CSE 410 Midterm Exam 5/03/10

Name ________________________________

Do **not** write your id number or any other confidential information on this page.

There are 6 questions worth a total of 100 points. Please budget your time so you get to all of the questions. Keep your answers brief and to the point.

You will want to use a copy of the "green card" from the textbook. We have additional copies if you did not bring one with you. Other than that, the exam is closed book, closed notes, closed calculators, closed laptops, closed twitter, closed telepathy, etc.

Please wait to turn the page until everyone is told to begin.

Score __________________ / 100

1. _______ / 14

2. _______ / 18

3. _______ / 14

4. _______ / 20

5. _______ / 22

6. _______ / 12

Question 1. (14 points) This question involves **12-bit signed, 2's complement** binary numbers.

(a) Give the 12-bit 2's complement binary and hexadecimal representations of the decimal number -320.

(b) Translate the 12-bit hexadecimal number 0xF00 to binary and then give its decimal value if we interpret it as a 2's complement binary integer.

Powers of 2 and 16 for reference.

Number	Hex	Decimal
2^0	16^0	1
2^1		2
2^2		4
2^3		8
2^4	16^1	16
2^5		32
2^6		64
2^7		128
2^8	16^2	256
2^9		512
2^{10}		1024
2^{11}		2048
2^{12}	16^3	4096

Question 2. (18 points) The pseudo-question. (But you have to provide a real answer)

Consider the following fragment of a MIPS assembly language program.

```
        bgt     $t3, $a0, there
        add     $t3, $t3, $a0
        move    $v0, $t3
there:
        li      $t0, 0x10010110
```

This fragment contains several assembler pseudo instructions as well as a real MIPS machine instruction or two (or maybe more, or maybe less).

Re-write this fragment of code so it uses only MIPS machine language instructions. You should replace each pseudo-instruction in the code with machine language instructions, as would be done by a MIPS assembler like SPIM.

Question 3. (14 points) Suppose we have a 32-bit MIPS word containing the value `0x000590C0`. We would like to know what MIPS machine instruction this represents.

(a) Write this instruction word in binary. Leave enough spaces between the digits for part (c) of the question.

(b) What is the format of this instruction? (circle)

R I J

(c) In your answer to part (a), draw boxes around the bits that make up the different fields of the instruction and then label the instruction fields (opcode, rs, etc.)

(d) Translate this instruction to assembly language. Use symbolic register names like `$t0` instead of absolute register numbers like `$8`.

Question 4. (20 points) Suppose we have the following four instructions in a fragment of a program.

```
add  $t1, $a0, $t4

add  $t5, $t4, $t2

lw   $t0, 12($t5)

add  $v0, $t0, $t1
```

(a) Identify all of the data dependencies in the above code. You can either write an answer below, or circle the affected registers in the above code and draw lines to indicate the data dependencies between instructions.

(b) Fill in the following pipeline timing diagram to show how those instructions would execute on a machine **with** internal forwarding of data from one instruction to the next. You should also assume that a register can be written with a new value at the beginning of a cycle and the new value can be read out at the end of the same cycle. Indicate a stall by leaving an entry blank or writing "stall" or "NOP". The first three cycles of the first instruction (only) are filled in for you. (You may not need all of the columns.)

cycle	1	2	3	4	5	6	7	8	9	10	11	12	13	14
`add`	IF	ID	EX											
`add`														
`lw`														
`add`														

(c) If there are any stall cycles in the timing diagram from part (b), is there any way to rearrange or alter these instructions to reduce the number of stalls? If so, how would you do it, and how many stall cycles (if any) are left in the final schedule?

Question 5. (22 points) A little string programming. For this problem, implement a MIPS assembly language procedure `strcat` that appends a copy of one string to the end of another. For instance, if strings `s` and `t` are

s	i	c	e	\0	?	?	?	?	?	?	?
t	c	r	e	a	m	\0	?	?	?	?	?

then after executing `strcat(s,t)` the strings should look as follows.

s	i	c	e	c	r	e	a	m	\0	?	?
t	c	r	e	a	m	\0	?	?	?	?	?

In the diagrams, `\0` indicates a byte containing binary 0 (`0x00`), and *?* indicates bytes whose contents are unknown.

Your code should use the standard MIPS calling conventions. You should assume that the first string has enough unused space at the end to hold a copy of the second string, and that both strings are properly terminated with a `\0` byte at the end. The `strcat` procedure has no return value (i.e., it would be a `void` function in C or Java).

Please supply reasonable comments so we can follow your use of registers and memory.

```
# append a copy of the second string argument to the first
# string argument.

strcat:
```

(additional room on the next page for the rest of your answer if needed)

Question 5. (cont) Additional room for the rest of your `strcat` procedure if needed.

Question 6. (12 points) When most programs are executed, their memory references exhibit a property known as *temporal locality*.

(a) What does this term mean? (You can be very brief here.)

(b) Why is this property useful when we design caches and memory hierarchies?