Homework 3B

1.Performance and CPI. Assume a typical program has the following instruction type breakdown:

- 30% loads
- 15% stores
- 50% adds
- 4% multiplies
- 1% divides

Assume the current-generation processor has the following instruction latencies:

- loads: 2 cycles
- stores: 6 cycles
- adds: 1 cycles
- multiplies: 14 cycles
- divides: 50 cycles

If for the next-generation design you could pick one type of instruction to make twice as fast (half the latency), which instruction type would you pick? Why?

I will choose to half the latency of **stores**. Because it has the most latency

Percentage	30%	15%	50%	4%	1%
cycles	2	6	1	14	50
	0.6	0.9	0.5	0.56	0.5
					3.06

2.Averaging. Assume that a program executes one branch every 8 instructions for the first 1M instructions and a branch every 16 instructions for the next 2M instructions. What is the average number instructions per branch for the entire program?

Total branch: 1M / 8 + 2M/16 = 250000

Average instruction per branch 3M / 250000 = **12**

3.Diminishing Returns (Amdahl's Law). A program has 20% divide instructions. All non-divide instructions take one cycle. All divide instructions take 20 cycles.

1 ***What is the CPI of this program on this processor?***

1*0.8+20*0.2 = 4.8

2 ***What percent of time is spent just doing divides?***

20*0.2/4.8 *100% = 83.33%

3 ***What would the speedup be if we sped up divide by 2x?***

If we speed up divide by 2x, each divide will need 10 cycles
CPI: 1*0.8 + 10 *0.2 = 2.8
Speedup: (1/2.8 - 1/4.8)/(1/4.8) *100% = 71.43%

4 ***What would the speedup be if we sped up divide by 5x?***

CPI: 1*0.8 +4*0.2 = 1.6
Speedup: (1/1.6 - 1/4.8)/(1/4.8)*100% = 200%

5 ***What would the speedup be if we sped up divide by 20x?***

CPI: 1*0.8 + 1*0.2 = 1
Speedup: (1 - 1/4.8)/(1/4.8)*100% = 380%

6 ***What would the speedup be if divide instructions were infinitely fast (zero cycles)?***

CPI: 1*0.8 + 0*0.2 = 0.8
Speedup: (1/0.8 - 1/4.8)/(1/4.8)*100% = 500%

4.Performance and ISA. Chip A executes the ARM ISA and has a 2.5Ghz clock frequency. Chip B executes the x86 and has a 3Ghz clock frequency. Assume that on average, programs execute 2.5 times as many ARM instructions than x86 instructions.

1 ***For Program P1, Chip A has a CPI of 2 and Chip B has a CPI of 3. Which chip is faster for P1? What is the speedup for Program P1?***

Suppose x86 chip execute X instructions per program, then the ARM chip will execute 2.5X instructions per program. We will calculate "secs per program" for both ARM chip and x86 chip
For x86 chip: X*1/3G * 3 = X * 10^-9
For ARM chip: 2.5X *1/2.5G * 2 = 2X *10^-9
x86 chip is **100% faster** than ARM chip

2 ***For Program P2, Chip A has a CPI of 1 and Chip B has a CPI of 2. Which chip is faster for P2? What is the speedup for Program P2?***

For ARM chip: 2.5X * 1/2.5G * 1 = X *10^-9
For x86 chip: X *1/3G*2 = 0.667X * 10^-9
X86 chip is **50% faster** than ARM chip

3 ***Assuming that Programs P1 and P2 are equally important workloads for the target market of this chip, which chip is "faster"? Calculate the average speedup.***

Suppose we have U P1 programs and U P2 programs.

For ARM chip: (U * 2X * 10^-9 + U * X * 10^-9)/(U + U) = 1.5X * 10^-9
For x86 chip: (U * 10^-9 + U * 0.667 * 10^-9)/(U + U) = 0.833X * 10^-9
X86 chip is faster, it's **20% faster** than the ARM chip.

5.ISA Modification and Performance Metrics.

You are the architect of a new line of processors, and you have the choice to add new instructions to the ISA if you wish. You consider adding a fused multiply/add instruction to the ISA (which is helpful in many signal processing and image processing applications). The advantage of this new instruction is that it can reduce the overall number of instructions in the program by converting some pairs of instructions into a single instruction. The disadvantage is that its implementation requires extending the clock period by 10% and increases the CPI by 20%.

1.Calculate under what conditions would adding this instruction lead to an overall performance increase.

Suppose the total instruction count is X, there are A pairs, base clock period is B, base CPI is C, then we have total execution time using the old ISA is B*C*X = BCX
The total execution time using the new ISA is (X-A)*1.1B*1.2C = 1.32BCX - 1.32BCA
We want the new time smaller than the old one, then we have
1.32BCA - 0.32BCX > 0
Then A/X > 0.242
Each pair contains two old instructions, so we need **48.4%** of the instructions can be paired in order to achieve performance increase.

2.What qualitative impact would this modification have on the overall MIPS (millions of instructions per second) of the processor?

Suppose we have B secs per cycle and C cycles per instruction, then we have BC secs per instruction. That's 1/BC instruction per secs.
The MIPS of the old ISA will be (10^6)/BC
For the new ISA, the MIPS will be (10^6)/(1.1B * 1.2 C)= (7.576*10^5)/BC
The MIPS decreases.

3.What does this say about using MIPS as a performance metric?

MIPS alone is not a accurate indicator of CPU performance. Combine the result of question 1 and question 2, we can see that although the new ISA has a lower MIPS, it can outperform the old ISA given that there are enough combinable instructions.

6.Caching.

Consider the following progression of on-chip caches for three generations of chips (loosely modeled after the Alpha 21064, 21164, and 21264).

1.Assume the first generation chip had a direct-mapped 8KB single-cycle first-level data cache. Assume T_{hit} for this cache is one cycle, T_{miss} is 100 cycles (to accesses off-chip memory), and the miss rate is 20%. What is the average memory access latency?

0.8 + 0.2 * 100 = 20.8

2.The second generation chip used the same direct-mapped 8KB single-cycle first-level data cache, but added a 96KB second-level cache on the chip. Assume the second-level cache has a 10-cycle hit latency. If an access misses in the second-level cache, it takes an additional 100 cycles to fetch the block from the off-chip memory. The second-level cache has a global miss rate of 4% of memory operations (which corresponds to a local miss rate of 20%). What is the average memory access latency?

0.8 + 0.04 * 10 + 0.16 * 100 = 16.12

3.The third generation chip replaced the two-levels of on-chip caches with a single-level of cache: a set-associative 64KB cache. Assume this cache has a 5% miss rate and the same 100 cycle miss latency as above. Under what conditions does this new cache configuration have an average memory latency lower than the second generation configuration?

Suppose the new chip's hit latency is X
X * 0.95 + 100 * 0.05 = 5 + 0.95X
We want 5 + 0.95X< 16.12
X < 11.7
When the hit latency is less than 11.7 cycle, the new chip will have an lower latency.

7.Cache Miss Rates

Consider two different cache configurations for an 8-bit processor. Both caches have two 16-byte blocks (for a total capacity of 32 bytes), but one is direct-mapped and the other is two-way set associative and uses the LRU replacement algorithm. All caches begin empty.

1 ***For the direct-mapped cache, how many bits are in the tag, index, and offset?***

Index 1bit, offset 4bit, tag 3bit

2 ***For the set-associative cache, how many bits are in the tag, index, and offset?***

Index 0bit, offset 4bit, tag 4bit

3 ***Give a short sequence of addresses (4 or fewer) in which the set-associative cache has a better hit rate than the direct-mapped cache. (Give the addresses as an 8-digit binary number). Give the miss rate for both the direct mapped cache and the set-associative cache.***

set	tag	data		set	tag	data
0	010		update →	0	010	
1	001			1	**010**	

way 0		way 1	
tag	data	tag	data
0101		0011	

Address:	01010000	hit in 2way associative, miss in direct map
	00110000	hit in 2way associative, miss in direct map because direct map was updated

4 ***Give a short sequence of addresses (4 or fewer) in which the set-associative cache has a worse hit rate than the direct-mapped cache. (Give the addresses as an 8-digit binary number). Give the miss rate for both the direct mapped cache and the set-associative cache.***

set	tag		data		
0	010				
1	001				
	way 0			way 1	
tag	data		tag		data
0101		1	0011		
Update:					
tag	data		tag		data
0101		**0**	**0100**		

Address:01000000	hit in direct map, miss in 2 way associative
Address:00110000	hit in the direct map, miss in 2 way associative because of the update.