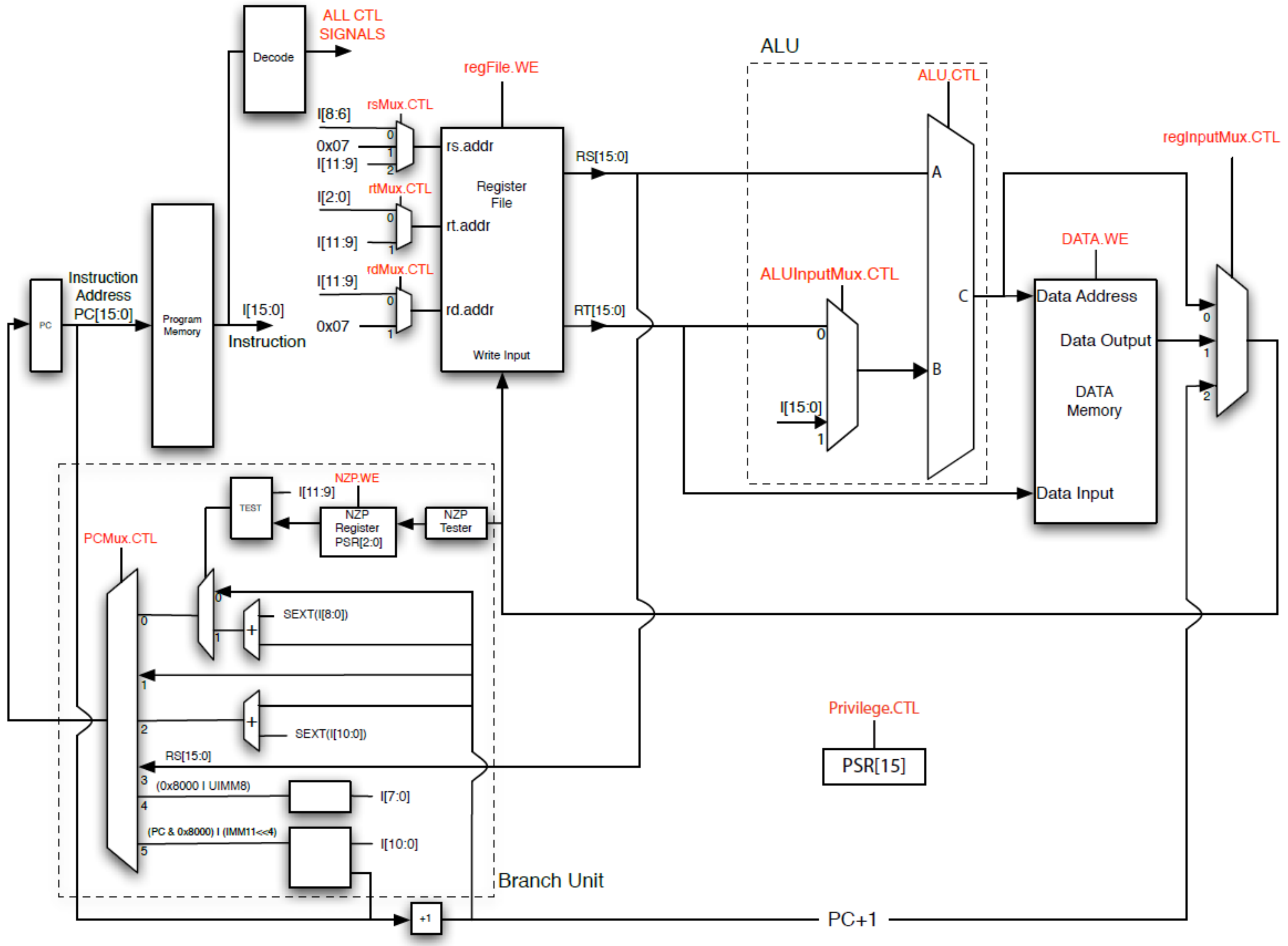
Lecture Outline

None of this is on the final exam or HW10/HW11

- Some History on Computer Systems
- Modern Processors & ISAs

Remember This?

Single Cycle Implementation of the LC4 ISA



“Single Cycle”

- In the past I’ve been talking about processor with the term “Single Cycle”
  - This means that one instruction is executed in one clock cycle.
  - That means the length of the program is directly proportional to the number of instructions executed

- “Single Cycle” is a convenient way for programmers to think about the processor, but modern processors are **not** like this
  - LC4 stands for “Little Computer 4”, provides a nicer environment to get used to systems concepts

Single Cycle Clock Speed

- When we guarantee each instruction completes in one cycle, the clock period must be timed to make sure the longest instruction still completes in one clock cycle
  - Instruction execution time is affected by the number of gates and the propagated gate delay

- Hypothetical Example:
  - Most LC4 instructions take 1 ns to fully execute
  - The longest instructions take 5 ns to fully execute
  - Clock period must be at least longer than 5 ns for our single cycle processor
  - In this case, most instructions will cause us to "waste" ~4ns of each clock cycle

Component Usage

- Our computer at this point is made of many components

- Not all components are used at all times
  - While an instruction is being fetched from memory only the program memory is being used
  - While an instruction is still being decoded, the control signals haven't been set properly and rest of the hardware is not being used for computation
  - Some instructions don't use all components
  - Etc.

Instruction Level Parallelism

- When multiple instructions are executed on hardware at the same point in time

- Many related concepts:
  - Pipelining
  - Superscalar execution
  - Out of Order Execution
  - Speculative Execution
  - …
  - More in CIS 4710 ☺

Pipelining

- What if we broke an instruction into stages and did one stage per clock cycle?
  - Example: Fetch -> Decode -> Compute -> Update Registers

- While one instruction is being fetched, another instruction is being decode, another is being computer, and another is updating registers

- We still complete one instruction per clock cycle, but are making progress on multiple instructions per clock cycle
  - Clock can be faster since we only need to consider maximum gate delay for a single stage

Superscalar Processors

- What if we had multiple instruction decoders, ALU's, and could fetch multiple instructions at a time?
  - We could run multiple instructions at the same time!

- Superscalar (and pipelining) have dependency issues:
  - Some instructions depend on the output of previous instructions, how do we know if we can execute two instructions at the same time and get the same output?

Additions are independent,
Could compute both at the same time

```
ADD R3, R2, R0
ADD R6, R7, R5
```

Second addition is dependent on first,
cannot run both in parallel

```
ADD R3, R2, R0
ADD R6, R7, R3
```

Modern ISAs

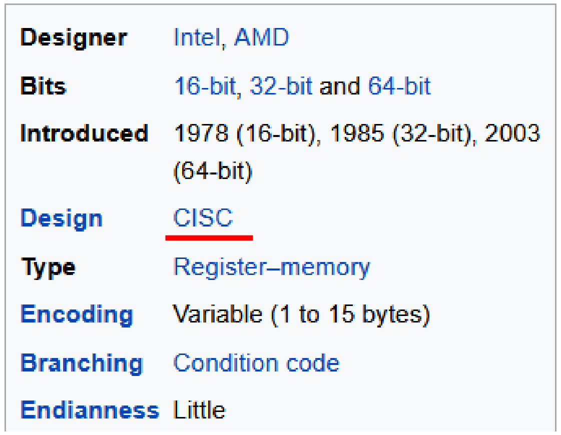
- Hardware is designed to support or "implement" a specific instruction set

- You can't really talk about modern processors without talking about modern Instruction set architectures

Modern ISAs

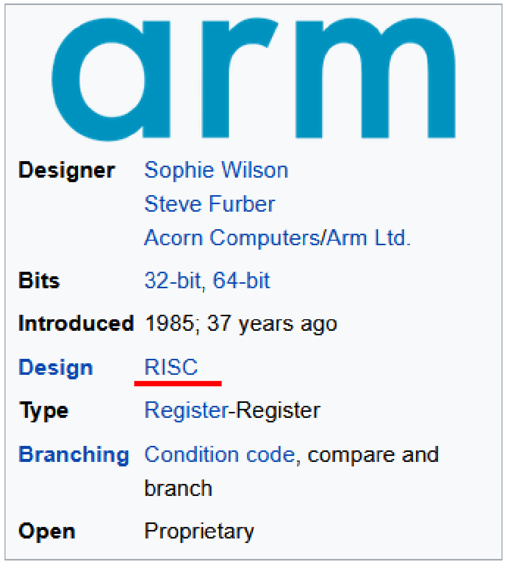
- There are other ISAs but the three most popular ISAs are:

x86



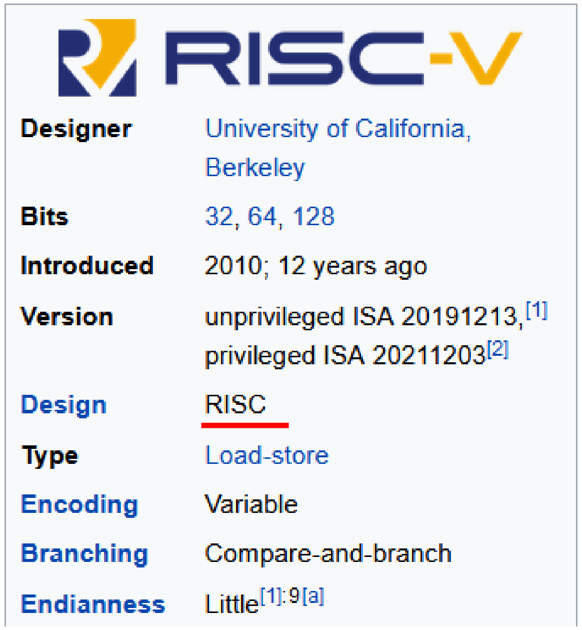
Designer	Intel, AMD
Bits	16-bit, 32-bit and 64-bit
Introduced	1978 (16-bit), 1985 (32-bit), 2003 (64-bit)
Design	CISC
Type	Register–memory
Encoding	Variable (1 to 15 bytes)
Branching	Condition code
Endianness	Little

ARM



Designer	Sophie Wilson Steve Furber Acorn Computers/Arm Ltd.
Bits	32-bit, 64-bit
Introduced	1985; 37 years ago
Design	RISC
Type	Register-Register
Branching	Condition code, compare and branch
Open	Proprietary

RISC-V



Designer	University of California, Berkeley
Bits	32, 64, 128
Introduced	2010; 12 years ago
Version	unprivileged ISA 20191213,[1] privileged ISA 20211203[2]
Design	RISC
Type	Load-store
Encoding	Variable
Branching	Compare-and-branch
Endianness	Little[1]:9[a]

I won’t go into details of writing ASM

- LC4 ASM will look similar (regarding syntax) to writing ASM in other languages.
  - This can be learned on your own
  - Would take more than this lecture to cover it all…

- LC4 : `ADD R0, R0, R1`
- ARM: `ADDS R0, R0, R1`
- RISC-V: `ADD x0, x0, x1`
- x86: `ADDQ %rdi, %rsi`

CISC and RISC

- **Complex Instruction Set Computing (CISC)**: Adds elaborate and specialized instructions to the ISA
  - This provides a lot of tools for programmers to use, but hardware needs to be able to support these instructions
  - Many of the complex instructions are not used frequently
  - x86 is CISC

- **Reduced Instruction Set Computing (RISC)**: Keep instructions relatively small and regular
  - Easier to build fast on hardware
  - Complex operations are composed of simpler operations
  - ARM, RISC-V, MIPS, LC4, etc are RISC

Instruction Complexity

- x86 is a CISC instruction set, has more complex instructions
- Example: `MOVSB` (Move byte from string to string)
  Does:
```
if (DF == 0)
   *(byte*)DI++ = *(byte*)SI++;
else
   *(byte*)DI-- = *(byte*)SI--;
```
- Building hardware to support instructions like this is more complicated to do than implementing hardware that only needs to support RISC instructions.
  - Hardware to support complex instructions could otherwise be used to make more common instructions faster

RISC vs CISC Memory Access

- In RISC, the only two types of instructions that can interact with memory are LDR and STR instructions

- CISC instructions can read/write memory in the same instruction as other operations
  - Example: `ADDQ %rdi, (%rsi)`
  - C pseudo-code: `rdi += *rsi;`

Instruction Length & Code Density

- x86 instructions are of variable length, longest ones being 15 bytes (120 bits)
- ARM instructions will be a mix of 16 bits or 32 bits

- Decoding x86 instructions is more complicated but each instruction can contain more information.
  - Programs can generally be written in fewer CISC instructions than RISC instructions.
  - Code density is useful when space to store the program is limited, also reduces time needed to go to memory to fetch an instruction

x86 Micro operations

- With more complex instructions, it is a lot harder to perform instruction level parallelism

- To help with this the x86 decoder breaks up instructions into micro-operations that can then be fed through a pipeline that supports ILP
  - These micro-operations end up being very “RISC” like

Business Model & Politics

- How the "intellectual property" of the instruction set architecture also has significant influence over which is used.
  - x86 is closed, only Intel and AMD can design and cell x86 CPUs
  - ARM is closed, owned by the ARM company who license the ISA to companies like Apple who then use it.
  - RISC-V is open, anybody can design and sell RISC-V processor. Those processors can be made open source or kept private.

Further Studies

- In the 2020's, it is unlikely you will have to write assembly once you graduate.
- If you do want to learn more about modern assembly languages, RISC instruction sets like ARM, RISC-V or MIPS will be most similar to LC4.
- Do not write LC4 on your resume, it only exists here.