COMP 230: Computer Architecture and Organization **Fall 2017**

Exam 2 (Practice)

Instructions:

- **This practice exam is meant to give you the flavor of questions that will be asked on the exam. Do not expect the real exam to be the same questions with only the numbers or MIPS instructions changed.**
- Your real exam will be open book and notes. However, you are not allowed to use laptops, cell phones, calculators, or other electronics.
- If you do not show your work, do not expect partial credit for incorrect answers.
- If you believe a problem is incorrectly or incompletely specified, make a reasonable assumption and solve the problem. The assumption should not result in a trivial solution.
- In all cases, clearly state any assumptions you make in your answers.

Name	

Question	**Points Possible**	**Grade**
1	10	
2	20	
3	20	
4	20	
5	30	
Total	100	

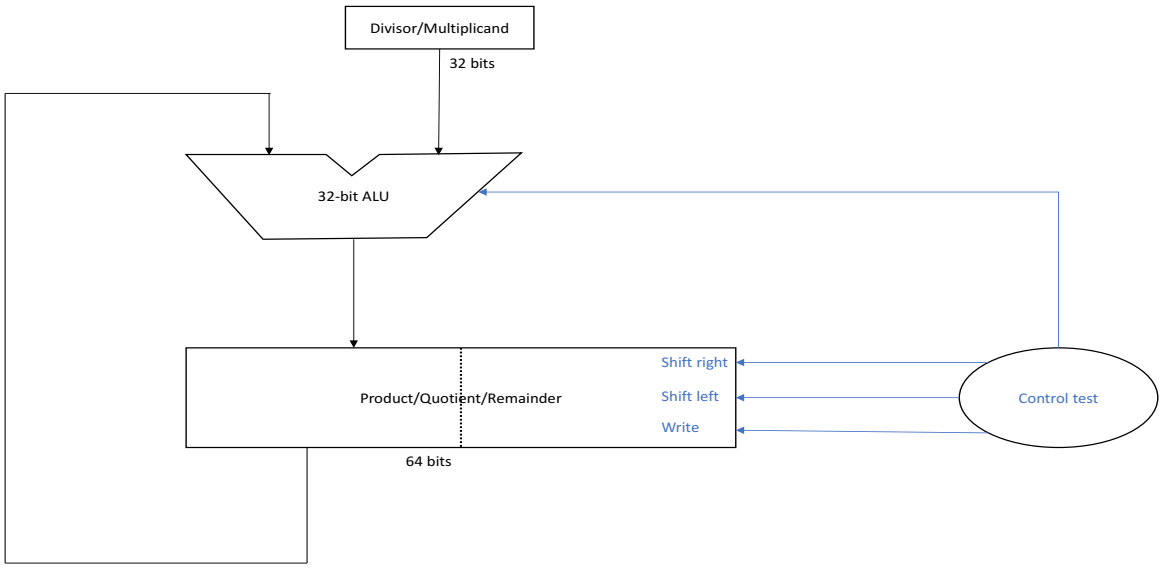

Figure 1: Hardware components for multiplication and division.

1. (a) Complete the following table to divide 12 by 5 using the hardware in Figure 1. Assume a machine that represents integers using only 4 bits.

Iteration	Step	8 Bits
0	initial	00001100
	shift	00011000
1	subtract?	−−−−
	shift	00110000
2	subtract?	1
	shift	01100000
3	subtract?	00010000
	shift	00100001
4	subtract?	−−−−
	shift	01000010
*	special shift	00100010

(b) If our MIPS machine operates on 4-bit words (so that we only have 4-bits to represent our integers), after a divide instruction, what instruction puts the quotient into `$t0`?

```
mflo $t0
```

2. Add 1.75 and 0.390625 using binary floating-point addition, assuming one round bit, one guard bit, one sticky bit, and using 5 fraction bits (5 bits of precision). Show your work. Also, what would the result be without a sticky bit?

$$1.75 = 1.11 \times 2^0 \quad = 1.11000 \mid 00$$
$$0.390625 = 0.011001 \times 2^0 \quad = 0.01100 \mid 10$$
$$10.00100 \mid 10$$

The result is 1.00010×2^1 no matter how you round. The sticky bit actually does not matter for this problem.

3. Write MIPS assembly code for a procedure `divides` that takes in two integers a and b as parameters. If a is divisible by b `divides` should return 1; otherwise 0.

```
divides:  div   $a0,  $a1
          mfhi  $v0
          beq   $v0,  $zero,  yes
no:       addi  $v0,  $zero,  1
          j     exit
yes:      add   $v0,  $zero,  $zero
exit:     jr    $ra
```

4. You have just written a new program for your company, which exhibits no pipeline stalls and has the following breakdown of executed instructions:

`add`	`addi`	`not`	`beq`	`lw`	`sw`
20%	20%	0%	25%	25%	10%

   (a) In what fraction of all cycles is the data memory used?

   $25\% + 10\% = 35\%$

   (b) In what fraction of all cycles is the input of the sign-extend circuit needed? What is this circuit doing in cycles in which its input is not needed?

   $20\% + 25\% + 25\% + 10\% = 80\%$

5. Consider the following sequence of instructions, executing using the pipeline shown in figure 2:

```
add  $s1,  $s2,  $s3
add  $s2,  $s1,  $s4
add  $s1,  $s1,  $s2
```

   (a) Assuming no forwarding, what are the hazards?

   3 data hazards: `$s1` RAW (read-after-write) between instructions 1 and 2, `$s2` RAW between instructions 2 and 3, and `$s1` RAW between instructions 1 and 3.

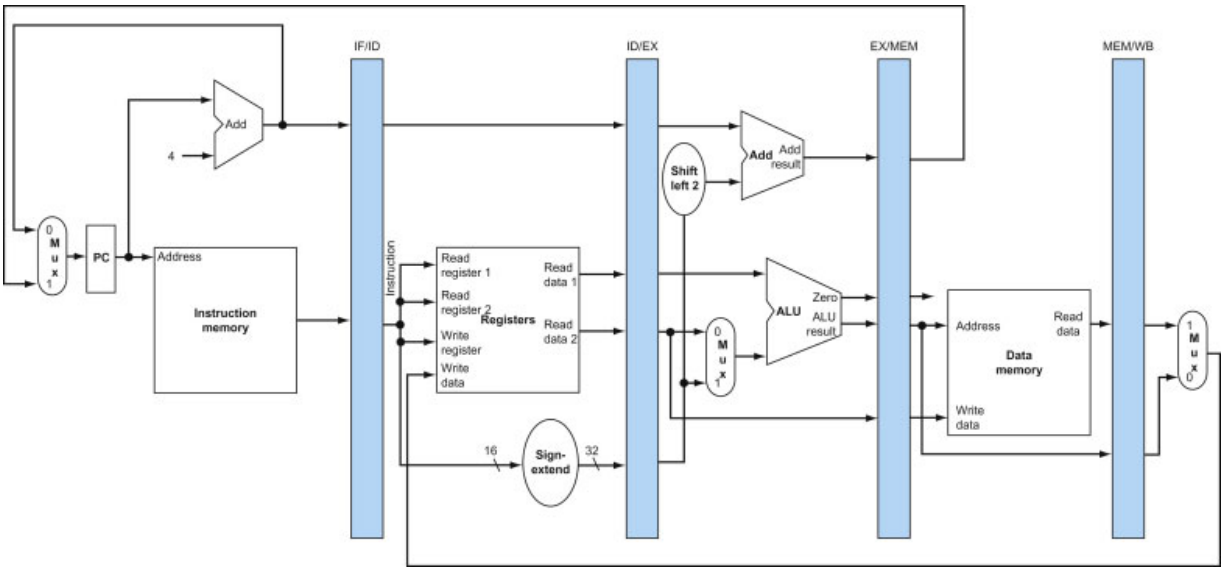

Figure 2: A simple MIPS pipeline.

(b) Assuming no forwarding, how many cycles does this sequence take to execute?

There are two bubbles after each add, so we can only start the last add on cycle 7. Then we need 4 more cycles to finish it, so 11 cycles total.

(c) Assuming full forwarding, what hazards remain?

None.

(d) Assuming full forwarding, how many cycles does this sequence take to execute?

Start the final add on the third cycle, then four more to finish it. In total, 7.