Homework #4 Oytun Kuday DURAN 28357

Assigned: 25/05/2023
Due: 08/06/2023

1. (**15 pts**) Consider a processor with the following parameters:

Base CPI, no stalls due to cache misses	0.5
Processor speed	2.5 GHz
Main memory access time	100 ns
First-level cache miss rate per instruction	3%
1st option for the second-level cache	
Direct-mapped: the latency	20 cycles
Local miss rate of the second level cache, direct mapped	50%
2nd option for the second-level cache	
8-way set associative: the latency	50 cycles
Local miss rate of the second level cache, 8-way set associative	40%

a. (**5 pts**) We are considering two alternatives for second-level cache memory as shown above. What are the global miss rates if we use second-level direct-mapped cache (1st option) and second-level 8-way set-associative cache (2nd option)?

Global miss rate if we use 1st option: 0.03 * 0.5 = 0.015 = 1.5%

Global miss rate if we use 2nd option: 0.03 * 0.4 = 0.012 = 1.2%

b. (**10 pts**) Calculate the CPI for the processor in the table using: i) only first-level cache, ii) a second-level direct mapped cache, and iii) a second-level 8-way set associative cache.

i) For the calculation of CPI, the given formula is: Base CPI + (miss rate per instruction * DRAM access time in clk cycles). DRAM access time in clock cycles is calculated by multiplying the clock frequency with the memory access time, which equals 250 clock pulses per DRAM access.

Therefore, the CPI of the processor is computed as: 0.5 (Base CPI) + 0.03 (miss rate per instruction) * 250 (DRAM access time in clk cycles) = 8 CPI

ii) Base CPI + (miss rate of L1 cache) * (hit rate of L2 cache) * latency + (miss rate of L1 cache) * (miss rate of L2 cache) * DRAM access time.

Substituting the given values, we get: 0.5 (Base CPI) + 0.03 (miss rate of L1 cache) * 0.5 (hit rate of L2 cache) * 20 (latency) + 0.03 (miss rate of L1 cache) * 0.5 (miss rate of L2 cache) * 250 (DRAM access time) = 4.55 CPI

iii) For the scenario where the miss rate of the L1 cache is 0.03 and that of the L2 cache is 0.4, we use the same formula and get the following result: 0.5 (Base CPI) + 0.03 (miss rate of L1 cache) * 0.6 (hit rate of L2 cache) * 50 (latency) + 0.03 (miss rate of L1 cache) * 0.4 (miss rate of L2 cache) * 250 (DRAM access time) = 0.5 + 3 + 0.9 = 4.4 CPI

2. (**15 pts**) Assume a direct-mapped cache with 16-byte cache lines. The following code is written in C programming language, where elements of integer arrays of **`A`** and **`B`** within the same row are stored contiguously in memory.

```
for(i=0; i<8; i++)
    for(j=0; j<8; j++)
        A[i][j] = A[i][j] + B[i]
```

Assuming there is no conflict and capacity misses, compute the miss rates for this code due to compulsory cache misses.

Given the cache lines are 16 bytes, each cache line can hold 16/4 = 4 words, since each word is 4 bytes. Hence, our block size is 4 words.

Examining the array A[i][j], we attempt to access a total of 64 different words. For every 4 words, a cache miss will occur (i.e., a miss for x, followed by hits for x+1 to x+3). Then, there will be 64/4 = 16 cache misses associated with A[i][j] operations. Out of the 64 memory accesses for A[i][j], 16 result in a miss.

For the array B[i], we attempt to access 8 different words, each 8 times. On the first access of B[1], a miss will occur, but the subsequent 7 accesses of B[1] will be hits. Similarly, the first access of B[5] will be a miss, but the next 7 accesses of B[5] will be hits. The accesses of B[2], B[3], B[4], B[6], B[7], and B[8] will all result in hits, for 8 iterations each. So, following this pattern, for B[i], we observe 64 memory accesses, with 2 resulting in a miss and 62 resulting in a hit.

Total miss / total access = 18/128 14%.

3. (**12 pts**) Consider Intrinsity FastMath Processor that implements MIPS 32 instruction set architecture. Its virtual addresses are 32-bit integers. It uses 16 KB pages. The processor has a cache with 256 entries where the block size is 16 words (64 B). Assume that a virtual address is 32-bit number, shown as Address[31:0]. Adopting the notation used for byte offset, Address[1:0] for instance, answer the following questions.

a. (**3 pts**) Give the address bits, namely Address[?:?], for cache index.

The cache index is indicative of the block's position within our cache. We have 256 cache lines and thus can represent this quantity with 8 bits (2^8 = 256).

The 32-bit cache address comprises tag bits, an 8-bit cache index, a 4-bit block offset, and a 2-bit byte offset. The byte offset is indicated by [1:0], the block offset by [5:2], and the cache index by [13:6].

b. (**3 pts**) Give the address bits, namely Address[?:?], for page offset.

Our page size is 16KB, corresponding to 2^4 * 2^10 B = 2^14 B. We need 14 bits to represent the position of 2^14 bytes of our data, ranging from 0 to (2^14)-1.

The page offset is thus [13:0].

c. (**3 pts**) Give the address bits, namely Address[?:?], for virtual page number.

Given that we have 32-bit virtual memory addresses and use 14 bits on page offset, the remaining bits are used for the virtual page number, i.e., 32 - 14 = 18 bits. (Virtual Page Number Address[31:14])

d. (**3 pts**) Give the address bits, namely Address[?:?], for block offset.

The block offset shows the position of words in our block. Each block contains 16 words, and we need 4 bytes to represent the positions of these words 16 = (2^4).
The 32-bit cache address comprises tag bits, an 8-bit cache index, a 4-bit block offset, and a 2-bit byte offset. The byte offset is indicated by Address[1:0] and the adress block offset by Address[5:2].

4. (**15 pts**) Consider a hard disk with the following parameters:

- Rotation speed: 7200 RPM
- Average seek time: 5.8 ms
- Transfer rate: 2 MB/s
- Controller overhead: 8 ms
- Sector size: 4096 B

a. **(7 pts)**What is the average time to read or write a sector from a disk?

Assuming there is no delay:

- Controller Time = 8 ms
- Avg Seek Time = 5.8 ms
- Rotational Latency = (60000/2) / 7200 = 4.167 ms
- Transfer Time = 4096 B / 2 MB/s = 2.04 ms

Summing these gives the total average time, which is 20 ms.

b. **(8 pts)** If the transfer rate increases to 4 MB/sec and the control overhead decreases to 6 ms, how much faster is the new disk system?

- Controller Time = 6 ms
- Avg Seek Time = 5.8 ms
- Rotational Latency = (60000/2) / 7200 = 4.16 ms
- Transfer Time = 4096 B / 4 MB/s = 1.02 ms

Summing these gives the total average time The total is 16.98 ms.

20/16.98 = 1.17 => 17% faster

5. Answer the following questions (**15 pts**)

a. (**5 pts**) Assume that you have a hard disk with MTTF = 400,000 hours/failure. Calculate the annual failure rate of the disk (AFR).

365*24 = 8765 hours
AFR = (1000 * 8765) / 400,000 = 21.9125 failures per 1000 disks, or 2.1%

b. (**5 pts**) You are asked to construct a **RAID1** disk system using two disks with capacities 100 GB and 150 GB, respectively. Find the total usable size.

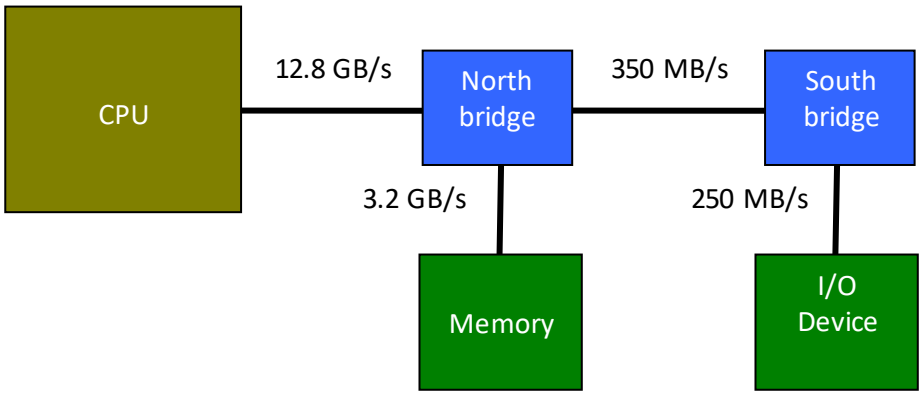
The total usable storage would be 100 GB. This is due to the principle of RAID1 systems, which keep mirrored copies of data on two separate disks. The usable storage is equal to the size of the smaller disk since copy will be kept.
Ans: min(100,150) = 100 GB.

c. (**5 pts**) Disk failures are not independent and if one disk fails the other fails with 50% probability, as well. What is the overall failure rate of **RAID1** you constructed in (b) using the AFR you find in (a)?

P(Failure of RAID1) = P(First Disk Fails) * P(Second Disk Fails | First Disk Fails) + P(Second Disk Fails) * P(First Disk Fails | Second Disk Fails)

P(Failure of RAID1) = 0.021 * 0.5 + 0.021 * 0.5 = 0.0105 + 0.0105 = 0.021, or 2.1%.

6. (**15 pts**) Consider the computer system shown in the figure below:



The bandwidths of the devices are:

Device	**Bandwidth (Byte/s)**
Front Side Bus	12.8×10^9
Memory	3.2×10^9
North-South Bus	350×10^6
I/O Device	250×10^6

Assume that the CPU needs an average of 4 bits from memory and 0.2 bits from the I/O device in order to execute an instruction while running a program.

a. (**10 pts**) Show the maximum instruction execution rates (in MIPS) that each device can sustain. What is the maximum sustainable MIPS of the system (assume that the CPU is not the bottleneck in the system)?

Front Side Bus:
Assuming no bottleneck because of CPU:
bandwidth = 12.8*10^9 Byte/s = 12.8*10^9 * 8 bit/s
Avg = 4bits/instruction for memory access and 0.2 bits/instruction for io.

Maximum sustainable instructions using avg: 12.8*10^9 * 8 / (4 + 0.2) = 2.44 * 10^11 ips => 2440000 MIPS

Memory:
Bandwidth = 3.2 * 10^9 * 8 bit/s.

4 bits form memory =>

Maximum sustainable instructions using avg inst: (3.2 * 10^9 * 8) / 4 = 6.4 * 10^9 ins/s = 6400 MIPS

North/South Bridge Bus:
BW = 350*10^6 * 8 bit/s

0.2 bits from I/O =>
Maximum sustainable instructions using avg inst: (350*10^6 * 8) / 0.2 = 1.4* 10^10 ins/s = 1.4*10^4 MIPS

I/O device:

BW = 250*10^6 * 8 bit/s
Maximum sustainable instructions using avg inst: 250*10^6 * 8 / 0.2 = 10^10 ins/s = 10^4 MIPS

Maximum sustainable MIPS is bottlenecked by memory (lowest) -> 6400 MIPS

b. (**5 pts**) Suppose that the CPU has a clock speed of 1.8 GHz and it can execute 3 instructions per cycle on average (since it is a multi issue processor). What is the bottleneck in the system now?

1.8 Ghz = 1.8 * 10^9 clock cycles per second => 3* 1.8 * 10^9 / 10^6 = 5400 MIPS supported.

CPU bottlenecks since has the lowest MIPS. (5400<6400) MIPS

7. (**15 pts**) Assume that a program executes in 100 seconds, where 80 seconds is CPU time and the rest of the time is spent for I/O operations. If the CPU performance improves by a speedup value of 1.5 and I/O performance improves by a speedup value of 1.1 per year, how much faster will the program run after 5 years?

80 s CPU time + 20 s IO time =>

Speedup = Old execution time/ New execution time

New execution time = Old execution time/ Speedup

New execution time2 = New execution time/ Speedup
=>
execution time = first execution time/ Speedup^n

CPU: X = 80 / 1.5^5 => X = 10.53 seconds

I/O: X = 20 / 1.1^5 => X = 12.41 seconds

Speedup:
100 / (10.53+12.41) = 100/22.94 = 4.36