Lab 4 preview

Hung-Wei Tseng

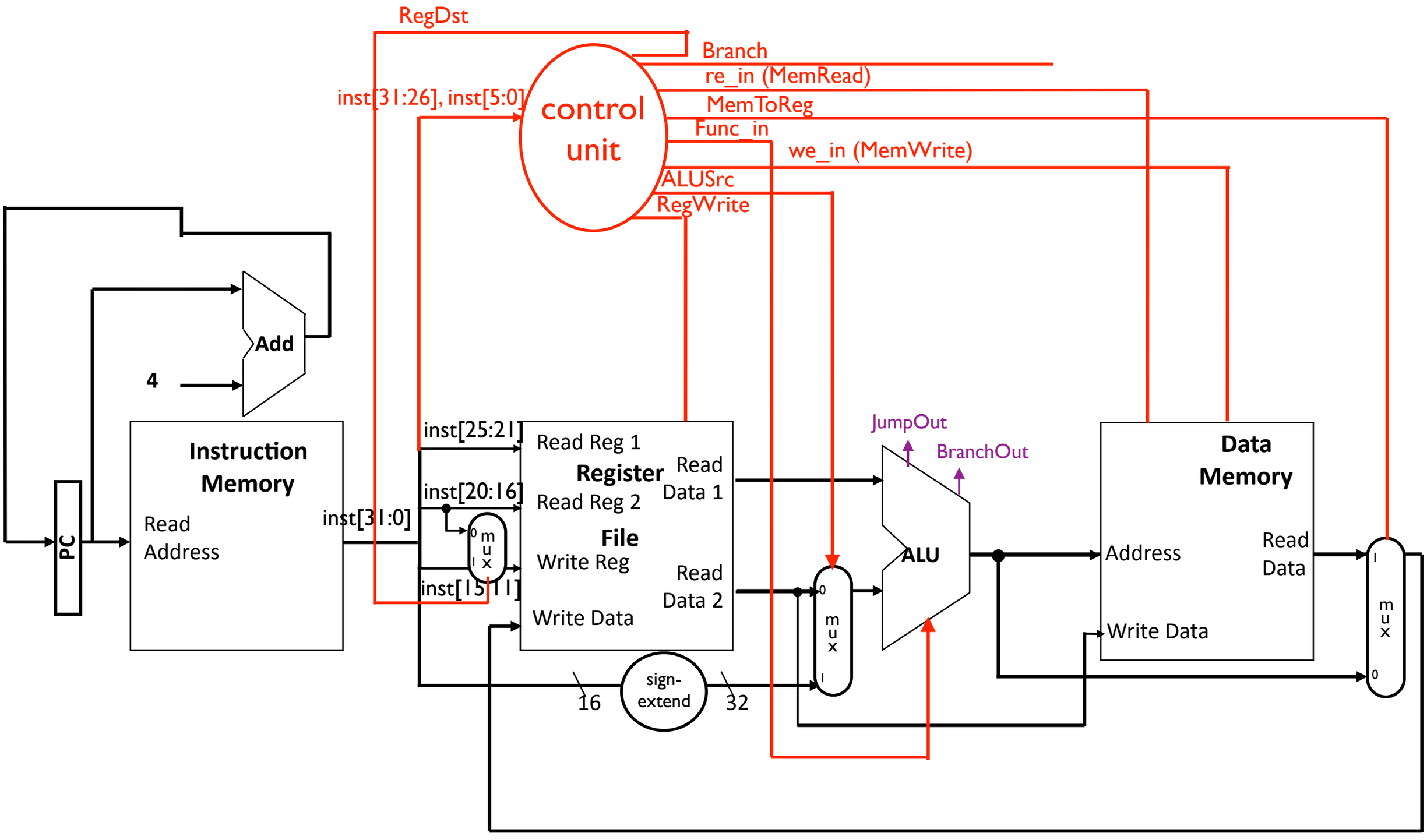
Announcement

- Lab 3 due tomorrow before 6pm
  - Interview with any of us

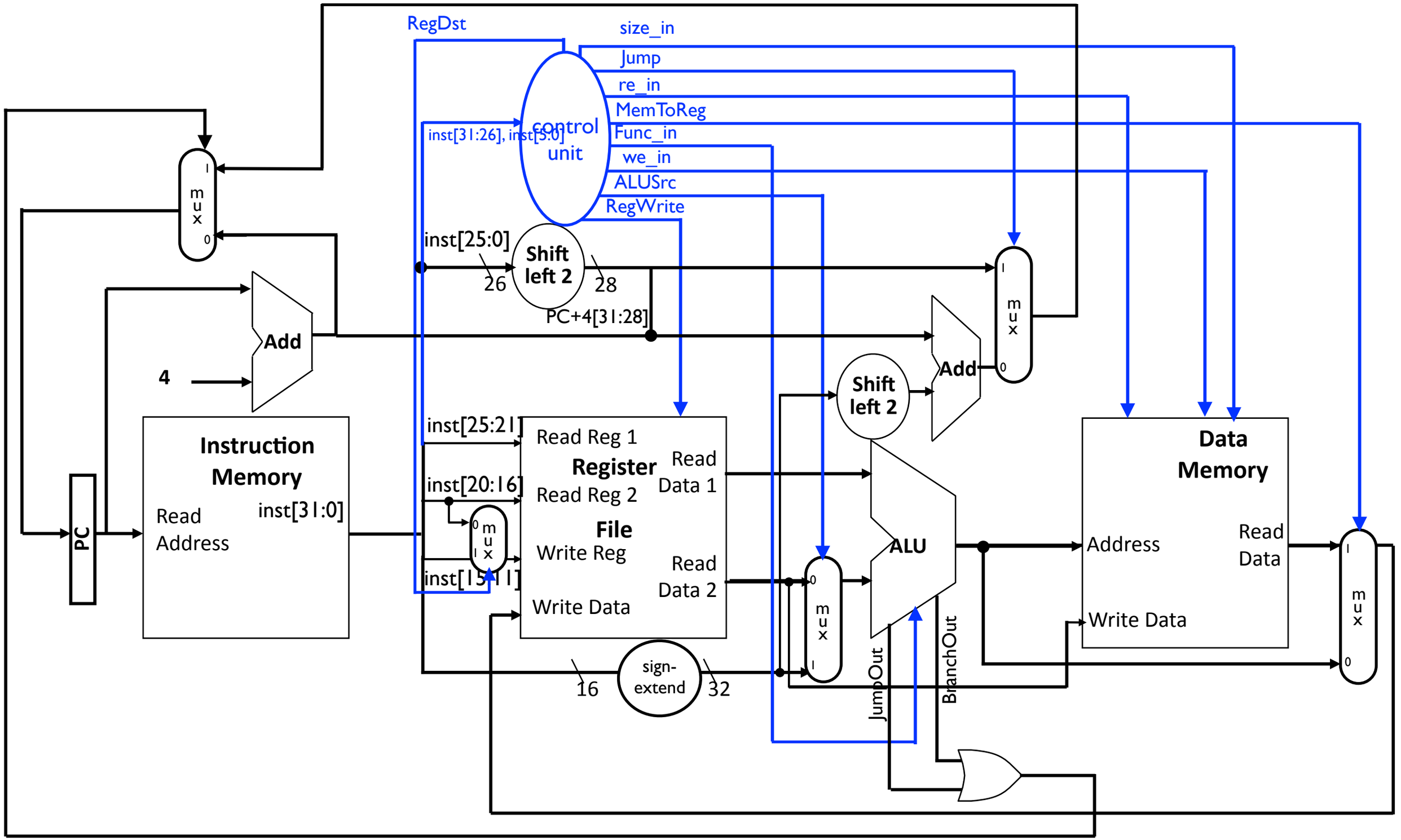
In Lab 4...

- You will be extending the datapath and control unit to support branch instructions!
  - The processor already support lw, sw, add, addi, sub, and, or nor, xor
  - We need to support
  - beq, bne, bltz, bgez, blez, bgtz, jump, jr, jal, jalr
  - lb, lh, sb, sh, lbu, lhu
  - addu, addiu, subu, andi, ori, xori, lui, slt, sltu

In lab 3, you have...



Lab 4!


RegDst
size_in
Jump
re_in
MemToReg
Func_in
we_in
ALUSrc
RegWrite
control unit
inst[31:26], inst[5:0]
inst[25:0]
26
Shift left 2
28
PC+4[31:28]
Add
4
Add
Shift left 2
Instruction Memory
Read Address
inst[31:0]
PC
inst[25:21]
inst[20:16]
inst[15:11]
Read Reg 1
Register File
Read Reg 2
Write Reg
Write Data
Read Data 1
Read Data 2
sign-extend
16
32
ALU
JumpOut
BranchOut
Data Memory
Address
Read Data
Write Data
mux

Control Unit (extended)

	instruction			control unit output								
	type	opcode inst[31:26]	funct inst[5:0]	func_in	RegDst	ALUSrc	RegWrite	MemRead	MemWrite	MemTo Reg	Jump	size_in
lb	I	0x20		100000	0	1	1	1	0	1	0	00
lh	I	0x21		100000	0	1	1	1	0	1	0	01
sb	I	0x28		100000	X	1	0	0	1	X	0	00
sh	I	0x29		100000	X	1	0	0	1	X	0	01
lbu	I	0x24		100000	0	1	1	1	0	1	0	00
lhu	I	0x25		100000	0	1	1	1	0	1	0	01
beq	I	0x4		111100	X	0	0	0	0	0	0	XX
bne	I	0x5		111101	X	0	0	0	0	0	0	XX
bltz	I	0x1		111000	X	0	0	0	0	0	0	XX
bgez	I	0x1		111001	X	0	0	0	0	0	0	XX
blez	I	0x6		111110	X	0	0	0	0	0	0	XX
bgtz	I	0x7		111111	X	0	0	0	0	0	0	XX

Control Unit (extended)

	instruction			control unit output								
	type	opcode inst[31:26]	funct inst[5:0]	func_in	RegDst	ALUSrc	RegWrite	MemRead	MemWrite	MemTo Reg	Jump	size_in
addu	R	0x0	0x21	100001	1	0	1	0	0	0	0	XX
addiu	I	0x9		100001	0	1	1	0	0	0	0	XX
subu	R	0x0	0x23	100011	1	0	1	0	0	0	0	XX
andi	I	0xC		100100	0	1	1	0	0	0	0	XX
ori	I	0xD		100101	0	1	1	0	0	0	0	XX
xori	I	0xE		100110	0	1	1	0	0	0	0	XX
slt	R	0x0	0x2A	101000	1	0	1	0	0	0	0	XX
sltu	R	0x0	0x2B	101001	1	0	1	0	0	0	0	XX
j	J	0x2		111010	0	0	0	0	0	0	1	XX
sll	R	0x0	0x0	100000	0	0	0	0	0	0	0	XX

nop

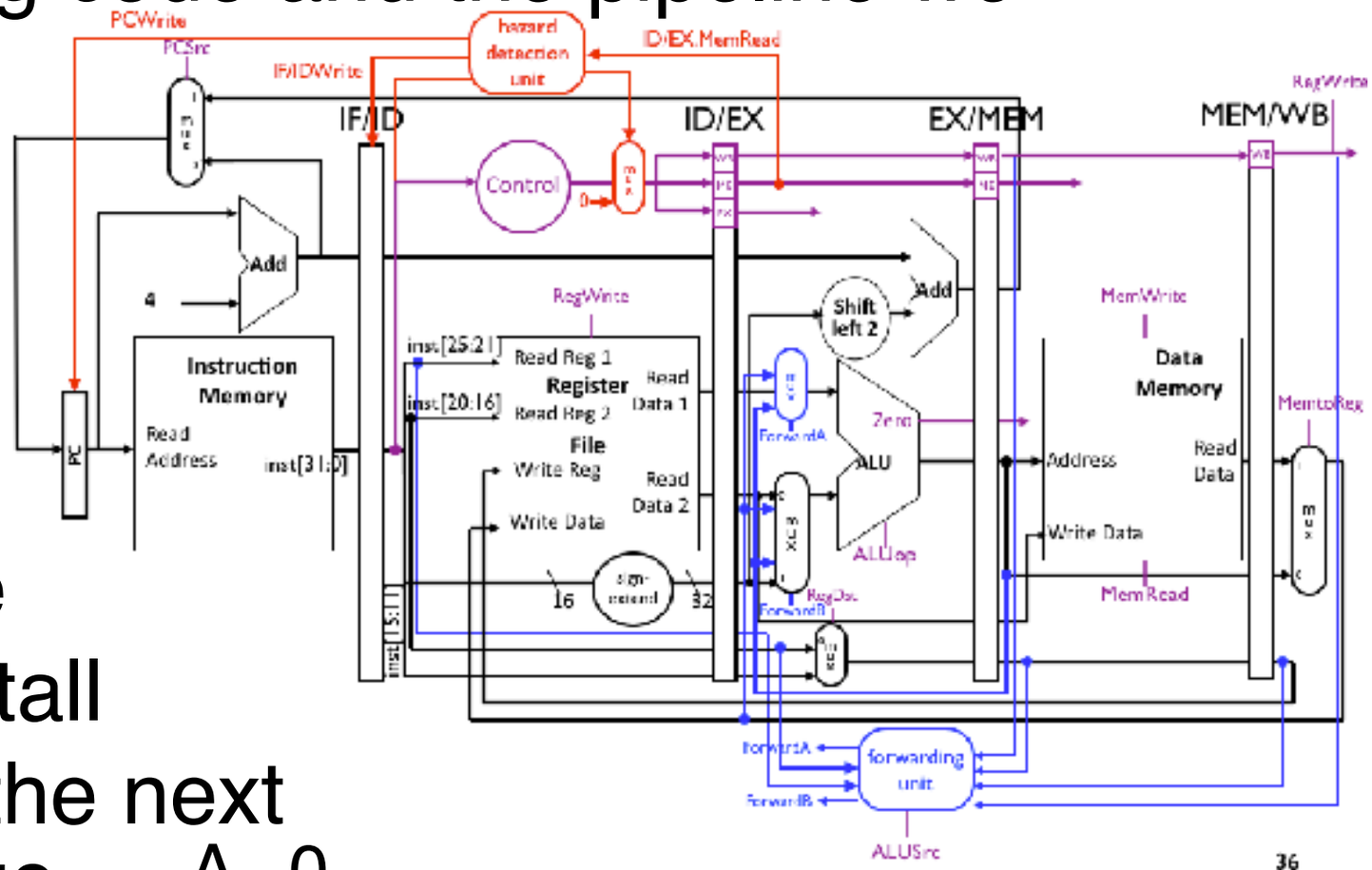
bgez and bltz

- opcode: 0x1
- rt
  - bgez: 1
  - bltz: 0

Control hazard

- Consider the following code and the pipeline we designed

```
LOOP: lw   $t3, 0($s0)
      addi $t0, $t0, 1
      add  $v0, $v0, $t3
      addi $s0, $s0, 4
      bne  $t1, $t0, LOOP
      sw   $v0, 0($s1)
```



How many cycles the processor needs to stall before we figure out the next instruction after “bne”?

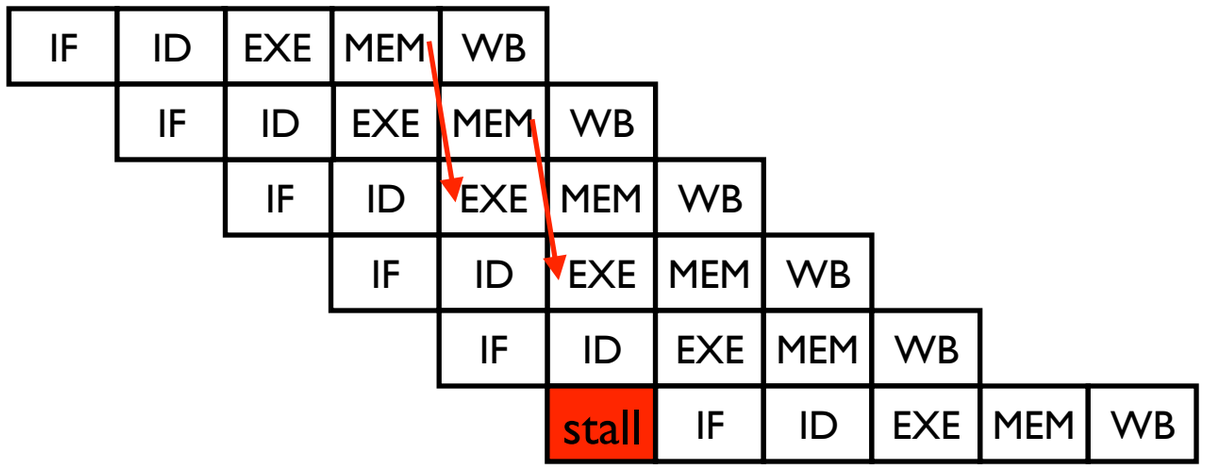
A. 0
B. 1
C. 2
D. 3
E. 4

Solution I: Delayed branches

```
LOOP: lw   $t3, 0($s0)
      addi $t0, $t0, 1
      add  $v0, $v0, $t3
      addi $s0, $s0, 4
      bne  $t1, $t0, LOOP
```

branch delay slot

```
LOOP: lw   $t3, 0($s0)
      addi $t0, $t0, 1
      add  $v0, $v0, $t3
      bne  $t1, $t0, LOOP
      addi $s0, $s0, 4
      lw   $t3, 0($s0)
```



6 cycles per loop

Solution I: Delayed branches

- An agreement between ISA and hardware
  - "Branch delay" slots: the next N instructions after a branch are **always** executed
  - Compiler decides the instructions in branch delay slots
    - Reordering the instruction cannot affect the correctness of the program
    - MIPS has one branch delay slot
- Good
  - Simple hardware
- Bad
  - N cannot change
  - Sometimes cannot find good candidates for the slot

We still need to support...

- lui (I-type)
  - $rt = {immediate, 16'b0}
- jr (R-type, func = 0x8)
  - PC = $rs
- jal (J-type)
  - $ra = PC+4
  - PC = {PC+4[31:28], imm << 2}
- jalr (R-type, func = 0x9)
  - $rd = PC+4
  - PC = $rs

Your task

- Modify the schematic to support all the required instructions
- Extend the control unit to support all the required instructions

Benchmarks

- In this lab, we provide three following benchmark programs in http://cseweb.ucsd.edu/classes/su19/cse141L-a/Media/lab4/lab4-files-2.zip
  - No branch hello world
  - ~~Hello world with branch~~
  - Fibonacci number
  - Start with PC 0x400000
    - The default PC could be 0x3FFFFC
    - But depends on your hardware design, you don't have to make it 0x3FFFFC.

Interview questions

- Show the schematics
- Show the waveforms of three benchmarks until the end
- Measure the IC, total cycles,CPI
- Report the Fmax
- We can calculate the performance of your processor now!

Q & A