ECE 6913 CSA Quiz - 2

Sahil Makwane (sm9127)

Q1. CPU Time = Clock Cycle * Instruction Count * CPI

600 = x * 300 * 1

2 ns = x

M's Clock Cycle needs to be shorter than **2 ns** in order to make his program run faster than O's

---

Q2. CPU Execution Time = (Clock $cycle_{CPU}$ + Memory Stall cycles) * Clock Cycle

= (IC * CPI + 0) * Clock Cycle

= IC * 1.0 * Clock Cycle

Memory Stall cycles = IC * (Memory accesses / Instruction) * Miss Rate * Miss Penalty

= IC * (1 + 0.25) * 0.05 * 50

= IC * 3.125

CPU Execution $Time_{cache}$ = ((IC * 1.0) + (IC * 3.125)) * Clock Cycle

= 4.125 * IC * Clock Cycle

Speedup = CPU Execution $Time_{cache}$ / CPU Execution Time

= 4.125 * IC * Clock Cycle / IC * 1.0 * Clock Cycle

= 4.125

The computer with no cache misses is **4.125** times faster

Q3. $CPU\ performance_B$ / $CPU\ performace_A$ = $CPU\ Time_A$ / $CPU\ Time_B$

6 = 3 / $CPU\ Time_B$

$CPU\ Time_B$ = 0.5

0.5 is the User CPU time

As I/O time does not get affected by increase in performance, I/O requires 1 second to be done.
Thus, 1 + 0.5 = 1.5 seconds would be required to run Program A on the faster processor CPU

1.5 is the Wallclock Time

MFLOPS = Number of Floating Point Operations * (10^6 / Wallclock Time)

10 = Number of Floating Point Operations * 10^6 / 4

40 * 10^6 = Number of Floating Point Operations

New System Performance = 40 * 10^6 / 1.5

MFLOP/sec = 26.6667

26.6667 is the MFLOP/sec performance

Q4. i) Data forwarding alone is not enough as by the time the first ALU instruction completes the second ALU (execute) step, the second ALU instruction is already in the first ALU step. Thus, it's too late.
Therefore, a NOP between the 2 instructions, as well as data forwarding between the second ALU step and the decode step is required.

ii).

	CC0	CC1	CC2	CC3	CC4	CC5	CC6	CC7	CC8
ADD x1, x2, x3	IF	ID	ALU 1	ALU 2	MEM	WB			
NOP									
ADD x5, x4, x1			IF	ID	ALU 1	ALU 2	MEM	WB	

Q5. Miss rate is 0.5 and block size is of 2 words (8 bytes)
The frequency of memory operations from processor is 10^9
and frequency of writes from processor is $0.25 * 10^9$
Bus transfers only 1 word at a given time to/from processor/memory
30% of cache blocks have been modified on average (must be written back in the case of the write back cache)
The cache is 'write allocate'

Thus,
Fraction of read hits = 0.75 * 0.95
= 0.7125

Fraction of write hits = 0.25 * 0.95
= 0.2375

Fraction of read misses = 0.75 * 0.05
= 0.0375

Fraction of write hits = 0.25 * 0.95
= 0.2375

Fraction of write misses = 0.25 * 0.05
= 0.0125

a. write back cache
On a read hit there is no memory access
On a read miss:
1. If replaced line is modified then cache must send 2 words to memory, and then memory must send 2 words to the cache
2. If replaced line is clean then memory must send 2 words to the cache

On a write hit there is no memory access
On a write miss:
1. If replaced line is modified then cache must send 2 words to memory, and then memory must send 2 words to the cache
2. If replaced line is clean then memory must send 2 words to the cache
Thus:
Average words transferred = 0.7125 * 0 + 0.0375 * (0.7 * 2 + 0.3 * 4) + 0.2375 * 0 + 0.0125 * (0.7 * 2 + 0.3 * 4) = 0.13
Average bandwidth used = $0.13 * 10^9$
Fraction of bandwidth used = $0.13 * 10^9 / 10^9 = 0.13$

b. write through cache

- ⇨ On a read hit there is no memory access
- ⇨ On a read miss memory must send 2 words to the cache
- ⇨ On a write hit the cache must send 1 word to memory
- ⇨ On a write miss memory must send 2 words to the cache, and then the cache must send 1 word to memory

Thus:
Average words transferred = 0.7125 * 0 + 0.0375 * 2 + 0.2375 * 1 + 0.0125 * 3 = 0.35
Average bandwidth used = $0.35 * 10^9$
Fraction of bandwidth used = $[0.35 * 10^9] / 10^9 = 0.35$

Therefore, comparing 1 and 2, we conclude that the write through cache uses more than twice the cache-memory bandwidth of the write back cache.