Introduction to Computer Architecture

Assignment 2

Due November 09, 2021

*****CHAPTER 1 - FUNDAMENTAL*****

01. [2 = 1 x 2]

a. Please describe the two kinds of parallelism in applications.

b. Please describe the four major ways to exploit the preceding two kinds of application parallelism.

02. [2 = 1 x 2]

Some microprocessors today are designed to have adjustable voltage, so a 15% reduction in voltage may result in a 15% reduction in frequency.

What would be the impact on dynamic energy and on dynamic power?

03. [2 = 1 x 2]

Please explain how the following techniques help modern microprocessors improve energy efficiency.

a. dynamic voltage-frequency scaling (DVFS);

b. overclocking.

04. [2 = 1 x 2]

Assume a disk subsystem with the following components and MTTF:

- 10 disks, each rated at 1,000,000-hour MTTF
- 1 ATA controller, 500,000-hour MTTF
- 1 power supply, 200,000-hour MTTF
- 1 fan, 200,000-hour MTTF
- 1 ATA cable, 1,000,000-hour MTTF

a. Using the simplifying assumptions that the lifetimes are exponentially distributed and that failures are independent, compute the MTTF of the system as a whole.

b. Using the components and MTTFs from above, calculate the reliability after adding one redundant power supply.

05. [3 = 1 + 2]

a. Please describe the equation of the Amdahl’s law;

b. Use the Amdahl’s law to reason about whether processor performance can be indefinitely improved. More specifically, given the faction of professor operations that can be enhanced, reason about why it is this enhancement fraction that decides the ultimate performance improvement.

06. [2 = 1 x 2]

Suppose we have made the following measurements:

- Frequency of FP operations = 25%
- Average CPI of FP operations = 4.0
- Average CPI of other instructions = 1.33
- Frequency of FPSQR = 2%
- CPI of FPSQR = 20

Assume that the two design alternatives are to decrease the CPI of FPSQR to 2 or to decrease the average CPI of all FP operations to 2.5.

Compare these two design alternatives using the processor performance equation.

07. [4 = 2 x 2]

You are designing a system for a real-time application in which specific deadlines must be met. Finishing the computation faster gains nothing. You find that your system can execute the necessary code, in the worst case, twice as fast as necessary.

a. How much energy do you save if you execute at the current speed and turn off the system when the computation is complete?

b. How much energy do you save if you set the voltage and frequency to be half as much?

08. [4 = 1 x 4]

Your company has just bought a new processor, and you have been tasked with optimizing your software for this processor. You will run two applications on this dual core, but the resource requirements are not equal. The first application requires 80% of the resources, and the other only 20% of the resources. Assume that when you parallelize a portion of the program, the speedup for that portion is 2.

a. Given that 40% of the first application is parallelizable, how much speedup would you achieve with that application if run in isolation?

b. Given that 99% of the second application is parallelizable, how much speedup would this application observe if run in isolation?

c. Given that 40% of the first application is parallelizable, how much overall system speedup would you observe if you parallelized it?

d. Given that 99% of the second application is parallelizable, how much overall system speedup would you observe if you parallelized it?

09. [5 = 1 x 5]

When parallelizing an application, the ideal speedup is speeding up by the number of processors. This is limited by two things: percentage of the application that can be parallelized and the cost of communication. Amdahl’s law takes into account the former but not the latter.

a. What is the speedup with *N* processors if 80% of the application is parallelizable, ignoring the cost of communication?

b. What is the speedup with 8 processors if, for every processor added, the communication overhead is 0.5% of the original execution time.

c. What is the speedup with 8 processors if, for every time the number of processors is doubled, the communication overhead is increased by 0.5% of the original execution time?

d. What is the speedup with *N* processors if, for every time the number of processors is doubled, the communication overhead is increased by 0.5% of the original execution time?

e. Write the general equation that solves this question: What is the number of processors with the highest speedup in an application in which *P*% of the original execution time is parallelizable, and, for every time the number of processors is doubled, the communication is increased by 0.5% of the original execution time?

APPENDIX A - INSTRUCTION

10. [3 = 1 x 3]

For each of R, I, S, U type RISC-V instructions:

a. draw its encoding format with each field marked;

b. explain the meanings of each field;

c. provide an example instruction and explain its operations.

11. [7 = 1 x 7]

Please describe the meaning of the following addressing modes and provide an example instruction for each.

a. Immediate

b. Displacement

c. Register indirect

d. Direct or absolute

e. Memory indirect

f. Scaled

g. PC-relative or position independence

12. [6 = 2 x 3]

For the following assume that values A, B, and C reside in memory. Also assume that instruction operation codes are represented in 8 bits, memory addresses are 64 bits, and register addresses are 6 bits.

For each instruction set architecture shown in Figure A.2 (*stack*, *accumulator*, *register-memory*, *register-register*), how many addresses, or names, appear in each instruction for the code to compute C = A + B, and what is the total code size?

13. [6 = 2 x 3]

Consider the following fragment of C code:

```
for (i = 0; i <= 100; i++)
{ A[i] = B[i] + C; }
```

Assume that A and B are arrays of 64-bit integers, and C and i are 64-bit integers. Assume that all data values and their addresses are kept in memory (at addresses 1000, 3000, 5000, and 7000 for A, B, C, and i, respectively) except when they are operated on. Assume that values in registers are lost between iterations of the loop.

a. Write the code for RISC-V.
b. How many memory-data references will be executed?
c. What is the code size in bytes?

14. [2 = 2 x 1]

Our favorite program runs in 10 seconds on compute A, which has a 4 GHz clock. We are trying to help a computer designer build a computer, B, that will run this program in 6 seconds. The designer has determined that a substantial increase in the clock rate is possible, but this increase will affect the rest of the CPU design, causing computer B to require 1.2 times as many clock cycles as computer A for this program.

What clock rate should we tell the designer to target?

APPENDIX C - PIPELINE

15. [5 = 1 x 5]

Please list the types of exceptions that might arise in various stages of a five-stage pipeline?

16. [2 = 1 x 2]

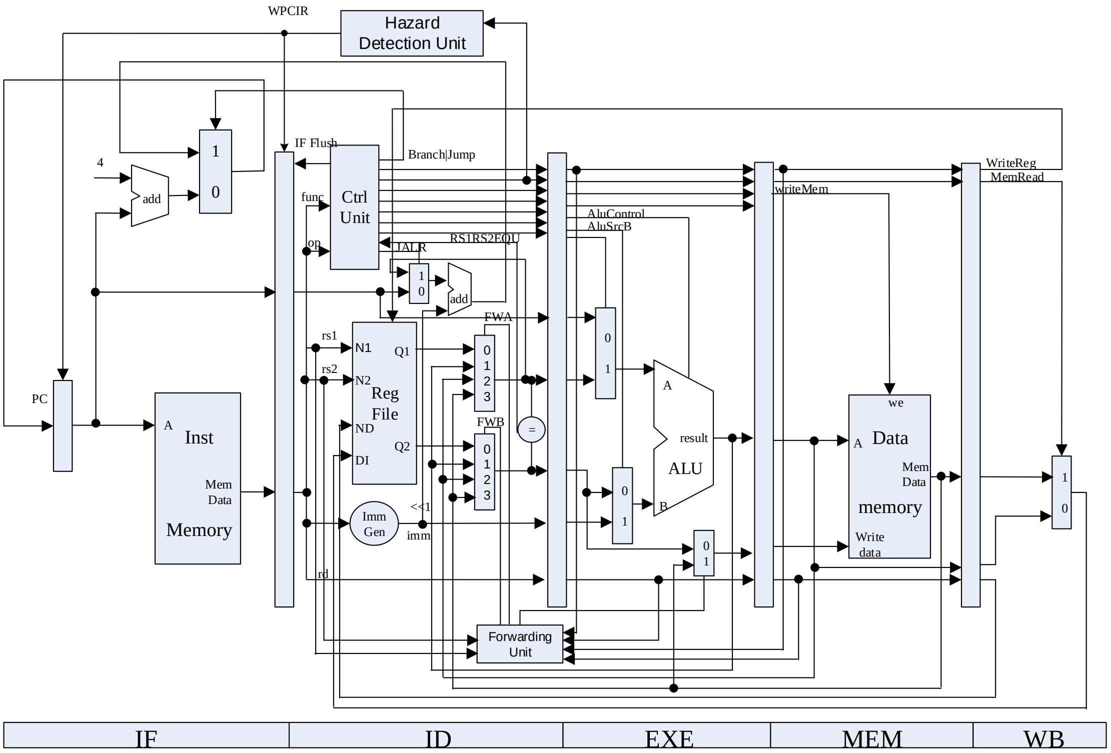
a. Please describe the two-memory technique that eases pipeline design.
b. Use an example to justify the benefit of it for improving pipeline performance.

17. [2]

Analyze why ideal pipelining with equal-length pipe stages yields the highest speedup.

18. [5 = 1 x 5]

Based on the following pipelined datapath, draw the links taken by each instruction:
add
ld
sd
beq
jal



APPENDIX B & CHAPTER 2 - MEMORY HIERARCHY

19. [5]

For the code below, assume we have an 8 KB direct-mapped data cache with 16-byte blocks, and it is a write-back cache that does write allocate. The elements of a and b are 8 bytes long since they a double-precision floating-point arrays. There are 3 rows and 100 columns for a and 101 rows and 3 columns for b. Assume they are not in the cache at the start of the program.

Determine the number of cache misses and which accesses cause them for the following codes with or without prefetching.

```
for (i = 0; i < 3; i = i+1)
        for (j = 0; j < 100; j = j+1)
             a[i][j] = b[j][0] * b[j+1][0];
```

```
for (j = 0; j < 100; j = j+1) {
        prefetch(b[j+7][0]);
        /* b(j,0) for 7 iterations later */
        prefetch(a[0][j+7]);
        /* a(0,j) for 7 iterations later */
        a[0][j] = b[j][0] * b[j+1][0];};
for (i = 1; i < 3; i = i+1)
        for (j = 0; j < 100; j = j+1) {
                prefetch(a[i][j+7]);
                /* a(i,j) for +7 iterations */
                a[i][j] = b[j][0] * b[j+1][0];}
```

20. [5 = 1 + 2 + 2]

The LRU replacement policy is based on the assumption that if address A1 is accessed less recently than address A2 in the past, then A2 will be accessed again before A1 in the future. Hence, A2 is given priority over A1. Discuss how this assumption fails to hold when the loop larger than the instruction cache is being continuously executed.

For example, consider a fully associative 128-byte instruction cache with a 4-byte block (every block can exactly hold one instruction). The cache uses an LRU replacement policy.

a. What is the asymptotic instruction miss rate for a 64-byte loop with a large number of iterations?
b. Repeat part (a) for loop sizes 192 bytes and 320 bytes.
c. If the cache replacement policy is changed to most recently used (MRU) (replace the most recently accessed cache line), which of the three above cases (64-, 192-, or 320-byte loops) would benefit from this policy?

21. [5 = 1 + 1 + 1 + 2]

You are building a system around a processor with in- order execution that runs at 1.1 GHz and has a CPI of 0.7 excluding memory accesses. The only instructions that read or write data from memory are loads (20% of all instructions) and stores (5% of all instructions). The memory sys- tem for this computer is composed of a split L1 cache that imposes no penalty on hits. Both the I-cache and D-cache are direct mapped and hold 32 KB each. The I-cache has a 2% miss rate and 32-byte blocks, and the D-cache is write- through with a 5% miss rate and 16-byte blocks. There is a write buffer on the D-cache that eliminates stalls for 95% of all writes. The 512 KB write-back, unified L2 cache has 64-byte blocks and an access time of 15 ns. It is connected to the L1 cache by a 128-bit data bus that runs at 266 MHz and can transfer one 128-bit word per bus cycle. Of all memory references sent to the L2 cache in this system, 80% are satisfied without going to main memory. Also, 50% of all blocks replaced are dirty. The 128-bit-wide main memory has an access latency of 60 ns, after which any number of bus words may be transferred at the rate of one per cycle on the 128-bit-wide 133 MHz main memory bus.

a. What is the average memory access time for instruction accesses?
b. What is the average memory access time for data reads?
c. What is the average memory access time for data writes?
d. What is the overall CPI, including memory accesses?

22. [5 = 1 + 2 + 2]

You are trying to appreciate how important the principle of locality is in justifying the use of a cache memory, so you experiment with a computer having an L1 data cache and a main memory (you exclusively focus on data accesses). The latencies (in CPU cycles) of the different kinds of accesses are as follows: cache hit, 1 cycle; cache miss, 105 cycles; main memory access with cache disabled, 100 cycles.

a. When you run a program with an overall miss rate of 5%, what will the average memory access time (in CPU cycles) be?
b. Next, you run a program specifically designed to produce com- pletely random data addresses with no locality. Toward that end, you use an array of size 256 MB (all of it

fits in the main memory). Accesses to random elements of this array are continuously made (using a uniform random number generator to generate the elements indices). If your data cache size is 64 KB, what will the average memory access time be?

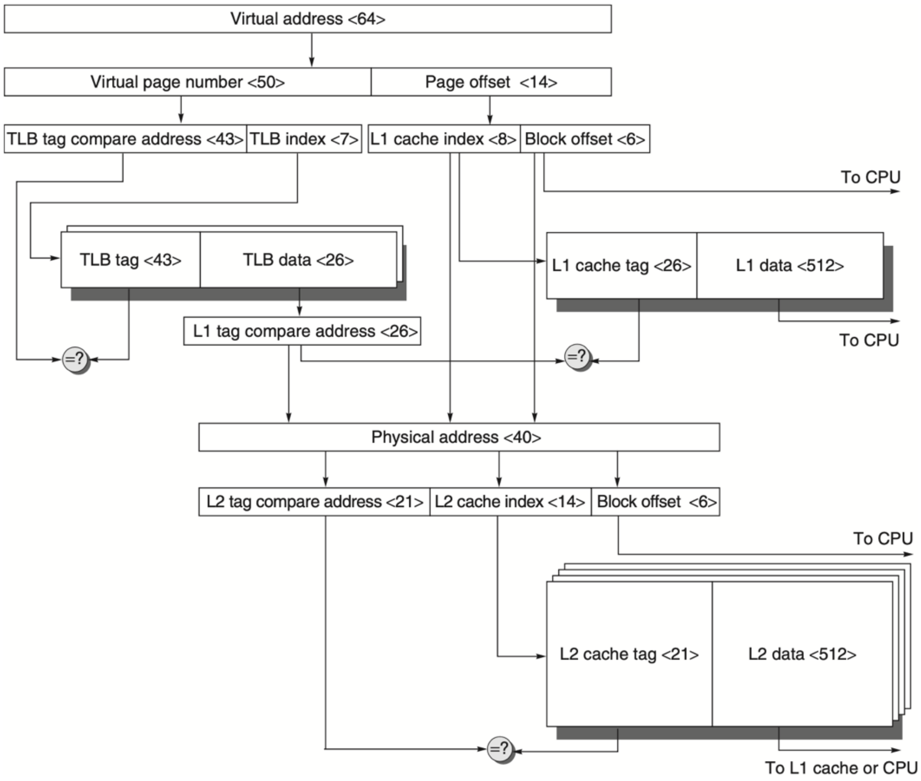
c. You observed that a cache hit produces a gain of 99 cycles (1 cycle vs. 100), but it produces a loss of 5 cycles in the case of a miss (105 cycles vs. 100). In the general case, we can express these two quantities as G (gain) and L (loss). Using these two quantities (G and L), identify the highest miss rate after which the cache use would be disadvantageous.

23. [5 = 2 + 2 + 1]

a. Describe the address translation process on the following memory system.

b. Consider a virtually indexed and physically tagged direct-mapped cache. Please reason about why the cache size should be no larger than the page size.

c. Please generalize the above conclusion to an n-way set associative cache: Given a virtually indexed and physically tagged n-way set associative cache, the cache size is no larger than n times of the page size.



24. [5]

Consider a 16-way set associative cache:
Data words are 64 bits long;
Words are addressed to the half-word;
The cache holds a 2 Mbytes of data;
Each block holds 16 data words;
Physical addresses are 64 bits long.

How many bits of tag, index, and offset are needed to support references to this cache?

25. [6] Commit & Enjoy

[requirements: a. append an extra blank page after your feedback for possible comments; b. submit an extra copy of Q25 via course.zju.edu.cn for ease of (anonymous) compilation; photocopy of a handwritten version is also welcome.]

I hope you really enjoyed the class (at least for some moments) so far. You decided to study Computer Architecture with me among several instructors. I truly appreciate your trust and your continuing support, understanding, tolerance, and cooperation. Ever since we met, I keep working on how to provide you with a satisfactory learning experience in return.

For almost all previous editions of this course, an assignment question sought to solicit thoughts and suggestions from the participants. For example, do you gradually understand strategies or things from different perspectives and weigh their tradeoffs? What do you think is the real challenge for you to learn this course? Do you consider interactions in class helpful? What held you back when you were trying to ask or answer questions in class? What suggestions (for better learning this course) would you like to provide to other students? What help or assistance would like from me or other students? All their thoughtful and constructive feedback has encouraged us toward a more rewarding computer architecture class.

This time, I would like you to introspect beyond this course. Taking a course should be rewarding in many possible ways. As we discussed in the first lecture session, even if you may barely practice computer architecture principles after the class ends, if whatever you learn through this class---whether it be computer architecture per se, a philosophy it implies, a study tip or inspirational quote we share, or a new friend you know---keeps driving you toward your greater self, this course fulfills its mission.

In particular, you are highly expected to develop a clear vision and be determined to strive for it. Most students in the computer architecture class are junior. The third year cannot be more decisive for your future. If you aim to work in an IT company after graduation, start practicing skills that help you ace the interview. If you plan to pursue a higher degree, join a research lab to cultivate research experience and application materials. If you have not decided, do both if your time and energy permit, see which one you really prefer. Try to make your decision based on your own goal and experience. Work with people who keep your interest and passion alive. It is really, really important to identify a right role model to emulate and learn from. It is also highly decisive to focus on what really is essential, especially given that some of constantly emerging paradigms might turn into hypes and fade away. Stick to golden rules that should be objective during the selection. It should not be simply about what one claims to be or what one claims others less so. “See the world not as it is, but as it should be.” Think of the proverb “shallow brooks babble loudest, still waters run deep” once in a while. (Or as sg put it: “shallow water hualahuala, deep water kckc.”) In return, be part of the initiative that motivates yourself and people around you.

Meanwhile, if you ever consider assistance from me as possibly helpful, whether course related or beyond, never hesitate to reach out. As I proudly claim on my webpage, I am always proud to be part of the journey for someone to excel. And I am pretty sure that I would also love to be part of yours as well.

???Therefore, with this question, I would like you to share your thoughts on your goal and plan. For example, what is your goal after graduation? What is your plan to achieve that goal and what challenges might be involved? How do you manage to be motivated and determined? What helpful advices or suggestions did you get from senior students and professors? What suggestions would you like to offer peers with similar goals? Or, it is possible that you are still finding your goals. Don’t rush and take your time. Being at such a young age, you have infinite possibilities to live your dreams. “Commit to something and commit hard. Doesn’t matter if you switch later. It’s easier to prove yourself if you’ve had to do it once before.”

“Where are you? Here.
  What time is it? Now.
  What are you? This moment.”

“Old urges continue to arise, but urges do not matter; only actions do.”
“A warrior is as a warrior does.”

“While they are deciding, make even more art.”

So pleeeeease, live a life you deeply enjoy and will remember.

If not now, when?