CS 2734, Org II, Spring 2003, First Exam, Answers

```
========================================================================
1.  (a) 76 (base 10) = 1001100 (base 2)
       -46 (base 10 = ~1001100 + 1 = 1111 1111 1011 0011 + 1 =
                                     1111 1111 1011 0100

========================================================================
2.
    # CS 2734, Computer Organization II, Spring 2003
    # MIPS program giving answer to Exam1, question 2
            .globl main
    main:   addu    $s7, $zero, $ra
            .data
    A:      .word 47, 23, 89, 52, 43, 0, 0, 0, 0, 0
            .text
    ########## Start of answer to Question 2 ################
            la      $s0, A          # address of A
            addi    $t0, $0, 0      # loop counter, start at 0
            addi    $t1, $zero, 5   # to terminate loop
            add     $t2, $0, $s0    # address of item in A
    Loop:   lw      $a0, 0($t2)     # load current A value into $a0
            li      $v0, 1          # print this value
            syscall
            li      $v0, 4          # print a blank
            la      $a0, Blank
            syscall
            addi    $t0, $t0, 1     # increment loop counter
            addi    $t2, $t2, 4     # increment pointer into A
            bne     $t0, $t1, Loop  # branch back to form loop
    ########## End of answer to Question 2 ##################
    ########## Finish main###################
            addu    $ra, $zero, $s7
            jr      $ra
    ########## write an array ###############
            .data
    Blank:  .asciiz " "
    ############### output #############################
    # ten42% spim -file exam1.s
    # 47 23 89 52 43 ten42%
    ####################################################
```

Alternatively, the following code uses the mul pseudo-instr:

```
    # CS 2734, Computer Organization II, Spring 2003
    # MIPS program giving answer to Exam1, question 2
            .globl main
    main:   addu    $s7, $zero, $ra
            .data
    A:      .word 47, 23, 89, 52, 43, 0, 0, 0, 0, 0
            .text
    ########## Start of answer to Question 2 ################
            la      $s0, A          # address of A
            addi    $t0, $0, 0      # loop counter, start at 0
            addi    $t1, $zero, 5   # to terminate loop
    Loop:   mul     $t2, $t0, 4     # pseudo-instr, mult $t0 by 4
            add     $t3, $t2, $s0   # add to start addr of A
            lw      $a0, 0($t3)     # load current A value into $a0
            li      $v0, 1          # print this value
            syscall
            li      $v0, 4          # print a blank
            la      $a0, Blank
            syscall
            addi    $t0, $t0, 1     # increment loop counter
            addi    $t2, $t2, 4     # increment pointer into A
            bne     $t0, $t1, Loop  # branch back to form loop
    ########## End of answer to Question 2 ##################
    ########## Finish main###################
            addu    $ra, $zero, $s7
            jr      $ra
    ########## write an array ###############
            .data
    Blank:  .asciiz " "
    ############### output #############################
    # ten42% spim -file exam1.s
    # 47 23 89 52 43 ten42%
    ####################################################
    ============================================
```

Finally, one could use the fact that there is a zero at the end:

```
    # CS 2734, Computer Organization II, Spring 2003
    # MIPS program giving answer to Exam1, question 2
            .globl main
    main:   addu    $s7, $zero, $ra
            .data
    A:      .word 47, 23, 89, 52, 43, 0, 0, 0, 0, 0
            .text
    ########## Start of answer to Question 2 ################
            la      $s0, A          # address of A
    Loop:   lw      $a0, 0($s0)     # load current A value into $a0
            beq     $a0, $0, Done   # quit when you get to zero
            li      $v0, 1          # print this value
            syscall
            li      $v0, 4          # print a blank
            la      $a0, Blank
            syscall
            addi    $s0, $s0, 4     # increment pointer into A
            b       Loop            # branch back to form loop
    Done:
    ########## End of answer to Question 2 ##################
    ########## Finish main###################
            addu    $ra, $zero, $s7
            jr      $ra
    ########## write an array ###############
            .data
    Blank:  .asciiz " "
    ############### output #############################
    # ten42% spim -file exam1.s
    # 47 23 89 52 43 ten42%
    ####################################################

========================================================================
3.
     # CS 2734, Computer Organization II, Spring 2003
     # MIPS program giving answer to Exam1, question 3
            .globl main
     main:  addu    $s7, $0,  $ra
     #### call to F#######################################
            addi    $a0, $0,  19     # 19 is input param
            jal     F               # call F
     #### end of code for call to F ####################
     #### must now finish program and return from main
     #### first print $v0 to show that code worked
            add     $a0, $v0, $0    # must move $v0 to $a0
            li      $v0, 1          # print an int
            syscall
     #### Finish main
            addu    $ra, $0,  $s7
            jr      $ra

     #### Code for the function F ######################
     F:     addi    $sp, $sp, -4    # make room on stack
```

```
            sw      $ra, 0($sp)     # save $ra on stack
            add     $v0, $a0, $a0   # double input parameter, return value
            lw      $ra, 0($sp)     # restore $ra from stack
            addi    $sp, $sp, 4     # restore stack
            jr      $ra             # return from call to F
     #### End of code for F #############################
     #### output ####
     # ten42% spim -file exam1_3.s
     # 38ten42%
     ################
```

==

4. Use

```
    (a)  add    $s1, $s2,   $0    or  addi $s1, $s2, 0
    (b)  addi   $s3, $0,   200
    (c)  beq    $0,  $0,  Loop
```

==

5.

```
# CS 2734, Computer Organization II, Spring 2003
       .globl main
main:
       addu   $s7, $zero, $ra

### First way uses addi and requires 0 for sign bit
       lui    $s3, 0x0344
       addi   $s3, $s3, 0x07ff
## output answer: 0x034407ff or 54790143
       li     $v0, 1
       move   $a0, $s3
       syscall
       jal    Newl
### Second way uses ori and and always works
       lui    $s3, 0x0344
       ori    $s3, $s3, 0x07ff
## output answer: 0x034407ff or 54790143
       li     $v0, 1
       move   $a0, $s3
       syscall
       jal    Newl
### Third method without lui:
       li     $s3, 0x0344
       sll    $s3, $s3, 16
       ori    $s3, $s3, 0x07ff
## output answer: 0x034407ff or 54790143
       li     $v0, 1
       move   $a0, $s3
       syscall
       jal    Newl
## final part of main
       addu    $ra, $0,  $s7
       jr      $ra
## Newl function, print a newline
Newl:  li     $v0, 4
       la     $a0, Newline
       syscall
       jr     $ra
       .data
Newline: .asciiz "\n"
#############################################
# Output:
# 54790143
# 54790143
# 54790143
```

==

6. With the given inputs, the upper two transistors are open (don’t conduct current), while the bottom two are closed. Thus C is connected to the ground at the bottom and not to any power, so C’s output is 0.

If either input is 0, C is 1, while both inputs 1 makes C a 0. Thus C is a NAND gate.