Tutorial 1

CSU 215 Computer Organization & Patterson

Winter 2008-2009

Reading. Patterson and Hennessy §2.5 – §2.7

1.1. Construct a control flow graph (like the one shown in Figure 2.11 of the text) for the following section of C or Java code:

```
for (i=0; i<x; i=i+1)
     y = y + i;
```

Implement this C code in MIPS. Assume that i, x, and y correspond to registers $s1, $s2, $s3 respectively.

1.2. MIPS to C. Assume $s3 = i, $s4 = j, $s5 = @A. Below is some MIPS code:

```
Loop: addi    $s4,  $s4, 1          # j = j + 1
      sll     $t1,  $s3, 2          # $t1 = 4 * i
      add     $t1,  $t1, $s5        # $t1 = @ A[i]
      lw      $t0,  0($t1)          # $t0 = A[i]
      addi    $s3,  $s3, 1          # i = i + 1
      slti    $t1,  $t0,   10       # $t1 = $t0 < 10?
      beq     $t1,  $zero, Loop     # goto Loop if >=
      slti    $t1,  $t0, 0          # $t1 = $t0 < 0?
      bne     $t1,  $zero, Loop     # goto Loop if <
```

Below is part of the corresponding C code:

```
do j = j + 1
   while (______);
```

What C code properly fills in the blank in the loop on the right?

```
do j = j + 1
   while (______);
```

What C code properly fills in the blank in the loop on the right?

A: [i++] >= 10?

B: A[i++] >= 10 | A[i] < 0?

C: A[i++] >= 10 & A[i] < 0?

D: A[i++] >= 10 || A[i] < 0?

Assignment 2 Page 2

Computer Science 240

E: A[i++] >= 10 && A[i] < 0?

F: None of the above

1.3. Add comments to the following MIPS code and describe in one sentence what it computes. Assume that $a0 and $a1 are used for the input and both initially contain the integers a and b, respectively. Assume that $v0 is used for the output.

```
        add   $t0,   $zero, $zero
loop:   beq   $a1,   $zero, finish
        add   $t0,   $t0, $a0
        sub   $a1,   $a1, 1
        j     loop
finish: addi  $t0,   $t0, 100
        add   $v0,   $t0, $zero
```

1.4. The following code fragment processes two arrays and produces an important value in register $v0. Assume that each array consists of 2500 words indexed from 0 through 2499, that the base addresses of the arrays are stored in $a0 and $a1, respectively, and their sizes (2500) are stored in $a2 and $a3, respectively. Add comments to the code and describe in one sentence what this code does. Specifically, what will be returned in

$v0?

```
sll      $a2, $a2, 2
         sll $a3, $a3, 2
         add $v0, $zero, $zero
         add $t0, $zero, $zero
outer: add $t4, $a0, $t0
         lw     $t4, 0($t4)
         add $t1, $zero, $zero
inner: add $t3, $a1, $t1
         lw     $t3, 0($t3)
         bne $t3, $t4, skip
         addi   $v0,   $v0,  1
skip       addi   $t1,   $t1,  4
         bne    $t1,   $a3,  inner
         addi   $t0,   $t0,  4
         bne    $t0,   $a2,  outer
```

1.5. Assume that the code from the previous exercise is run on a machine with a 2 GHz clock that requires the following number of cycles for each instruction:

Instruction	Cycles
add, addi, sll	1
lw, bne	2

In the worst case, how many seconds will it take to execute this code?

1.6. The following program tries to copy words from the address in register $a0 to the address in register $a1, counting the number of words copied in register $v0. The program stops copying when it finds a word equal to 0. You do not have to preserve the contents of registers $v1, $a0, $a1. This terminating word should be copied but not counted.

```
addi $v0, $zero, 0                    # Init to avoid counting 0 word
loop: lw, $v1, 0($a0)                  # Read next word from source
      sw     $v1, 0($a1)             # Write to destination
      addi $a0, $a0, 4               # Advance pointer to next source
      addi $a1, $a1, 4               # Advance ptr to next dest
      beq $v1, $zero, loop# Loop if word copied != zero
```

There are multiple bugs in this MIPS program; fix them and turning a bug-free version.

1.7. Write a MIPS procedure to compute the nth Fibonacci number F(n) where

F(n) = 0, if n = 0;
1, if n = 1;
F(n-1) + F(n-2), otherwise.

Base your algorithm on the straightforward but hopelessly inefficient procedure below, which generates a recursive process:

```
int fib(int n){
      if (n == 0)
            return 0;
      else if (n == 1)
            return 1;
      else
            return fib(n-1) + fib(n-2);
```

Be sure to use good programming style (comments, meaningful variable names etc...) At the beginning of your program include a comment stating whether or not your program works. Turn in a copy of your program and, if your program works, a printout of your program’s output for an input of 6.