Sample Exam I  
PAC II

Please answer questions 1 and 2 on this paper and put all other answers in the blue book.

1. True/False. Please circle the correct response.
   a. T F In the C and assembly calling convention that we used for this class, arguments to a function call are pushed in reverse (i.e. right-to-left) order.
   b. T F In C, for the expression \((x \mid \text{THE\_MASK})\), where \text{THE\_MASK} has at least one bit that is not zero, the result will be zero if all the bits of \(x\) are zero.
   c. T F In x86 assembly, the instruction \texttt{mov eax, [ebp+12]} (or \texttt{movl 12(%ebp), %eax} in AT&T syntax) will add 12 to the value contained in the ebp register in order to compute a memory address.
   d. T F In x86 assembly on a 32-bit machine, the instruction \texttt{pop eax} adds 4 to the value of the \texttt{esp} register.
   e. T F In assembly, performing a right arithmetic shift operation, SAR, on a negative number results in a negative number.
   f. T F Any circuit constructed using only AND, OR, or NOT gates can also be constructed using only NAND gates (which perform an AND and then a NOT).
   g. T F Branch prediction is used to predict which way a conditional branch (jump) instruction will go in order to avoid pipeline stalls.
   h. T F A multiplexer uses \(2^N\) select lines to select from \(N\) input lines.
   i. T F A 32-bit adder can be used for subtraction if each bit of the second operand is first sent through a (32-bit) NOT gate and the carry-in to the adder is set to 1.
   j. T F In an unclocked latch, setting both the S and the R inputs to 0 will cause the output Q to retain its current value.

2. 
   a. Write the number AB31 hex in binary: ________________________________.
   b. Write the number 73 decimal in hex: ______________ and in binary: ____________.
   c. \log_{16} G = ____________.
   d. In order to access all bytes in a 64MB memory, an address must have at least _____ bits.

3. Write a C procedure, \textbf{int foo(int x)}, that returns the index of the most significant bit of \(x\) whose value is 1. For example, if bit 15 of \(x\) is the most significant bit of \(x\) whose value is 1, then \(\text{foo}\) should return 15. If no bits are 1, then \(\text{foo}\) should return -1.

(Please turn page over)
4. Write an x86 assembly procedure `bar` that takes two parameters, an integer `n` and a pointer `p` to an integer array (i.e. `p` is the address of the start of the array), and adds up the first `n` integers of the array and returns the result.

5. a. Write in C a recursive version of the bubble sort procedure (hint: it should take as parameters an array `a[]` and an integer size that gives the size of the portion of the array that is currently unsorted).

   b. Write the recurrence relation that gives the running time, `T(n)`, of the bubble sort algorithm.

   c. Solve the recurrence relation in order to determine the asymptotic complexity, expressed in “big-Oh” notation, for bubble sort.

6. a. Build, from AND, OR, and NOT gates, a circuit that represents the two-bit “less-than” function. That is, it has two two-bit inputs, A and B, and a single one-bit output, R, such that R is true when A < B. [Hint: enumerate the possible inputs for which the output is true.]

   b. As you saw in class, a clocked latch is built from an unclocked latch as shown below. Why are flip-flops used for storing bits in a CPU rather than clocked latches?

   ![Unclocked Latch Diagram]

   c. Build from gates, multiplexers, decoders, and/or adders (including 32-bit versions of each) a circuit that takes two 32-bit inputs, X and Y, and outputs the value of the larger of X and Y. That is, it computes the equivalent of the following C code:

   \[
   \text{output} = \text{X}>\text{Y} \ ? \ \text{X} \ : \ \text{Y};
   \]

7. As you probably remember from the project, the MIPS jalr instruction (jump-and-link, MIPS' version of the call instruction),

   \[
   \text{jalr} \ \$rs
   \]

   has the following effect:

   \[
   \text{register[31]} = \text{pc} + 4
   \]

   Build the data path for this instruction. Assume that the MIPS register file, the PC register, and any adder or ALU you want already exists (i.e. you don't need to build any of these, you can use them). Show all the wires that are involved in executing the instruction and be sure to the label the values that are carried on any wires that you draw.