The Processor: Datapath & Control

- **READING**: 4.1 – 4.4
- We’re ready to look at an implementation of the MIPS
- Simplified to contain only:
  - memory-reference instructions: `lw`, `sw`
  - arithmetic-logical instructions: `add`, `sub`, `and`, `or`, `slt`
  - control flow instructions: `beq`, `j`
- Generic Implementation:
  - use the program counter (PC) to supply instruction address
  - get the instruction from memory
  - read registers
  - use the instruction to decide exactly what to do
- All instructions use an ALU after reading the registers – why?
  - memory-reference?
  - arithmetic?
  - control flow?
Our Timing Methodology

• An edge triggered methodology
• Typical execution:
  – read contents of some state elements,
  – send values through some combinational logic
  – write results to one or more state elements

Single Cycle Implementation

First, Datapath
Later, Control
Our Simple Implementation

- Let’s start putting our pieces together to form our single-cycle implementation.
- Our pieces include:
  - 1. Fetching the instruction
  - 2. Performing an operation (R-type)
  - 3. Loading and storing data
  - 4. Branching

- We will discuss one piece at a time.
Partial Datapath #1 – for fetching

Partial Datapath #2 – for R-type instructions
Partial Datapath #3 – for load and store (#1)

Partial Datapath #3 – for load and store (#2)
Partial Datapath #4 – for branch

How do we tie them together?

• Strategy:
Unified Datapath – copy #3

Unified Datapath – copy #4