```
 1 # FILENAME: scpDemos.s  BY: durant@msoe.edu      BEGAN: 15 April 2009
 2 # $Id: scpDemos.s,v 1.3 2010/05/09 14:08:01 durant Exp durant $
 3 # PROVIDES:  CE2930 test programs for the single-cycle processor
 4
 5 # implemented instructions, arithmetic/logic,
 6 # R-format: add  addu  and  nor or  slt  sltu  sll srl sub subu
 7 # I-format: addi addiu andi     ori slti sltiu
 8 # branch,   I-format: beq bne
 9 # transfer, I-format: lw sw
10
11 # demo procedure
12 # 1. get your assigned program number
13 # 2. assemble and test in a simulator (use MARS for correct branch offsets)
14 # 3. if bugs exist, correct and document on cover sheet, o.w. indicate no bugs exist
15 # 4. simulate on your processor in Quartus, including internal signals as needed
16 # 5. identify key results in the simulation and show to professor
17
18 # scoring if your demo doesn't work
19 # you can demo a program of your choice but not the given one: -20% on demo
20 # you can't demo on the due date: -10% first day, -5% each additional weekday
21
22         .text   # Executable code section
23 main:          # User program entry point
24
25         # Sum natural numbers from 10 down - LW5 example
26 P1:     addi    $t0,$zero,10    # t0 = 10        0x2008 000a
27         add     $t1,$zero,$zero # t1 = 0         0x0000 4820
28 L1_1:   add     $t1,$t1,$t0     # t1 += t0       0x0128 4820
29         addi    $t0,$t0,-1      # --t0           0x2108 ffff
30         bne     $t0,$zero,L1_1  # while(t0 != 0) 0x1500 fffd
31 P1_end: beq     $zero,$zero,P1_end #            0x1000 ffff
32
33         # Load/store and memory offsets
34 P2:     addi    $t4,$zero,5     # t4 = 5         0x200c 0005
35         addi    $t8,$zero,8     # t8 = 8         0x2018 0008
36         sw      $t4,0($t8)      #                0xaf0c 0000
37         addi    $t4,$t4,6       # t4 += 6, = 11  0x218c 0006
38         sw      $t4,-4($t8)     # M[4] = 11      0xaf0c fffc
39         addi    $t8,$t8,-8      # t8 = 0         0x2318 fff8
40         lw      $t6,4($t8)      # t6 = M[4] = 11 0x8f0e 0004
41 P2_end: beq     $zero,$zero,P2_end #            0x1000 ffff
42
43         # slt/sltu - okay if you have not fully implemented S/U behavior,
44         # but be sure you can EXPLAIN your result is correct for a full implementation.
45 P3:     addi    $t0,$zero,3     # t0 = 3                  0x2008 0003
46         sub     $t1,$zero,$t0   # t1 = -3                 0x0008 4822
47         sltu    $t2,$t1,$t0     # U, t1-t0=big- 3 =big, F 0x0128 502b
48         slt     $t3,$t1,$t0     # S, t1-t0=-3 - 3 =-6 , T 0x0128 582a
49         slt     $t4,$t0,$t1     # S, t0-t1= 3 - -3= 6 , F 0x0109 602a
50 P3_end: beq     $t0,$t0,P3_end  #                        0x1108 ffff
51
52         # logic
53 P4:     ori     $t0,$zero,0xA5C3        # t0=1010 0101 1100 0011        0x3408 a5c3
54         ori     $t1,$zero,0xC3A5        # t1=1100 0011 1010 0101        0x3409 c3a5
55         nor     $t2,$t0,$t1             # t2=0001 1000 0001 1000=0x1818 0x0109 5027
56         sll     $t2,$t2,2               # t2=0110 0000 0110 0000=0x6060 0x000a 5080
57         and     $t2,$t1,$t2             # t2=0100 0000 0010 0000=0x4020 0x012a 5024
58 P4_end: beq     $zero,$zero,P4_end      #                               0x1000 ffff
59 # END OF PROGRAM
```