Computer Science Comprehensive Examination
Computer Architecture
[60 points]

This examination is open book. Please do all of your work on these sheets. Do not do your work in a blue book.

Number: _____

Problem	Max Score	Your Score
1	20	
2	20	
3	20	
TOTAL	60	

Problem 1: Short Answer [20 points]

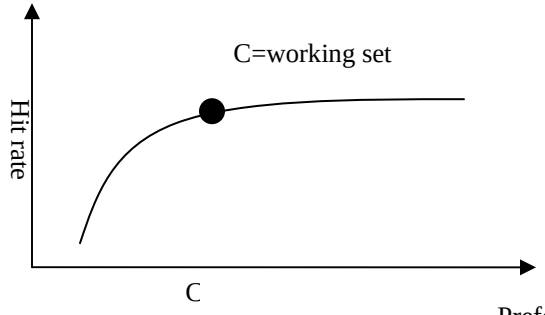
A. [7 points] Assume a pipelined processor with N pipeline stages. As we increase N, briefly explain that happens to the following (increase/decrease + 1 sentence reasoning)

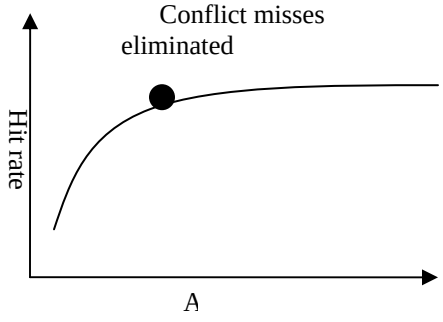
- Clock cycle time: Decreases – `fewer gates per pipeline stage`
- Cycles per instructions: Increases – `more data and control hazards`

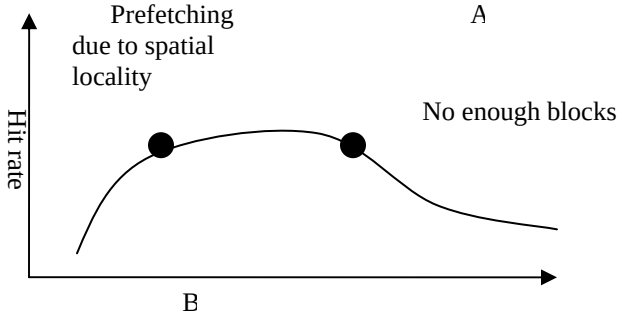
Considering clock cycle time only, is there a limit or point of diminishing returns for the number of pipeline stages in a processor? Why?

`Yes there is. Clock cycle is limited by clock skew and pipeline register clocking overhead. As we subdivide the work for each instruction in more pipeline stages, eventually the clock cycle will become equal to the skew+overhead. Deeper pipelining, will not lead to additional clock cycle benefits`.

[7 points] Assume a first level data cache with capacity (C), associativity (A), and block size (B). Draw the expected effect on hit rate as C, A, and B increase respectively. In each case, we vary one of the three parameters, while the other two are kept constant. Mark any interesting points on the graphs with 3-4 words to provide some reasoning. .







B. [6 points] What does the following MIPS code do? What is the instruction cache miss rate? What is the data cache miss rate? Assume the caches are initially empty and have a 16 byte line size.

```
        move  $t0, $a0
        addi  $t1, $t0, 4000
loop:   Sw    $t0, 0($t0)
        addi  $t0, $t0, 4
        slt   $t2, $t0, $t1
        bne   $t2, $zero, loop
exit:
```

The code intitializes a 1000 words to the memory address of the word

Icache miss rate = 2/(2 + 1000*4) = 2/4002 = 0.05%
Dcache miss rate = 1/4 = 25%

Problem 2: Instruction-Set Architecture Design [20 Points]

Your task is to redesign the way branches work in the MIPS ISA to create a new architecture called MIPS-new. You decide to add branches with arbitrary compares which are called complex branches. You are given the following information about MIPS and MIPS-new

Instruction type	Frequency
compares	23%
ALU ops (not compares)	20%
loads/stores	30%
conditional branches	25%
jumps	2%

Instruction mix on MIPS

Branch type	Frequency
NE/EQ compare to nonzero	30%
NE/EQ compare to zero	25%
LT/LE compare	25%
GT/GE compare	20%

Frequency of branch compares on MIPS-new

A. [10 points] Using the information given above, what is the ratio of instruction count (IC) of MIPS to MIPSnew.

```
IC_MIPS-new = IC_MIPS - compares eliminated
compares eliminated = branch compares except NE/EQ compares to zero
IC_MIPS-new = IC_MIPS - %branches(%branch compares eliminated)
IC_MIPS-new = IC_MIPS (1 - 25%(75%)) = IC_MIPS81.25%
IC_MIPS/IC_MIPS-new = 1.23
```

B. [10 points] Complex branches in MIPS-new force you to move the branch decision point from the ID stage to the EX stage. To maintain compatibility with MIPS, you decide to use delayed branching for the first cycle of branch delay and to predict taken for any subsequent cycles of branch delay. Assume that the target PC is calculated in the ID stage. If the delay slot is usefully filled 50% of the time and branches are taken 70% of the time, what is the ratio of CPIs of MIPS to MIPS-new? Assume that noops do not count as instructions and all instructions except branches and jumps have a CPI of 1.0.

CPI_MIPS = 1 + CPI_branch-delay + CPI_jump-delay

In MIPS the single delay slot is usefully filled 50% of the time

CPI_branch-delay + CPI_jump-delay = 25% (0.5) + 2%(0.5)

CPI_MIPS = 1.14

CPI_MIPS-new = 1 + CPI_branch-delay + CPI_jump-delay

In MIPS-new the first cycle of branch delay behaves like MIPS and the second cycle of branch delay is hidden when the branch is taken (70% of the time) and all the time for jumps.

You must renormalize the instruction mix to account for the compares that have been eliminated

MIPS-new branch frequency = 25%/81.25% = 31%

MIPS-new jump frequency = 2%/81.25% = 2.5%

CPI_branch-delay + CPI_jump-delay = 31%(0.5 + (1-70%)) + 2.5%(0.5)

CPI_MIPS-new = 1.26

CPI_MIPS/CPI_MIPS-new = 1.14/1.26 = 0.9

Problem 3: Memory Hierarchy Design [20 Points]

You have a computer with two levels of cache memory and the following specifications:

- Processor: 2GHz, 64-bit RISC CPU
- On-chip L1 caches
  - split instruction & data cache, blocking, single-ported
  - write-through & non-write allocate
  - 1 CPU cycle access time
  - Block size = 32 bytes
- Off-chip L2 cache off-chip
  - unified single-ported cache, blocking
  - write-back
  - 10 CPU cycles access time (L1 miss penalty) for both reads and writes
  - Block size = 32 bytes
- Main memory:
  - Bus: 64-bit data transfers at 400 MHz
  - Latency: 15+5+5+5 CPU cycles access time for 32 bytes (L2 miss penalty – includes latency of both DRAM and memory bus)

A. What is the:

- Peak L1 data cache bandwidth available to CPU (assuming 0% L1 misses)?
- Peak L2 cache bandwidth available to L1 cache (assuming 0% L2 misses)?
- Main memory bandwidth available to L2 cache?

Report the bandwidths in Gbytes/sec, i.e. 2^{30} bytes/sec.

Peak L1 cache bandwidth:
8 bytes / 1 CPU cycle access time = 8 bytes/0.5ns = 16 Gbytes/sec

Peak L2 cache bandwidth:
32 bytes / 10 CPU cycle access time = 32 bytes/5ns = 6.4 Gbytes/sec

Peak memory bandwidth:
32 bytes / 30 CPU cycle access time = 32 bytes/15ns = 2.13 Gbytes/sec

B. You are given the following L1 cache statistics for a program executing on this system

Metrics	Access Type: Total	Instrn	Data	Loads	Stores
Accesses	10000000	7362210	2637790	1870945	766845
Misses	52206	8466	43740	36764	6976

Words Read From Lower-levels	180920 (i.e. 45230 cache lines)
Words Written-back to Lower-levels	766845
Total Traffic	947765

How long does an average instruction take to execute (in ns), assuming 1 clock cycle per instruction in the absence of memory hierarchy stalls, no write buffering at the L1 cache level, and 0% L2 miss rate? Ignore register dependencies between instructions.

Any instruction that hits in the cache will not be penalized by the misses. (Unless of course there are data dependencies, but since the problem tells us to ignore this). Thus, we just need to find average miss penalty of an instruction since that will incur extra latency. Note, that since we have a 0% L2 miss rate, we never incur any main memory accesses.

Average instruction latency $= \text{latency}_{ideal} + \text{extra latency}_{stalls}$
$= 1 + \text{instruction stall cycles} + \text{data stall cycles}$

Instruction stall cycles = *I$ miss rate * L1 miss penalty*
Data stall cycles = (*load rate * load miss rate * L1 miss penalty*) + (*store rate * write penalty*)

The L1 miss penalty is 10 cycles. Since L1 is write-through, we assume the write penalty is the same as the L1 miss penalty which is 10 cycles. Also, since the L2 is single-ported, there may be the case where a instruction miss and a data miss occurs at the same time and there will be a structural hazard. In that case, one of the misses will incur an extra 10 cycle penalty. However the chance of this happening is almost zero: *I$ miss rate * D$ miss rate* = 0.115%*1.66% ~= 0%. Therefore we've neglected this in our equation.

The *I$ miss rate* is given in the table as: 8466/7362210 = 0.115%
The *load rate* is given in the table as: 1870945/7362210 = 25.41%
The *load miss rate* is given in the table as: 36764/1870945 = 1.965%
The *store rate* is given in the table as: 766845/7362210 = 10.42%

Thus,
Instruction stall cycles = 0.115% * 10 = 0.0115 cycles
Data stall cycles = 25.41% * 1.965% * 10 + 10.42% * 10 = 1.09 cycles
Average instruction latency = 1 + 0.0115 + 1.09 = 2.1015 cycles
So, average instruction time = Average latency * cycle time = 2.1015 * 0.5ns = 1.05ns

C. You are considering replacing the L2 cache with a victim cache. Given the information provided to you, compute a measure of "speed" for each alternative and

indicate which is the faster solution. Justify the metric you choose to compare the two alternatives and state your assumptions. Assume the performance statistics are:

- L2 cache local miss ratio = 0.18
- Victim cache miss ratio = 0.23
- Victim cache transport time from L1 miss = 2 CPU clock

Hint: Use a metric that's simple and is representative of the common case.

Given the information provided, it's clear that the common case is a cache read. The reads, i.e. instruction fetch and loads, account for (7362210+2637790)/10000000 = 92.3% of total L1 cache accesses. Also, the write miss rate is much lower than the load miss rate. Thus, one metric that we can use for comparison then is **AMAT of a read**.

The L1 read miss rate = (instruction & load misses) / (instruction & load accesses)
= (8466 + 36764) / (7362210 + 1870945) = 0.4899%

1. L2 Cache,

$AMAT_{L2}$ = 10 + L2 miss rate * 30 = 10 + 0.18*30 = 15.4 cycles
$AMAT_{L1}$ = 1 + L1 read miss rate * $AMAT_{L2}$ = 1 + 0.004899*15.4 = 1.0754 cycles

2. Victim Cache,

$AMAT_{VC}$ = 2 + VC miss rate * 30 = 1 + 0.23*30 = 8.9 cycles
$AMAT_{L1}$ = 1 + L1 read miss rate * $AMAT_{L2}$ = 1 + 0.004899*8.9 = 1.0436 cycles

So, it would seem that the victim cache would be a better choice in this case.