Exercise 4.9

In this exercise we examine the operation of the single-cycle datapath for a particular instruction. Problems in this exercise refer to the following MIPS instruction:

	Instruction
a.	`SW R4,-100(R16)`
b.	`SLT R1,R2,R3`

4.9.1 [10] <4.4> What is the value of the instruction word?

4.9.2 [10] <4.4> What is the register number supplied to the register file's "Read register 1" input? Is this register actually read? How about "Read register 2"?

4.9.3 [10] <4.4> What is the register number supplied to the register file's "Write register" input? Is this register actually written?

Exercise 4.9

In this exercise we examine the operation of the single-cycle datapath for a particular instruction. Problems in this exercise refer to the following MIPS instruction:

	Instruction
a.	`SW R4,-100(R16)`
b.	`SLT R1,R2,R3`

4.9.1 [10] <4.4> What is the value of the instruction word?

4.9.1

	Binary	Hexadecimal
a.	101011 10000 00100 0000000001100100	AA040064
b.	000000 00010 00011 00001 00000 101010	0043082A

In this exercise we examine the operation of the single-cycle datapath for a particular instruction. Problems in this exercise refer to the following MIPS instruction:

	Instruction
a.	`SW R4,-100(R16)`
b.	`SLT R1,R2,R3`

4.9.2 [10] <4.4> What is the register number supplied to the register file's "Read register 1" input? Is this register actually read? How about "Read register 2"?

4.9.2

	Read Register 1	Actually Read?	Read Register 2	Actually Read?
a.	16 (10000_b)	Yes	4 (00100_b)	Yes
b.	2 (00010_b)	Yes	3 (00011_b)	Yes

In this exercise we examine the operation of the single-cycle datapath for a particular instruction. Problems in this exercise refer to the following MIPS instruction:

	Instruction
a.	`SW R4,-100(R16)`
b.	`SLT R1,R2,R3`

4.9.3 [10] <4.4> What is the register number supplied to the register file's "Write register" input? Is this register actually written?

4.9.3

	Read Register 1	Register Actually Written?
a.	Either 4 (00100_b) or 0 (don't know because RegDst is X)	No
b.	1 (00001_b)	Yes

Different instructions require different control signals to be asserted in the datapath. The remaining problems in this exercise refer to the following two control signals from Figure 4.24:

	Control Signal 1	Control Signal 2
a.	ALUSrc	Branch
b.	Jump	RegDst

4.9.4 [20] <4.4> What is the value of these two signals for this instruction?

4.9.4

	Control Signal 1	Control Signal 2
a.	ALUSrc = 1	Branch = 0
b.	Jump = 0	RegDst = 1

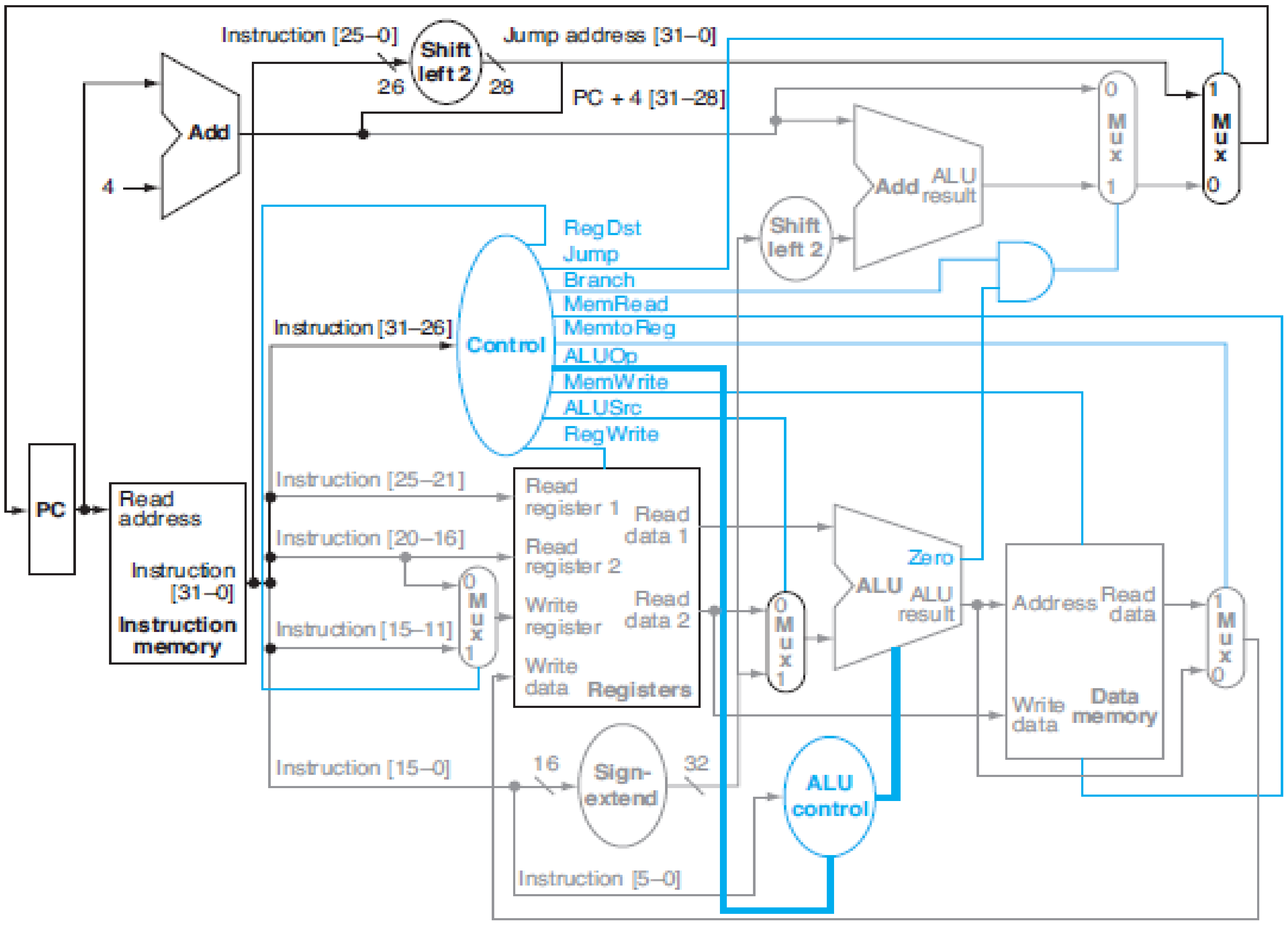

FIGURE 4.24 The simple control and datapath are extended to handle the jump instruction. An additional multiplexor (at the upper right) is used to choose between the jump target and either the branch target or the sequential instruction following this one. This multiplexor is controlled by the jump control 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.

Different instructions require different control signals to be asserted in the datapath. The remaining problems in this exercise refer to the following two control signals from Figure 4.24:

	Control Signal 1	Control Signal 2
a.	ALUSrc	Branch
b.	Jump	RegDst

4.9.5 [20] <4.4> For the datapath from Figure 4.24, draw the logic diagram for the part of the control unit that implements just the first signal. Assume that we only need to support LW, SW, BEQ, ADD, and J (jump) instructions.

4.9.5 We use I31 through I26 to denote individual bits of Instruction[31:26], which is the input to the Control unit:

a.	ALUSrc = I31
b.	Jump = (NOT I31) AND I27

Different instructions require different control signals to be asserted in the datapath. The remaining problems in this exercise refer to the following two control signals from Figure 4.24:

	Control Signal 1	Control Signal 2
a.	ALUSrc	Branch
b.	Jump	RegDst

4.9.6 [20] <4.4> Repeat 4.9.5, but now implement both of these signals.

4.9.6 If possible, we try to reuse some or all of the logic needed for one signal to help us compute the other signal at a lower cost:

a.	ALUSrc = I31
	Branch = I28

b.	RegDst = NOT I31
	Jump = RegDst AND I27