CprE 381: Computer Organization and Assembly Level Programming

Pipelining

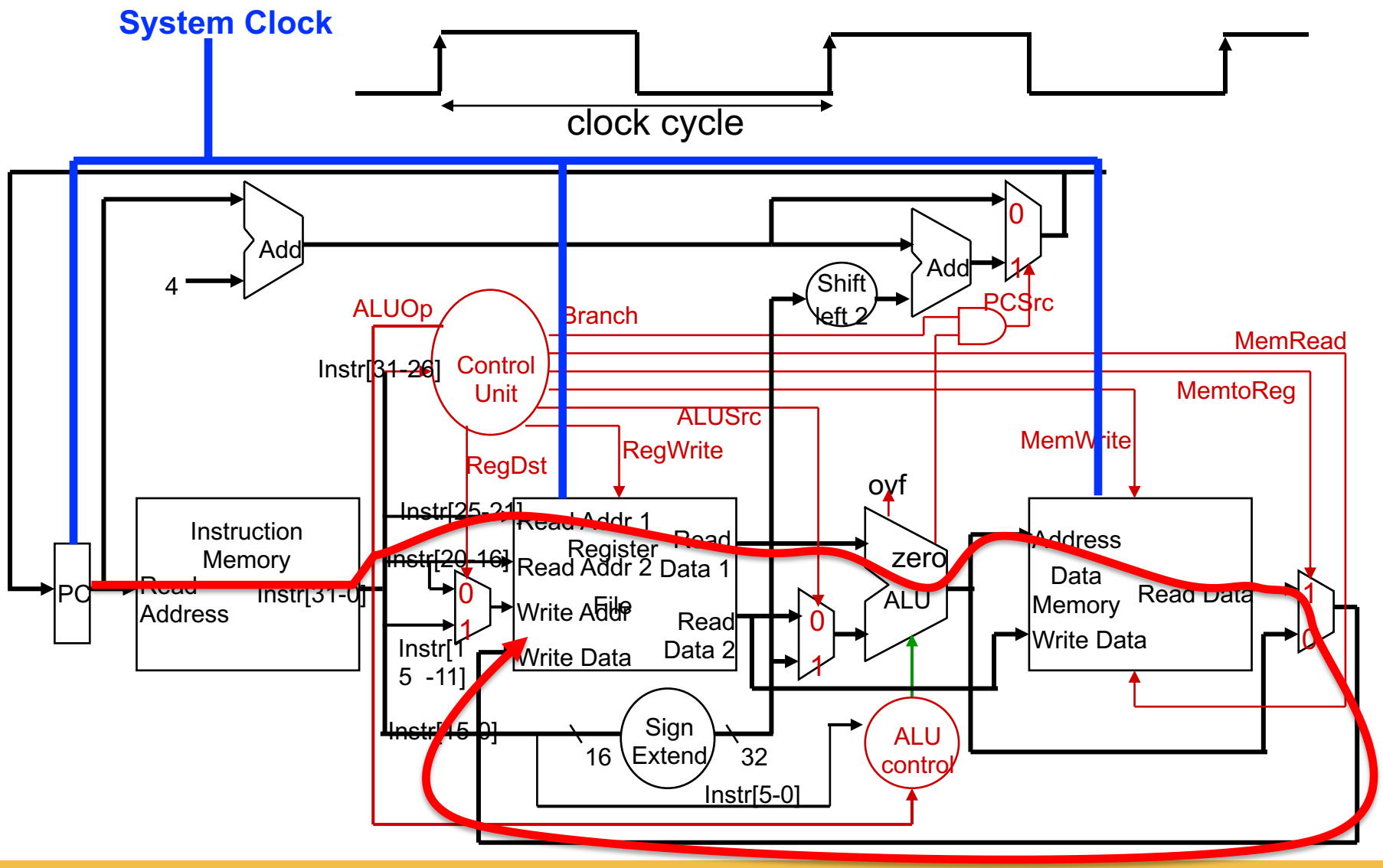
Henry Duwe
Electrical and Computer Engineering
Iowa State University

Administrative

- HW6 posted due Wed Mar 13
  - Make sure you have the new one

- Term Project
  - Part 2a due BEFORE Spring Break

Review: Worst Case Timing (Load Instruction)



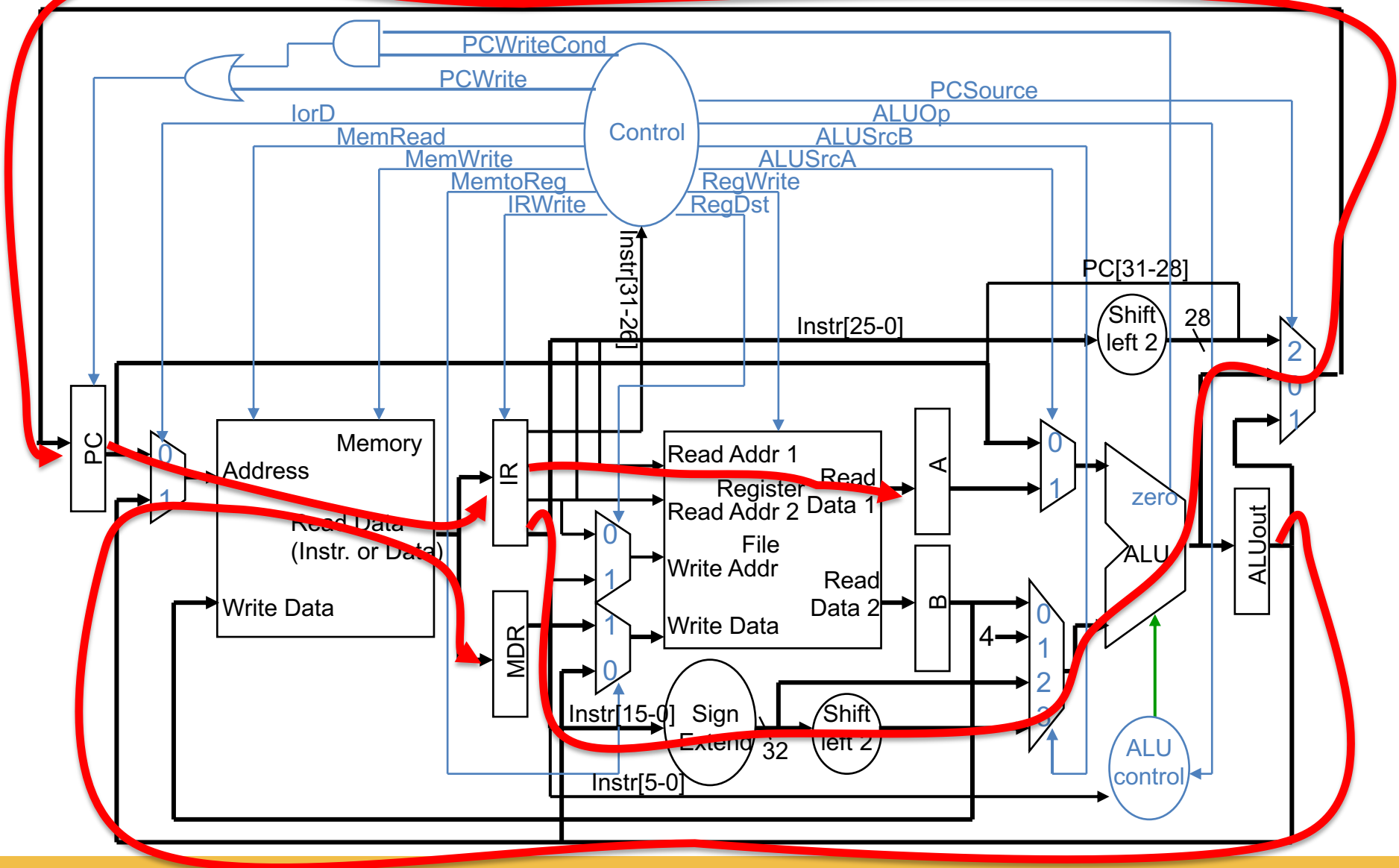
Review: Execution Time

- Drawing on the previous equation:

$$\textit{Execution Time} = \#\ \textit{Instructions} \times \frac{\textit{Cycles}}{\textit{Instruction}} \times \frac{\textit{Seconds}}{\textit{Cycle}}$$

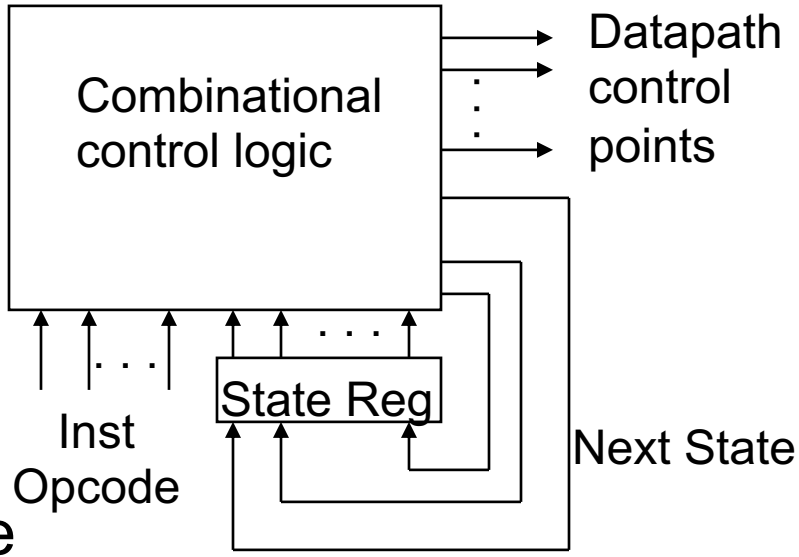
- To improve performance (i.e., reduce execution time)
  - Increase clock rate (decrease clock cycle time) OR
  - Decrease CPI OR
  - Reduce the number of instructions
- Designers balance cycle time against the number of cycles required
  - Improving one factor may make the other one worse...

Review: Multicycle Processor



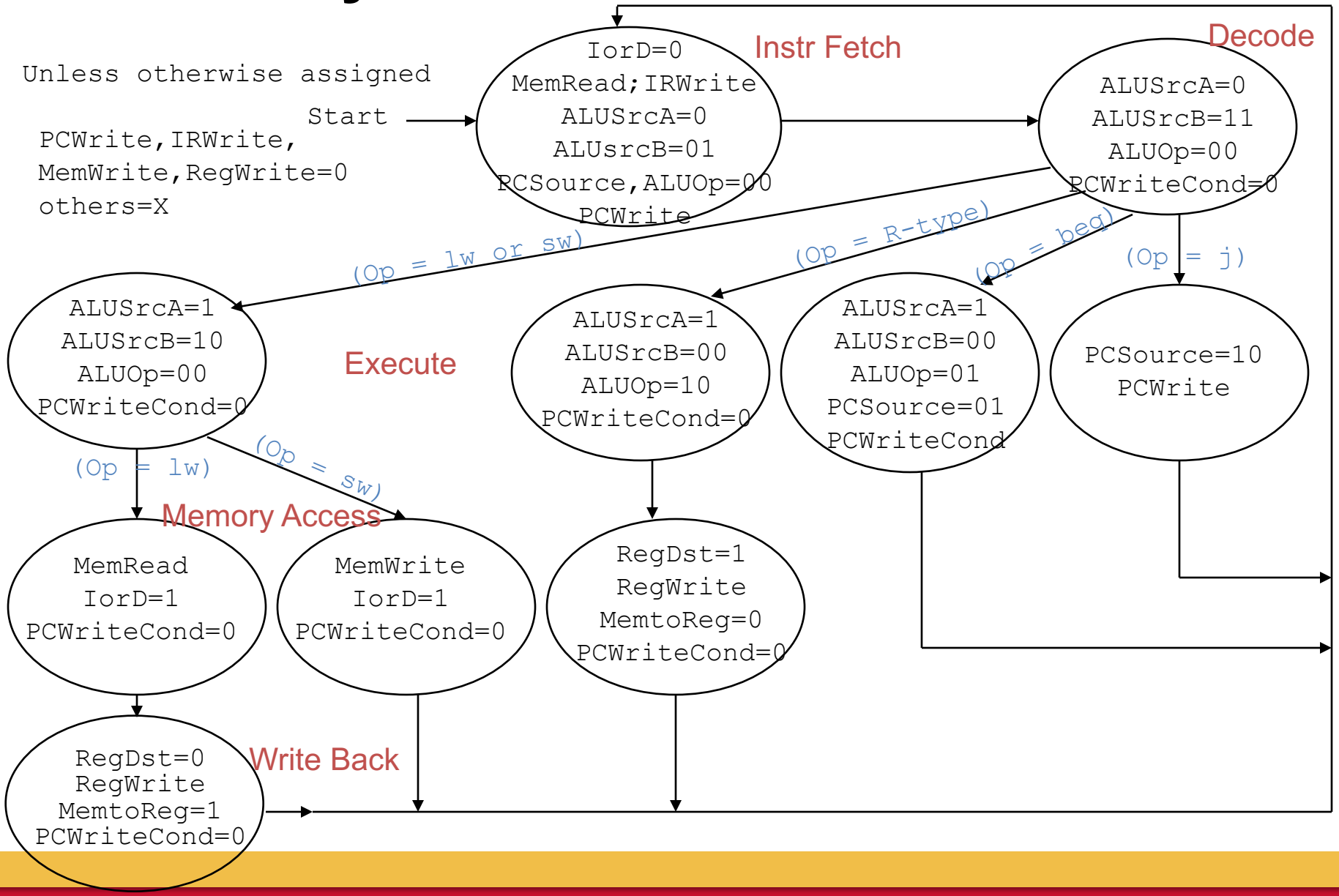
Review: Multicycle Control

- Multicycle datapath control signals are not determined solely by the bits in the instruction
  - e.g., op code bits tell what operation the ALU should be doing, but *not* what instruction cycle is to be done next
- We can use a finite state machine for control
  - A set of states (current state stored in State Register)
  - Next state function
    - determined by current state and the input
  - Output function
    - determined by current state)



- So we are using a Moore machine (datapath control signals based only on current state)

Final Multicycle Control FSM



Simplifying the Control Unit Design

- For an implementation of the full MIPS ISA instructions can take from 3 clock cycles to 20+ clock cycles
  - Resulting in finite state machines with hundreds to thousands of states with even *more* arcs (state sequences)
  - Such state machine representations become impossibly complex
- Instead, can represent the set of control signals that are asserted during a state as a low-level control "instruction" to be executed by the datapath

Microinstructions

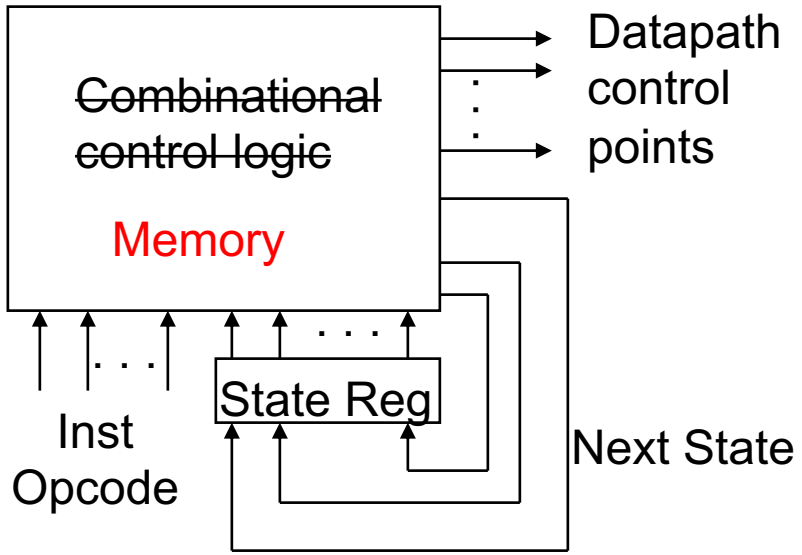
- "Executing" the microinstruction is equivalent to asserting the control signals specified by the microinstruction

Microprogramming (50k feet)

- A microinstruction has to specify
  - what control signals should be asserted
  - what microinstruction should be executed next
- Each microinstruction corresponds to one state in the FSM and is assigned a state number (or "address")
  1. Sequential behavior – increment the state (address) of the current microinstruction to get to the state (address) of the next
  2. Jump to the microinstruction that begins execution of the next MIPS instruction (fetch state)
  3. Branch to a microinstruction based on control unit input using dispatch tables
     - need one for microinstructions following decode state
     - need another for microinstructions following memory address gen state
- The set of microinstructions that define a MIPS assembly language instruction (macroinstruction) is its microroutine

Microprogramming (50k feet)

- A microinstruction has to specify
  - what
  - wh
- Each
  1. Se
     cu
  2. Ju
     M
  3. Br
     di
     -
     -



in the
ess")
of the
of the next
f the next
put using
address gen

- The set of microinstructions that define a MIPS assembly language instruction (macroinstruction) is its microroutine

Multicycle Advantages & Disadvantages

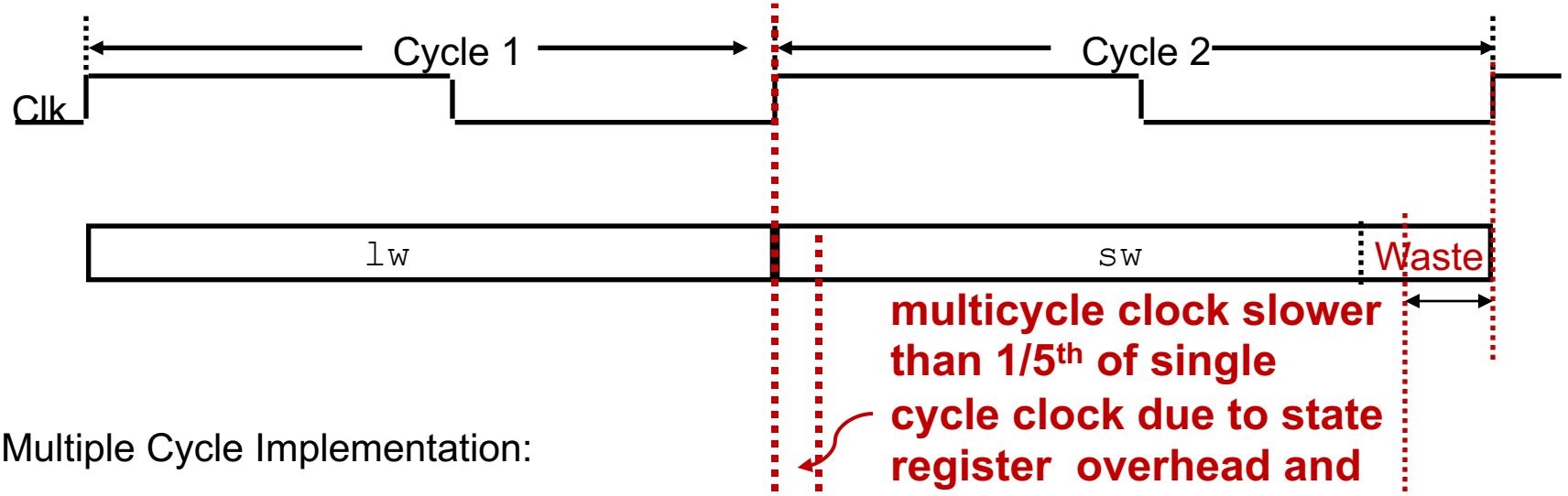
- Uses the clock cycle efficiently – the clock cycle is timed to accommodate the slowest instruction step
  - Balance the amount of work to be done in each step
  - Restrict each step to use only one major functional unit
- Multicycle implementations allow
  - Faster clock rates
  - Different instructions to take a different number of clock cycles
  - Functional units to be used more than once per instruction as long as they are used on different clock cycles

But

- Requires additional internal state registers, muxes, and more complicated (FSM) control
- **CPI > 1**

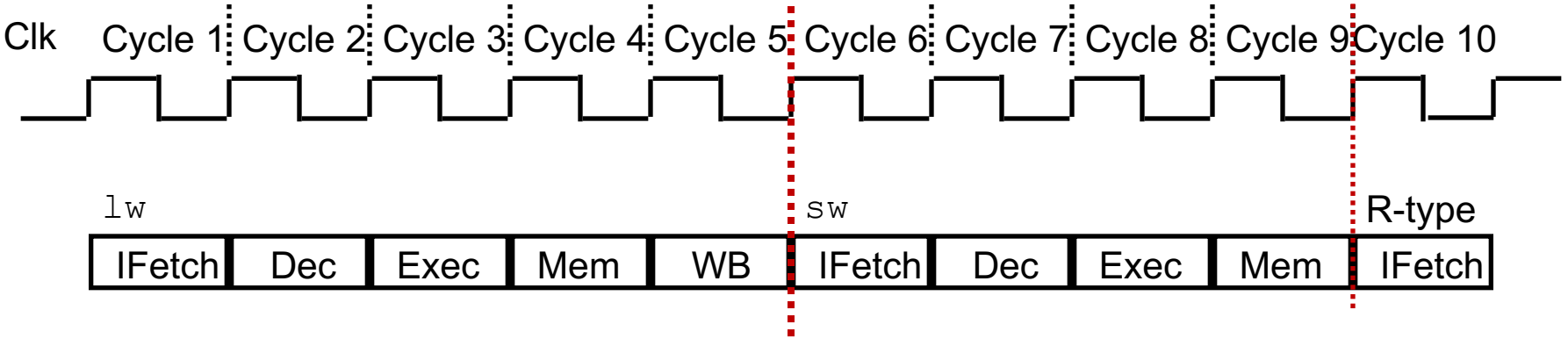
Single Cycle vs. Multiple Cycle Timing

Single Cycle Implementation:



multicycle clock slower than 1/5th of single cycle clock due to state register overhead and imbalance in stages

Multiple Cycle Implementation:

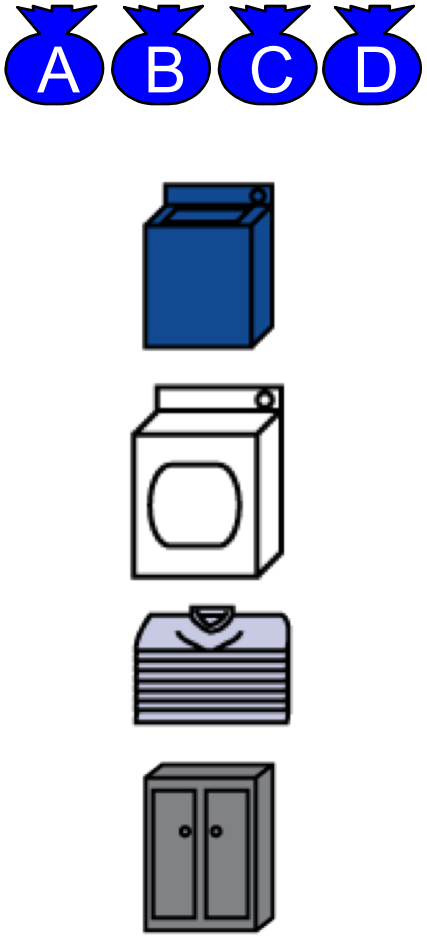


Increasing Parallelism

- Problem with the multi-cycle processor:
  - Each functional unit used once per instruction
  - Most of the time it is sitting waiting for its turn
- It's a circuit so it's calculating all the time, but it's waiting for valid data
  - Wasted time/energy
- Making instructions take more cycles can make machine faster!
  - Each instruction takes roughly the same time
    - While the CPI is much worse, the clock freq is much higher
  - IDEA: Overlap execution of multiple instructions at the same time
    - Different instructions will be active at the same time
  - This is called "Pipelining"
  - We will look at a 5 stage pipeline
    - Modern machines span the range from 3 (embedded micro-controllers) to 30+ (high performance processors) stages/instruction

Pipelining: You Do it All the Time!

- Ann, Brian, Cathy, Dave each have one load of clothes to wash, dry, fold, and put away
- Washer takes 30 minutes
- Dryer takes 30 minutes
- "Folder" takes 30 minutes
- "Stasher" takes 30 minutes to put clothes into drawers



Tech Update: Folder Exists!

FoldiMate Home How it Works Speaking Technically FAQ Blog Check Status Want a FoldiMate?

RESTART

Speaking Technically...

AGE 6-XXL + SIZE S-L Learn More >

Estimated Price: $980

We strive to make FoldiMate as affordable as we can. The estimate doesn't include tax and shipping, as regulations vary by region.

First Shipments: Late 2019 Why? >

Starting in US then internationally based on demand.

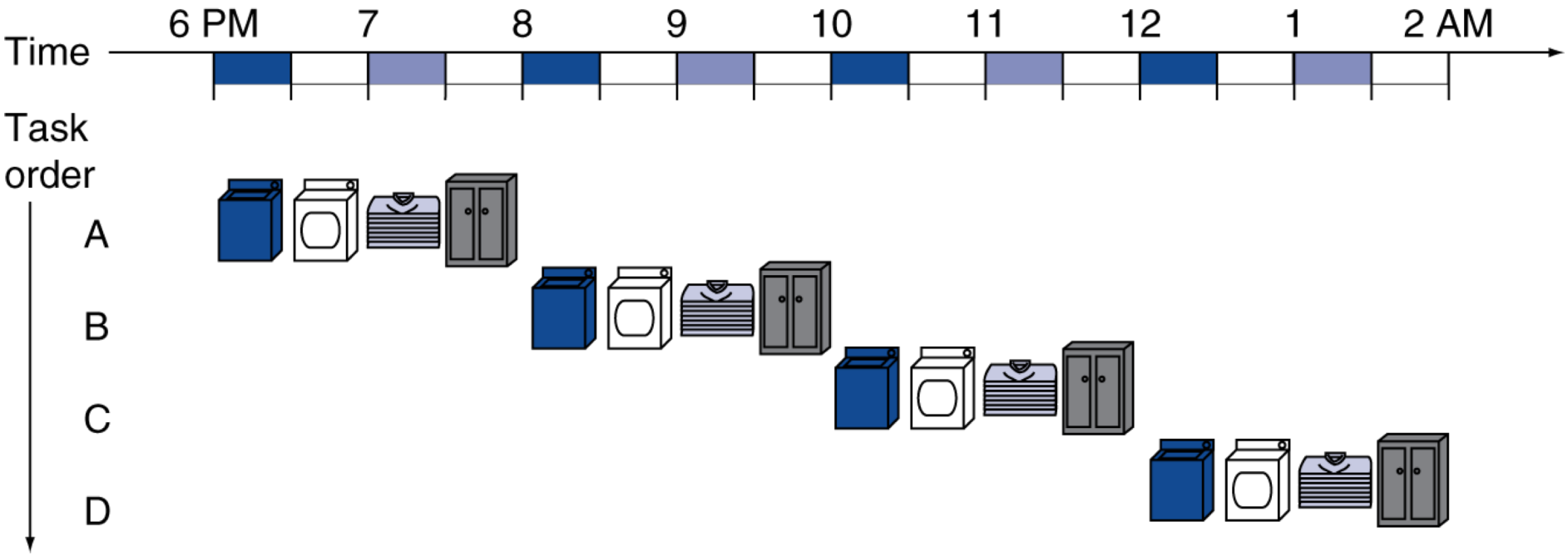
Speed: Under 4 minutes

Imagine, getting your laundry folding done in less time than it takes to brew a cup of coffee!

Capacity: Unlimited

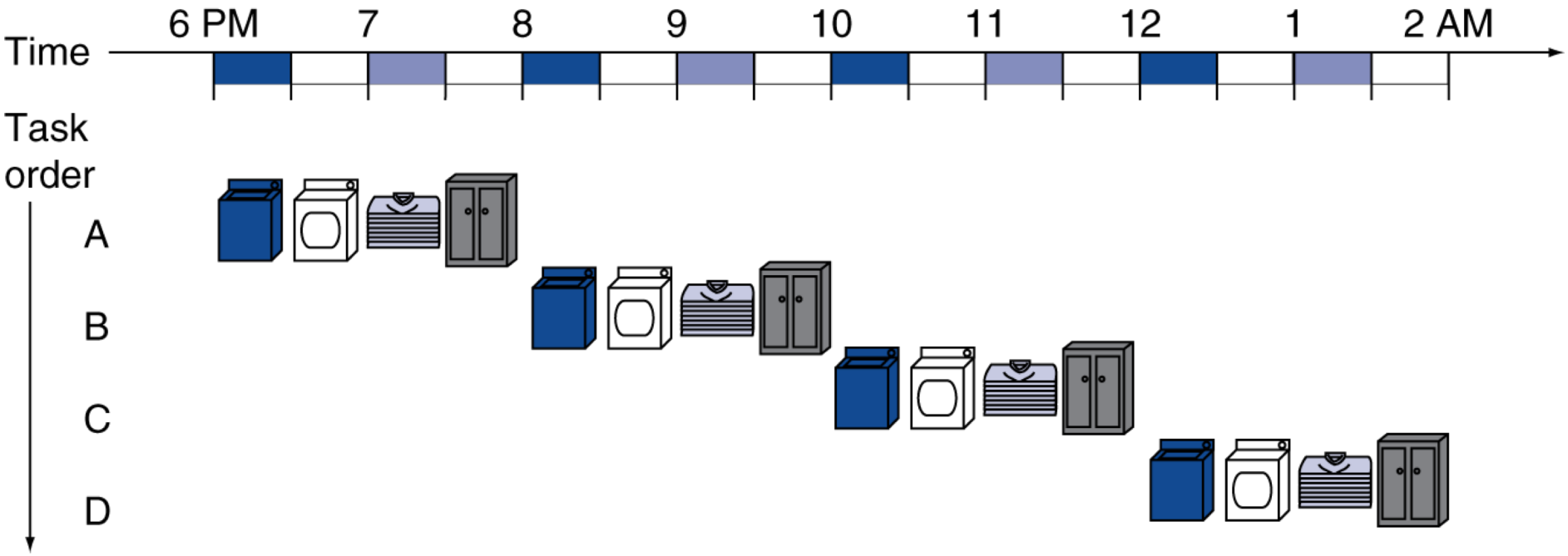
FoldiMate folds as long as you continue clipping.

Sequential Laundry



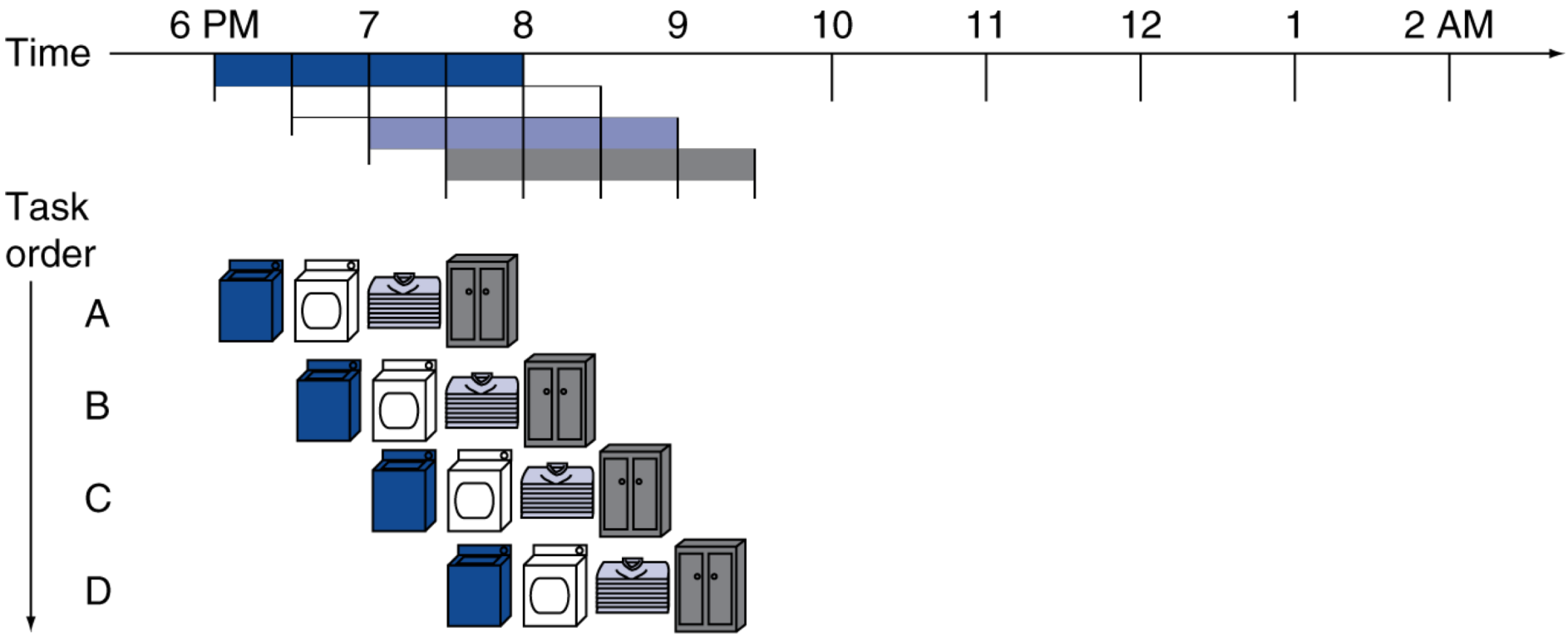
- Drawback: Sequential laundry takes 8 hours for 4 loads

Sequential Laundry



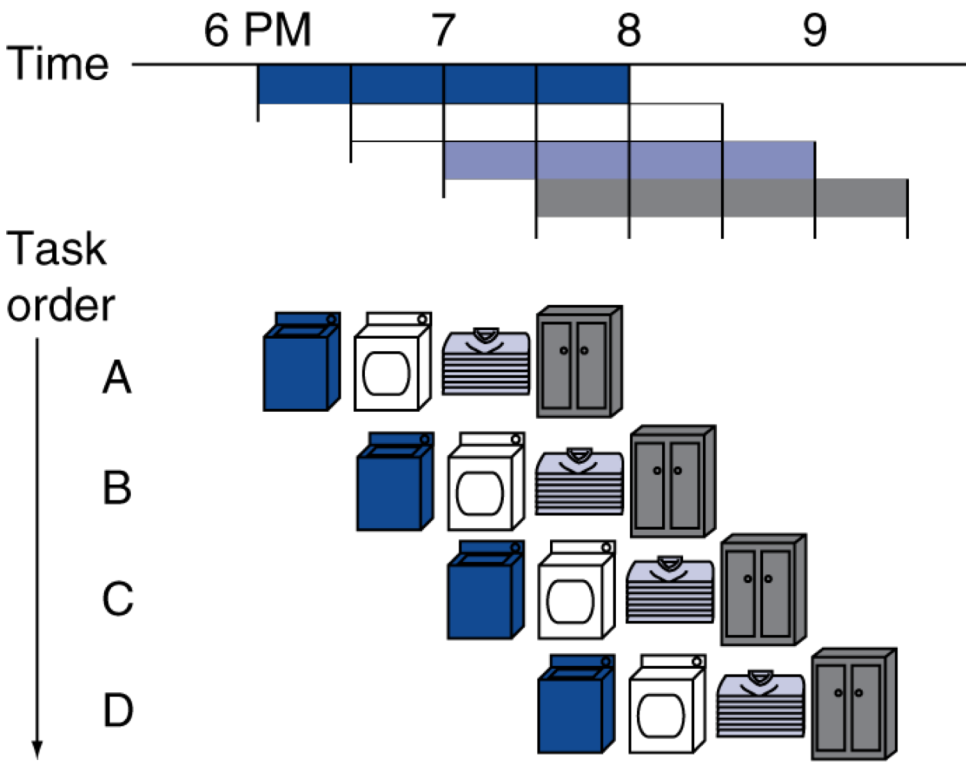
- Benefit: Allows Dave to work on 381 for 6 hours on Friday evening

Pipelined Laundry



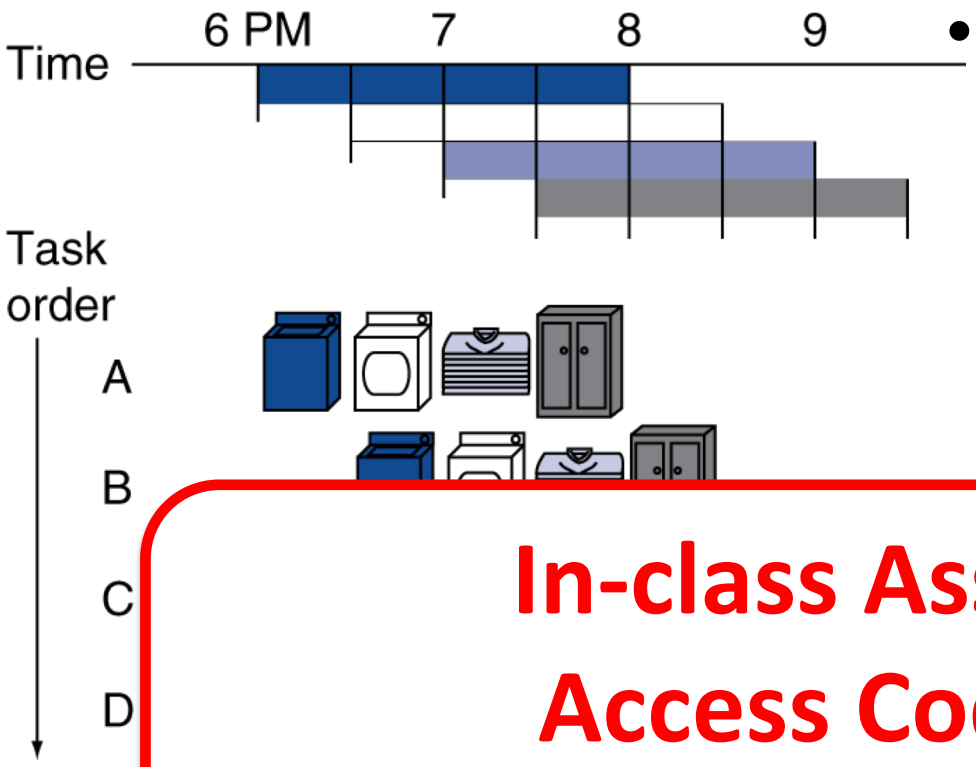
- Pipelined laundry takes 3.5 hours for 4 loads!
- Speedup = 8 / 3.5 = 2.3
- General case: 0.5*4*n / (0.5*n +1.5) ≈ 4 (# of stages)

Pipelining Lessons



- Pipelining doesn't help latency of single task, it helps throughput of entire workload
- Multiple tasks operating simultaneously using different resources
- Potential speedup = Number pipe stages
- Time to "fill" pipeline and time to "drain" it reduces speedup:
  2.3X v. 4X in this example

Pipelining Lessons (cont.)



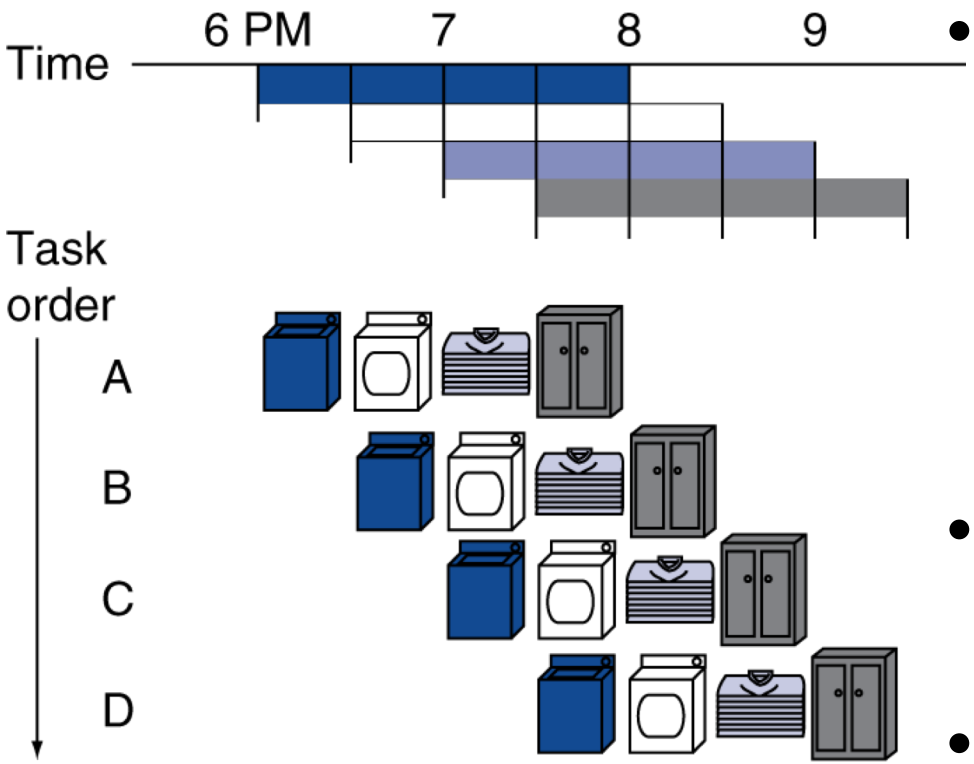
- Suppose new Washer takes 20 minutes, new Stasher takes 20 minutes. How much faster is pipeline?

In-class Assessment!

Access Code: iStack

Note: sharing access code to those outside of classroom or using access while outside of classroom is considered cheating

Pipelining Lessons (cont.)



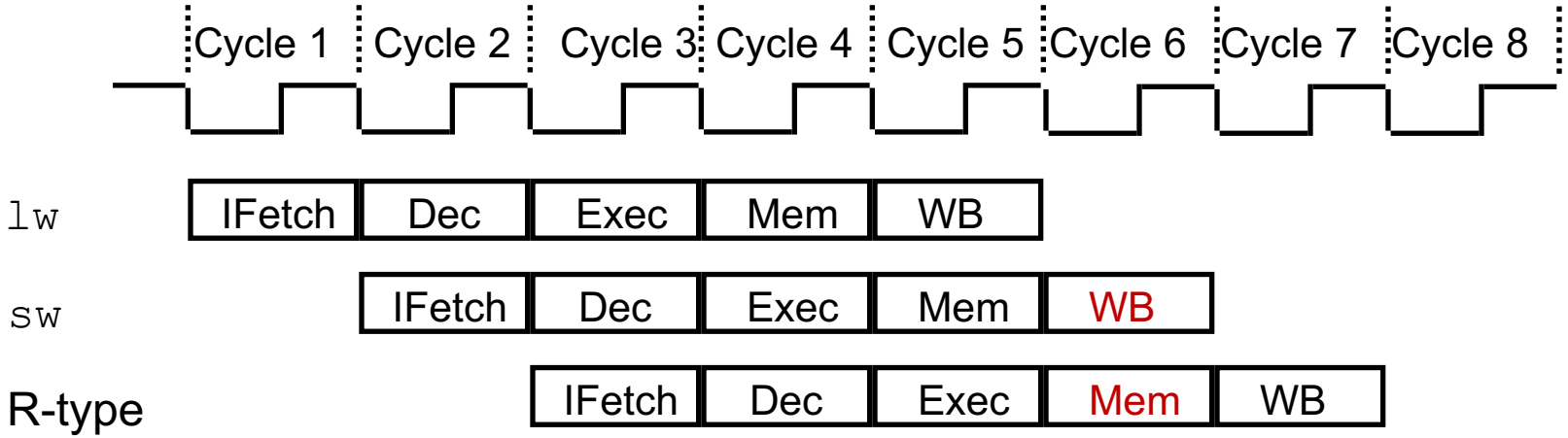
- Suppose new Washer takes 20 minutes, new Stasher takes 20 minutes. How much faster is pipeline?
- Pipeline rate limited by slowest pipeline stage
- Unbalanced lengths of pipe stages reduces speedup

Remember: 5 Stage MIPS Execution

- IF: Instruction Fetch
  - Fetch the instruction from memory
  - Increment the PC
- RF/ID: Register Fetch and Instruction Decode
  - Fetch base register
- EX: Execute
  - Calculate base + sign-extended offset
- MEM: Memory
  - Read the data from the data memory
- WB: Write back
  - Write the results back to the register file

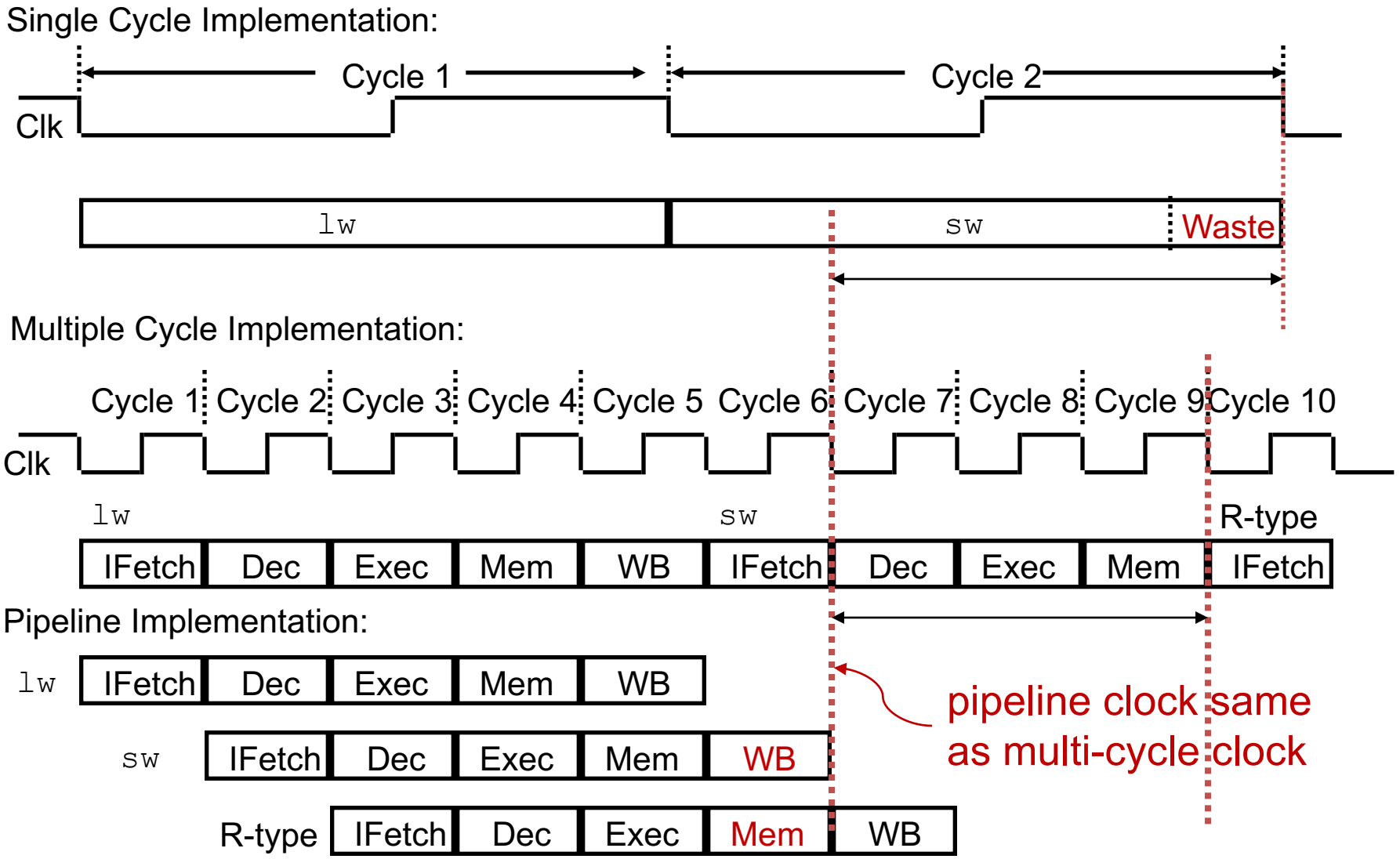
A Pipelined MIPS Processor

- Start the next instruction before the current one has completed
  - Improves throughput - total amount of work done in a given time
  - Instruction latency (execution time, delay time, response time - time from the start of an instruction to its completion) is *not* reduced



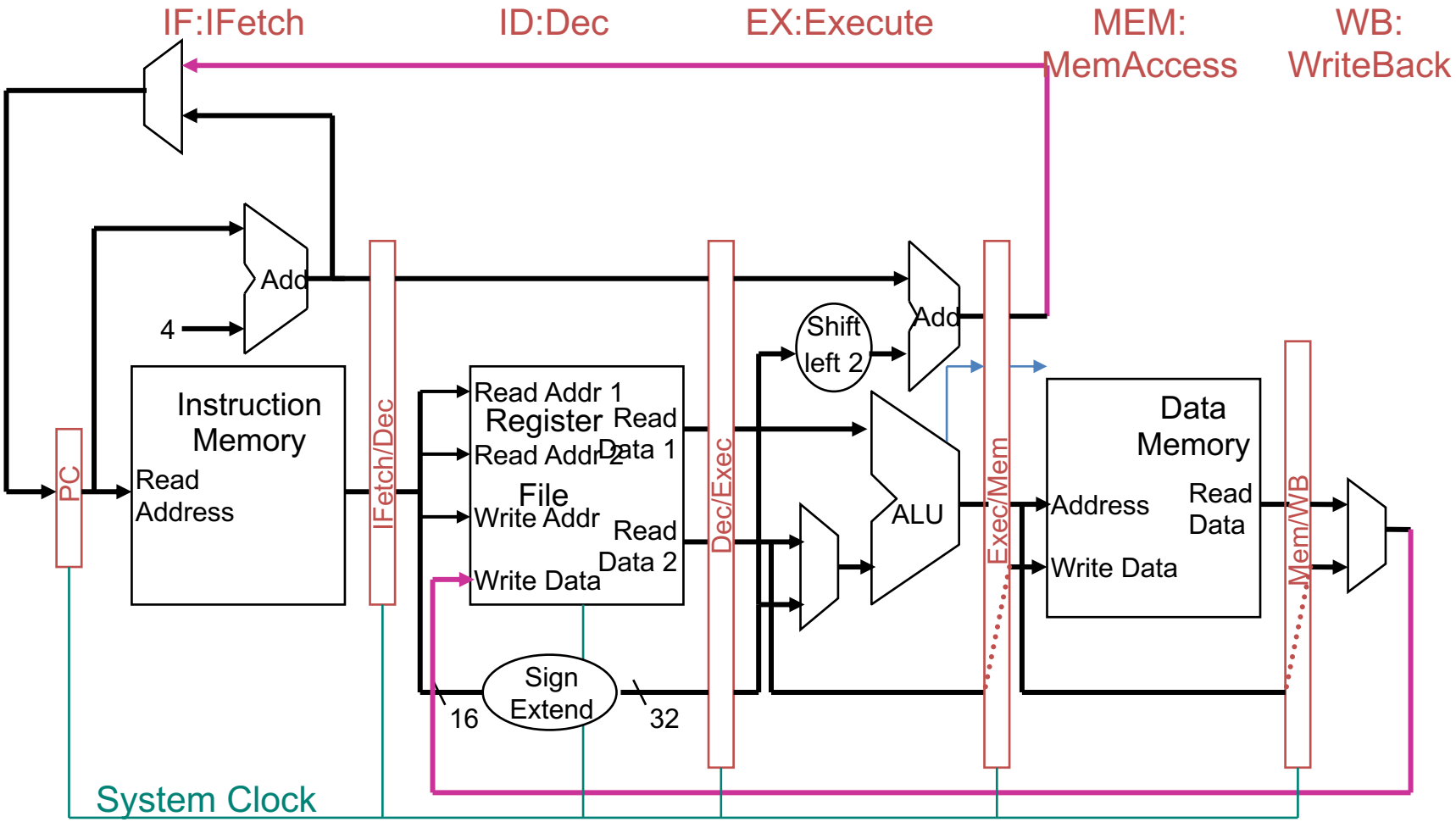
  - Clock cycle (pipeline stage time) is limited by the slowest stage
  - For some instructions, some stages are wasted cycles

Single Cycle, Multiple Cycle, vs. Pipeline



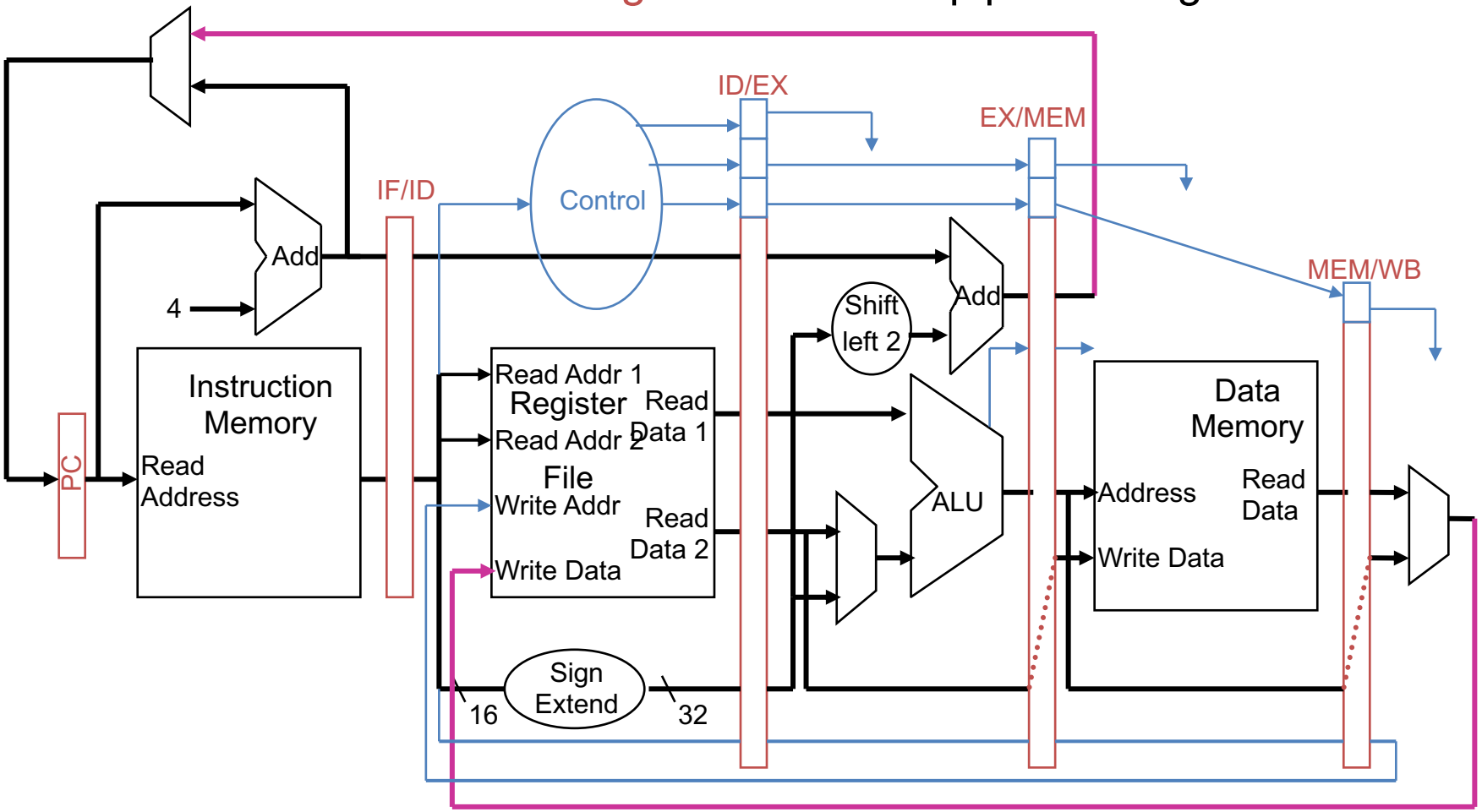
MIPS Pipeline Datapath Modifications

- What do we need to add/modify in our MIPS datapath?
  - State registers between each pipeline stage to isolate them



Preview: MIPS Pipeline Control Path Modifications

- All control signals can be determined during Decode
  - And held in the state registers between pipeline stages



Acknowledgments

- These slides contain material developed and copyright by:
  - Joe Zambreno (Iowa State)
  - David Patterson (UC Berkeley)
  - Mary Jane Irwin (Penn State)
  - Christos Kozyrakis (Stanford)
  - Onur Mutlu (Carnegie Mellon)
  - Krste Asanović (UC Berkeley)