Ruchita

3/21/2017

CSCE 530

Midterm Exam #1

1. Consider a processor with the following specifications: (1) mean CPI of FP instructions = 5.0, (2) mean CPI of all other instructions = 2.0, and (3) CPI of FPSQR instructions = 24.0. Assume that FP operations account for 20% and FPSQR operations account for 4%. If the processor is to be enhanced by either (1) decreasing the mean CPI of its FPSQR down to 6.0 or (2) decreasing the mean CPI of all its FP operations down to 3.4. If these two enhancement alternatives cost equally to realize, which one of the two is preferred? (Show your work, 10%)

2. Given a multi-core system to run an application with 80% parallelizable (when all cores receive the same workload to execute in parallel) while a single-core system is sufficient for the application (in terms of the response time), power savings can result from lowering its operating frequency. How much dynamic power savings will be from an eight-core system whose operating voltage may not decrease by more than 33% of the original voltage (called the *voltage floor*), when compared with its single-core counterpart? Note that voltage (V) decreases linearly with frequency (f) until it reaches the voltage floor, and any further reduction in frequency does not bring down the voltage below its floor. the expression of dynamic power is proportional to $V^2 \times f$. During the sequential execution portion, all but one core are powered down to save energy. (Show your work. 10%)

3. Show the diagram of correlated branch prediction which involves 1K bits in total for all predictors, each with 2 bits and correlated under 3-bit global branch results. (The address bits used for decoding have to be provided as well. 10%)

??

$n=2$ 64bits

$2^3 \cdot 2 \times \boxed{2^6} = 1K = \frac{2^{10}}{2^4} = \boxed{2^6}$

$\frac{1024}{8}$ 128

4. Explain briefly how to realize speculative execution (using speculative instructions) with exceptions handled properly (but not necessarily precisely) via associating poison bits with registers, using the code fragment below as an example by speculating the 2nd load instruction. (15%)

=1K

512 1024 bits / 2

```
            if (A==0) A = B; else A = A+4;

        where A is at 0(R3) and B is at 0(R2):

                    LD      R1,0(R3)    ;load A
                    BNEZ    R1,L1       ;test A
                    LD      R1,0(R2)    ;then clause
                    J       L2          ;skip else
            L1:     DADDI   R1,R1,#4    ;else clause
            L2:     SD      R1,0(R3)    ;store A
```

5. Briefly explain how trace scheduling works for unrolling a loop twice, where one branch instruction is resided in the loop body. What kind of instructions can be rescheduled ahead of the branch instruction and what are compensation codes for? (10%)

Scanned with CamScanner

6. Consider the MIPS code fragment given below for execution on a processor without a branch target buffer (BTB) so that a branch delay slot exists, with the following numbers of execution cycles: (i) L.D & S.D: 2, (ii) ADD.D & SUB.D: 4, (iii) MUL.D: 3, (iv) integer DADD, DSUB, DADDI, and BNE: 1 -- 2

```
Loop:   L.D       F6, 32(R2)
        L.D       F2, 8(R3)
        L.D       F4, 0(R1)
        MUL.D     F0, F2, F4
        SUB.D     F6, F2, F6
        ADD.D     F6, F6, F0
        S.D       F6, 32(R2)
        S.D       F0, 8(R3)
        DADDI     R1, R1, #-8
        DADDI     R2, R2, #-8
        DADDI     R3, R3, #-8
        BNEZ      R3, Loop
```

where R2, R3, and R1 are initialized before the loop body. The processor issues one instruction per cycle and contains one FP adder (responsible for FP addition and FP subtraction operations) plus one FP multiplier. Its execution unit is pipelined, with results forwarded from one execution stage to another (or itself), whereas FP functional units are not pipelined.

Schedule the code fragment statically by reordering its instructions as best as possible, after register renaming, if necessary, to improve its execution performance. How many cycles per iteration (including the branch delay slot, if any) does your reordered code take? (20%)

✓ 7. Show a stall-free software-pipelined version of the following code fragment (without the start-up code nor the wind-down code), assuming that R1, R2, and R3 are initialized and that the instruction execution cycles follow those specified in Problem 3 above, except for MUL.D which takes 8 cycles for execution. (15%)

```
Loop:   L.D       F2, 0(R1)
        L.D       F4, 0(R2)
        L.D       F0, 0(R3)
        MUL.D     F4, F4, F2
        ADD.D     F4, F4, F0
        S.D       F4, 0(R2)
        DADDI     R1, R1, #-8
        DADDI     R2, R2, #-8
        DADDI     R3, R3, #-8
        BNEZ      R1, Loop
```

-1 S(n)

no renaming

Redo the preceding code fragment, with the instruction execution cycles as what are specified in Problem 3, except for MUL.D which takes 10 cycles for execution. (10%)

Good Luck!

Scanned with CamScanner