ECE 571 – Advanced Microprocessor-Based Design Lecture 1

Vince Weaver

https://web.eece.maine.edu/~vweaver

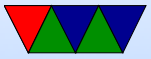
vincent.weaver@maine.edu

4 September 2024

Welcome to ECE571!

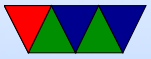
We're going to learn all about modern Microprocessors!

https://web.eece.maine.edu/~vweaver/classes/ece571_2024f/



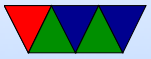
Syllabus – Instructor Info

- I'm Professor Weaver
- Go over syllabus
- QR-Code: Should you trust it?
- Office is 203 Barrows
- Tentative Office hours 11am-noon Tues/Thurs.
  Feel free to stop by if door open
- Lecture notes will be posted to website usually within a day or so



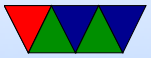
Pre-reqs / Requirements

- ECE471 or permission
  - Linux knowledge helps
  - ECE473 comp arch helps
  - I'll review a lot of this as we go
- Optional Textbook – when we review computer architecture, Patterson and Hennesey optional readings are posted. Can "check out" for free from Umaine Library webpage.

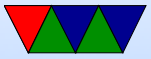


Syllabus – Grading

- 11 homeworks (5% each), one dropped
  - You will be given accounts on Linux machines. Please use them responsibly.
  - Generally due Friday be beginning of class. Will have week to do them.
  - Submit via e-mail
  - Will send out e-mail when posted to website
  - Will reply with grades
- class participation (5%)

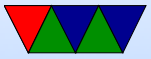


- 1 midterm exam (20% of total)
  Tentatively October 30th
- 1 final project (25% of total)
  last week of classes
  work in groups
  More details as get closer
- No final exam



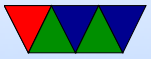
Syllabus – Late Work / Regrade

- Late work penalty. I will consider late work, but best to turn in what you have at time.
- Make regrade requests via e-mail.



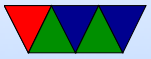
Covid/Mask Policy

- Follow UMaine Guidance
- I feel this year is more dangerous than previous years
- If you test positive for Covid please don't come to class and let me know and we can make sure you get the work done
- If you are sick for any reason but still coming to class I encourage you to wear a mask



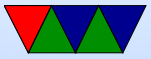
Syllabus – Academic Honesty

- Less of an issue than with other classes as we won't be coding much
- Do not copy answers from other students, either current or from previous years.
- Asking help from the professor/TA is fine
- Asking for general help, or discussing with classmates is fine
- Try to avoid giving completed assignment to someone else as a reference as in my experience it's too tempting



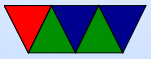
and the person will "accidentally" submit it as their own

- Just don't copy someone else's assignment and submit it as your own
  This includes cut-and-paste or retyping
- Also don't copy answers off the internet (again, looking for advice online is fine, but copying code directly is not)
- Don't use AI tools that do the homework for you! (Like Microsoft/Github Co-pilot/ChatGPT)



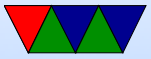
Why not AI?

- You'll note that I'm not a huge fan of AI
- Makes me unusual as it's the current fad
- You're here to become an expert on embedded systems
- AI can be subtly wrong, and you can only catch it if you actually know what's going on



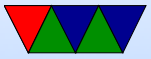
Syllabus – Boilerplate

- Go over boilerplate

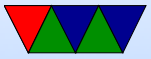


Advanced Microprocessor Based Design

- *NOT* a direct continuation of ECE471 (Embedded Systems) No blinking LEDs on embedded boards. More of a mix of 471 and 473 ideas.
- Modern CPU architecture, DRAM, GPU, storage
- Power and Energy concerns on modern systems.
- Will involve some computer architecture. Don't worry if not a Computer Engineer, will try to review completely.
- Will involve reading some papers.
- Will involve logging into Linux servers and running

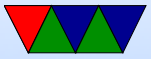


experiments.



Modern CPU Related Topics

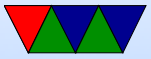
- Modern CPUs (x86? Intel? AMD? ARM? RISCV?)
- Memory (DDR4/DDR5?), NVRAM
- Disk (SSD)
- Graphics (GPUs)



Advanced Microprocessor Based Design

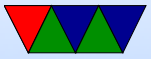
What is an Advanced Microprocessor?

- Desktop?
- Server?
- Supercomputer?
- Embedded?
- They are all converging.



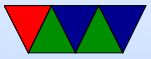
Moore's Law

- Memory Wall
- Power Wall
- Tiny tiny transistors
- More and More Cores
- Something's Got To Give

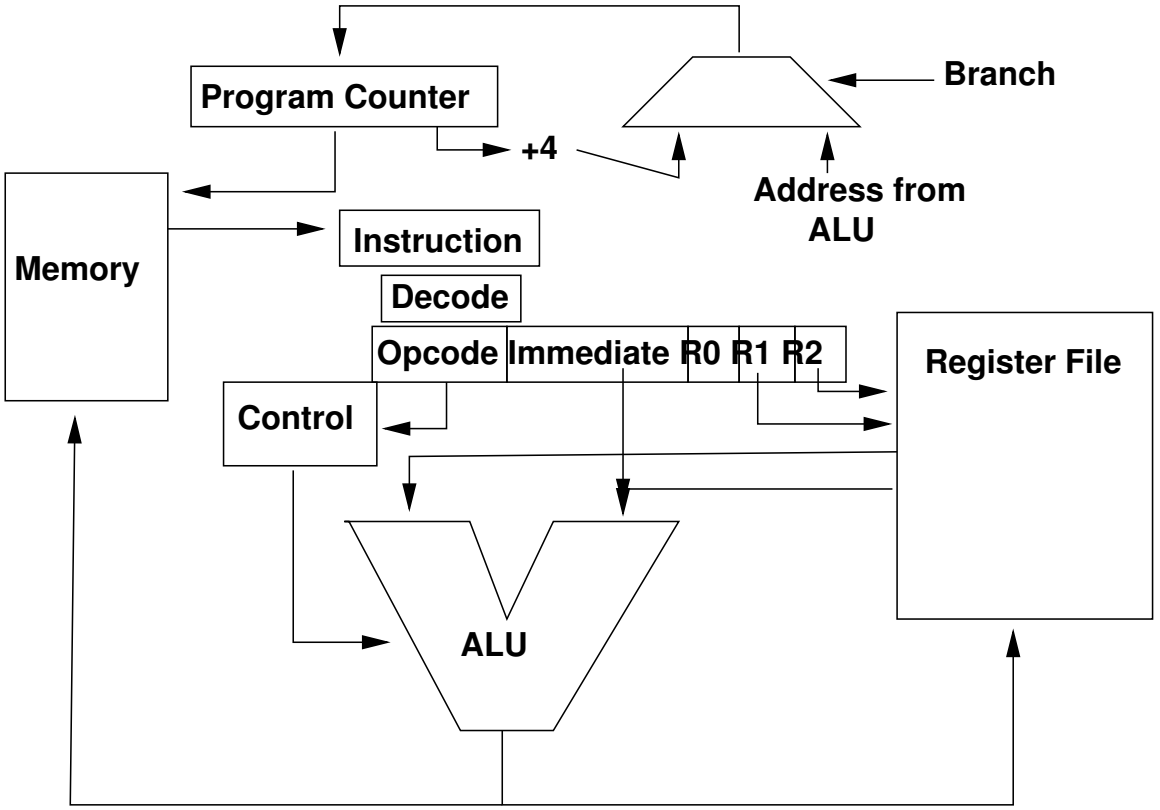


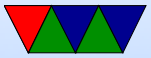
Microprocessors

- Also known as Central Processing Unit (CPU)
- Do the general purpose calculations in a system
- Originally big, multi-cabinet, multi-board, multi-chip
- The first "micro" processor fit on one chip.
  Often regarded as the 4-bit Intel 4004. (history?)
- In the old days you could buy a discrete CPU, plop onto circuit board, hook up some memory and a terminal, and you had a computer.
- These days things are a lot more complex.



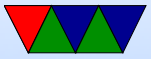
Simple CPU


Program Counter
+4
Branch
Address from ALU
Memory
Instruction
Decode
Opcode
Immediate
R0
R1
R2
Register File
Control
ALU



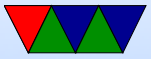
Simple CPU Notes

- Can CPU run high-level code directly?
  - First compiler change to assembly
  - Assembler change to machine code
  - This is the pattern of 1s and 0s CPU expects
- Where does code live?
  - Harvard vs Von Neumann (code and data separate vs combined)
  - Program Counter (PC) or Instruction Pointer (IP) point to next instruction

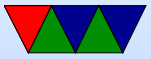


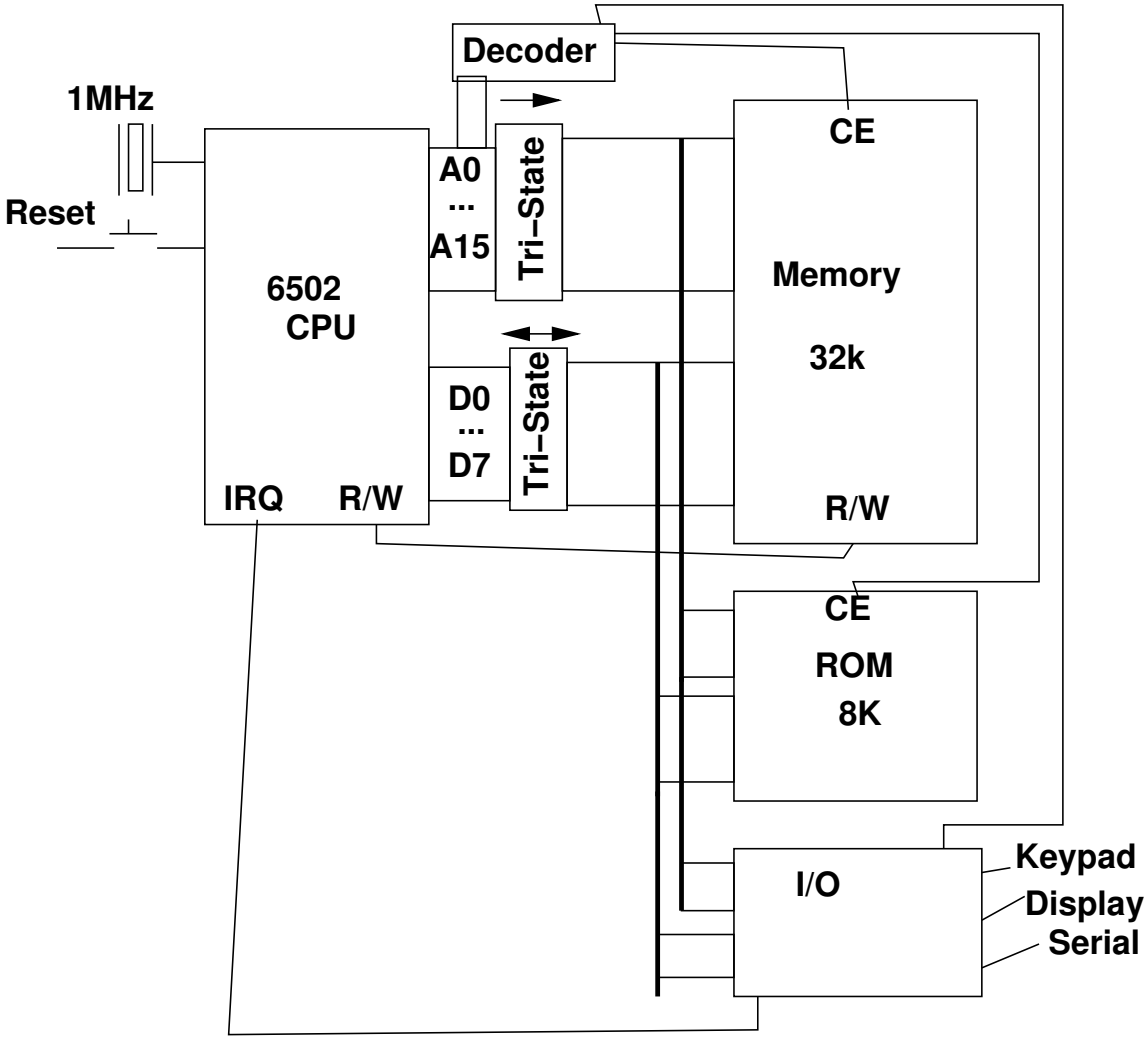
- Next instruction found with incrementer or by taking target of branch
- How do you find next instruction? Is it always 4 bytes?

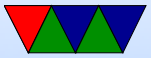
- Instruction is decoded
  - Simple architectures can easily find opcode, registers
  - Register file scratch area for fast access to small number of values
  - ALU does arithmetic and logic, results back to register file
  - Control flow (branches, jumps, function calls)
  - Load/store unit to get values from memory



Simple System




Decoder
1MHz
Reset
6502
CPU
IRQ
R/W
A0
...
A15
Tri–State
D0
...
D7
Tri–State
CE
Memory
32k
R/W
CE
ROM
8K
I/O
Keypad
Display
Serial



Describe some early simple systems

- KIM-1
- Atari 2600
- Apple II
- These all have 8-bit 6502 processors
  - Designed by Chuck Peddle, UMaine Alumnus
  - 1MHz, 3 8-bit registers
  - Up to 64k of RAM (could get more with bank switch)
  - 3500 transistors
- Modern Raptor Lake estimated 25 billion transistors