NAME:
ID:

EE M116C/CS 151B

Computer Systems Architecture

Fall 2008 Final Exam

Instructor: Prof. Gupta

One handwritten double-sided sheet, MIPS cheat-sheet and calculator allowed.
Total allotted time: 2 hours
Total points: 100

Please show your *relevant* work for partial credit. Showing at least some line of reasoning to arrive at an answer is required for anything that is not trivially obvious. Without it you may not be given full credit even if your answer is correct. Also erase/cross-out any irrelevant work. Again incorrect extraneous work on the answer sheet may lead to deduction of points.

There are total seven pages including this cover page. Check that you have all pages. If not, let the proctor know right now.

Q 1: _____ of 25 points

Q2: _____ of 20 points

Q 3: _____ of 30 points

Q4: _____ of 17 points

Q 5: _____ of 8 points

Total: _____ of 100 points

Q1. 25 points

You are running a benchmark on your company's processor, **Mbase**, which runs at 400 MHz and has the following characteristics:

Instruction Type	Frequency	Cycles
A	40%	2
B	30%	3
C	20%	3
D	10%	5

(a) What is the CPI rating for **Mbase**?

Answer:

Effective CPI = 40%X2+30%X3+20%X3+10%X5=2.8

(b) You ask the hardware team if they can improve the processor design. They tell you that they could make this processor run at 500 MHz, however they would have to increase the number of cycles for instruction type C to 4. (All the other instruction types still take the same number of cycles). Call this machine **Mopt**.

Instruction Type	Frequency	Cycles
A	40%	2
B	30%	3
C	20%	4
D	10%	5

What is the CPI rating for **Mopt**?

Answer:

Effective CPI = 40%X2+30%X3+20%X4+10%X5=3

(c) How much faster is **Mopt** than **Mbase** ?

Answer:

$$\frac{2.8 \times {}^{1}/_{400 \times 10^{6}}}{3 \times {}^{1}/_{500 \times 10^{6}}} = 1.17$$

d) Is there an instruction mix that makes **Mbase** faster than **Mopt**? If so, suggest such a mix. (Note: The mix doesn't have to contain all the instruction types.)

Answer:

In such an instruction mix

Instruction Type	Frequency	**Mbase** Cycles	**Mopt** Cycles
A	5%	2	2
B	5%	3	3
C	85%	3	4
D	5%	5	5

Mbase's effective CPI=5%X2+5%X3+85%X3+5%X5=3.05
Mbase's effective CPI=5%X2+5%X3+85%X3+5%X5=3.9

Then

$$\frac{3.9 \times {}^{1}/_{500 \times 10^{6}}}{3.05 \times {}^{1}/_{400 \times 10^{6}}} = 1.02$$

So Mbase is 1.02 times faster than Mopt.

(e) If instead of multi-cycle implementation as above, the processor **Mbase** has a pipelined implementation with 5 stages, what will be the latency of instruction A (assume that there is no overhead from pipeline registers)? Assuming no hazards, what will be the effective CPI for a very long program running on this pipeline?

Answer:

Since the longest instruction needs 5 cycles, each stage need 1 cycle. Therefore, the latency of instruction A is 5.

The effective CPI will be 1 for a very long program running on the pipeline.

Q2. 20 points

(a) Generate code that avoids pipeline stalls for the following sequence:

a= b + c

d= e – f

Assume that six registers, R1 through R6, are available. Assume the availability of Load, Store, Add, and Sub instructions. Assume forwarding too.

Answer:

lw R1, b

lw R2, c

lw R3, e

lw R4, f

add R5, R1, R2

sub R6, R3, R4

sw a, R5

sw d, R6

(b) The following piece of code has pipeline hazard(s) in it. Reorder the instructions and insert the least number of NOOPs to make it hazard free. Do not change the instructions themselves and assume all forwarding logic and a branch delay slot exist.

```
haz:  move   $5, $0
      lw     $10, 1000($20)
      addi   $20, $20, -4
      add    $5, $5, $10
      bne    $20, $0, haz
```

Answer:

```
haz:  addi   $20, $20, -4
      bne    $20, $0, haz
      lw     $10, 1000($20)
      move   $5, $0
      add    $5, $5, $10
```

Q3. 30 points

(a) The page size of a computer is 16 **Kbytes**; the block size is 32 **words** and the machine is **byte**-addressable. The cache size is 1 **Kbtye**, and it is 4-way set associative. The virtual addresses are 42 bits and physical addresses are 36 bits long. Calculate the sizes (number of bits) of the following fields: (12 points)

i) block offset
ii) set index
iii) tag
iv) page offset
v) virtual page number
vi) physical page number

Answer:

i) 7
ii) block number = 1KB/128B =8 set number = 8/4=2, so set index needs 1 bit
iii) 36-1-7=28
iv) 14
v) 42-14=28
vi) 36-14=22

(b) For a cache with 4 total blocks and block size of 4 bytes, following memory reference sequence is given: 00000 00100 01001 11000 11100 01100 11100 What are the possible values of associativity for this cache? If the last access is a miss, what is the cache design ? Explain how you got to the answer. (13 points)

Answer:

The associativity can be 1,2, and 4. Because there are 4 blocks totally.

If the last access is a miss, the associativity is 1. That the last access is a miss means 11100 is replaced by 01100. Then 11100 and 01100 must be in the same set, and in each set, there is just one block.

(c) A processor has a 4 entry TLB and 4KB pages. What is the maximum memory a program should access to maximize its performance?

Answer:

16KB.

Q4. 17 points

(a) To get to 3TB storage with 500GB disks, how many disks will I need in (i) RAID0 (ii) RAID1 (iii) RAID5 configurations ? What is the minimum and maximum number of disk failures that the RAID1 configuration can tolerate ?

Answer:

(i) 3TB/500GB=6 disks

(ii) 3TBx2/500GB=12 disks

(iii) 7 disks. RAID 5 needs one additional disk.

For RAID1, it can tolerate 1 to 6 disk failures

(b) If the overheads of polling and interrupt handling are 100 and 1000 cycles respectively for a processor with clock frequency 100MHz, which is a better approach (in terms of CPU time wasted) for a mouse which is moved at most once every millisecond but on average moves only once every 100milliseconds (i.e. 10 times per second).

Answer:

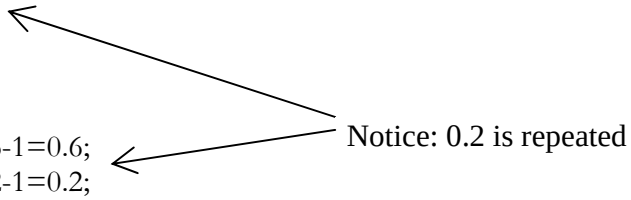
Interrupts waste 1000*10 cycles every second. For polling, CPU has to poll once every ms to be safe. Polling overhead is 100*1000 cycles every second.

Therefore, interrupt is better than polling.

Q5. 8point

(a) Convert 0.1 decimal to IEEE single-precision floating point.

Answer:



0.1*2=0.2<1, the 1st bit is 0;
0.2*2=0.4<1, the 2nd bit is 0;
0.4*2=0.8<1, the 3rd bit is 0;
0.8*2=1.6>1, the 4th bit is 1, 1.6-1=0.6;
0.6*2=1.2>1, the 5th bit is 1, 1.2-1=0.2;
The binary number of 0.1 is infinite.
$0.1=0.0001100110011\ldots_b=0.0\dot{0}01\dot{1}_b=1.10011001100110011001100*2^{-4}$
e=-4.
$E=127-4=123=01111011_b$
Then floating point number of 0.1 is represented as:

0	01111011_b	$10011001100110011001101_b$

Sign bit E M

(The last bit of Mantissa is 1 because of rounding.)

(b) The following piece of C code does not print "equal". Why ?

```
float a = 1;
float b = 1 + 10/100;
float c = 1.1;
if(c==b)cout<<"equal";
```

Answer:

Because 10/100=0.1, and according to (a), we know the binary number of 0.1 is infinite, which will be rounded. As a result, in the computer, 10/100≠0.1. Therefore c≠b.

Or you can say 10/100 is integer division, so b=1, and c≠b.