# CS3375: Computer Architecture

Instructor: Dr. Sunho Lim (Ph.D., Assistant Professor)

Lecture 00

*sunho.lim@ttu.edu*



## Administration

- Class Meetings:
  - M/W/F, 1:00 PM – 1:50 PM, 226 Human Science Building
- Instructor: Dr. Sunho Lim (Ph.D., Assistant Professor)
  - Office: 310 Engineering Center
  - Tentative office hours: M/W, 2:00 PM – 3:30 PM, or by appointment
  - E-mail: sunho.lim@ttu.edu
  - Class homepage,
    - **http://www.myweb.ttu.edu/slim/teaching/ttu_cs3375**
    - Check any update
- Textbook :
  - **Computer Organization and Design: The Hardware/Software Interface**, by D. A. Patterson and J. L. Hennessy, 5th edition, Morgan Kaufmann
  - Additional references or materials will be included in the lecture note

# Course Description

- Descriptions:
  - This course is an introduction to computer architecture concepts and recent developments.
  - Key topics will include computer abstractions and technology trends, performance models, computer arithmetic, organization and architecture of pipelined processor systems, instruction level parallelism, memory organization, and fundamentals of multiprocessor systems and parallelism.
- Prerequisites:
  - CS2350: Computer Organization and Assembly Language Programming



# Tentative Course Outline

- **Ch. 1: Computer Abstractions and Technology**
  - Performance
- **Ch. 2: Instructions: Language of the Computer**
  - Operations, Operands, and MIPS
- **Ch. 3: Arithmetic for Computers**
  - Addition, Subtraction, Multiplication, and Division
  - Floating Point
- **Ch. 4: The Processor**
  - Datapath and Control
  - Pipelined Datapath and Control
  - Data Hazards and Control Hazards
- **Ch. 5: Large and Fast: Exploiting Memory Hierarchy**
  - Caches and Cache Performance
  - Memory Hierarchy



# Methods of Assessment

- No make-up exam!
- 1st Midterm Exam: 15%
  - **Feb 24th (Monday) during the class**
- 2nd Midterm Exam: 15%
  - **Mar 30th (Monday) during the class**
- Final Exam: 20%
  - **May 7th (Thursday), 1:30 PM – 4:00 PM (226 Human Science Building)**
- Quiz/Review: 20%
  - Quiz/Review may NOT be announced in advance
  - There will be NO make-up quiz/review
- Homework/Project: 30%
  - Homework/Project should be turned-in IN CLASS on the due date. NO late homework will be receive



# In addition,

- **NO** laptop/electric-device/cellphone is used during the class
- In case of missing class,
  - Send me an email later
- **NO** phone call discussion on programming homework
- **NO** chatting/ **NO** yawning ☺
- Utilize office hours
  - Instructor: **M/W 2:00 PM - 3:30 PM, 310 ENG CTR**
  - Grader: TBA



# Computer Abstractions and Technology

Instructor: Dr. Sunho Lim (Ph.D., Assistant Professor)

Lecture 01

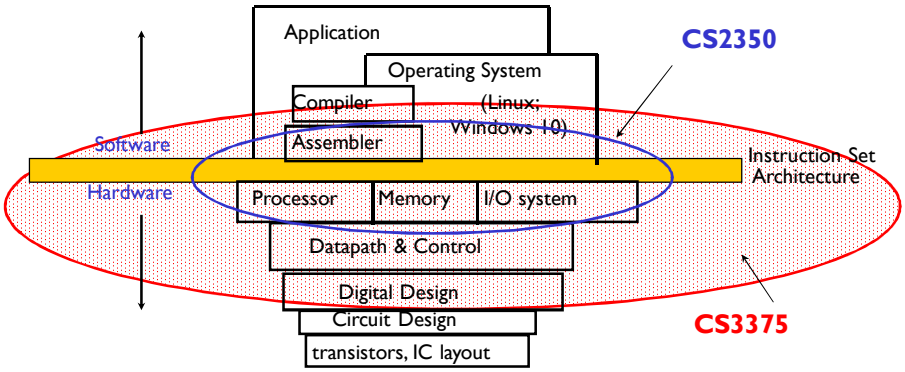
*sunho.lim@ttu.edu*

*Adapted partially from Computer Organization and Design, Patterson & Hennessy and from Prof. Mary Jane Irwin at The Pennsylvania State University*





## Overview





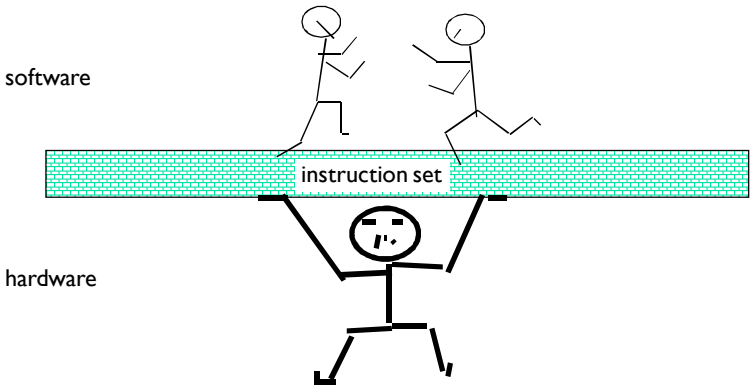

# What is "Computer Architecture"?

- Computer Architecture =
  - **Instruction Set Architecture (ISA):**
    - The one "true" language of a machine,
    - Boundary between hardware and software,
    - The hardware's specification,
    - Defines "what" a machine does
  - **Machine Organization:**
    - The "guts" of the machine,
    - "how" the hardware works,
    - The implementation,
    - Must obey the ISA abstraction



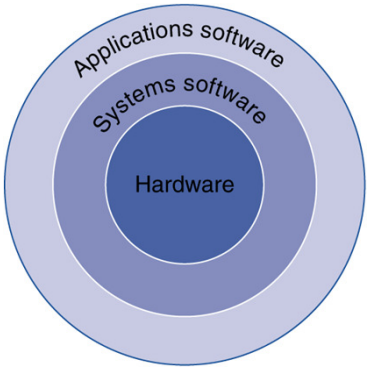


# ISA: Critical Interface



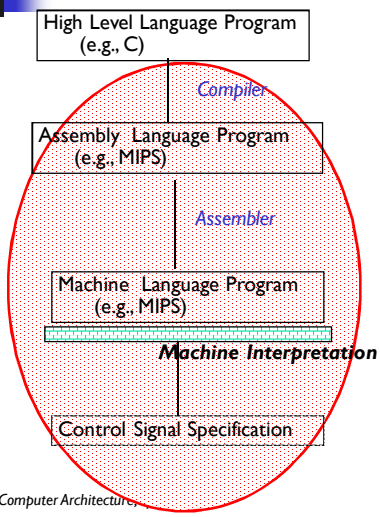
# Below Your Program…

- Application software
  - Written in high-level language
- System software
  - Compiler: translates HLL code to machine code
  - Operating System: service code
    - Handling input/output
    - Managing memory and storage
    - Scheduling tasks & sharing resources
- Hardware
  - Processor, memory, I/O controllers






# Levels of Abstraction




```
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;
```

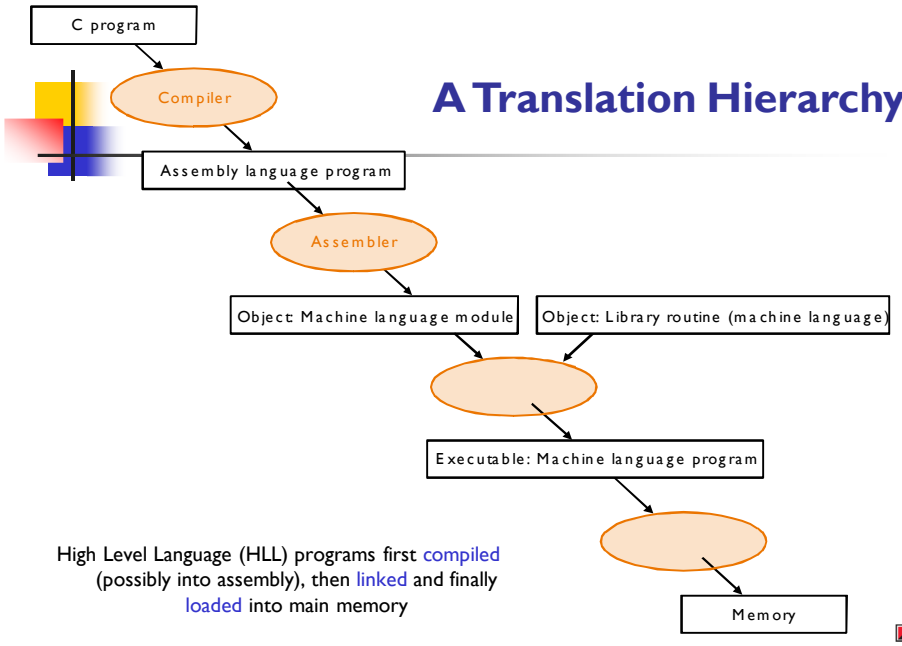
```
lw  $15,  0($2)
lw  $16,  4($2)
sw  $16,  0($2)
sw  $15,  4($2)
```

```
0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101 1000 0000 1001
0101 1000 0000 1001 1100 0110 1010 1111
```

```
ALUOP[0:3] <= InstReg[9:11] & MASK
```



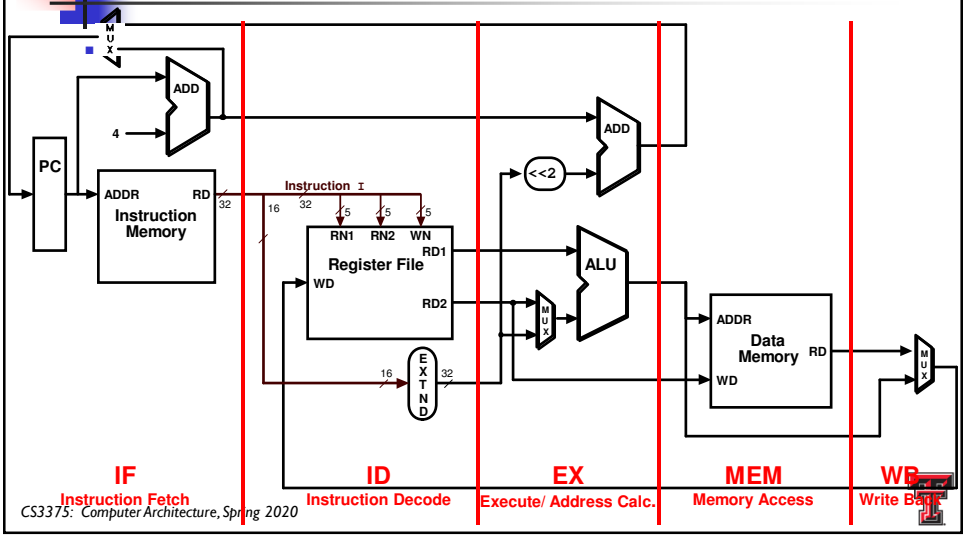
# A Translation Hierarchy

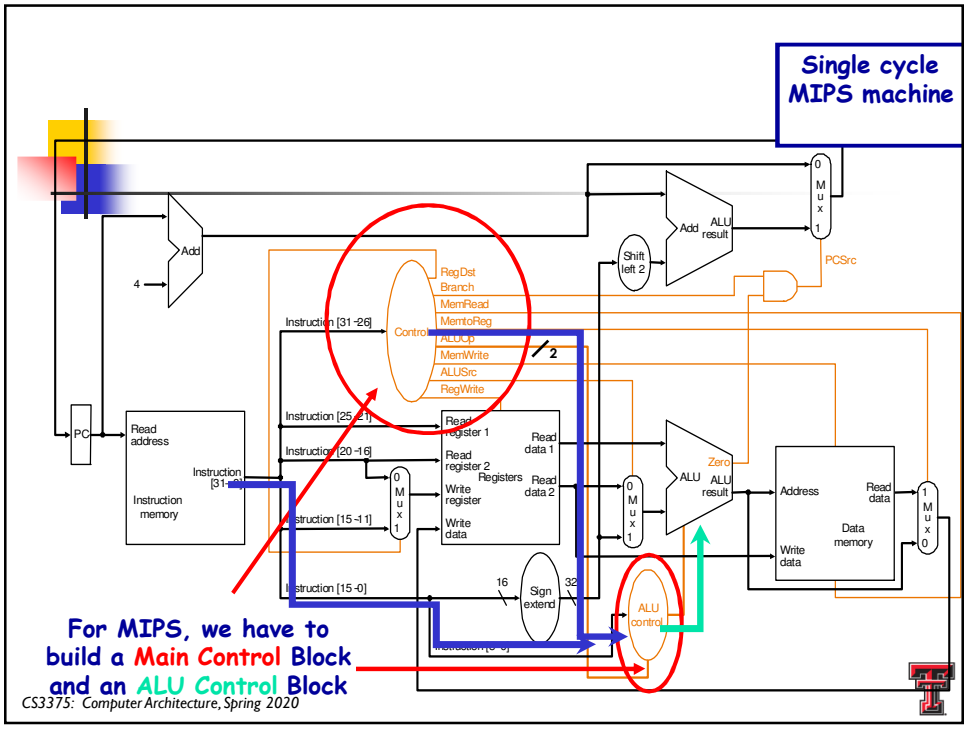



High Level Language (HLL) programs first compiled (possibly into assembly), then linked and finally loaded into main memory



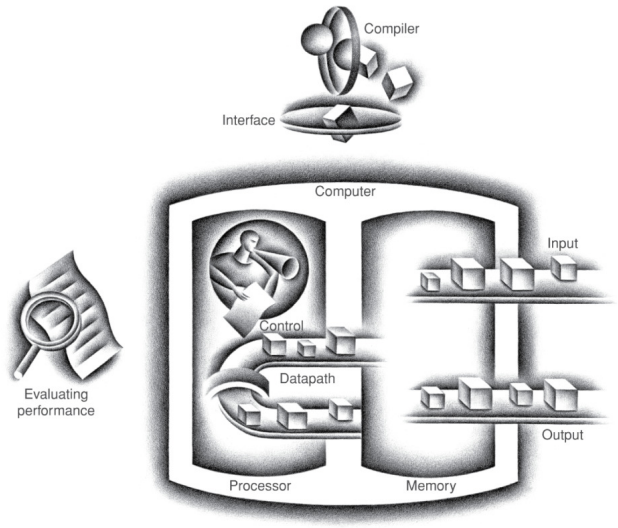
# How to really implement???
# Datapath



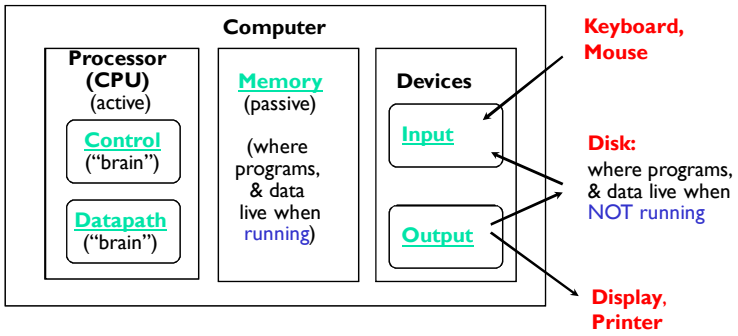





# Components of a Computer
## The Big Picture



# Machine Organization: 5 classic components of any computer




The components of every computer, past and present, belong to one of these five categories.



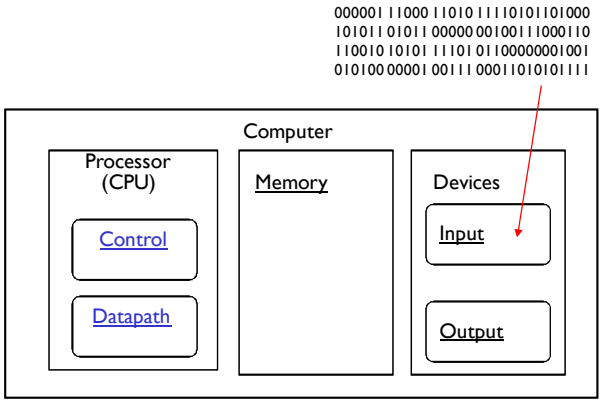
# Processor Organization

- **Control** needs to have the,
  - Ability to input instructions from memory
  - Logic and means to control instruction sequencing
  - Logic and means to issue signals that control the way information flows between datapath components
  - Logic and means to control what operations the datapath's functional units perform
- **Datapath** needs to have the,
  - Components needed to execute instructions
    - Functional units (e.g. adder) and storage locations (e.g., register file)
  - Components interconnected so that the instructions can be accomplished
  - Ability to load data from memory, and store data to memory



# Machine Organization (cont.)

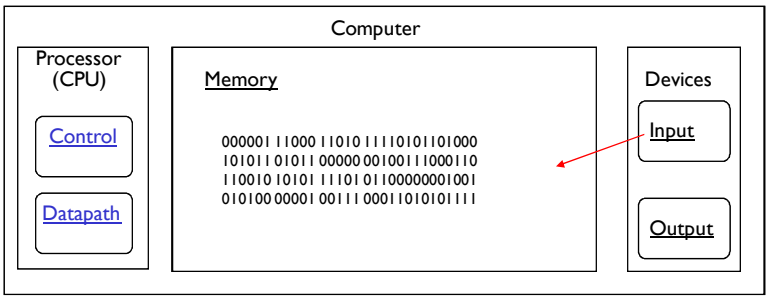
- Input device inputs object code






# Machine Organization (cont.)

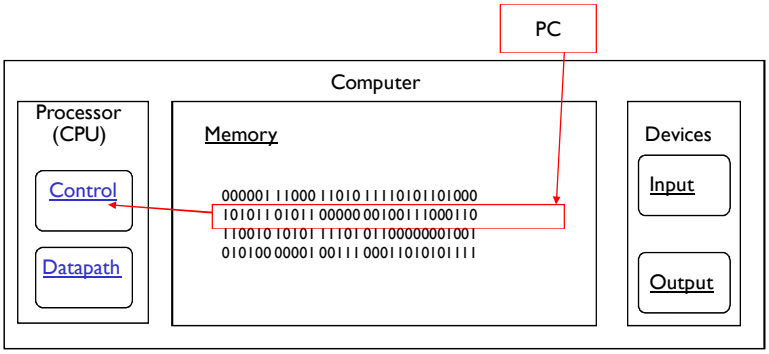
- Object code is stored in memory






# Machine Organization (cont.)

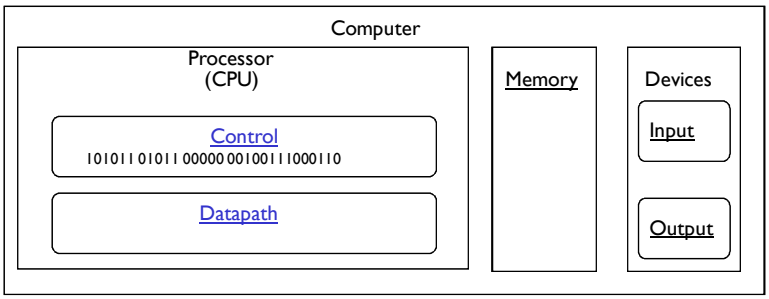
- Processor fetches an instruction from memory






# Machine Organization (cont.)

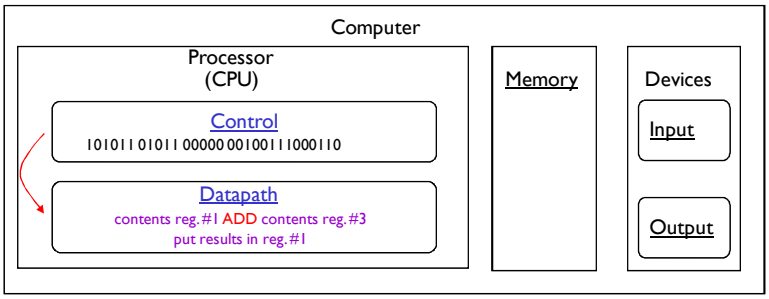
- Control decides the instruction to determine what to execute






# Machine Organization (cont.)

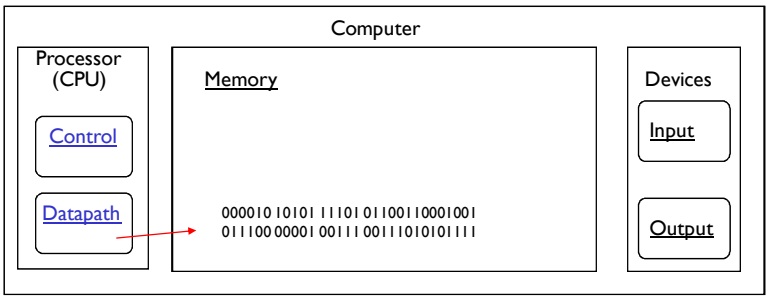
- Datapath executes the instruction as directed by control






# Machine Organization (cont.)

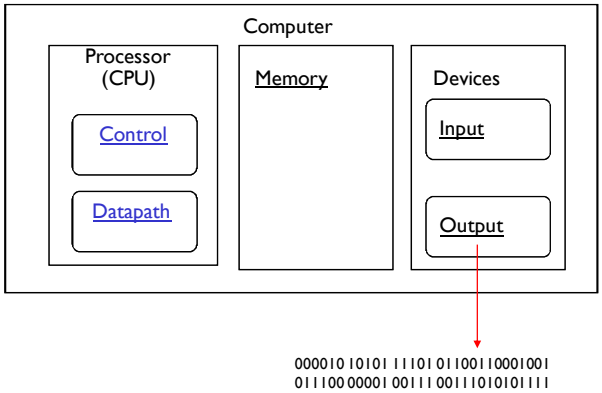
- Output data stored in the memory






# Machine Organization (cont.)

- Output device outputs data






# R(Reduced)ISC Vs. C(Complex)ISC

- **RISC philosophy**:
  - Fixed instruction lengths
  - Limited addressing modes
  - Limited operations
    - e.g. **Intel(Compaq)**, MIPS, Sun SPARC, HP PA-RISC, IBM PowerPC, Alpha, …
- **CISC philosophy**:
  - Increased capability of each instruction
  - Lead to more addressing modes
    - e.g. Motorola 68000 → 14 addressing modes
    - e.g. Motorola 68020 → 25 addressing modes
  - Variable length instructions
  - Instructions execution time variable



# R(Reduced)ISC Vs. C(Complex)ISC (cont.)

- CISC provides a large and powerful range of instructions
  - 32 Jump Instructions in a 8086 processor !
    - JA Jump if Above
    - JAE Jump if Above or Equal
    - JB Jump if Below ...
    - JZ Jump if Zero
- By contrast, in the RISC,
  - Only 3 jump instructions in MIPS
    - J Jump
    - Jal Jump And Link
    - Jr Jump register



# R(Reduced)ISC Vs. C(Complex)ISC (cont.)

- Tradeoff:
  - With **CISC** processors,
    - The programs (sometimes called the Application Program) are much [shorter? or longer?]
    - Each line in the program takes much [shorter? or longer?] to interpret and execute
  - With **RISC** processors,
    - The programs require [more? or less?] instruction lines
    - But each instruction takes much [shorter? or longer?]