CPE 221: Computer Organization

13 Microarchitecture

rahul.bhadani@uah.edu

Rahul Bhadani

The Instruction Set Architecture (ISA)

- ★ The ISA is the interface between what the software commands and what the hardware carries out
- ★ The ISA specifies
  - The memory organization
    - Address space
    - Addressability: word or byte-addressable
- ★ Instruction set specifies
  - Opcodes
  - Data types
  - Addressing modes
  - Length and format of instructions

What is Microarchitecture?

An implementation of ISA.

Many such implementations:

- **MIPS**: R2000, R3000, R4000, R6000, R8000, R10000, ⋯
- **x86**: Intel 80486, Pentium, Pentium Pro, Pentium 4, Kaby Lake, Coffee Lake, Comet Lake, Ice Lake, Golden Cove, Sapphire Rapids, AMD K5, K7, K9, Bulldozer, BobCat, Ryzen X
- **POWER**: 4, 5, 6, 7, 8, 9, 10 (IBM), ⋯
- **PowerPC**: 604, 605, 620, ⋯
- **ARM**: Cortex-M*, ARM Cortex-A*, NVIDIA Denver, Apple A*, M1, ⋯
- **Alpha**: 21064, 21164, 21264, 21364, ⋯
- **RISC-V**

Many Different ISAs Over Decades

What are the fundamental differences?

- ★ E.g., how instructions are specified and what they do
- ★ E.g., how complex are instructions, data types, addr. modes

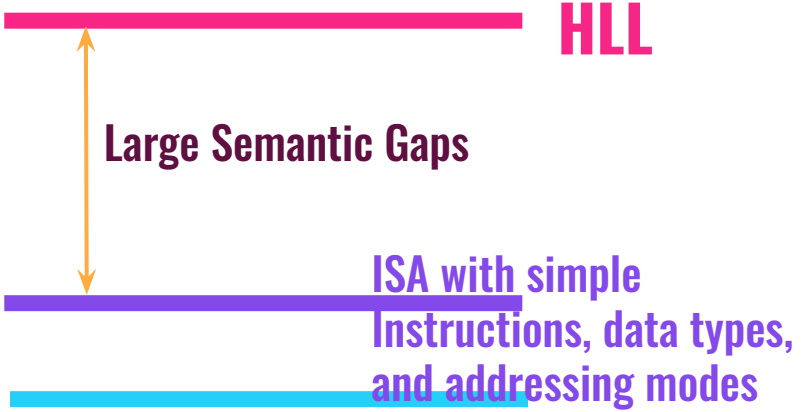
Semantic Gaps

How close instructions & data types & addressing modes are to high-level language (HLL)

Type 1

Easier mapping of HLL to ISA. Less work for software designer. More work for hardware designer. Optimization burden on HW

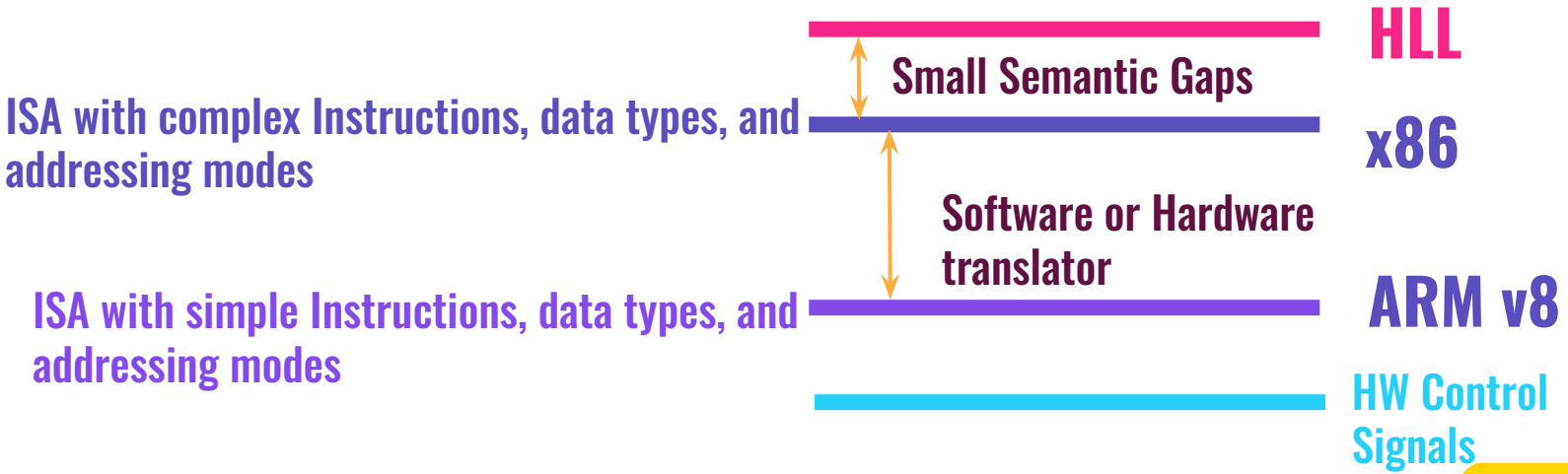
Type 2



Harder mapping of HLL to ISA. More work for software designer. Less work for hardware designer. Optimization burden on SW

Change the Semantic Gap Tradeoffs

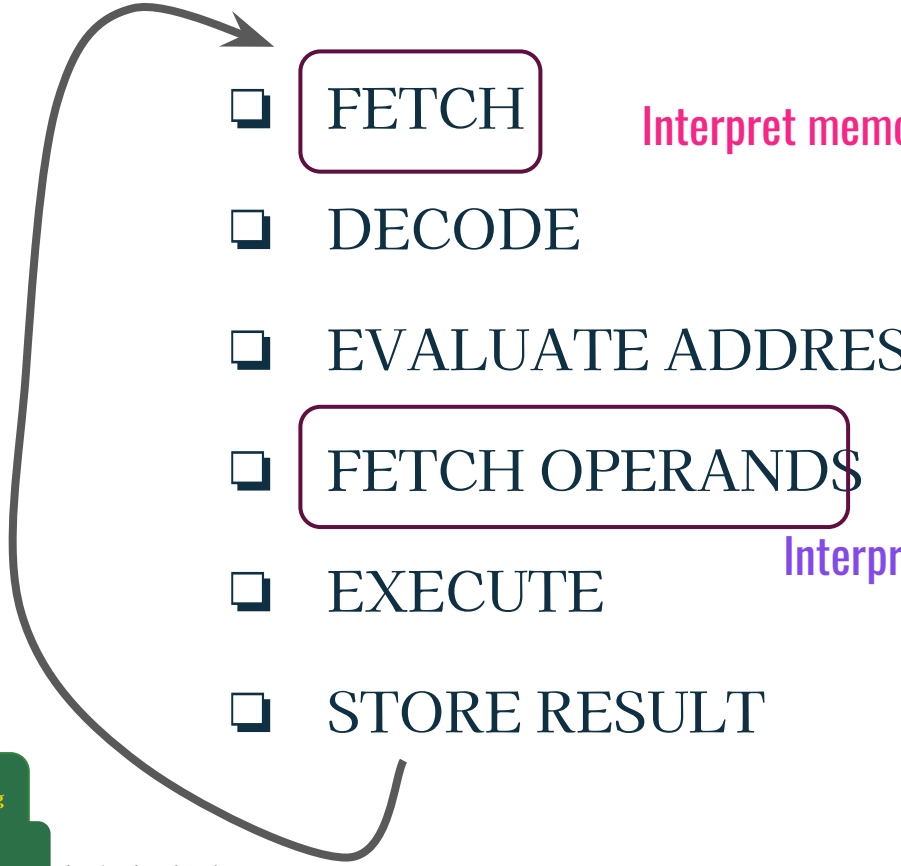
- Translate from one ISA into a different “implementation” ISA
-



The von Neumann Model/Architecture

- ★ Von Neumann model is also called stored program computer (instructions in memory). It has two key properties:
  - Stored program
    - Instructions stored in a linear memory array
    - Memory is unified between instructions and data
    - The interpretation of a stored value depends on the control signals
  - Sequential instruction processing
    - One instruction processed (fetched, executed, completed) at a time
    - Program counter (instruction pointer) identifies the current instruction
    - Program counter is advanced sequentially except for control transfer instructions

Instruction Cycle



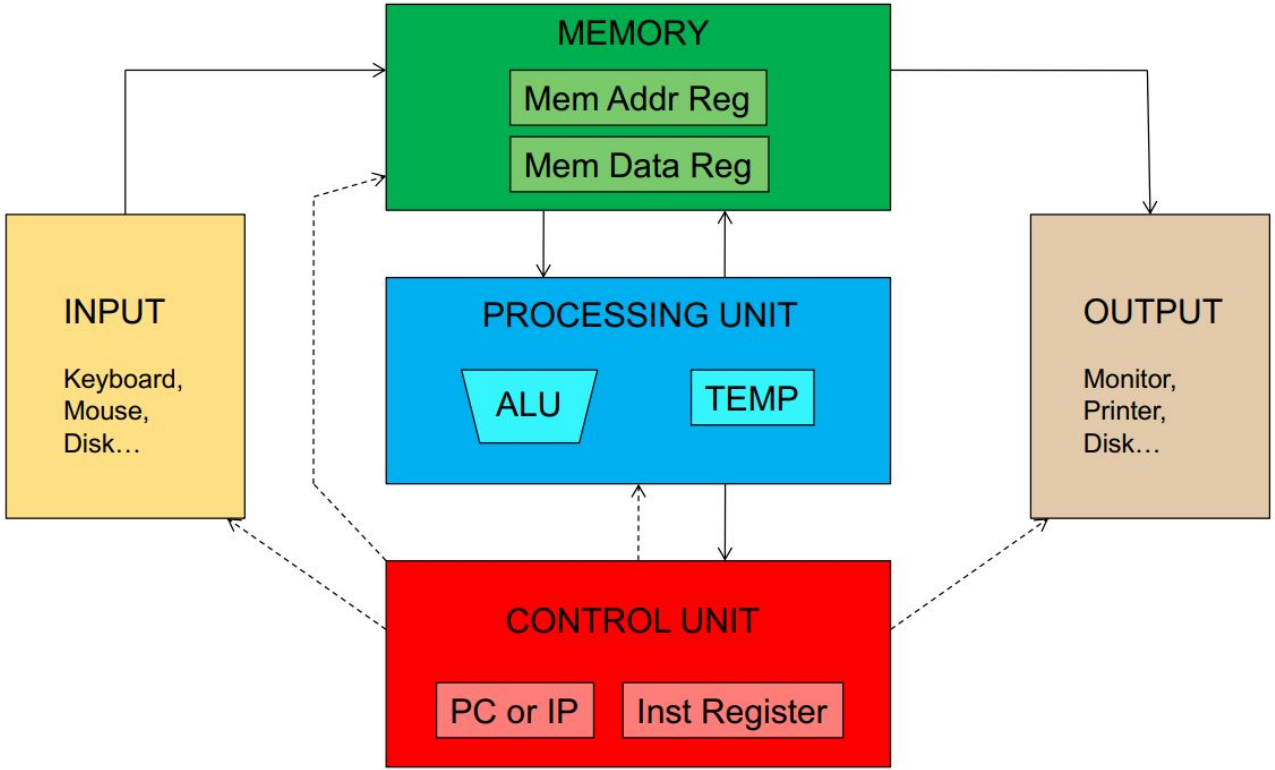
- FETCH
- DECODE
- EVALUATE ADDRESS
- FETCH OPERANDS
- EXECUTE
- STORE RESULT

Interpret memory value as Instruction

Whether a value fetched from memory is interpreted as an instruction depends on when that value is fetched in the instruction processing cycle.

Interpret memory value as Data

The Von Neumann Model (of a Computer)



ISA vs. Microarchitecture

ISA:

- Agreed upon interface between software and hardware
  - SW/compiler assumes, HW promises
- What the software writer needs to know to write and debug system/user programs

Microarchitecture

- Specific implementation of an ISA
- Not visible to the software

ISA: What Does It Specify?

Instructions

- Opcodes, Addressing Modes, Data Types
- Instruction Types and Formats
- Registers, Condition Codes

Memory

- Address space, Addressability, Alignment
- Virtual memory management

- Call, Interrupt/Exception Handling
- Access Control, Priority/Privilege
- I/O: memory-mapped vs. instructions
- Task/thread Management
- Power & Thermal Management
- Multithreading & Multiprocessor support

Implementing the ISA: Microarchitecture Basics

Questions to ask

- How do we implement it?
- How do we design a system that obeys the hardware/software interface?

How Does a Machine Process Instructions?

- What does processing an instruction mean?

$\mathcal{A}$: Architectural (programmer visible) state before an instruction is processed

$\mathcal{B}$: = Architectural (programmer visible) state after an instruction is processed

Microarchitecture implements how $\mathcal{A}$ is transformed to $\mathcal{B}$

- Processing an instruction: Transforming $\mathcal{A}$ to $\mathcal{B}$ according to the ISA specification of the instruction.

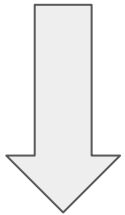
Programmer's Visible State

- Memory: array of storage locations indexed by an address
- Registers:
  - given special names in the ISA (as opposed to addresses)
  - general vs. special purpose
- Program Counter: memory address of the current (or next) instruction

A Very Basic Instruction Processing Engine

- Each instruction takes a single clock cycle to execute
- Only combinational logic is used to implement instruction execution
  - No intermediate, programmer-invisible state updates

$\mathcal{A}$: Architectural (programmer visible) state before an instruction is processed

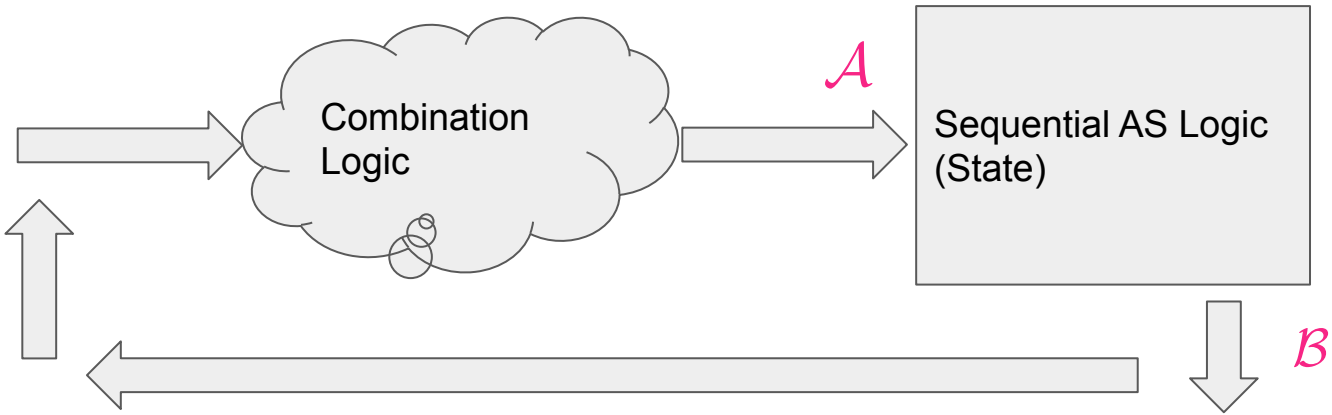


Process Instructions in one clock cycle

$\mathcal{B}$: = Architectural (programmer visible) state after an instruction is processed

A Very Basic Instruction Processing Engine

- Single-cycle Machine
- What is the clock cycle time determined by?
- What is the critical path (i.e., longest delay path) of the combinational logic determined by?



Single-cycle vs. Multi-cycle Machines

- **Single-cycle machines**
  - Each instruction takes a single clock cycle
  - All state updates made at the end of an instruction's execution
  - **Big disadvantage:** The slowest instruction determines cycle time → long clock cycle time
- Multi-cycle machines
  - Instruction processing broken into multiple cycles/stages
  - State updates can be made during an instruction's execution
  - Architectural state updates made at the end of an instruction's execution
  - Advantage over single-cycle: The slowest "stage" determines cycle time

Instruction Processing "Cycle"

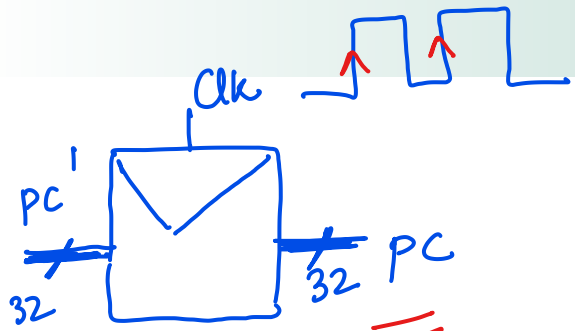
- Instructions are processed under the direction of a "control unit" step by step.
- Instruction cycle: Sequence of steps to process an instruction Fundamentally, there are six steps:
  - FETCH
  - DECODE
  - EVALUATE ADDRESS
  - FETCH OPERANDS
  - EXECUTE
  - STORE RESULT

Not all instructions require all six steps

Instruction Processing Engine

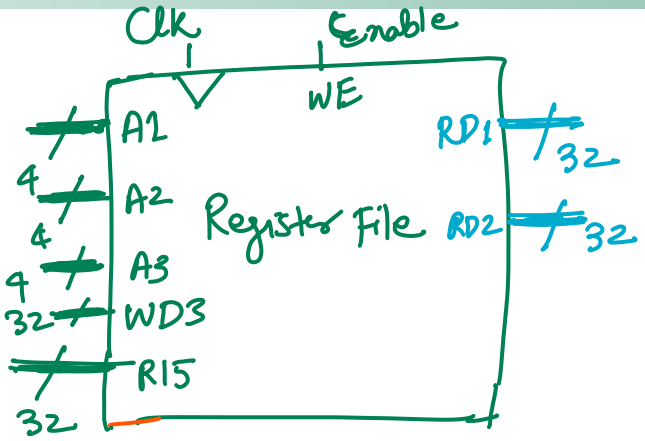
- **Datapath:** Consists of hardware elements that deal with and transform data signals
  - functional units that operate on data
  - hardware structures (e.g., wires, muxes, decoders, tri-state bufs) that enable the flow of data into the functional units and registers
  - storage units that store data (e.g., registers)
- Control logic: Consists of hardware elements that determine control signals, i.e., signals that specify what the datapath elements should do to the data

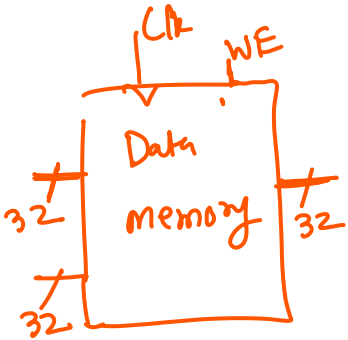
State Elements of the ARM Processor.

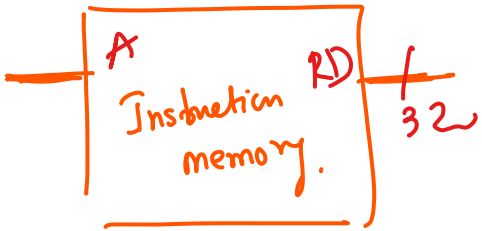


address for the next instruction

address for the current instructions.







UMULL
UMLAL
SMULL

} Requires two destination

Spring
2025

Rahul Bhadani

THE UNIVERSITY OF
ALABAMA IN HUNTSVILLE

Spring
2025

Rahul Bhadani

THE UNIVERSITY OF
ALABAMA IN HUNTSVILLE

Spring
2025

Rahul Bhadani

THE UNIVERSITY OF
ALABAMA IN HUNTSVILLE

Spring
2025

Rahul Bhadani

THE UNIVERSITY OF
ALABAMA IN HUNTSVILLE

Spring
2025

Rahul Bhadani

THE UNIVERSITY OF
ALABAMA IN HUNTSVILLE

Spring
2025

Rahul Bhadani

THE UNIVERSITY OF
ALABAMA IN HUNTSVILLE

Spring
2025

Rahul Bhadani

THE UNIVERSITY OF
ALABAMA IN HUNTSVILLE

Spring
2025

Rahul Bhadani

THE UNIVERSITY OF
ALABAMA IN HUNTSVILLE

Spring
2025

Rahul Bhadani

THE UNIVERSITY OF
ALABAMA IN HUNTSVILLE