Chapter Exam

Chapter 4-The Processor

2012/06/26

1. According Figure 1, 2 and 3, please fill the results in blank (1)~(36) Table 1. And there is an instruction that it it "sub $s0, $s3, s4" and its binary code is " 000000 10011 10100 10000 00000 100010". Please fill the results in the blank (37)~(40) according points A~D in the table 2. Assume that $s3=16, $s4=4, and PC=12 . (You must draw the Table on your answer paper and fill the results) (20%).

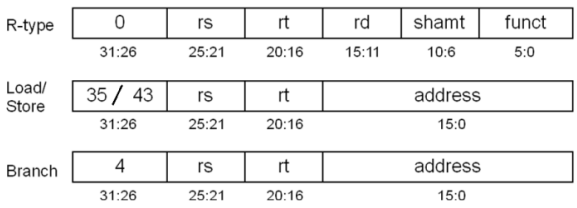

Figure 1

opcode	ALUOp	Operation	funct	ALU function	ALU control
lw	00	load word	XXXXXX	add	0010
sw	00	store word	XXXXXX	add	0010
beq	01	branch equal	XXXXXX	subtract	0110
R-type	10	add	100000	add	0010
		subtract	100010	subtract	0110
		AND	100100	AND	0000
		OR	100101	OR	0001
		set-on-less-than	101010	set-on-less-than	0111

Figure 2

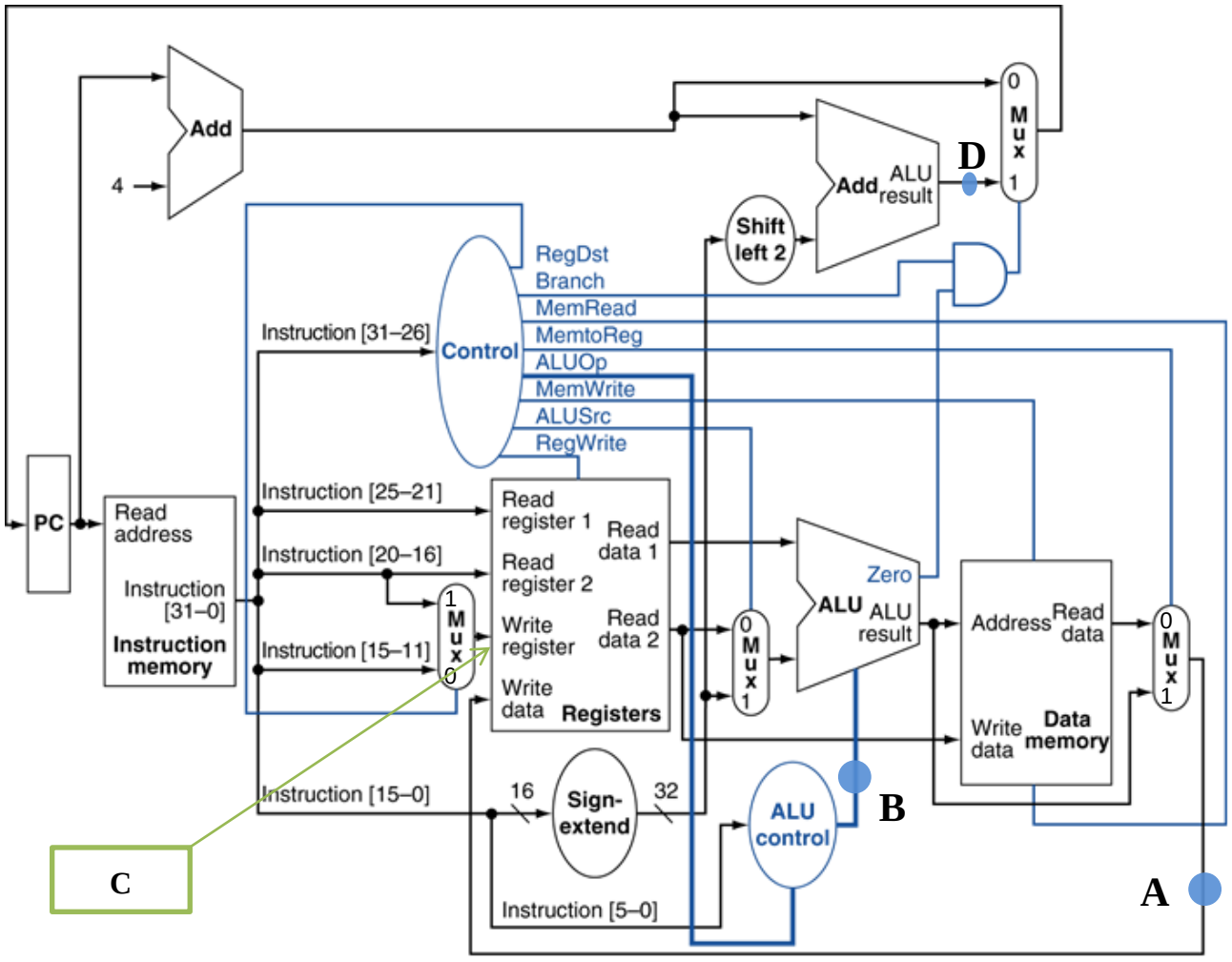

Figure 3

instruction	RegDst	ALUSrc	Memto-Reg	Reg Write	Mem Read	Mem Write	Branch	ALU Op1	ALU Op0
R-format	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
lw	(10)	(11)	(12)	(13)	(14)	(15)	(16)	(17)	(18)
sw	(19)	(20)	(21)	(22)	(23)	(24)	(25)	(26)	(27)
beq	(28)	(29)	(30)	(31)	(32)	(33)	(34)	(35)	(36)

Table 1

point	Value
A	(37)
B	(38)
C	(39)
D	(40)

Table 2

2. Consider executing the following code on the MIPS pipeline

```
sub   $1, $3, $2
lw    $2, 10($1)
add   $2, $6, $5
and   $8, $4, $2
or    $9, $8, $3
```

(1) If the forwarding unit is available, what instruction will be in each of the MIPS pipeline stages at the end of the fifth cycle of execution (IF, ID, EXE, MEM, WB)? If it has to stall, you can insert nop instruction. (5%)

(2) If the forwarding unit is not available, what instruction will be in each of the pipeline stages at the end of the fifth cycle for this instruction sequence to be executed correctly? If it has to stall, you can insert nop instruction. (5%)

3. The textbook presents the following development issues in a specific sequence (from simple to complex), in order to show a typical research and development (R&D) process. Please fill in the sequence number for the corresponding issue (1 being the simplest and 9 being the most complex). (10%)

Development Issue	Sequence (1…9)
Hazard Detection Unit	
Data hazards	
Single cycle implementation	
2-bit Dynamic Branch Prediction	
Delayed Branch	
Exception handling	
Control Hazards	
Pipelining	
Forwarding (bypassing) Unit	

4. This exercise examines the accuracy of various branch predictors for the following repeating pattern (e.g., in a loop) of branch outcomes: (T: taken, NT: not taken)

	Branch outcomes
a.	T, T, T, T, NT

What is the accuracy of the two –bit predictor if this pattern is repeated forever? (10%)

5. In this exercise we examine how latencies of individual components of the datapath affect the clock cycle time of the entire datapath, and how these components are utilized by instructions. For problems in this exercise, assume the following latencies for logic blocks in the datapath:

	I-Mem	Add	Mux	ALU	Regs	D-Mem	Sign-extend	Shift-left-2
a.	500ps	150ps	100ps	180ps	220ps	1000ps	90ps	20ps

(1) What is the clock cycle time in Figure 3 datapath if the only type of instructions we need to support are ALU instructions (add, and, etc)? (5%)

(2) What is the clock cycle time in Figure 3 datapath if we only had to support lw instruction? (5%)

6. In the pipeline structure, there are situation called hazards that the next instruction cannot execute in the following clock cycle. Briefly describe the following concepts (1) Pipeline (2) The three types of hazards defined in the textbook (10%)

7. The following program is increment the content of an array in a loop fashion.

```
Loop:   lw    $t0, 0($s1)      //$t0=array element
        addu  $t0, $t0, $s2    //add scalar in $s2
        sw    $t0, 0($s1)      //store result
        addi  $s1, $s1, -4     //decrement pointer
        bne   $s1, $zero, Loop //branch $s1!=0
```

(1). Looping unrolling is a technique to enhance instruction level parallelism. The following program adopts the loop unrolling technique by unrolling the loop twice to enhance the instruction level parallelism. Please fill in the proper values in the blank areas (1, 2, 3, 4, 5) in the program:

```
Loop:   addi  $s1, $s1, (1)
        lw    $t0, (2)($s1)
        addu  $t0, $t0, $s2
        sw    $t0, (3)($s1)
        lw    $t1 (4)($s1)
        addu  $t1 $t1 $s2
        sw    $t1 (5)($s1)
        lw    $t2 4($s1)
        addu  $t2 $t2 $s2
        sw    $t2 4($s1)
        bne   $s1, $zero, Loop
```

(2). Schedule the above loop unrolled program into a two-issue MIPS machine and fill in the proper values (6, 7, 8, 9, 10). The two-issue MIPS can execute ALU or branch instructions in one pipeline, and execute the data transfer instructions in another pipeline. (20%)

	ALU or branch instruction	data transfer instruction	clock cycle
Loop:	addi $s1,$s1, **(6)**	lw $t0, **(7)**($s1)	1
		lw $t1, **(8)**($s1)	2
	addu $t0, $t0, $s2	lw $t2, 4($s1)	3
	addu $t1, $t1, $s2	sw $t0, **(9)**($s1)	4
	addu $t2, $t2, $s2	sw $t1, **(10)**($s1)	5
	Bne $s1, $zero, Loop	sw $t2, 4($s1)	6

8. Given this instruction sequence,

```
40   sub   $11, $2, $4
44   and   $12, $2, $5
48   or    $13, $2, $6
4C   add   $1, $2, $1
50   slt   $15, $6, $7
54   lw    $16, 50($7)
…
```

assume the interrupt vectors to be invoked on an exception begin like this:

```
80000180   sw   $25, 1000($0)   //arithmetic overflow jumps to this address
80000184   sw   $26, 1004($0)   //load abort jumps to this address
```

If an overflow exception occurs in the add instruction. What are contains of blank (1)~(5) in the Figure 4 and 5? (20%)

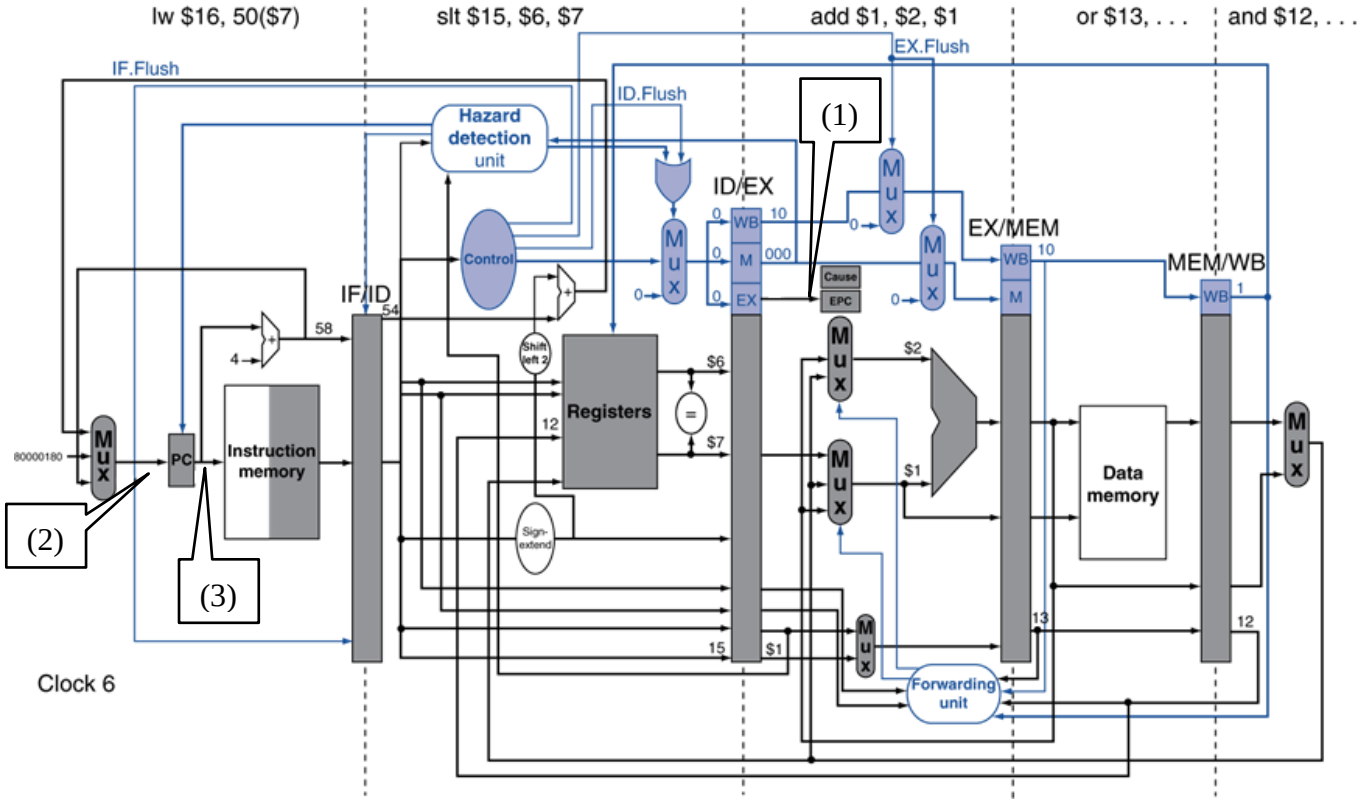

Figure 4

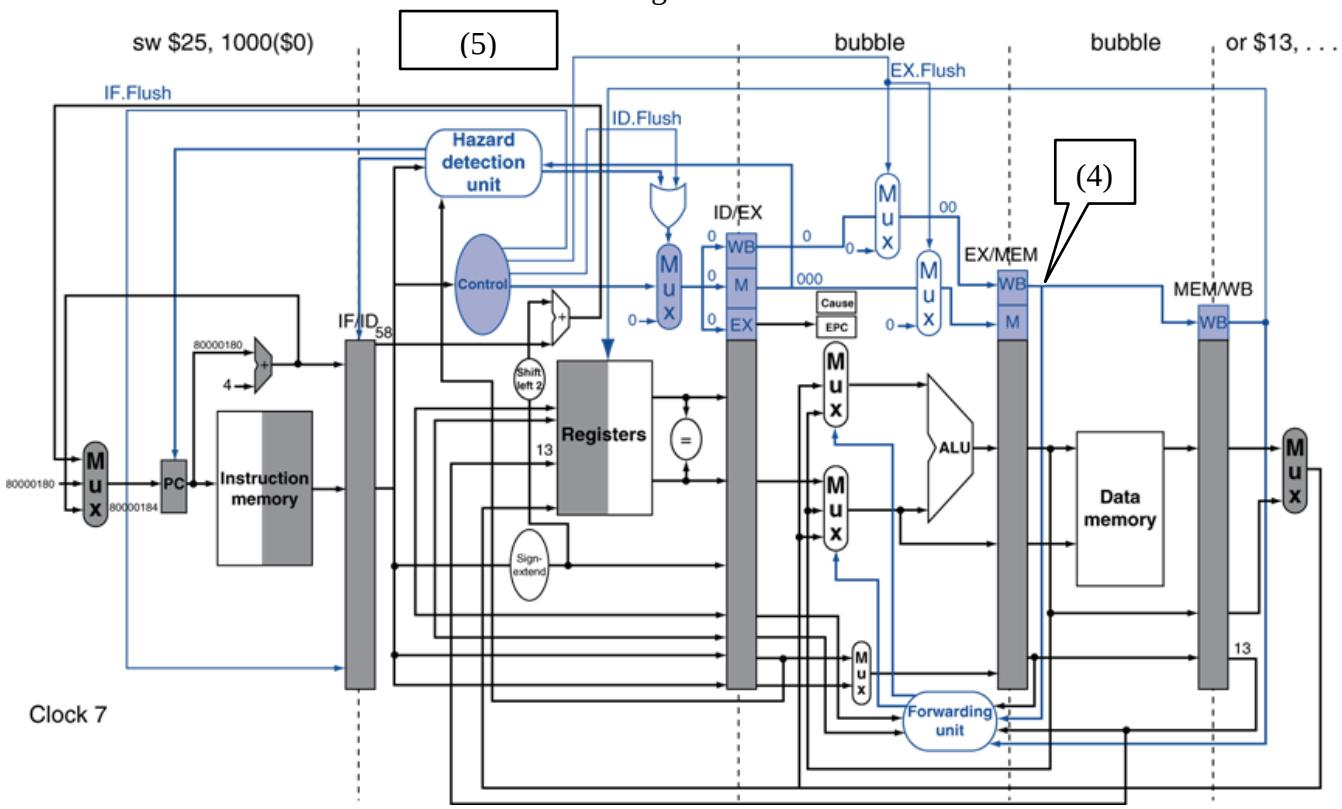

Figure 5