Indian Institute of Technology Kharagpur

AUTUMN Semester 2021
COMPUTER SCIENCE AND ENGINEERING

Computer Organization and Architecture

Students: 133 Date: 04 October 2021
Total points: 80 Time: **Opens** 11:55 AM, 04-10-2021
Credit: 30% **Closes** 1:55 PM, 04-10-2021

INSTRUCTIONS: This is an OPEN-BOOK, OPEN-NOTES test. You may use calculators if required. This question paper has three pages. **Answer all questions**.

Submission of answers: The questions are such that they require either numerical answers or short answers. Please write ONLY THE ANSWERS on a sheet of paper, convert it to **pdf** directly (or by scanning), and submit it on Moodle by **2:10 pm** on Monday, 04 October 2021.

DO NOT FORGET TO WRITE YOUR **NAME** AND **ROLL NUMBER** AT THE TOP OF YOUR ANSWER SHEET.

---

1. (**5 points**) Let $t1, $t2 hold two 32-bit integers in two's complement. We want to determine whether a *final* carry (emitted out of the 32nd bit, which is usually discarded) is being generated after addition of these two integers. A register ($t3) should be set to 0 if the final carry bit is 0, else it should be set to 1. Overflow, if any, should be ignored. Write a MIPS code to accomplish this, with *justification why it works correctly*. This can be done using only two lines of codes! (2 + 3)

2 (**10 points**). We have a CPU with an 8-bit ALU equipped with an 8-bit integer adder, registers, shift-registers, and other basic control blocks but hardware circuits for multiplier, divider, or for performing bit-wise logical operations (AND, NOT, etc.), or floating-point operations, are **not available**. We need to compute $X = \lfloor((34 \times A) + B)/7\rfloor$ where A, B, X are all integers expressed in *unsigned* 8-bit numbers. In particular, assume that A = 0000 0011, and B = 0001 1001. Show a procedure that computes X with the exact desired result on this CPU while **minimizing** the number of basic operations for these special values of A and B. Report the number of operations that you have used, as well.

3 (**10 points**). Consider that two numbers, M and N represented in 2's complement notation, where M = 0xABCDEF76 and N = 0x543765DF. We need to multiply them using the Booth's Multiplication Algorithm, using *fewest* addition/subtraction operations.
(a) State, with justification, how many times addition, subtraction, shift, and bit-pair test operations (whether two consecutive bit-pairs are 00, 01, 10 or 11) need to be performed in order to complete the task.

(b) In the multiplier circuit, the delays of hardware modules are as follows: Adder/subtractor: 0.25 *ns*; Combinational Shifter: 0.05 *ns*; Bit-pair testing: 0.10 *ns*. Ignore initialization time required for loading M and N.

(i) In one implementation of the multiplier, each of these operations needs one clock cycle. What is the CPU-time needed to finish the above multiplication?
(ii) To speed up the operation of the multiplier, in another implementation each operation is allowed to be completed in multiple clock-cycles. How much CPU-time is needed to finish the above multiplication for this case? Your approach should attempt to minimize CPU-time.
(6 + (2 + 2))

4. (**10 points**) Consider the following two normalized floating-point (FP) numbers A and B in IEEE 754 single-precision format, which we want to add:

A: 0 1111 1110 1111 0000 0000 0000 0000 001

B: 0 1111 1011 1100 0000 0000 0000 0001 001

(a) What will be the values of guard, round, and sticky bits?
(b) What will be the result of (A + B) in 32-bit IEEE 754 format? Show your work and justify your answers. (3 + (4 + 3))

5. (**10 points**) Let N be a normalized FP-number in 32-bit IEEE 754 format. Both the exponent field (E) and mantissa (M) are extracted and stored in two 32-bit registers $t1 and $t2, respectively, in right-justified fashion, i.e., with all blank bits on the left filled with 0's. Sketch an algorithm in the light of MIPS which can check whether or not N is exactly an integer expressible as 32-bit 2-complement number. You do not need to write any MIPS code, just describe what conditions need to be checked so that they can be implemented using MIPS instructions.

6. (**10 points**) Consider a special floating-point format, which is an 8-bit normalized format, with 1 sign bit, 4 exponent bits, and 3 mantissa bits. Except for being only 8-bit wide, this scheme is analogous to the standard IEEE-754 32-bit or 64-bit format in terms of the meaning of fields, bias, and special encodings.
(a) Represent $(0.00110111)_2$ in this 8-bit special format with the nearest-even rounding.
(b) Represent $(16.0)_{10}$ in this 8-bit special format. (5 + 5)

7. (**5 points**)
(a) What is the maximum difference between two consecutive normalized FP-numbers (ignoring $\pm \infty$) that are representable in 32-bit IEEE 754 format?
(b) A and B are two 32-bit FP-number, where A(B) is a normalized (denormal) number. Given A = 0 1111 1110 1111 0000 0000 0000 0000 001, find B such that (A – B) is maximum. Write your result in 32-bit IEEE 754 FP-format.
(c) In FP-arithmetic, while performing addition, the normalization step may require either left or right shift of the sum of two significands. On the other hand, during multiplication, the normalization step may require *only right shift* of the product of two significands. Justify the reason. (2 + 2 + 1)

8. (**10 points**)
Let M denote hardware realization of a 32-bit array multiplier for integers operands. M uses AND gates for computing partial products and 1-bit FA-blocks for additions. The architecture deploys a *standard* carry-save-adder for adding partial products (not Wallace-tree type). The last row of adders is implemented with a 4-stage carry-select-adder, each stage comprising a 4-bit ripple carry adder (RCA). Assume the delay of each 1-bit FA block is 1 *ns*. Ignore delays of all AND-gates MUX-es, other logic, if any, and interconnecting wires.

(a) What is the worst-case delay of M?

(b) How many 1-bit FA-blocks do you need? (5 + 5)

9. (**10 points**)

(a) Two (nk)-bit numbers $A = a_{nk}\, a_{nk-1}\ldots a_1$ and $B = b_{nk}\, b_{nk-1}\ldots b_1$ are being added using the following scheme: The bits are partitioned into n groups, each group consisting of k bits. For each group of k bits, a PPA (Brent-Kung Parallel Prefix Adder) is employed to compute the sum. These PPA's are then serially cascaded as in ripple-carry adders. Estimate the cost and delay of the proposed adder in terms of n and k.

(b) We want to multiply two integers A = 57 and B = 58, using Karatsuba multiplication algorithm (KMA). In *decimal space*, show the steps that lead to the correct result for (A × B) with reduced computational effort. Go down only up to *one level* of recursion for illustration. In general, what is the complexity of KMA for multiplying two n-bit binary integers?

((2 + 3) + (3 + 2))