CS 2734, Computer Organization II
Spring Semester, 2002
First Examination

1. Below are questions about number representations and conversions: **(25)**

   (a) Convert the (decimal) number -46 to 16-bit two's complement binary. (The binary representation for 46 is 101110.)

   (b) Consider the floating point number (a `double`) with representations:

   ```
   1011 1111 1101 1100  (48 more 0's)  (binary)
   b    f    d    c     (12 more 0's)  (hex)
   ```

   i. What is the sign of this number?

   ii. What is its exponent (power of 2)? (Remember that the bias for a `double` is 1023, and that an exponent of 1 is represented by `100 0000 0000`.)

   iii. What is the significant part?

   iv. Put i, ii, and iii together to get the number.

---

2. Consider the following MIPS code fragment: **(25)**

```
        .data
# stored in A are squares of first 7 primes
A:      .word  4, 9, 25, 49, 121, 169, 289
        .text
# insert MIPS instructions here.
```

For insertion at the comment, write MIPS instructions that will do the following:

   (a) Put the starting address of `A` into register `$s1`.

   (b) Use a loop to add the values of the array `A` and to leave the result in register `$s2`. [You must use a loop for this.]

   (c) Print the resulting sum, using `syscall`. [Recall that `syscall` requires `$v0` equal to 1 to print the value in `$a0`.]

   Your MIPS code should do what is asked for above and *nothing more*.

---

3. Write MIPS functions `F` and `G` so that **(25)**

   (a) `F` saves registers `$a0`, `$a1`, and `$ra` on the stack.

   (b) `F` calls a function `G` with input parameters 14 and 47, following MIPS conventions for input parameters.

(c) `G` adds its parameters and returns the sum, following MIPS conventions for returning a value.

(d) `F` takes the value that `G` returns, adds 1 to this value, and returns the new quantity.

(e) `F` restores the three registers saved above.

(f) Return from `F`.

Note: You should not give code for the call to `F`. Just give the definition of the function `F` and the function `G` that do the above items and *nothing more*.

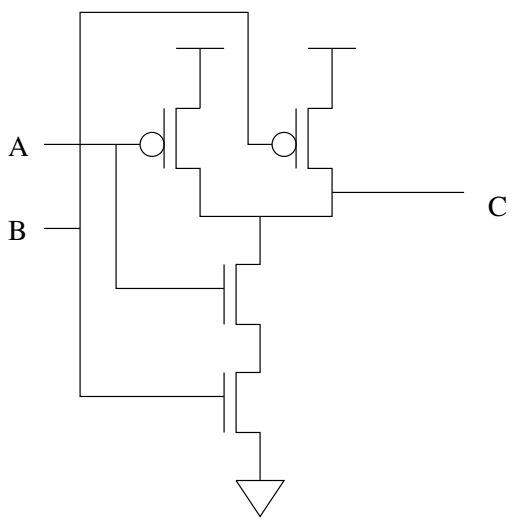
---

4. The MIPS assembler can use actual machine instruction to create other *pseudo-instructions*. In each case below, show how the given instruction could be rendered using an actual instruction (examples of actual instructions include `add`, `addi`, `slt`, `beq`, and `bne`): **(10)**

```
(a)   move $s1, $s2          # $s1 = $s2
(b)   li   $s3, 200          # $s3 = 200
(c)   b    $s1, $t1, Loop    # unconditional branch
```

---

5. Consider the following logic gate constructed out of CMOS transistors. **(15)**

(a) In case A is a 1 (voltage high) and B is also a 1, what will be the output at C? Explain your answer in terms of the diagram and the properties of the transistors. (Show which switches are open (don't conduct current), which are closed (conduct current), and explain why the output at C is what it is.)

(b) Say what kind of logic gate this represents.