

Computer Organization Test 2

Question 1. (18 points) Translate the following high-level language code segment to MIPS assembly language. Use the registers indicated in the code.

a) for \$4 = 0 to 100 by steps of size 10 do
 if (\$3 < \$4) OR (\$2 >= 50) then
 \$2 = \$2 + \$3
 end if
end for

b) while (\$8 > 20) do
 if (\$8 > 100) AND (\$8 < 200) then
 \$7 = \$8
 \$8 = \$8 - 10
 else
 \$8 = \$8 - \$7
 end if
 \$7 = \$6 + 4
end while

Question 2. (12 points) Suppose you have the following .data area in MIPS assembly language:

```
.data
array: .word 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25
```

For each of the following assembly language segments, what value is loaded into register \$t2?

<pre>a) li \$t0, 5 la \$t1, array mul \$t3, \$t0, 4 add \$t1, \$t1, \$t3 lw \$t2, 0(\$t1)</pre>	<pre>b) la \$t1, array lw \$t2, 8(\$t1)</pre>	<pre>c) li \$t0, 5 la \$t1, array sll \$t3, \$t0, 2 # shift left logical add \$t1, \$t1, \$t3 lw \$t2, 0(\$t1)</pre>
--	--	---

Question 3. (8 points) For the .data area in question 2, complete the translation of the following high-level code segment to MIPS assembly language.

```
for i = 0 to 14 do
    array[i+1] = array[i]
end for

                                la $t1, array
for:
                                li $t0, 0
                                li $t2, 14
for_compare:
                                bgt $t0, $t2, end_for

                                j for_compare
end_for:
```

Question 4. (8 points) Assume that a two-dimensional array A is stored in memory using column-major order, i.e., column 0 is followed by column 1, etc. Write a formula to calculate the address of element A[i][j].

Question 5. Consider the following selection sort subprogram that utilizes a function Max to search for the largest element in the unsorted part of the array.

```

procedure selectionSort(numbers - array of integers, count - integer)
  local integer variables: lastUnsortedIndex, maxIndex, temp

  for lastUnsortedIndex = (count-1) downto 1 do
    maxIndex = Max(numbers, 0, lastUnsortedIndex)
    temp = numbers[lastUnsortedIndex]
    numbers[lastUnsortedIndex] = numbers[maxIndex]
    numbers[maxIndex] = temp
  end for
end selectionSort
    
```

a) (6 points) Using the MIPS register conventions (\$a0-\$a3, \$t0-\$t9, \$s0-\$s7, \$v0-\$v1, \$sp, \$ra), what registers would be used to pass each of the following parameters to selectionSort:

base address of "numbers" array	count

b) (6 points) Using the MIPS register conventions, which of these parameters ("numbers", "count", or both of them) should be moved into \$s-registers?

c) (6 points) Using the MIPS register conventions, what registers should be used for each of the local variables:

lastUnsortedIndex	maxIndex	temp

d) (4 points) In addition to the above registers, the value of "numbers[maxIndex]" will need to be stored into a register. Using the MIPS register conventions, what register should be used to hold this value?

e) (6 points) For the registers indicated above, write the MIPS instructions to set up the selectionSort call-frame on the run-time stack (i.e., move the stack pointer and save registers, etc.).

f) (8 points) For the registers indicated above, write the assemble language code to call the Max function ("maxIndex = Max(numbers, 0, lastUnsortedIndex)"). Include the MIPS instructions to setup the parameters to Max and assigning "maxIndex" the value returned. (You do not need to write the Max function code just the code to call it)

g) (8 points) Using the registers you indicated, write the MIPS assembly language statements to perform the statements:

```
temp = numbers[lastUnsortedIndex]
numbers[lastUnsortedIndex] = numbers[maxIndex]
numbers[maxIndex] = temp
```