Indian Institute of Technology, Delhi

FALL, 2013

CSL 211: COMPUTER ARCHITECTURE

Minor 2

One and Half Hours

NOTE:
- All answers need to be brief and to the point.
- Please make any assumptions that you deem to be reasonable.
- We need exquisite detail, and clarity in every answer.
- You need to write your answer in the answer box (Ans:). We will **NOT GRADE** your answer, if it is not written in the box.
- Every answer needs to be written neatly and cleanly in the space provided for it.
- Use proper handwriting, and do not write anything in the margins.
- Note that in most questions, there is no part marking.
- You can use rough sheets. Do not attach them with this paper.
- You are not allowed to carry any electronic gadget including calculators.
- This question paper **NEEDS TO BE SUBMITTED**. Do not take it with you.

Total Marks: 50
Total Number of Pages : 8

Name:	Group No:	Entry No:

Marks:

1	2	3	4	5	6	7	8	Total

Easy

1. ***NO PART MARKING***

Answer the following: (15 marks)

i) Which algorithm is faster, restoring or non-restoring? $\boxed{\text{Ans: non-restoring}}$

ii) What is the time complexity of floating point division?
$\boxed{\text{Ans: log(n), or log(n)} \times \text{log(p) (p is the precision), or } log(n)^2}$

iii) The four IEEE 754 rounding modes are round to $+\infty$, $-\infty$, to 0, and round to $\boxed{\text{Ans: nearest/even}}$.

iv) How many stages does a *SimpleRisc* processor have? $\boxed{\text{Ans: 5}}$

v) The store instruction saves the value to be stored in the $\boxed{\text{Ans: rd}}$ field of the instruction packet.

vi) A pipeline latch is triggered on the $\boxed{\text{Ans: negative edge}}$ of a clock.

vii) The $flags$ register is updated by the $\boxed{\text{Ans: cmp}}$ instruction.

viii) Is it required to have a forwarding path from the EX to OF stage? (Yes/No) $\boxed{\text{Ans: No}}$

ix) We need to insert 3 bubbles for a taken branch. (Yes/ No) $\boxed{\text{Ans: No}}$

x) How many microinstructions did we define in class? $\boxed{\text{Ans: 8}}$

xi) List the registers that are exposed by the memory unit to the shared bus in a microprogrammed processor. $\boxed{\text{Ans: } \textit{mar}, \textit{mdr}, \textit{ldResult}}$

xii) Which micro-instruction implements a jump to the corresponding index in the microprogram memory for a given program instruction? $\boxed{\text{Ans: } \textit{mswitch}}$

xiii) Performance is completely dependent on the frequency of a processor. (true/false) $\boxed{\text{Ans: false}}$

xiv) What are the two factors that determine the frequency of a processor (options: compiler, technology, architecture). $\boxed{\text{Ans: technology, architecture}}$

xv) Does compiler technology have an effect on IPC (yes/ no). $\boxed{\text{Ans: yes}}$

Medium

2. Implement the *load* instruction in micro-assembly. Do not specify the preamble. (5 marks)

ld instruction

```
/* transfer rs1 to register A */
mmov regSrc, rs1, <read>
mmov A, regVal

/* calculate the effective address */
mmov B, immx, <add> /* ALU operation */

/* perform the load */
mmov mar, aluResult, <load>

/* write the loaded value to the register file */
mmov regData, ldResult
mmov regSrc, rd, <write>

/* jump to the beginning */
mb .begin
```

3. Answer the following questions. (6 marks)

i) What are the six possible forwarding paths in our *SimpleRisc* processor? (2 marks)
$EX \rightarrow OF$, $MA \rightarrow EX$, $RW \rightarrow MA$, $RW \rightarrow EX$, $MA \rightarrow OF$, $RW \rightarrow OF$

ii) Which four forwarding paths, are required, and why? (Give examples to support your answer). (4 marks)

Path	Reason
$MA \rightarrow EX$	Forwarding across consecutive instructions with a RAW dependence. The first instruction cannot be a load instruction. add r1, r2, r3 sub r4, r1, r2
$RW \rightarrow MA$	Fowarding from a load to a store instruction ld r1, 10[r4] st r1, 20[r3]
$RW \rightarrow EX$	Forwarding across two stages. ld r1, 10 [r4] add r8, r9, r10 add r4, r1, r2
$RW \rightarrow OF$	Forwarding across three stages. ld r1, 10 [r4] add r8, r9, r10 add r6, r9, r10 add r4, r1, r2

Note to TAs 1 mark per case. Just one example, or a little bit of reasoning is sufficient. Award 0 marks to subparts that give wrong examples.

4. Write pseudo-code for detecting and handling the branch-lock condition? (without delayed branches) (4 marks)

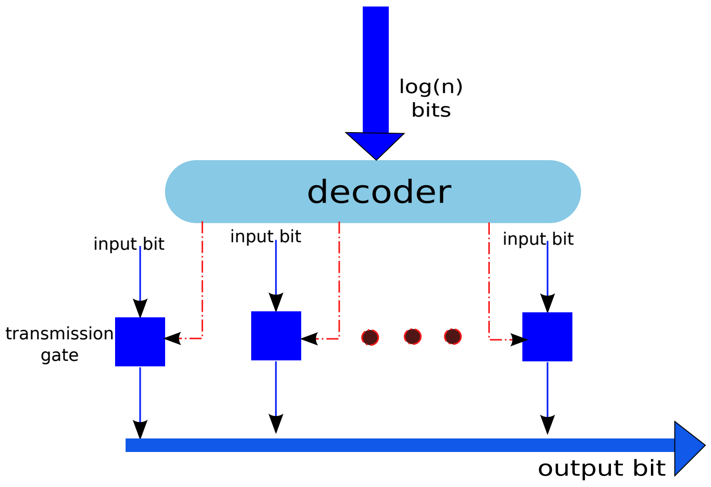
If the instruction in the EX stage is a taken branch, convert the instructions in the IF and OF stage to bubbles. We find out if a branch is taken by inspecting the *isBranchTaken* signal.

Note to TAs This is a very easy question. We do not need more than this basic insight.

Hard

5. How would you implement a multiplexer with transmission gates? (show a circuit diagram, and explain why your idea will work) (3 marks)

A multiplexer takes n input bits, and has $log(n)$ select bits. We first use a decoder to expand the set of $log(n)$ select bits to n control bits. Each control bit controls a transmission gate.



Note to TAs Use of the decoder (1 mark), Insight: Connecting all the transmission gates in a wired OR configuration (1 marks), final solution (1 mark)

6. Assume that we have an instruction immediately after a call instruction that reads ra. We claim that this instruction will get the correct value of ra in a pipeline with forwarding. Is this true? Prove your answer. (3 marks)

A *call* instruction is a taken branch (1 mark), and thus the two slots after it will contain bubbles (1 mark). $RW \rightarrow OF$ will ensure that the ra register gets the right value (1 mark).

7. Answer the following:

NO PART MARKING

```
add r1, r2, r3      1        1         1     1     1
sub r4, r1, r6      4        2         2     2     2
ld  r5, 10[r4]      7        3   3     3     3     3
add r6, r5, r5      10       5         5     5     5
sub r8, r8, r9      11       6         6     6
mul r10, r10, r11   12       7         7     7
cmp r8, r10         15       8         8     8
beq .label          18       9         9     9
add r5, r6, r8      21       12
st  r3, 20[r5]      24       13              10
ld  r6, 20[r5]      27+1     14              11
ld  r7, 20[r6]      30+1     16              12
lsl r7, r7, r10     33+1     18              ~~13~~14
                                             16
```

(6 marks)

i) Assuming a traditional *SimpleRisc* pipeline, how many cycles will this code take to execute in a pipeline with just interlocks? Assume that time starts when the first instruction reaches the RW stage. This means that if we had just one instruction, then it would have taken exactly 1 cycle to execute (Not 5). Moreover, assume that the branch is not taken. [Assumptions: No forwarding, No delayed branches, No reordering] Ans: 34

ii) Now, compute the number of cycles with forwarding (no delayed branches, ~~no reordering~~). Ans: 16

iii) Compute the minimum number of cycles when we have forwarding, and we allow instruction reordering. We do not have delayed branches, and in the reordered code, the branch instruction cannot be one of the last three instructions. Number of cycles = Ans: 15

iv) Compute the minimum number of cycles when we have forwarding, allow instruction reordering, and have delayed branches. Here, again, we are not allowed to have the branch instruction as one of the last three instructions in the reordered code. Number of cycles = Ans: 16

Note to TAs (a) 1, (b) 1.5 , (c) 1.5, (d) 2. For ± 1, give a third of the marks (2/3 = 0.5, 1.5/3 = 0.5, 1/3 = 0).

Research

8. We wish to compute the square root of a floating point number in hardware using the Newton Raphson method. Outline the details of an algorithm, prove it, and compute its computational complexity. Follow the following sequence of steps. (8 marks)

(a) Find an appropriate objective function.
(b) Find the equation of the tangent, and the point at which it intersects the x-axis.
(c) Find an error function.
(d) Calculate an appropriate initial guess for x.
(e) Prove that the magnitude of the error is less than 1.
(f) Prove that the error decreases at least by a constant factor per iteration.
(g) Evaluate the asymptotic complexity of the algorithm.

Answer:

(a) Objective function: $f(x) = x^2 - b$ Assume that $1/2 \leq b < 2$. Any floating point number in the normal form can be brought to this form by rounding the exponent to the nearest even number. Example. $1.5 \times 2^{-3} = 0.75 \times 2^{-2}$.

(b) Slope = 2x. Point at which the tangent intersects the x-axis (x_2):

$$\begin{aligned} \frac{x_1^2 - b}{x_1 - x_2} &= 2x_1 \\ \Rightarrow x_2 &= \frac{x_1^2 + b}{2x_1} \end{aligned} \tag{1}$$

(c) Error function: $E(x) = x - \sqrt{b}$

$$\begin{aligned} E(x_2) &= x_2 - \sqrt{b} \\ &= \frac{x_1^2 + b}{2x_1} - \sqrt{b} \\ &= \frac{x_1^2 + b - 2x_1\sqrt{b}}{2x_1} \\ &= \frac{(x_1 - \sqrt{b})^2}{2x_1} \\ &= \frac{E(x_1)^2}{2x_1} \end{aligned} \tag{2}$$

(d) Initial guess for x: x = 1

(e) Initial value of the error: $1 - \sqrt{b}$.

$$(1/2 \leq b < 2) \Rightarrow (1 - \sqrt{2}) < (1 - \sqrt{b}) \leq (1 - 1/\sqrt{2})$$

Hence, $\mid 1 - \sqrt{b} \mid < 1$

(f) Let us prove by induction that $x_n \geq 1/2$, and $b \geq x_n/2$. This is true for the base case $x_1 = 1$. Assume the hypothesis is true for $x_n = x'$. Let us try to prove that the hypothesis holds for $x_{n+1} = x''$.

$$\begin{aligned} x'' &= \frac{x'^2 + b}{2x'} \\ &= \frac{x'}{2} + \frac{b}{2x'} \end{aligned} \tag{3}$$

Now, $x'/2 \geq 1/4$ (because, $x' \geq 1/2$), and $b/2x' \geq 1/4$ (because $b \geq x'/2$). Hence, $x'' \geq 1/2$.

Let us now prove that $b \geq x''/2$. We have:

$$\begin{aligned} & b \geq x''/2 \\ \Leftrightarrow & b \geq \frac{x'^2 + b}{4x'} \\ \Leftrightarrow & b \geq \frac{x'}{4} + \frac{b}{4x'} \end{aligned} \tag{4}$$

By the induction hypothesis, $b/2 \geq x'/4$. If we prove that $b/2 \geq b/4x'$, we are done.

$$\begin{aligned} & \frac{b}{2} \geq \frac{b}{4x'} \\ \Leftrightarrow & 1 \geq \frac{1}{2x'} \\ \Leftrightarrow & x' \geq \frac{1}{2} (TRUE) \end{aligned} \tag{5}$$

We have thus proved the induction, and we can conclude that $x_n \geq 1/2$ for all values of n. Hence, $\forall n, 2x_n \geq 1$. We thus have:

$$E(x_{n+1}) \leq E(x_n)^2$$

(g) We have $log(n)$ steps because, we assume that the precision is till n bits. In each step, we have to do a left shift and round (log(n)), addition (log(n)), and division ($log(n)^2$).

Thus, the total time complexity is $\boxed{\text{Ans: } O(log(n)^3)}$

Note to TAs: : (a and b) should have a reduced form of b (1 mark) (c) 1 mark (d and e) 1 mark (f) 4 marks (g) 1 mark. Take a look at part (f) carefully, it is the crux of the algorithm. The proof should be crystal clear (for full marks). If there is any ambiguity, the benefit of doubt goes to the TA. For this answer, we need to assume that the time complexity of a division is $O(log(n)^2)$ (This is a research question).