

COEN 122 Computer Architecture

QUIZ 2 ON MONDAY! DO HW3 but no turn in

- Diagram will be given

From Last Time

- Single Cycle datapath
- Execution Time

Excercise: Cycle time

- Data analysis
- Assume
 - ALU delay=2ns
 - Adder (PC+4) delay = x ns
 - Adder (branch address) delay = y ns

R-type: x

$2 + 2 + 2 + 2 = 8 \rightarrow \text{execution time max}[x, 8]$

IF Reg ALU REG

beq: $\max(x, 2) + y$
 $2 + 2 + 2 = 6$

IF REG ALU

so $\max[6, \max[x, 2] + y]$

because we have more parallel things going on

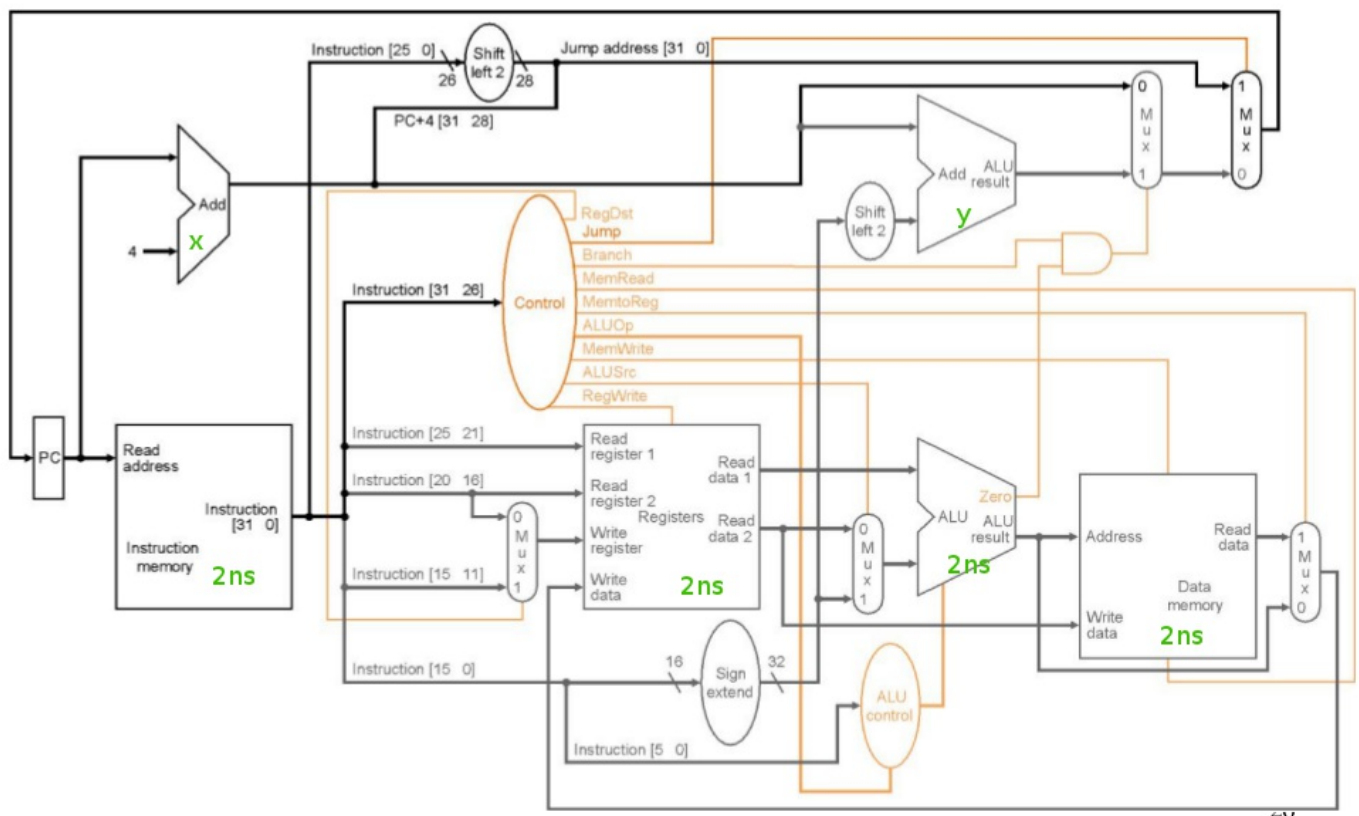
lw

sw

$x=1, y=9$

so $R = 9$ ns cycle time

beq = 11 ns cycle time

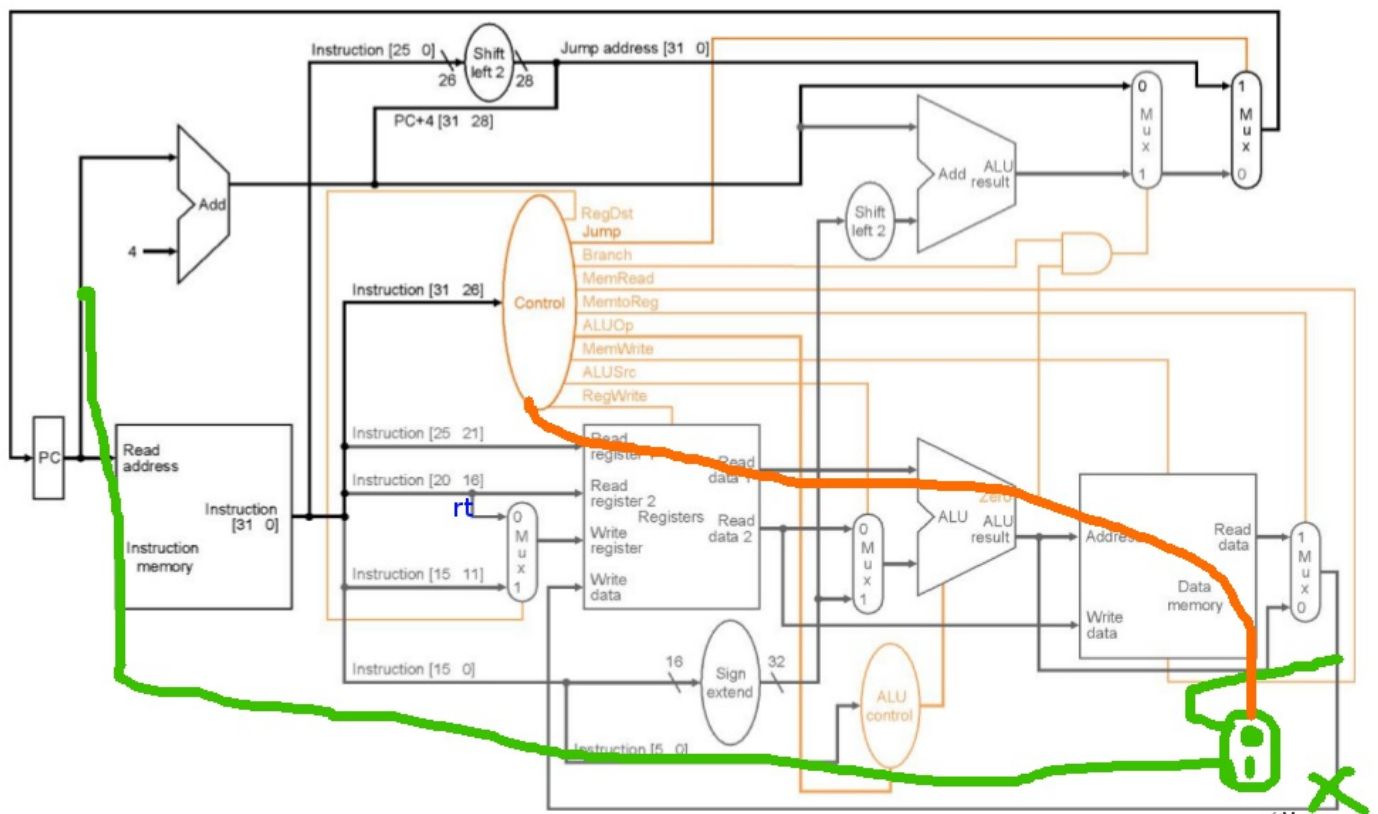


Review Quiz 2

- Don't memorize the table
- Pay attention to the control names/orders
- Mux may be reversed

2) label -> wai 5

- \$5 <- label



3) Assume we have a single stuck-at-1 fault for ALUSrc
+ ALUSrc = 1 always

- what works still? consider only 5 instructions
- look at R, lw, sw, beq, jump
- I-type instructions will work
 - lw works
 - sw works
 - j works

+ Zero=1

- R-type will work
- lw will work
- sw will work
- jump will work
- BRANCH WILL NOT

Quiz 2

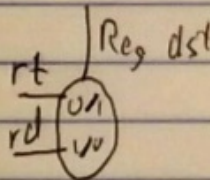
2) wai 5

$\$5 \leftarrow \text{label} \quad (\$5 \leftarrow PC)$

[wai | ^{rs}X | ^{rt} | 5 | ^{const}X]

	reg wrt	reg dst	mem to Reg	ALU wrt	ALU op (2)	mem	mem	b	j	wai
wai	1	0	X	X	XX	0	0	0	0	1

check the datapath \Rightarrow before filling in the table



0 is safe to replace don't care

Pipeline - will be on the quiz

Pipeline - Outline

- Basic idea
- Pipelined datapath
- Pipelined control
- Data hazards
- Three methods to handle data hazards - use compiler

Instruction fetch

- lw -> | 2ns | Reg/ID | Ex | mem | wrtBk |
 - | IF | ID | EX | | WB |
 - | 2ns | 2ns | 2ns | 2ns | 2ns |
- speedup will be 5, buffer between stages
- 10 ns total

Basic idea of pipelining

- roughly 5 steps for single cycle
 - IF, ID, EX, MEM, WB
- For the current step, hardware for other steps are idle
- Use the idle hardware to work on other instructions

MIPS Pipe: 5 stages

- IF: instruction fetch
- ID: instruction decode and register fetch
- EX: address calculation/excecute
- MEM: memory access
- WB: write back
- USE BUFFERS between modules

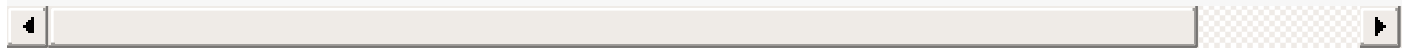
Pipelining: Control

- control passed through buffers between modules
- Size of buffer: 32 for instruction, 64 for PC+4
 - so IF/ID buffer size 64

- ID/EX buffer size $32 * 4 + 5$ (5 from read register)
- EX/MEM buffer size $32 * 3 + 1 + 5$
- MEM/WB buffer size $32 * 2 + 5$

Pipeline Control

- We have 5 stages, What needs to be controlled with each stage
 - IF and PC++ << no control
 - ID/Reg << no control
 - Exec << ALUsrc, ALUop(2), and RegDst can be moved
 - Mem << mRd, mWrt, branch
 - WB << regwrt, regDst, memtoReg
- Generate all control signals in the ID stage and pipeline 1



END