HW4

Aidan Lopez

Assignment: Processor Control and Datapath

Instructor: Elaheh Sadredini

elaheh@cs.ucr.edu

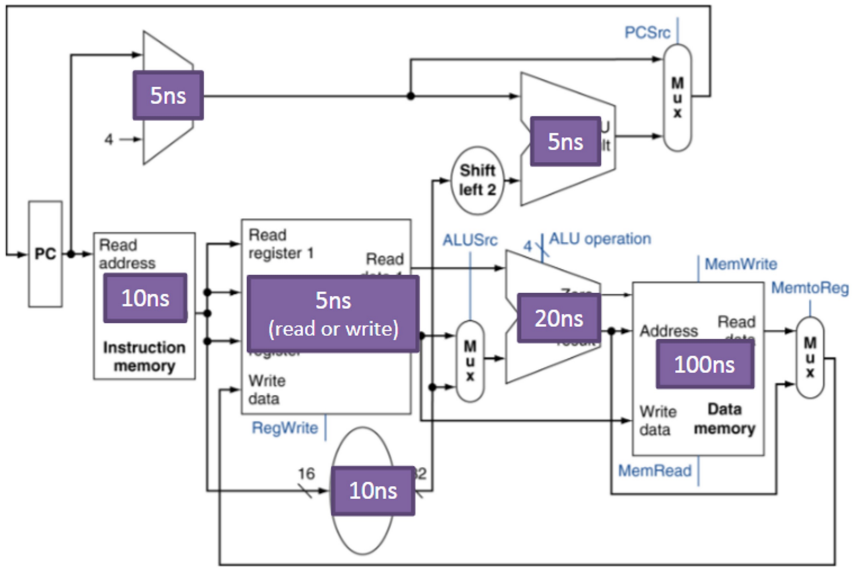
University of California, Riverside

Grading (total 4.2 points)

- Question 1: 1.2 points (part A – each part 0.3 point), 0.3 point (part B)
- Question 2: 1.2 points (each part 0.3 point)
- Question 3: 1.8 points

Question 1: Logic Path and Delay

A. Calculate the time to execute the “addi”, ”add”, “beq”, and “lw” instructions. Show the calculation. Ignore the wire delay.
B. What is the maximum speed of the processor? (hint: maximum speed is calculated based on the slowest instruction. For example, if the slowest instruction takes 50ns, the maximum processor speed is 1/50ns=20MHz).



A. Wire delay for:
   1. addi:
   2. add:
   3. beq:
   4. lw:

B. Maximum speed:

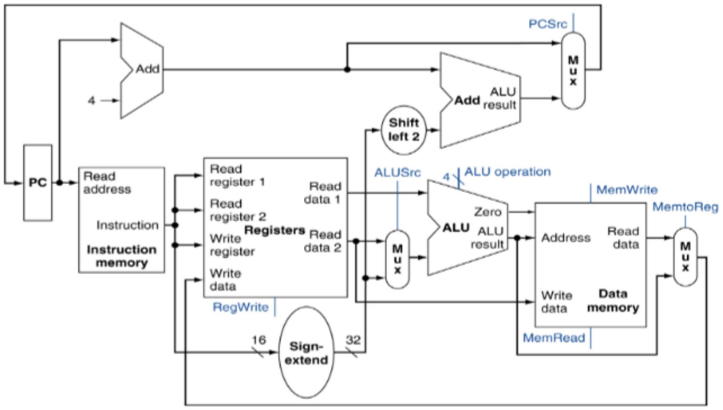
Answer:

A. addi: 5 + 10 + 5 + 10 + 20 = 50ns
   add: 5 + 5 + 10 + 20 = 40ns
   beq: 5 + 10 + 5 = 20ns
   lw: 5 + 10 + 10 + 20 + 100 = 145ns

B. Max speed = 1/145ns = 6.7MHz

Question 2

- What are the following control signals will be set into for “bne”? Explain your answer for each part.
  A. RegWrite
  B. ALUSrc
  C. ALUOp
  D. MemWrite



Answer:

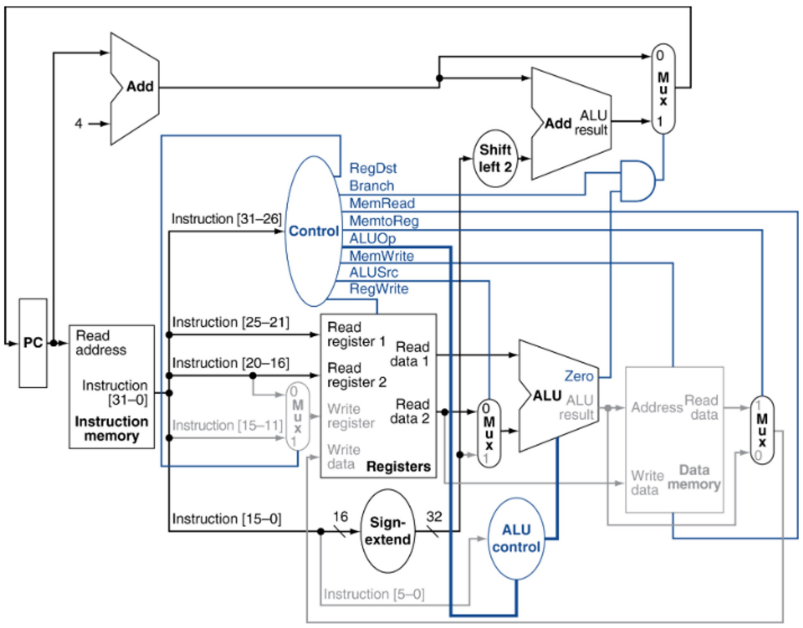
A. 0, because bne does not write to any registers, it just compares values.
B. 0, because it is comparing between registers not registers and immediates.
C. 0010 (add), uses this operation to compare the values of the registers to see if they are equal or not.
D. 0, because bne does not write anything to memory.

Question 3: Adding unconditional jump

We want to add unconditional jump to the processor datapath shown here. Here is what we need and where they are coming from.

Question 3: Adding unconditional jump

We want to add unconditional jump to the processor datapath shown here. Here is what we need and where they are coming from.



What do you need?

Address for the jump

Control signal to jump

MUX to choose the jump address

Where do they come from?

Address comes from the **instruction**

Control signal comes from the **control logic**

Question 3: Adding unconditional jump

- Question: How do you change the datapath below to enable unconditional jumps (j)? **Draw** the necessary wires, logic units, and control signals in the datapath, and explain your design.
  - Hint: "Jump" control signal will be generated from the controller, and it is true if instruction opcode is represent "J-type" instruction. Where the control signal will be connected to?

Review for Q3: addresses in branches and jumps

- **Branch instructions**
  - **bne/beq** I-format **16 bit imm**

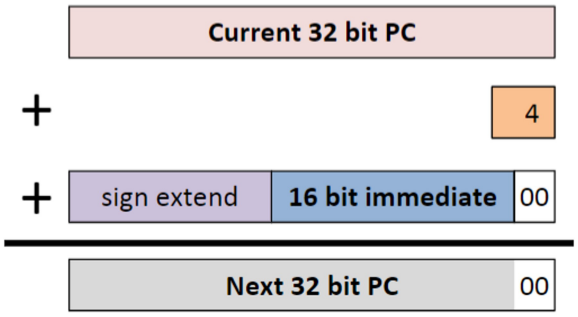
Name	Bit Fields						Notes
	6 bits	5 bits	5 bits	5 bits	5 bits	6 bits	(32 bits total)
R-Format	op	rs	rt	rd	shmt	funct	Arithmetic, logic
I-format	op	rs	rt	address/immediate (16)			Load/store, branch, immediate

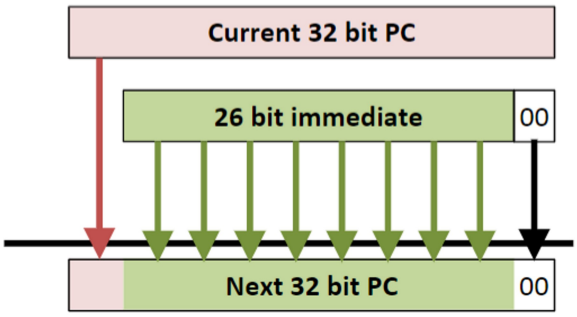
Review for Q3: addresses in branches and jumps

- **Branch instructions**
  - **bne/beq** I-format — **16 bit imm**
  - **j** J-format — **26 bit imm**

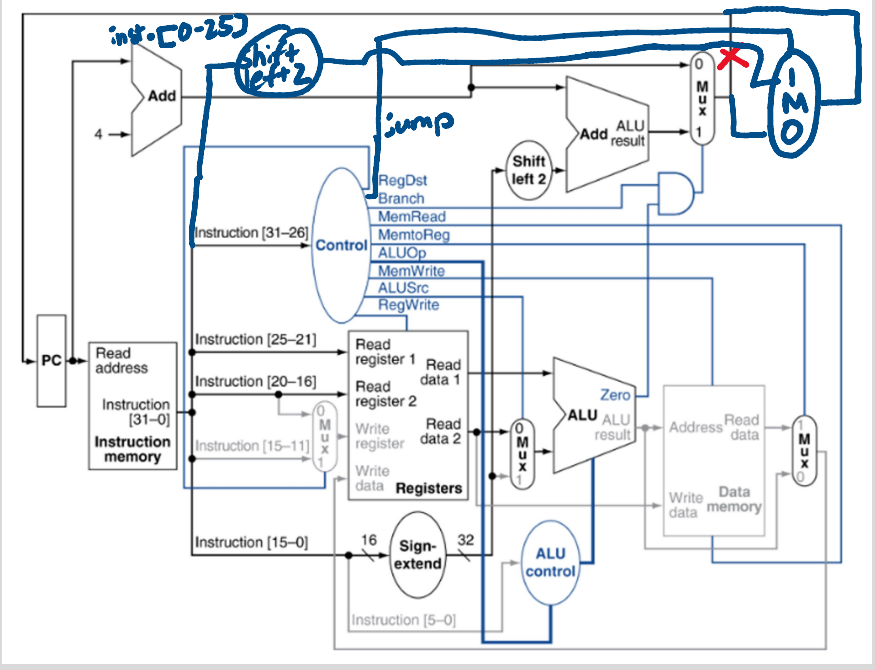
Name	Bit Fields						Notes
	6 bits	5 bits	5 bits	5 bits	5 bits	6 bits	(32 bits total)
R-Format	op	rs	rt	rd	shmt	funct	Arithmetic, logic
I-format	op	rs	rt	address/immediate (16)			Load/store, branch, immediate
J-format	op	target address (26)					Jump

- **But addresses are 32 bits!** How do we handle this?
  - Treat bne/beq as **relative offsets** (**add** to current PC)
  - Treat j as an **absolute value** (**replace** 26 bits of the PC)





Answer:



The design is mainly the same except I added another wire for jump and the immediate and a mux. The new mux will determine whether to increase the pc a normal amount or to jump to the 26 bit immediate address.