ECE 2035 Programming HW/SW Systems Fall 2017

3 problems, 6 pages Exam One Solutions 22 September 2017

Problem 1 (4 parts, 30 points) **Loops**

Consider the following C code fragment:

```
int H[100] = {1997, 2, -7, 1, 2010, 4, 3, 6, …, 17};
int i, x, y, z;
for (i = 1; i<100; i = i+4){
  x = H[i];
  y = H[i+1];
  <code block A>
 }
z = i;
```

The body of this fragment contains `<code block A>` to indicate additional instructions not shown. This block uses *`i`*, but does not change the value of *`i`*. It also does not contain *continue* or *break* statements.

Part A (4 points) How many times is `<code block A>` executed? Answer: **25** .

Part B (4 points) What is the minimum value of *i*? Answer: **1**.*(note: i is allocated but not initialized before the loop.)*

Part C (4 points) What is the final value of z? Answer: **101** .

Part D (18 points) Write a MIPS code fragment that is equivalent to the C code above. **Use the following register assignments: $1: i, $2: x, $3: y, $4:z**. Use additional registers if necessary. Use "<code block A>" in your MIPS code to indicate where the instructions for this code block go. *For maximum credit, include comments.* (Note: there are more blank lines provided than you need.)

Label	Instruction	Comment
	`.data`	
`H:`	**`.word 1997, 2, -7, 1, 2010, …, 17`**	**`# int H[100]={1997, 2, …, 17};`**
	`.text`	
	`addi $1, $0, 1`	**`# initialize i=1`**
`Loop:`	**`slti $5, $1, 100`**	**`# is i<100?`**
	`beq  $5, $0, Exit`	**`# if not, exit the loop`**
	`sll  $5, $1, 2`	**`# scale i by 4 for word offset`**
	`lw   $2, H($5)`	**`# x = H[i]`**
	`addi $5, $5, 4`	**`# 4(i+1) for next word offset`**
	`lw   $3, H($5)`	**`# H[i+1]`**
	`<code block A>`	**`# uses i`** ***`(not scaled by 4 or i+1)`***
	`addi $1, $1, 4`	**`# i = i+4`**
	`j    Loop`	**`# loop back`**
`Exit:`	**`addi $4, $1, 0`**	**`# z = i`**

Problem 2 (2 parts, 35 points) **Conditionals: Compound Predicates**

Consider the following C code fragment, where the variables `Hs, He, Ss, and Se` are integers:

```
if ((Se >= Hs) && (He >= Ss))
  {
    <code block A>
  }
else
  {
    <code block B>
  }
<code block C>
```

Part A (15 points) Write the equivalent C fragment using a logical *or* (`||`) instead of logical *and* (&&). Hint: Use DeMorgan's Theorem and swap the then and else clauses. It may be helpful to draw the control flow graph.

```
if ((Se < Hs) || (He < Ss))
  {
    <code block B>
  }
else
  {
    <code block A>
  }
<code block C>
```

Part B (20 points) Turn the C code fragment into the equivalent MIPS code. **The variables are held in these registers: $1: Hs, $2: He, $3: Ss and $4: Se**. Use additional registers if necessary. Use "*<code block A/B/C>*" to indicate where the instructions for these code blocks go. *For maximum credit, include comments and use a minimal number of instructions.* (More blank lines are provided than you need.)

Label	Instruction	Comment
	`slt $5, $4, $1`	`# Se < Hs?`
	`bne $5, $0, DoB`	`# if so, do B`
	`slt $5, $2, $3`	`# He < Ss?`
	`bne $5, $0, DoB`	`# if so, do <code block B>`
	`<code block A>`	
	`j DoC`	`# jump to <code block C>`
`DoB:`	`<code block B>`	
`DoC:`	`<code block C>`	

Problem 3 (4 parts, 35 points) **Understanding Code**

Part A (9 points) What values are in registers $1 and $2 after this MIPS code fragment executes? Express your answers in hexadecimal.

```
lui  $1, 0x1
addi $2, $1, 0xABCD
```

# $1: 0x10000 *(note:0x1 = 0x0001)*	
# $2: 0xABCD	

```
addi sign extends the immediate:
$1: 0x00010000
+    0xFFFFABCD
$2: 0x0000ABCD
```

Part B (9 points) Given the following MIPS code:

```
.data
Input: .word 0xAABBCCDD

.text
       addi $3, $0, Input
```

Write a single MIPS instruction that is equivalent to the original MIPS fragment. Assume *little endian* byte ordering.

Original:	Equivalent MIPS instruction:
addi $4, $0, 8	**lbu $5, 1($3)**
lw $5, 0($3)	
srlv $5, $5, $4	
andi $5, $5, 0xFF	

Part C (8 points) What are the values of the variables x, y, z, and w after the following C code fragment executes? Express your answers in decimal. Hint: remember how C implements compound predicates.

```
int x, y, z;
x = y = z = 33;
int w = 10;
if ((x == 44) || (y = 5) || (z = 8))  // note “==” vs “=”
  w = 77;
```

Variable:	Value:
x	**33**
y	**5**
z	**33**
w	**77**

Part D (9 points) What does the following code fragment print?

```
int V[] = {1, 5, 7, 6, -9, 17, -20, 0, -3};
int j, i=0;
while(V[i] != 0)
  {
    printf(“V[%d]: %d\n”, i,V[i]);
    if (V[i] < 0)
      {
        for (j=0; j<i; j++)
          {
            if (j == 2)
              continue;
            printf(“j: %d\n”, j);
          }
        break;
      }
    i++;
  }
```

```
V[0]: 1
V[1]: 5
V[2]: 7
V[3]: 6
V[4]: -9
j: 0
j: 1
j: 3
```