

EE 2310 Homework #5 Solutions – MIPS Instructions and Simple Programming

Below are snapshots of MIPS registers, a data declaration, and a SPIM memory display from the data section. Answer the following questions with respect to the displays shown.

NOTE: Changes to a register or memory location in one problem do not carry to any other problem.

<code>.data</code>
<code>n: .word</code>
<code>p: .word</code>
<code>q: .word</code>
<code>r: .word</code>
<code>s: .word</code>
<code>t: .word</code>
<code>u: .word</code>
<code>v: .word</code>
<code>w: .word</code>
<code>x: .word</code>
<code>y: .word</code>
<code>z: .word</code>
<code>str:</code>
<code>.asciiz</code>
<code>"hello</code>
<code>world\n"</code>

MIPS Registers

R0 (r0): 0x00000000	R8 (t0): 0x0f0f0f0f	R16 (s0): 0x00000000	R24 (t8): 0x00000000
R1 (at): 0x10010000	R9 (t1): 0x0000ffff	R17 (s1): 0x00000000	R25 (t9): 0x00000000
R2 (v0): 0x0000000b	R10 (t2): 0x00000000	R18 (s2): 0x00000058	R26 (k0): 0x00000000
R3 (v1): 0x00000000	R11 (t3): 0x10010020	R19 (s3): 0x00000000	R27 (k1): 0x00000000
R4 (a0): 0x00000058	R12 (t4): 0x100100f0	R20 (s4): 0x00400020	R28 (gp): 0x10008000
R5 (a1): 0x10010010	R13 (t5): 0x10010030	R21 (s5): 0x00000000	R29 (sp): 0x7ffffeff0
R6 (a2): 0x0000000c	R14 (t6): 0x80000080	R22 (s6): 0x800c1001	R30 (s8): 0x00000000
R7 (a3): 0x00000010	R15 (t7): 0xffff0000	R23 (s7): 0x00000050	R31 (ra): 0x00400070

Data

`[0x10000000]...[0x1000fffc] 0x00000000`

<code>[0x10010000]</code>	<code>0x5350494d</code>	<code>0x20697320</code>	<code>0x61207573</code>	<code>0x6566756c</code>
<code>[0x10010010]</code>	<code>0x2073696d</code>	<code>0x756c6174</code>	<code>0x6f722066</code>	<code>0x6f72206c</code>
<code>[0x10010020]</code>	<code>0x6561726e</code>	<code>0x696e6720</code>	<code>0x8d495053</code>	<code>0x20617373</code>
<code>[0x10010030]</code>	<code>0x68656c6c</code>	<code>0x6f20776f</code>	<code>0x726c640a</code>	<code>0x00000000</code>
<code>[0x10010040]</code>	<code>0x726f6772</code>	<code>0x616d6d69</code>	<code>0x6e672061</code>	<code>0x6e642063</code>
<code>[0x10010050]</code>	<code>0x6f6d7075</code>	<code>0x74657220</code>	<code>0x61726368</code>	<code>0x69746563</code>

`[0x10010060]...[0x10020000] 0x00000000`

1. After: `add $s3, $a1, $a3`, what are the contents of \$s3? 0x 1001 0020
2. After: `and $t2, $t1, $t0`, what are the contents of \$t2? 0x0000 0f0f
3. After: `or $s8, $t6, $t4`, what are the contents of \$s8? 0x 9001 00f0
4. After: `sub $t2, $t5, $t3`, what are the contents of \$t2? 0x 0000 0010
5. After: `move $t2, $sp`, what are the contents of \$t2? 0x7fff eff0

6. After: `mul $t2, $a2, $a3`, what are the contents of \$t2? 0x 0000 00c0
7. After: `not $t2, $a3`, what are the contents of \$t2? 0x ffff ffef
8. After: `addi $t2, $a2, 12` what are the contents of \$t2? 0x 0000 0018
9. After: `neg $t2, $a3`, what are the contents of \$t2? 0x ffff fff0
10. After `lw $t1, v`, what are the contents of \$t1? 0x 6f72 206c
11. After `lw $t0, 20($t5)`, what are the contents of \$t0? 0x 616d 6d69
12. After `lb $t0, 10 ($a1)` , what are the contents of \$t0? 0x 0000 0020
13. After `srl, $t0, $sp, 16`, what are the contents of \$t0? 0x 0000 7fff
14. After `sra, $t0, $t6, 16`, what are the contents of \$t0? 0x ffff 8000
15. After `rol, $t0, $t7, 8`, what are the contents of \$t0? 0x ff00 00ff
16. After `ror, $t0, $a1, 8`, what are the contents of \$t0? 0x 1010 0100
17. After `sll, $t0, $ra, 24`, what are the contents of \$t0? 0x 7000 0000
18. After `ble $t2, $t1, kloop`, is the branch to “kloop” taken? YES.
19. After `bgtz $s6, calc`, is the branch to “calc” taken? NO
20. After `slt $t0, $t3, $t4; beqz $t0, loop`, is the branch to “loop” taken? NO.

21. The program at the right does a mathematical and logical calculation. Use the data declaration shown on page 1 of this homework. Answer the following questions:

21.1. Where is the result stored?

\$t2

21.2. What is the result (state in hex numbers)?

0x18

```
.text
main: lw $t0,n
      andi $t0,$t0,0xffff
      srl $t0,$t0,12
      sll $t0,$t0,1
      add $t2,$t0,0x10
      li $v0,10
      syscall
```

22. Write a program to do the following (again, no data declaration is necessary): Load the data word `v` into `$a0`. Do a left rotate six places, then test the word to see if it is negative. If it is, output it to the console and do a syscall 10. If positive or zero, simply stop the program with a syscall 10.

```
.text
main: lw $a0, v
      rol $a0,$a0,6
      bgez $a0, over
      li $v0, 1
      syscall
over: li $v0,10
      syscall
```

23. Using the data declaration on page 1, write a program to do the following: Load the word “y” into register `$t0`. Test to see if the number is negative. If it is, output it to the console. If it is positive, take the 2’s complement and then output it to the console. If it is zero, simply end the program. Output a CR/LF after the word, if output, and then end the program.

23.1. Is a number output?

Yes.

23.2. Was the number originally positive or negative?

Negative.

```
.text
main: lw $t0, y
      beqz $t0,done
      bltz $t0,prt
      neg $t0,$t0
prt:  move $a0,$t0
      li $v0, 1
      syscall
      li $a0,0x0a
      li $v0,11
      syscall
done: li $v0,10
      syscall
```