University of New Brunswick

Computer Science

CS3853: Computer Architecture and Organization

Instructor: Joannah Nanjekye, jnanjeky@unb.ca

Due Date: Aug 02, 2024 – 11:59 PM

ASSIGNMENT 3

Submission instructions:

- Submit a pdf file to the Desire2Learn dropbox

Problem 1. Given a computer system with the following characteristics:

- 4 GB of main memory
- 8 MB of cache
- 128 Bytes in each block

Show the address format for direct mapping Show the address format for 4-way associative mapping

Problem 2. Consider the following statement: "Theoretically, a fully associative cache will have a better hit ratio than a 8-way set associative one." Do you think this is true? Briefly explain why, or why not. Why is a fully associative cache never implemented in practice?

Problem 3. Consider the C program snipped below (one int is 4 bytes).

```
    int v[1024], w[1024], i;
    ...
    for(i = 0; i < 1024; i++) {
        v[i] = w[i] + 5*i;
    }
}
```

The cache has the following characteristics:

- 32-byte block size
- 8 KB of cache
- 4-way set associative

(a) What is the hit ratio of the data? If possible, give the exact ratio.

(b) What is the hit ratio of the data if the cache is reduced to 2 KB? Everything else stays the same. Explain!

(c) Consider the following cache:

- 32-byte block size
- 4 KB of cache
- 2-way set associative

What is the hit ratio of the data? If possible, give the exact ratio. If you are missing information, make reasonable assumptions—and state them.

Problem 4. Consider the C program snipped below (one int is 4 bytes).

```
    int v[1000], w[1000], i;
    ...
    for(i = 0; i < 1000; i++) {
        v[i] = w[i] + 5*i;
    }
}
```

The cache has the following characteristics:

- 32-byte block size
- 4 KB of cache
- direct mapping

What is the hit ratio of the data? If possible, give the exact ratio. If you are missing information, make reasonable assumptions—and state them.

Problem 5. Given a memory of 2GB, 8MB cache, blocks size of 64B show the address format for

(a) direct mapping

(b) associative mapping

(c) 8 way set associative mapping

Problem 6. Describe a cache for which this program will not run well (lots of cache misses). Suggest a way to correct the problem — without changing the cache characteristics.

```
int a[8192], b[8192], c[8192];
int i;
for(i = 0; i < 8192; i++)
        c[i] = a[i] + b[i];
```

Assume that each int is 4 bytes. Assume `a`, `b`, and `c` are allocated successively in memory.

Problem 7. When the processor modifies the cache, main memory must reflect this change. Describe two approaches to keep the cache consistent with main memory.

Problem 8. Consider the following code:

```
int a[100], b[1000], c[1000];
// get some data for a, b, and c
for(i = 0; i < 100; i++)
       for (j = 0; j < 10; j++)
             a[i] += c[j*100 + i] * b[j*100 + i] ;
```

(a) Comment on the spacial locality in the data.

(b) Comment on the temporal locality in the data.

(c) Comment on the spacial locality in the machine code generated by the program.

(d) Comment on the temporal locality in the code.

(e) Write the code that does the same but has better cache hits (Assume that the cache can hold at most 500 integer);

Problem 9. Can a direct mapped cache sometimes have a higher hit rate than a fully associative cache with an LRU replacement policy (on the same reference pattern and with the same cache size)? If so, give an example. If not explain why not?

Problem 10. Consider the C program below

```
int main(){
    int v[10000], i, max;
    init(v);   // somehow v is initialized
    max = v[0];
    for(i = 1; i < 10000; i++) {
       if ( v[i] > max )
          max = v[i];
    }
}
```

(a) Estimate the cache hit ratio for the code with direct mapping. The code is 256 bytes this includes init() which is adjacent to the other code. The cache 8K bytes, and a cache line has 128 bytes. You may assume, that the cache is for code only. Show your work! Each instruction is 4 bytes long. You don't have to give an exact number. The number of cache misses may be useful in this case.

(b) Estimate the cache hit ratio for the data with direct mapping. An int is 4 bytes, the cache has 8K bytes, and a cache line is 128 bytes. You may assume, that the cache is for data only. Show your work!

You may assume that `i` and `max` are stored in a register (no memory access is required for them).

(c) For the above example, describe the impact of going from a direct mapping to a 2-way set associative, and a 4-way set associative mapping.

Problem 11. Consider a processor with a PC-relative branch instruction. Memory addresses are given as 32 bits.

(a) Consider branch instruction is a location A00F1AB0 (in hex). The target address branch is A00F3B08. Determine the displacement in the instruction. All instructions are 4 bytes.

(b) Why do we need to know the instruction size?

(c) The displacement is given as 16 bit signed integer. What is the range of the branch?

(d) Do you think the range of relative branches (as specified above) is adequate? Justify your answer. With current software engineering practices it will be very rare that a program is larger that 32K—you could still jump from the start to the end of the subprogram.

Problem 12. Consider the CPUs with the following instruction set characteristics:

- **CompA:** Instruction ADD, SUB, MUL, DIV have 3 memory addresses, the MOV instruction has 2 (target, source);
- **CompB:** Instruction ADD, SUB, MUL, DIV have 2 memory addresses[1], the MOV instruction has 2 as well (target, source);
- **CompC:** Instruction ADD, SUB, MUL, DIV have 2 registers (R0, R1, ... R15) the MOV instruction has one register and one memory address (target, source);

[1] ADD M1, M2 means M1 = M1 + M2

The size of each instruction is determined as follows:

- Opcode – 1 byte
- Memory Address – 4 bytes
- Register - 4 bits

Write assembly code for each of the CPU's given above to make the following calculation:

$$X = \frac{(A - B)(C - D)}{A + B}$$

(a) How much memory does the code for each program take? Show your work.

(b) How many memory accesses are required? Include the loading of the code (assume that each memory access will get 4 bytes.)

Problem 13. Given a RAID 3 (bit-interleaved parity) with 9 disks—8 for data one for parity.

(a) Explain how a single bit error in any byte can be detected.

(b) Explain how errors can be corrected—assume one of the disks has crashed.

(c) Why does this RAID configuration perform poorly for high I/O request rates?