❖ As you wait for class to start, answer the following question:

❖ What is important in a computer? What features do you look for when buying one?

Clock Speed (GHz)

Storage — Persistent storage (TB)
— Memory (GB)
cache

OS

Price

GPU

Display

# of CPUs / Cores

Power / battery life

Type of CPU Intel, ARM

ECE/CSE 469: Computer Design and Organization

Professor Scott Hauck, ECEB-307Q, hauck@uw.edu

Office hours: email w/schedule

TA: Shruti Chakraborty shrutic2@uw.edu
Ember Chow emmychow@uw.edu
Brady Lindell bdl231@uw.edu

Office hours: up-to-date times on website

Book:

Patterson, Hennessy, *Computer Organization and Design: The Hardware/Software Interface – ARM Edition*, 2017, Morgan Kaufmann.

Grading (approximate):

20% - Homeworks   30% - Design Project   20% - Midterm   30% - Final Exam

2 weekly   Nov 5th   Dec 8   8:30

Prerequisites

Basic Logic Design and Boolean Algebra

  AND, OR, NAND, NOR gates

  Boolean Algebra

  D flip-flops, registers, and memories

  Binary numbers, 2's complement, negation, overflows

Verilog

C/C++/Java programming

If you don't know this material, <u>DO NOT TAKE THE CLASS</u>

If you don't remember this material, <u>REVIEW NOW.</u>

Joint Work Policy

The processor design and homeworks will be done in groups of 1-2.

Groups may not collaborate on the specifics of homework or on the projects.
All submitted student work must be from their own efforts, and not from any other source.
Can use Piazza to form groups.

OK:

Studying together for exams
Discussing lectures or readings
Talking about general approaches
Help in debugging, CAD tools peculiarities, etc.

Not OK:

Developing a design between groups
Implementing the CPU between groups
Checking homework answers between groups

Violation of these rules is _at minimum_:

Loss of twice the points of that assignment.
Report of Academic Misconduct to Dean's Level.
Potentially fail class, be expelled from UW.

Late Policy, Lab Policy

All homework assignments due by the end of the class period
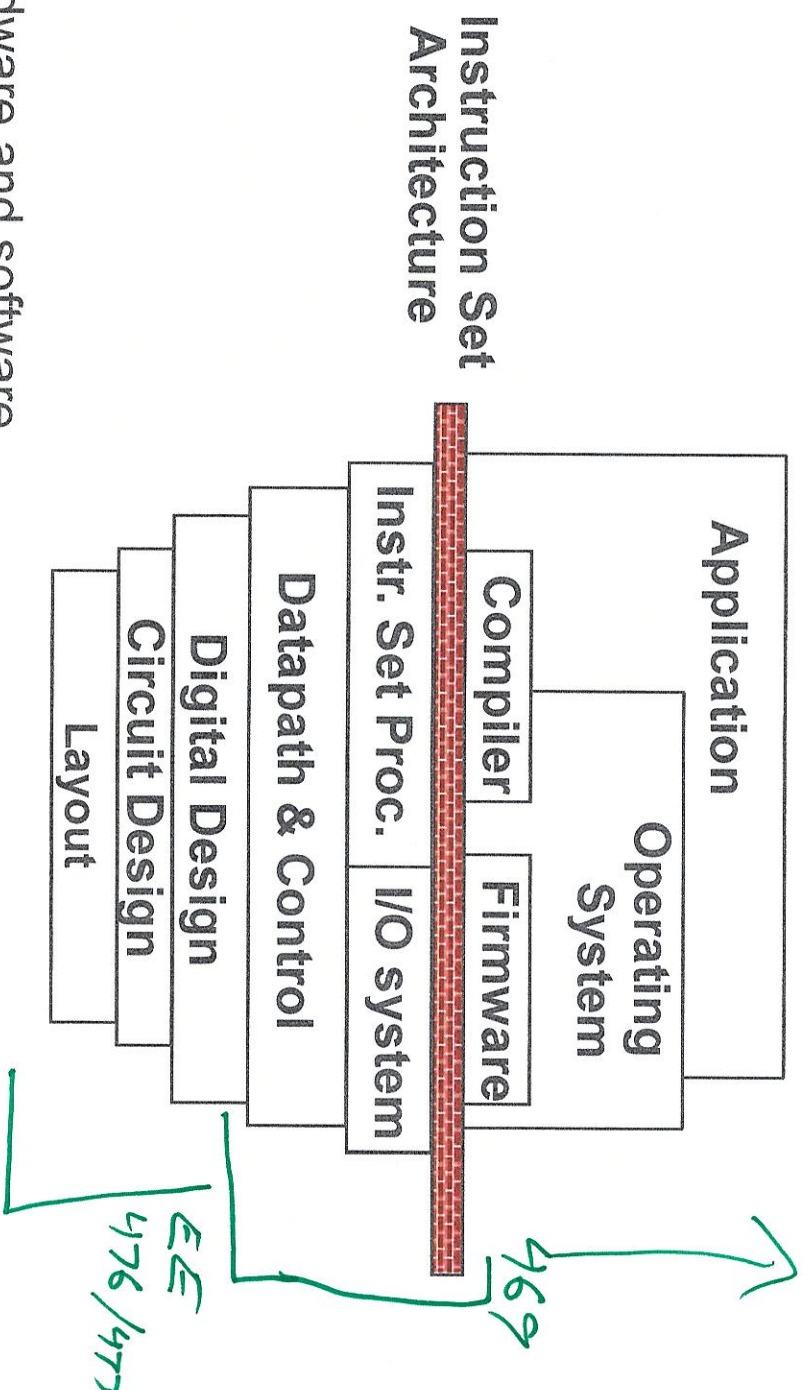
Late penalties (homework and lab):
-10% for the first 24 hours
-20% for the second 24 hours (total –30%)
-30% for the third 24 hours (total –60%)
-40% for all additional hours (total –100%)

Labs are an integral portion of the class learning. Failure to make a good-faith effort at the labs is grounds for failing the class.

Computer Architecture

Readings: 1.1-1.4

Instruction Set
Architecture

| Application |
| Operating System |
| Compiler | Firmware |
| Instr. Set Proc. | I/O system |
| Datapath & Control |
| Digital Design |
| Circuit Design |
| Layout |

469

EE
476/47

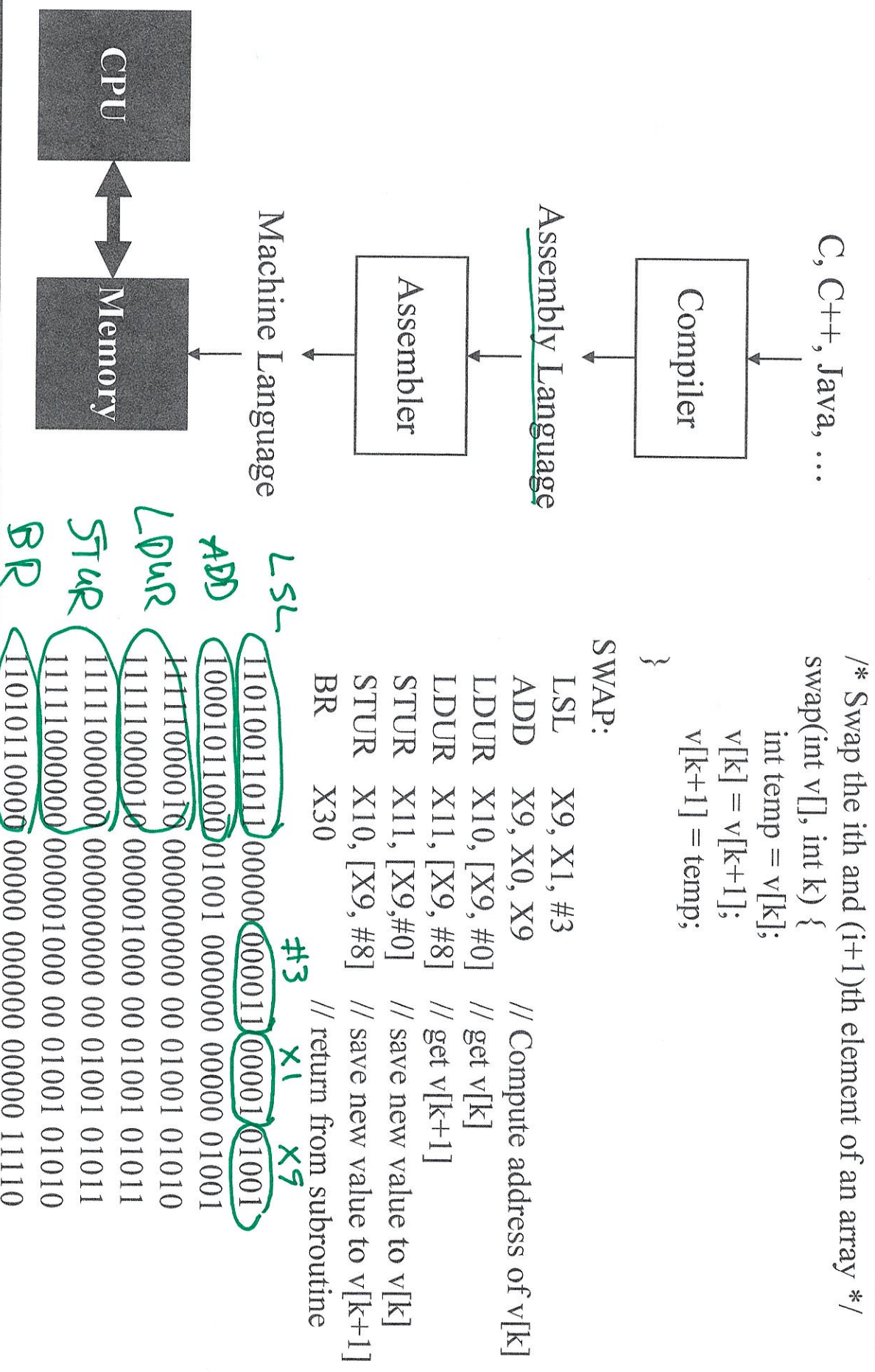Interaction between hardware and software

Hardware sets realities, requirements

Area, power, performance

Software places demands on hardware

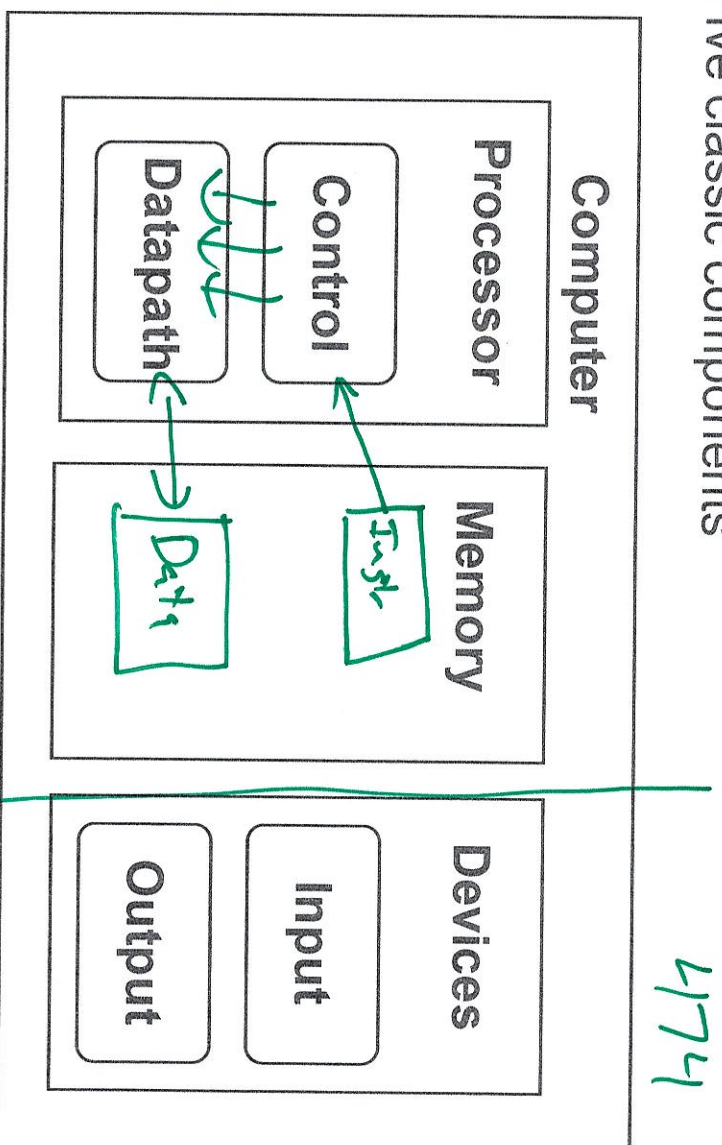Processor only as good as software it runs

Implementing Software – The Compilation Process

C, C++, Java, … → Compiler → Assembly Language → Assembler → Machine Language → Memory ↔ CPU

```
/* Swap the ith and (i+1)th element of an array */
swap(int v[], int k) {
    int temp = v[k];
    v[k] = v[k+1];
    v[k+1] = temp;
}
```

```
SWAP:
    LSL    X9, X1, #3      // Compute address of v[k]
    ADD    X9, X0, X9
    LDUR   X10, [X9, #0]   // get v[k]
    LDUR   X11, [X9, #8]   // get v[k+1]
    STUR   X11, [X9, #0]   // save new value to v[k]
    STUR   X10, [X9, #8]   // save new value to v[k+1]
    BR     X30             // return from subroutine
```

```
LSL
1101001101 000000 0000011 00001 01001
ADD
1000101011000 001001 000000 00000 01001
LDUR
11111000010 00000000 00 01001 01010
LDUR
11111000010 00000001000 01001 01011
STUR
11111000000 00000000 00 01001 01011
STUR
11111000000 00000001000 01001 01010
BR
11010110000 00000 000000 1000 00000 11110
```

#3   X1   X9

Computer Organization

Five classic components

Memory: Store instructions, data
Datapath: Perform operations (Add, subtract, ...)
Control: Orchestrate operations (who does what when)
Input: Get information from the outside world
Output: Provide results

Instruction Fetch — Obtain instruction from program storage

Instruction Decode — Determine required actions and instruction size

Operand Fetch — Locate and obtain operand data

Execute — Compute result value or status

Result Store — Deposit results in storage for later use

Next Instruction — Determine successor instruction