

Name: \_\_\_\_\_

# Single Cycle Implementation Homework

1. For each instruction below, determine the value at each of the labeled points in Figure 1. Assume that register  $x$  has the value  $1000 * x + 10 * x$ . For example, assume \$0 is 0, \$1 is 1010, \$2 is 2020, \$31 is 31310, etc. In addition: Assume the current instruction is at address 0x0040001C, label fwd6 is 0x00400034, and bwd6 is 0x00400004. Represent the contents of memory location  $x$  as  $M[x]$ . Some values (e.g., the output of the ALU when doing a jump) are undefined. List all well-defined values — even if that value isn't used by the instruction.

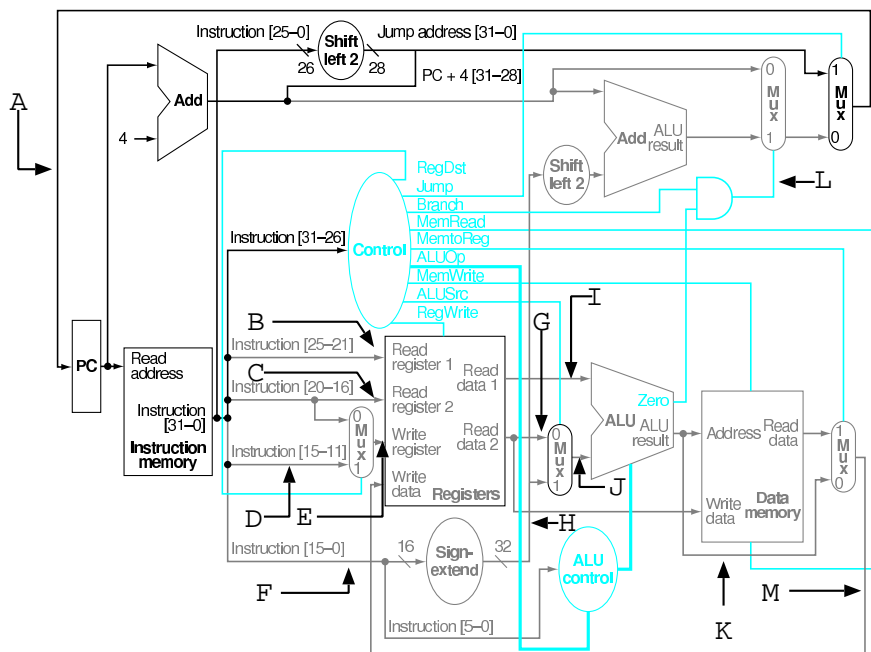
Point	add \$12, \$3, \$2	addi \$3, \$2, 131	addi \$3, \$2, -16	lw \$5, 0x5ABC(\$24)
A				
B				
C				
D				
E				
F				
G				
H				
I				
J				
K				
L				
M				

Point	sw \$4, -8(\$2)	beq \$3, \$2, fwd6	beq \$3, \$3, bwd6	j fwd6
A				
B				
C				
D				
E				
F				
G				
H				
I				
J				
K				
L				
M				

Name: \_\_\_\_\_

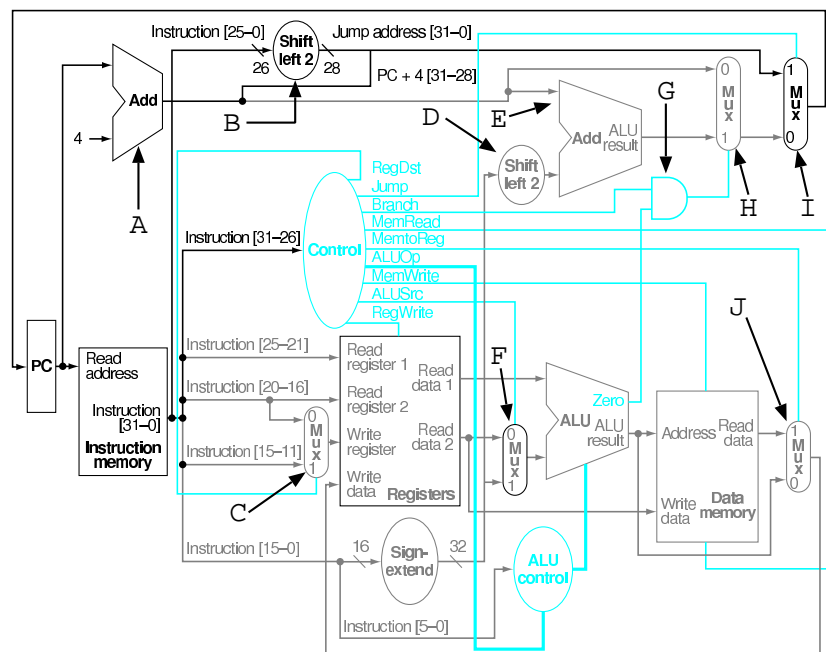
2. What is the purpose of the ALU labeled “A” in Figure 2?
3. Why does the input to the ALU labeled “A” in Figure 2 have the constant value 4?
4. What is the purpose of the “shift-left 2” labeled “B” in Figure 2?
5. Why are jump targets encoded as a word address instead of a byte address?
6. What is the purpose of the sign extender in Figure 2? Why is it necessary?
7. What is the purpose of the mux labeled “C” in Figure 2?
8. What is the purpose of the “shift-left 2” labeled “D” in Figure 2?
9. What is the purpose of the mux labeled “F” in Figure 2?
10. What is the purpose of the and gate labeled “G” in Figure 2?
11. What is the collective purpose of the muxes labeled “H” and “I” in Figure 2?
12. Does it matter whether mux “H” comes first or mux “I”?
13. What is the purpose of the mux labeled “J” in Figure 2?
14. Add components to Figure 2 to show how to add a “branch if not equal” instruction to the single-cycle CPU.

Version: 2013:vgh66



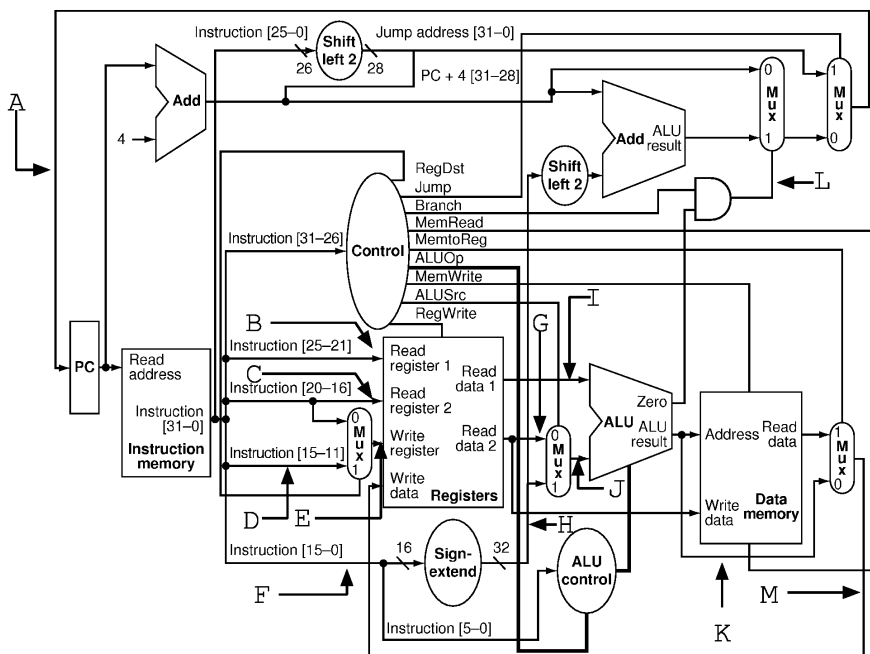
**FIGURE 4.24** The simple control and datapath are extended to handle the jump instruction. An additional multiplexor (at the upper right) is used to choose between the jump target and either the branch target or the sequential instruction following this one. This multiplexor is controlled by the jump control signal. The jump target address is obtained by shifting the lower 26 bits of the jump instruction left 2 bits, effectively adding 0 as the low-order bits, and then concatenating the upper 4 bits of PC+4 as the high-order bits, thus yielding a 32-bit address. Copyright © 2009 Elsevier, Inc. All rights reserved.

Figure 1: Single Cycle CPU with labeled points



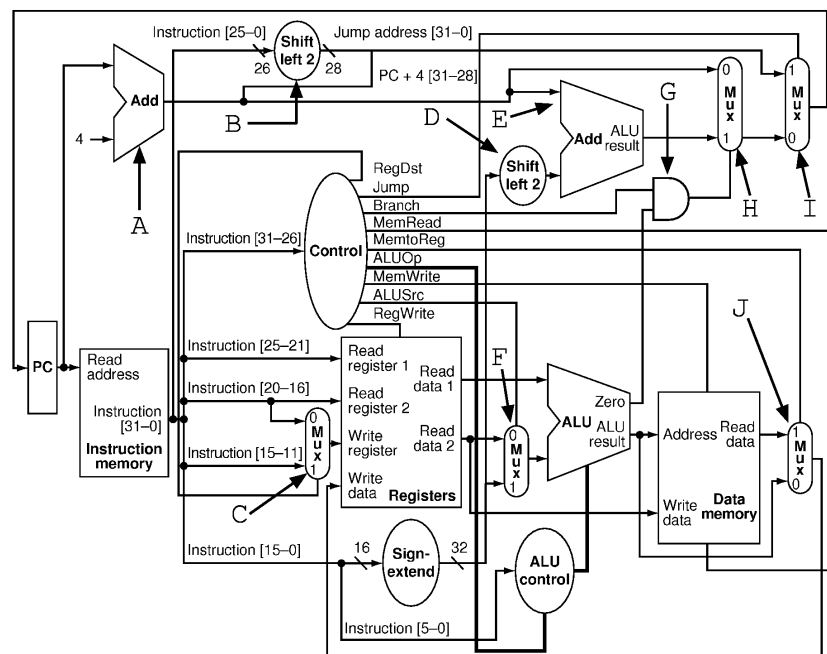
**FIGURE 4.24** The simple control and datapath are extended to handle the jump instruction. An additional multiplexor (at the upper right) is used to choose between the jump target and either the branch target or the sequential instruction following this one. This multiplexor is controlled by the jump control signal. The jump target address is obtained by shifting the lower 26 bits of the jump instruction left 2 bits, effectively adding 0 as the low-order bits, and then concatenating the upper 4 bits of PC+4 as the high-order bits, thus yielding a 32-bit address. Copyright © 2009 Elsevier, Inc. All rights reserved.

Figure 2: Single Cycle CPU with labeled parts



**FIGURE 4.24** The simple control and datapath are extended to handle the jump instruction. An additional multiplexor (at the upper right) is used to choose between the jump target and either the branch target or the sequential instruction following this one. This multiplexor is controlled by the jump control signal. The jump target address is obtained by shifting the lower 26 bits of the jump instruction left 2 bits, effectively adding 0 as the low-order bits, and then concatenating the upper 4 bits of PC + 4 as the high-order bits, thus yielding a 32-bit address. Copyright © 2009 Elsevier, Inc. All rights reserved.

Figure 3: Single Cycle CPU with labeled points (black & white)



**FIGURE 4.24** The simple control and datapath are extended to handle the jump instruction. An additional multiplexor (at the upper right) is used to choose between the jump target and either the branch target or the sequential instruction following this one. This multiplexor is controlled by the jump control signal. The jump target address is obtained by shifting the lower 26 bits of the jump instruction left 2 bits, effectively adding 00 as the low-order bits, and then concatenating the upper 4 bits of PC+4 as the high-order bits, thus yielding a 32-bit address. Copyright © 2009 Elsevier, Inc. All rights reserved.

Figure 4: Single Cycle CPU with labeled parts (black & white)