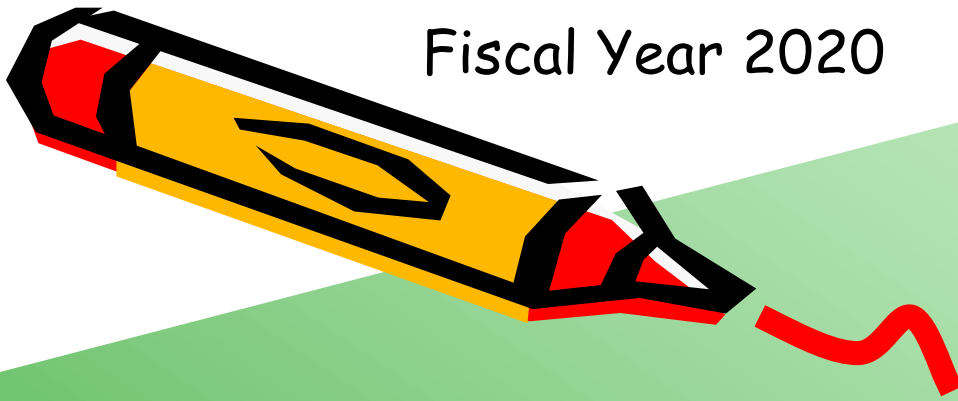


Fiscal Year 2020

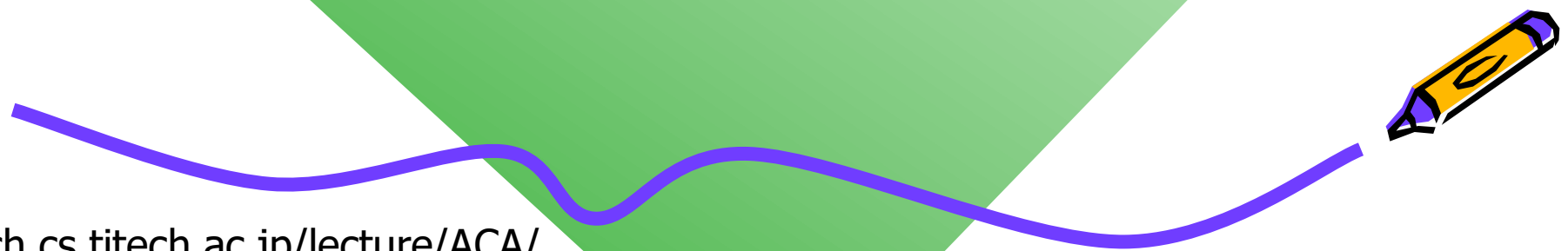
Ver. 2020-12-11a



Course number: CSC.T433
School of Computing,
Graduate major in Computer Science

Advanced Computer Architecture

4. Pipelining



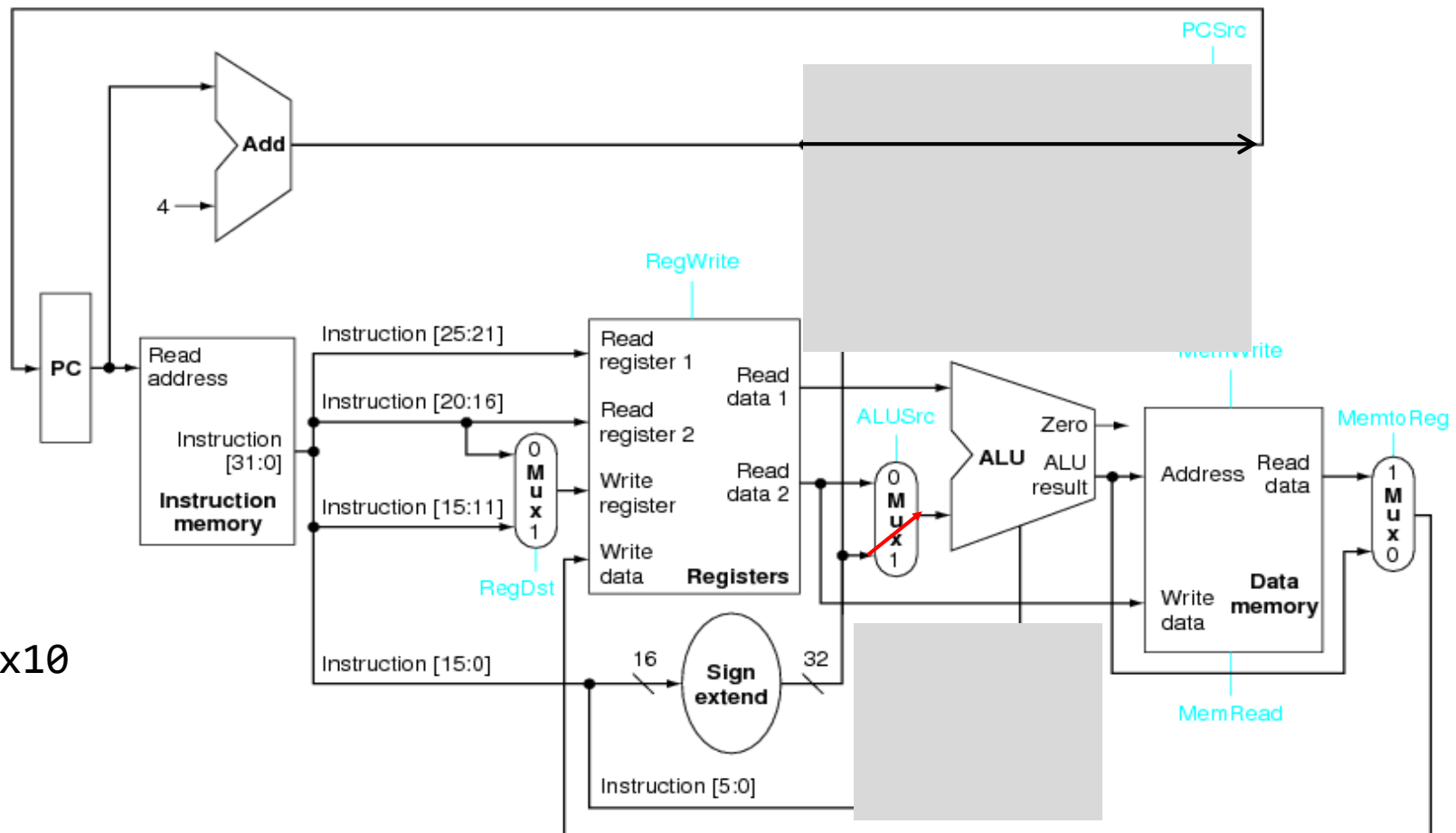
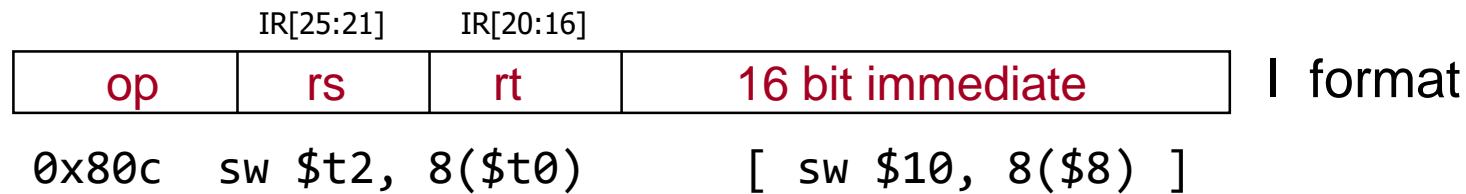
www.arch.cs.titech.ac.jp/lecture/ACA/
Room No.W936
Mon 14:20-16:00, Thr 14:20-16:00

Kenji Kise, Department of Computer Science
kise_at_c.titech.ac.jp

ACRi Room demonstration



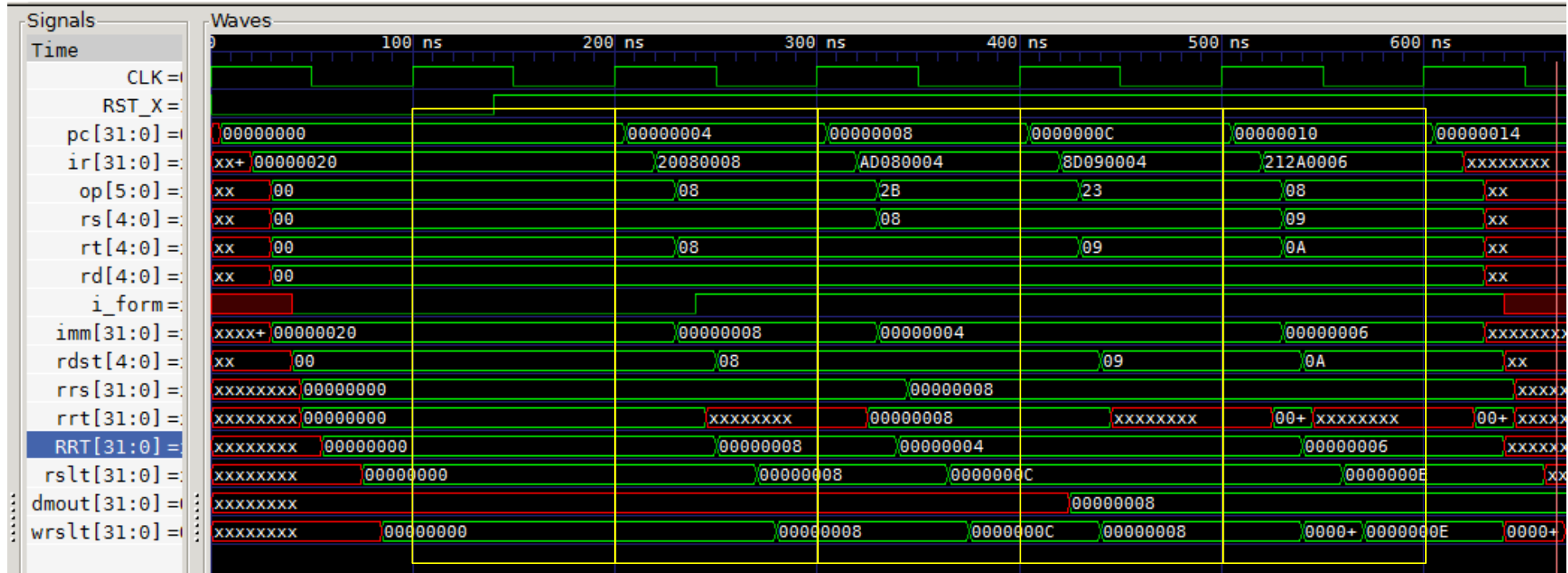
Datapath of processor supporting **ADD, ADDI, LW, SW**



\$8 = 0x10
\$10 = 2



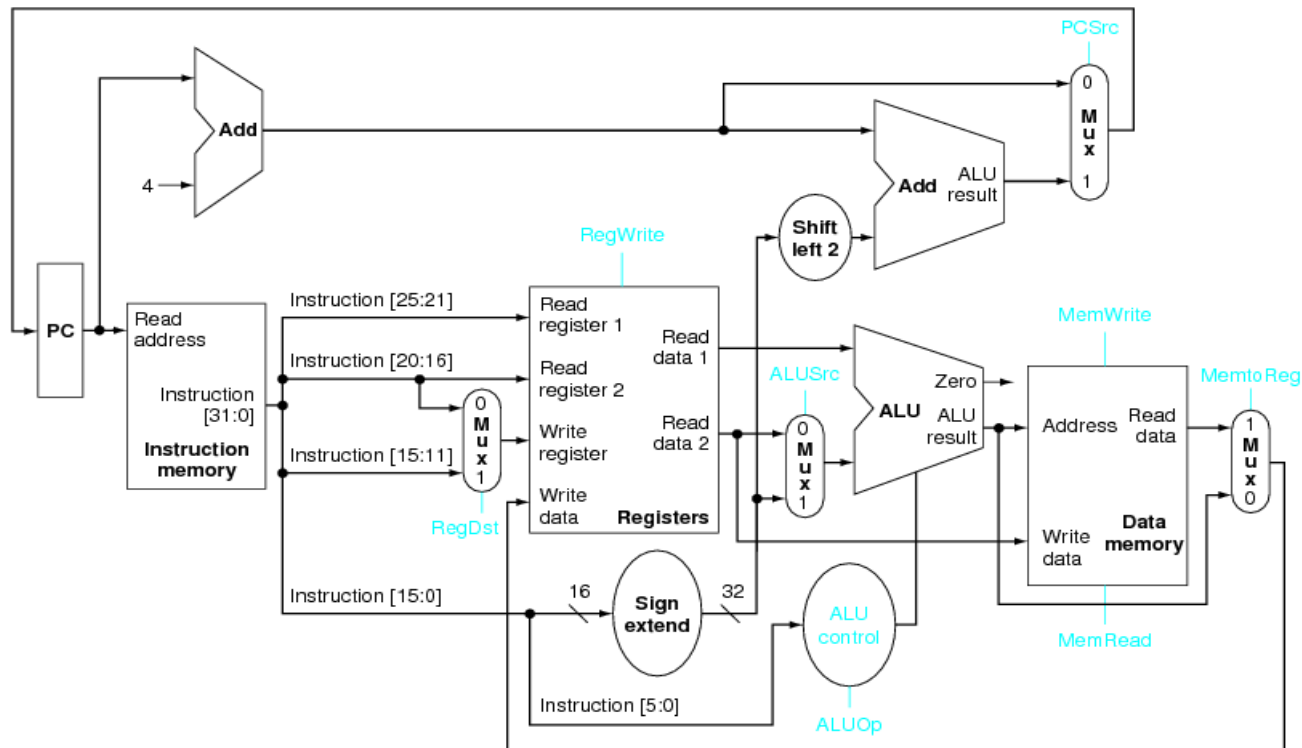
Waveform of proc04 (Assignment2)



- add \$0, \$0, \$0 # NOP {6'h0, 5'd0, 5'd0, 5'd0, 5'd0, 6'h20}
- addi \$t0, \$zero, 8 # {6'h8, 5'd0, 5'd8, 16'd8} \$t0 = 8
- sw \$t0, 4(\$t0) # {6'h2b, 5'd8, 5'd8, 16'd4} mem[12] = 8
- lw \$t1, 4(\$t0) # {6'h23, 5'd8, 5'd9, 16'd4} \$t1 = mem[12] = 8
- addi \$t2, \$t1, 6 # {6'h8, 5'd9, 5'd10, 16'd6} \$t2 = 8 + 6

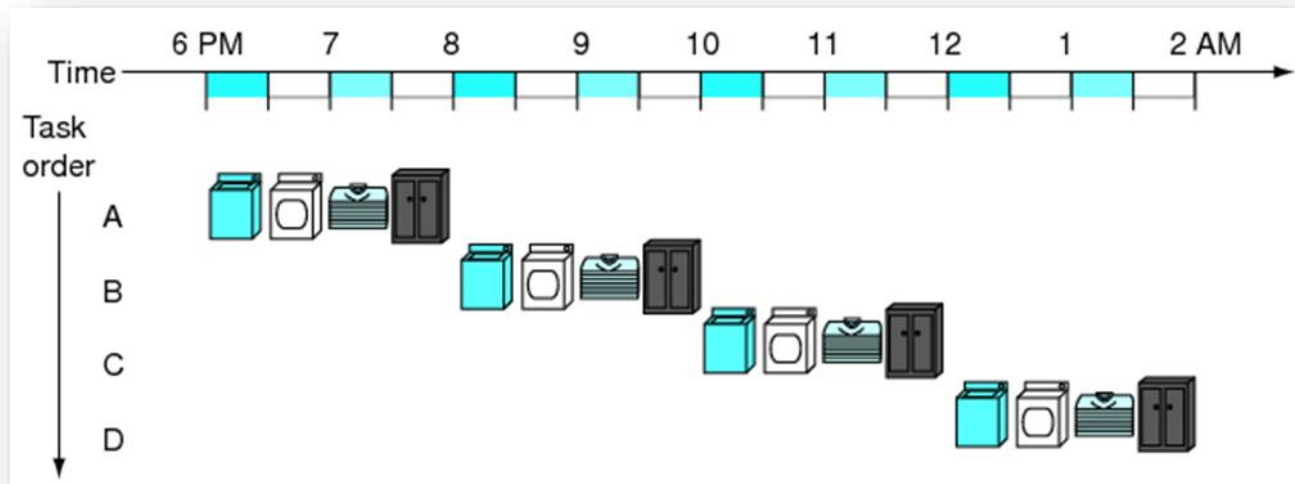
Single-cycle implementation of processors

- Single-cycle implementation also called **single clock cycle implementation** is the implementation in which an instruction is executed in one clock cycle. While easy to understand, it is too slow to be practical.



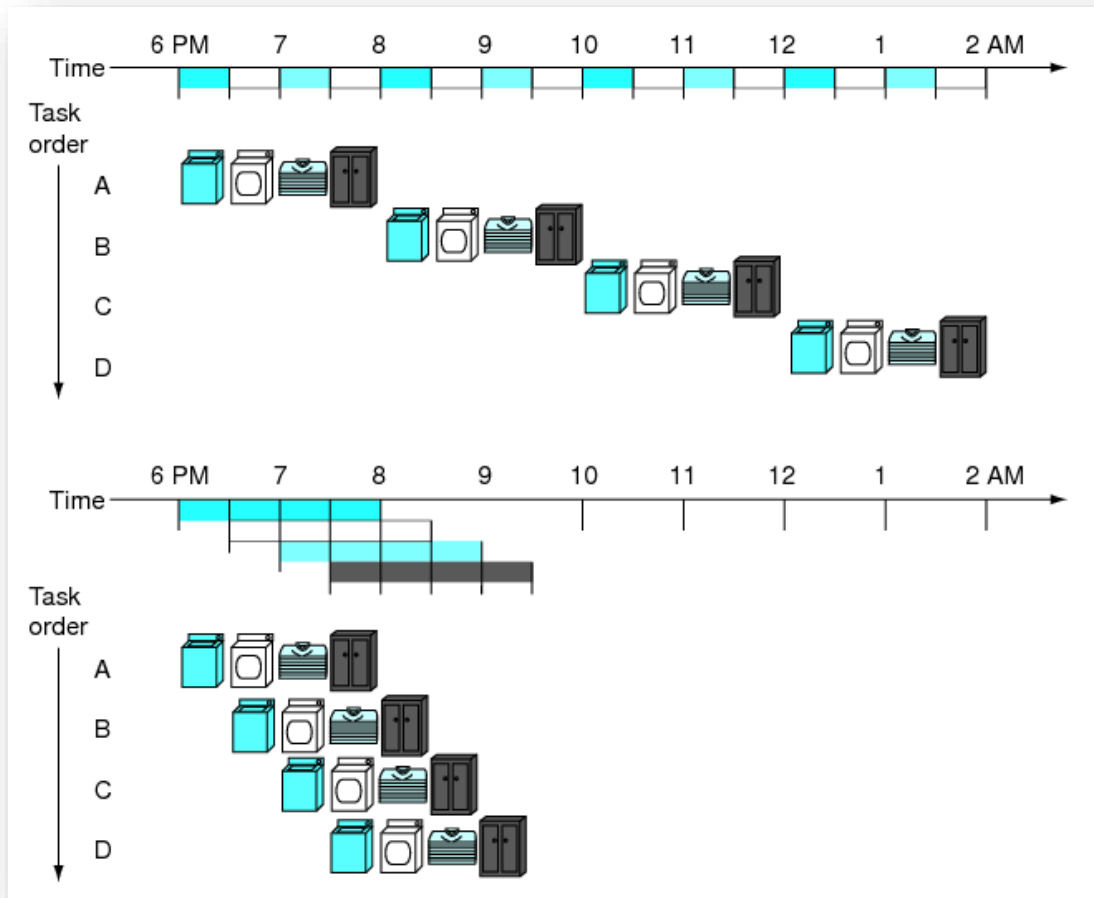
Single-cycle implementation of laundry

- (A) Ann, (B) Brian, (C) Cathy, and (D) Don each have dirty clothes to be *washed, dried, folded, and put away* where each takes 30 minutes.
- Cycle time is 2 hours.
- Sequential laundry takes 8 hours for 4 loads.

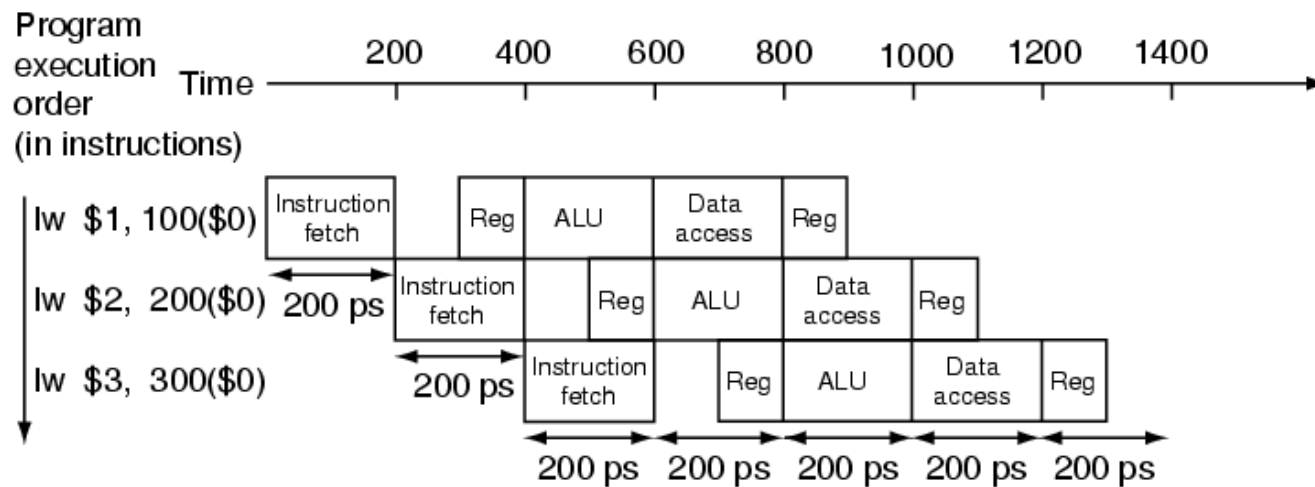
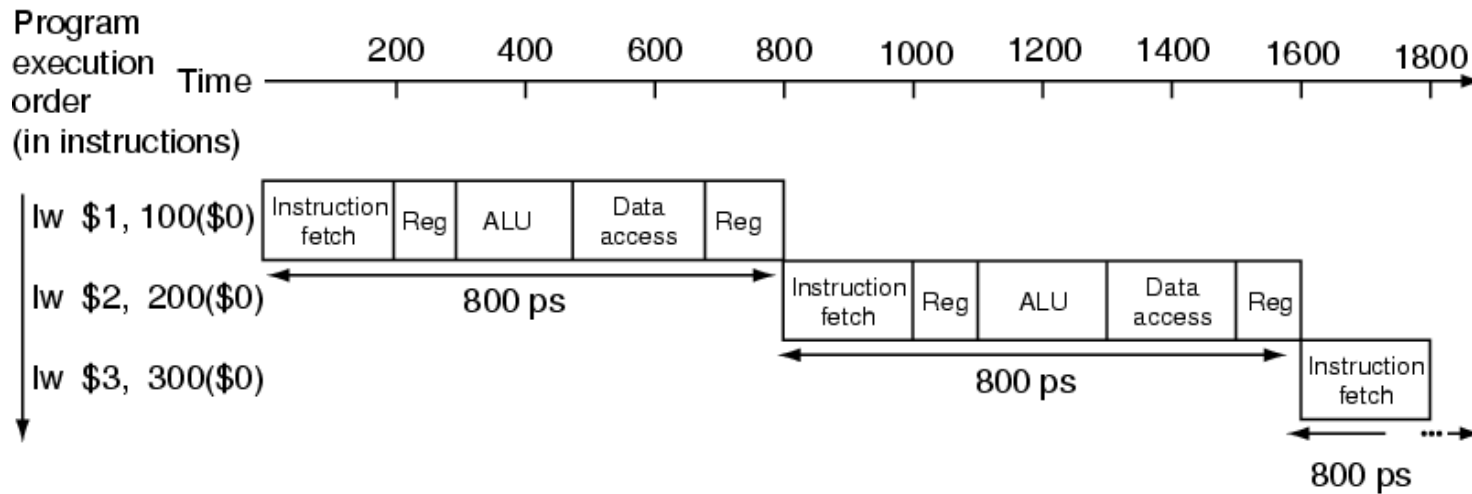


Single-cycle implementation and **pipelining**

- Pipelined laundry takes 3.5 hours just using the same hardware resources. Cycle time is 30 minutes.
- What is the latency of each load?

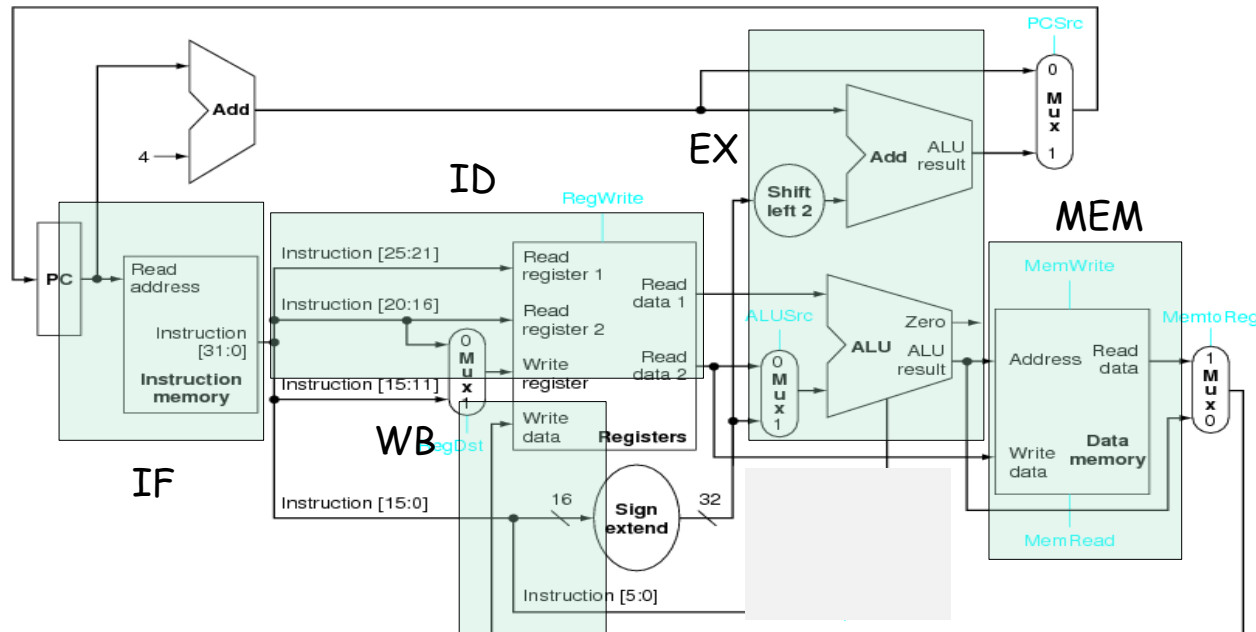


Single-cycle and pipelined processors



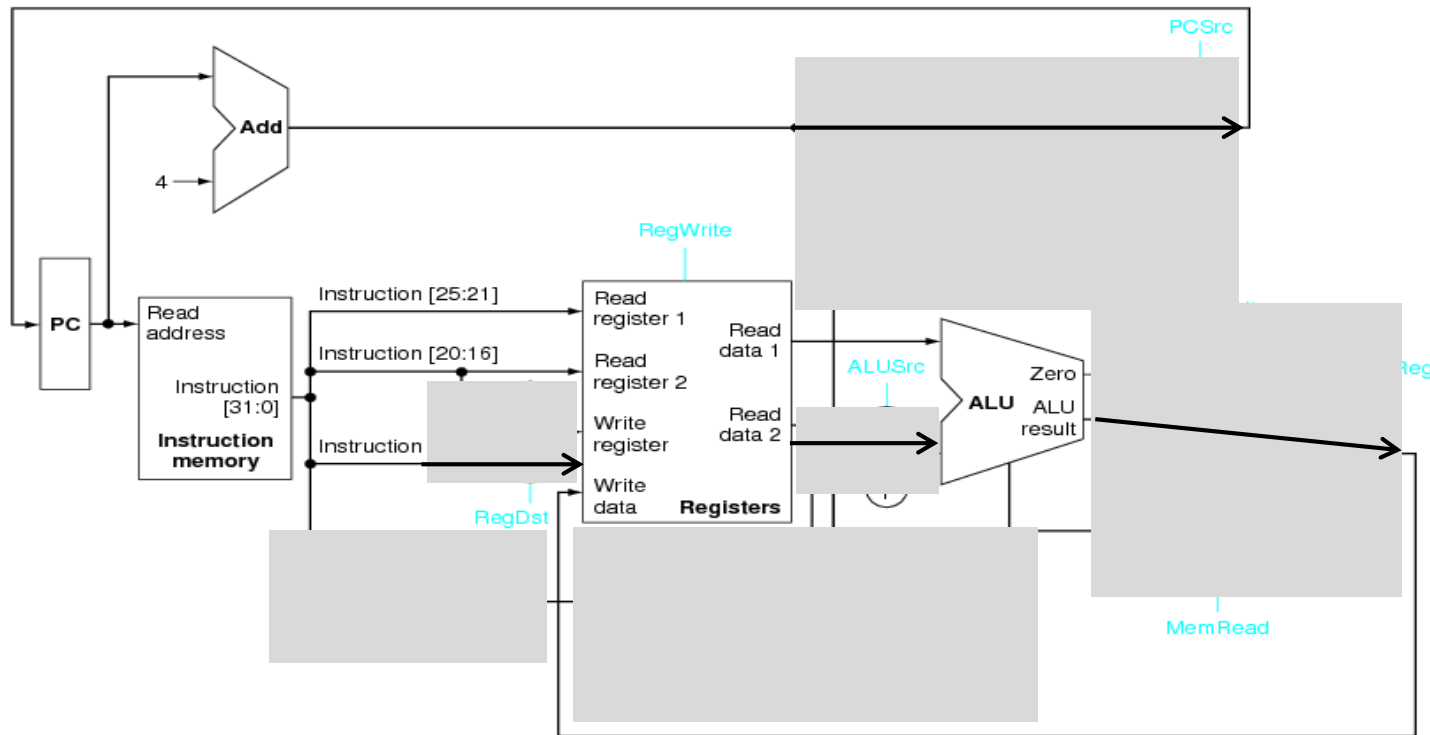
Conventional five steps (stages) of MIPS

- **IF**: Instruction Fetch from instruction memory
- **ID**: Instruction Decode and operand fetch from regfile (register file)
- **EX**: EXecute operation or calculate address for load/store or calculate branch condition and target address
- **MEM** (MA): MEMory access for load/store
- **WB**: Write result Back to regfile



Towards four stage pipelined one supporting ADD

- **IF**: Instruction Fetch from instruction memory
- **ID**: Instruction Decode and operand fetch from regfile
- **EX**: EXecute operation
- **WB**: Write result Back to regfile

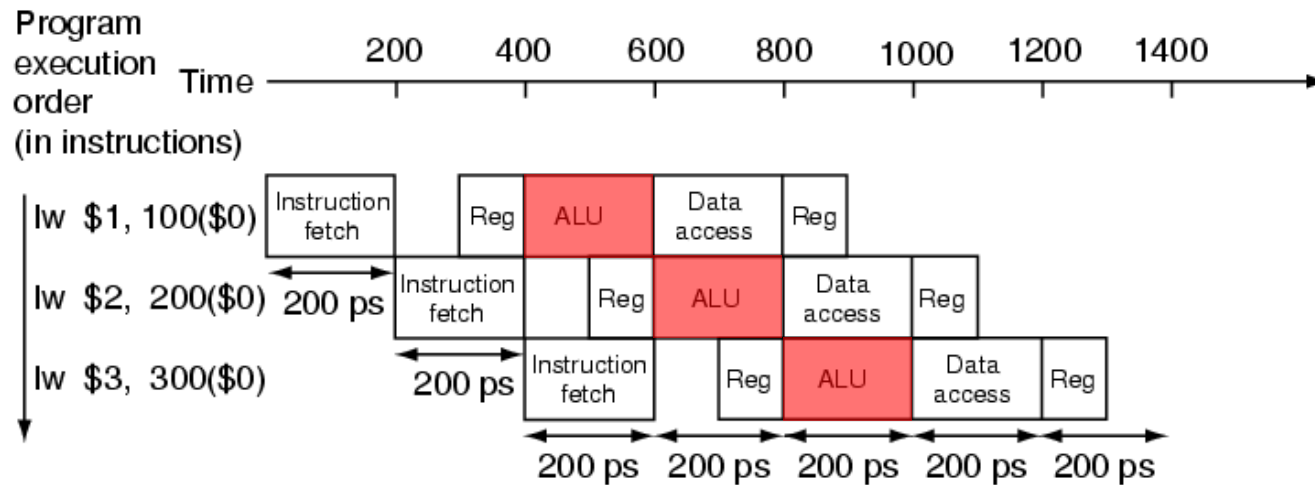
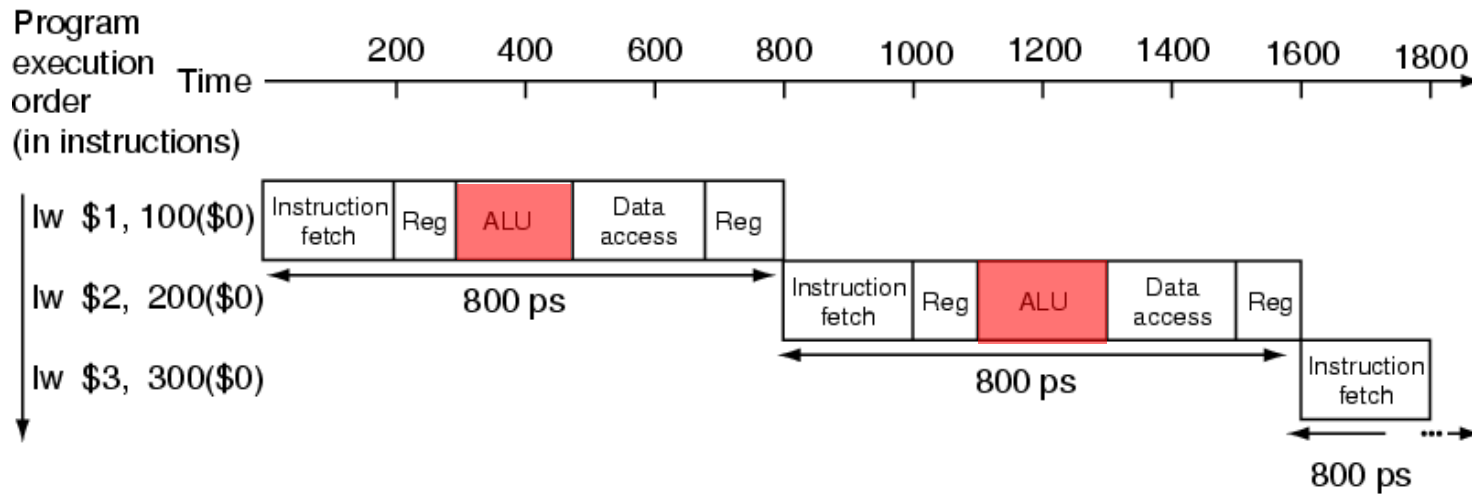








Single-cycle and pipelined processors



Assignment 3



1. Design a four stage pipelined processor supporting MIPS **add** instructions in Verilog HDL. Please download **proc01.v** from the support page and refer it.
2. Verify the behavior of designed processor using following assembly code
assuming initial values of $r[1]=22$, $r[2]=33$, $r[3]=44$, and $r[4]=55$
 - `add $0, $0, $0 # NOP {6'h0, 5'd0, 5'd0, 5'd0, 5'd0, 6'h20}`
 - `add $1, $1, $1 #`
 - `add $2, $2, $2 #`
 - `add $3, $3, $3 #`
 - `add $4, $4, $4 #`
3. Submit **your report** in a PDF file via E-mail by the beginning of the next lecture.
 - The report should include a block diagram, a source code in Verilog HDL, and obtained waveforms of your design.
 - E-mail address : `report@arch.cs.titech.ac.jp`
 - E-mail title: Assignment of Advanced Computer Architecture



MIPS Control Flow Instructions

- MIPS **conditional branch** instructions:

bne \$s0, \$s1, Lbl # go to Lbl if \$s0≠\$s1

beq \$s0, \$s1, Lbl # go to Lbl if \$s0=\$s1

- Ex: **if (i==j) h = i + j;**

bne \$s0, \$s1, Lbl1

add \$s3, \$s0, \$s1

Lbl1: ...

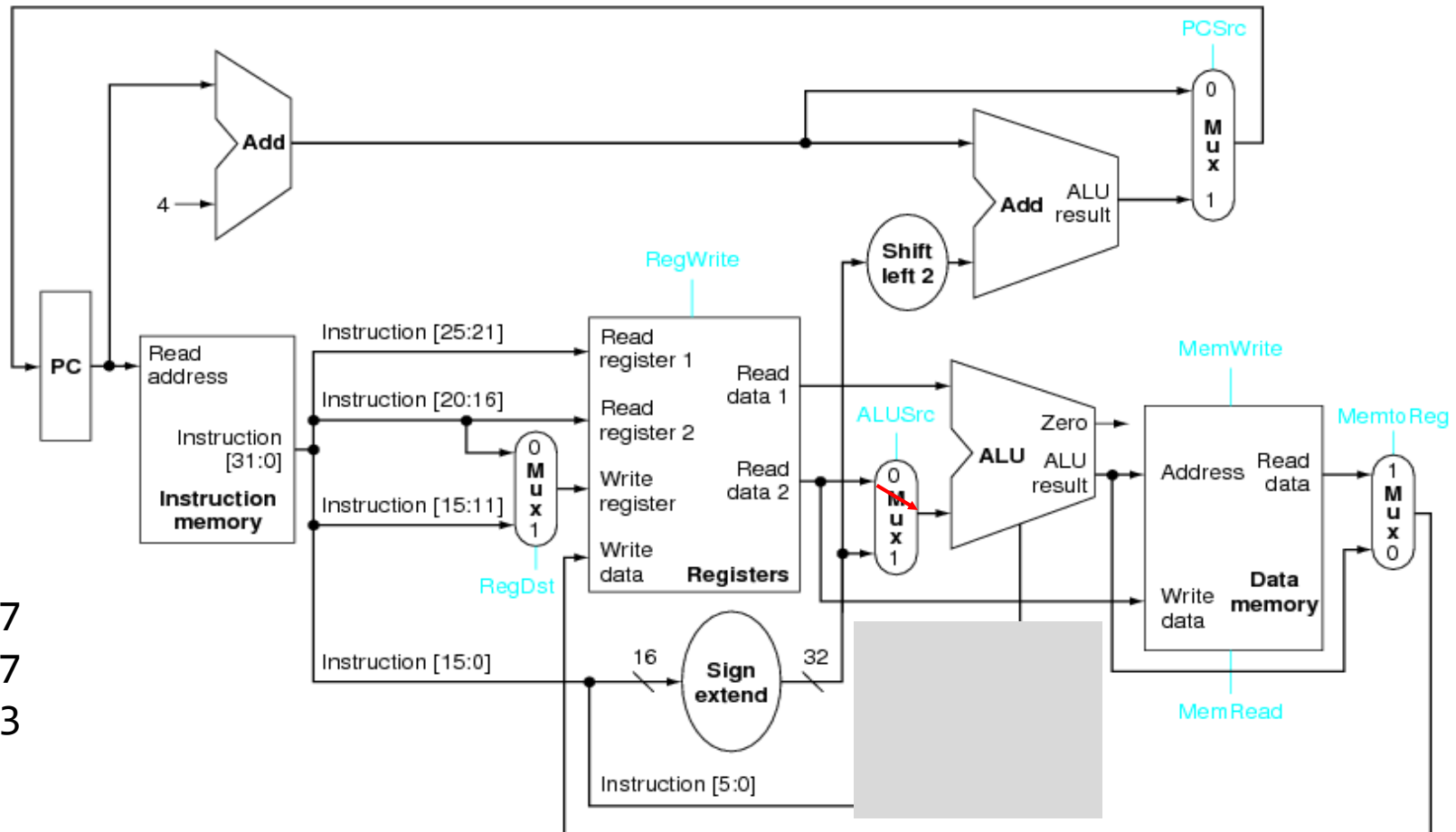
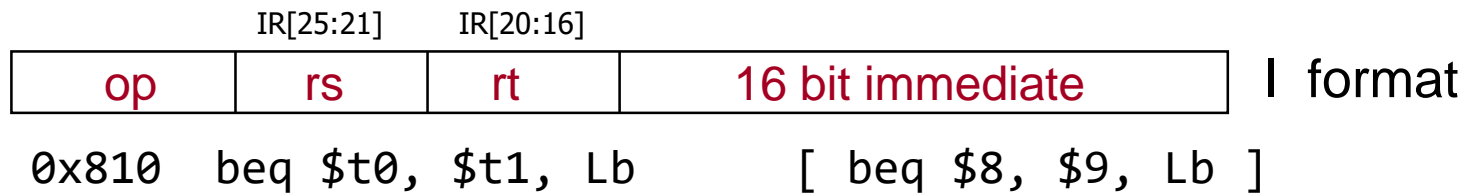
- Instruction Format (**I** format):

op	rs	rt	16 bit offset
----	----	----	---------------

- How is the branch destination address specified?



Datapath of processor supporting **ADD, ADDI, LW, SW, BNE, BEQ**



\$8 = 7
\$9 = 7
imm = -3

