Expected / Projected / Actual Class Progression

STUPID ZOOM UI DESIGN

I apologize for the videos become out of sync towards the end of the semester. Stupidity on the part of Zoom is amazing. From one of the links here, you cannot get back to the video's title / date. That's right, you can't relate videos from here back to time / calendar.

Unbelievable.

Week 1 - 2/1

- Syllabus
- What's already assigned
- Install
- Questions
- Recording

Week 2 - 2/6 2/8

- Tuesday's Recording
- Thursday's Recording
- Apple Silicon
- Windows
- Intel Mac - get the distro, get QEMU, follow instructions for Windows except use your plain old terminal instead of WSL.
- Binary
- Powers of 2 up to 216
- Signed and Unsigned Integers
- 1's Complement and 2's Complement
- Registers
  - Integer Registers w & x
  - Why Have Registers
    - Speed of Processors Relative to RAM
  - *Up to this point was Tuesday 2/6. Thursday's class follows.*
  - Special Registers

        * Program Counter - pc
        * Stack Pointer - sp

Week 3 - 2/13 2/15

- Tuesday's Recording
- Thursday's Recording
- Floating Point Registers ***h***, s, d, v & q
  - h are half floats - not used much - are least significant half of s's
  - s are single precision values - least significant half of d's
  - d are double precision values - are least significant half of v's
  - v's are a vector of something
  - q's are a single 128 bit value
- Floating Point Construction
  - Floats / Doubles are approximations
  - Normalized scientific notation
    * Sign
    * Exponent
    * Mantissa
  - Single Precision - how above are implemented
  - Double Precision - how above are implemented
- Why Have Registers (Continued)
  - Steps Needed to Execute an Instruction
  - Pipelined Execution
- Aside:
  - Bit fields in C/C++
  - Unions in C/C++
- **Above this was covered Tuesday. Below this covered Thursday.**
- Special Registers (other than the *really* special registers)
  - Frame Pointer - x29
  - Link Register - x30
- How linking works - what is an object file

- Assembly Language!
  - `bl` branch with link (x30)
  - `ret` return (uses x30)
  - `and` bitwise and
  - `cbnz` compare and branch if non-zero
  - `cmp` compare (is actually a subtraction)
  - `b` unconditional branch
  - `.p2align` power of 2 alignment
  - `.text` what comes next is code
  - `.global` add "I have _____" to object file TOC
  - `str`, `stp`, `ldr`, `ldp` store to memory, load from memory
  - `beq` branch if the previous `cmp` is zero
  - `add` add two registers together and store result in a register
  - `mov` copy a value into a register
  - `.end` nothing else should come after this
  - `.asciz` put an ASCII string with null terminator into memory

Week 4 - 2/20 2/22

- 2/20/2024
- 2/22/2024
- **Tuesday**
  - Assembly Language
  - File descriptors
  - system calls using CRT vs making them directly
- **Thursday**
  - Assembly Language
  - Going if, for, while, continue, break

Week 5 - 2/27 2/29

- **Tuesday**
  - Review

  - 2/27/2024
  - All essays graded. 17 P1 left to grade - been quite sick so progress has been slow
  - Discuss essay
  - Common biggest error seen so far in P1 is calling write assuming that x0 through x7 are not corrupted.
    * Demonstrate `regs` a program designed to drive this point home.
  - P2 is assigned
  - Go over P2
  - `nm` demonstrated to demonstrate the "toc" i.e. the symbol table showing "have" and "need"
  - demonstrate running `as` directly
  - demonstrate running `cpp` directly
  - demonstrate asking c++ to leave behind a .S file
  - began discussion of structs
- **Thursday**
  - 2/29/2024
  - All P1 graded
  - Review
  - What is x29
  - malloc() - how it works
  - free() - how it works
  - **brief** introduction to virtual memory
    * history - none, fixed, static relocation, dynamic relocation (segmentation)
    * paging
    * linear page tables
  - P2 questions

"Spring Break"

Week 6 - 3/12 3/14

- **Tuesday**

Class canceled due to injury to instructor.

- **Thursday**
  - The Debugging Talk

    Recording

Week 7 - 3/19 3/21

- **Tuesday**
  - Review and guided coding
  - Recording
- **Thursday**
  - We begin introducing hardware concepts
  - Spinning Disks
    - Speed and latencies
    - Construction
    - RAID (part 1)
  - Recording

Week 8 - 3/26 3/28

- **Tuesday**

  RAID (part 2)

  Spaghetti Code

  Review of Calling and Making "functions"

  Recording
- **Thursday**

  No class.

Week 9 - 4/2 4/4

- **Tuesday**

  Review

  Review of calling Assembly from C and C++.

  Review of function calls and parameters.

  Writing a recursive function. Example: factorial.

Demonstration of recursion using GDB.

SSDs

Recording

- **Thursday**

  Review.

  What is the "this" pointer?

  Going over Project 4.

  Sample program doing floating point math.

  Recording

Week 10 - 4/9 4/11

- **Tuesday**

  Recording

- **Thursday**

  No class due to illness.

Week 11 - 4/16 4/18

- **Tuesday**

  Going over Project 5

  Introduced SIMD and SIMD instructions

  Please stop using direct system calls

  Introduced Intrinsics

  Discussion of which floating point registers are "safe"

  Recording

- **Thursday**

  Precomputing

  - Factorials in detailed example
  - Sin / Cos etc. in concept
  - Interpolation - not exactly Comp Org but useful knowledge - all in the context of pre-computation
    * Nearest Neighbor
    * Linear

        * Quadratic using Lagrange Polynomials
    - What's with float literals?

  Recording

Week 12 - 4/23 4/25

- **Tuesday**
    - Cache Systems

  Recording

- **Thursday**
    - Logic Gates
    - Half Adder
    - Full Adder

  Recording

Week 13 - 4/30 5/2

- **Tuesday**
- **Thursday**

Week 14 - 5/7 5/9