CDT204 – Computer Architecture

Date: Aug. 17th 2023

Time: 8:30 – 12:30

Help: Small calculator is allowed in the exam

The exam has 37 points. And 4 BONUS points. The grades will be awarded as follows:

- 3 : 19 Points
- 4 : 27 Points
- 5 : 33 Points

Important Notes:

- Give as full an answer as possible to obtain full marks. All calculations, approximations, assumptions and justifications must be reported for full credit unless stated otherwise. Please use figures and examples to clarify.
- If you do not understand a question clearly, you are allowed to call the teacher and ask.
- Write the question and part number on each page clearly.
- Answer each question on a separate page.
- In case you might have forgotten:

  1GB = 2^{10}MB = 2^{20}KB = 2^{30}B

  1 sec = 10^3 ms = 10^6 μs = 10^9 ns = 10^{12} ps

Live long and prosper

Task 1 - General (6p)

This is a multichoice question. Use the answer sheet at the end of the exam to mark the correct answer. Please note that each part of this question has exactly one correct answer. You will receive +1 point for the correct answer and -1/4 point for a wrong answer, and 0 points for no answer.

A. Suppose you guess randomly on one of the questions on this exam. Assuming there is exactly one correct answer (out of five options), what is the expected number of points you will receive for that one question?
   a. 1/5 point.
   b. -1/5 point.
   c. 1/4 point.
   d. 0 points.
   e. It depends on the question.

B. A program spends 75% of its time doing multiply instructions. If the part of the hardware that performs the multiplication is sped up by 3x with a new design, how much faster does the application run?
   a. 2x
   b. 3x
   c. 9/4x
   d. 2.25x
   e. 3.75x

C. What is the limit on the maximum speedup if the multiplier in part B was infinitely fast?
   a. ∞
   b. 3x
   c. 4x
   d. 75x
   e. 16x

D. Consider that 10100110 represents an 8-bit signed integer represented in two's complement. What is its value in base 10?
   a. -38
   b. -90
   c. 166
   d. -166
   e. 89

E. Which of the following is correct?
   a. For forwarding we only need to look into the data available in WB stage.
   b. A dynamic branch predictor is always better than a static branch predictor.
   c. The average number of memory accesses per instruction is less than one, if not all instructions are loads or stores.
   d. Increasing the number of pipeline stages, normally leads to an increase in latency.
   e. A perfect branch predictor combined with data forwarding would allow a processor to always keep its pipeline full.

F. Consider $(F1A)_{16}$ as an unsigned integer value. What is its value in base 10?

a. 267
b. 15110
c. 26
d. -266
e. 3866

Task 2 - Performance (6p)

Execution time can be measured as:

$$\boldsymbol{Execution\ Time = Instruction\ Count \times Avergae\ CPI\ \times Cycle\ Time}$$

For each of the design changes presented at the end of the task, discuss the effect of the change on the following factors:

1. Instruction Count (IC),
2. CPI,
3. Cycle Time (CT).

Say whether, all other things being equal, the change cannot affect the factor, will tend to increase it, or decrease it. Give a justification or explanation for your choice. To get full credit for any part, you need to get ***both*** the type of change, if any, and the reason correct.

A. Design change: More pipeline stages. (3p)
B. Design change: A larger cache. (3p)

Task 3 – Instruction Sets (8p)

Consider a hypothetical machine called SIC, for Single Instruction Computer. As its name implies, SIC has only one instruction: *subtract and branch if negative*, or sbn for short. The sbn instruction has three operands, each consisting of the address of a word in memory:

```
sbn a, b, c     #Mem[a] = Mem[a] - Mem[b]; if (Mem[a]<0) go to c
```

The instruction will subtract the number in memory location b from the number in location a, and place the result back in a, overwriting the previous value. If the result is greater than or equal to 0, the computer will take its instruction just after the current instruction. If the result is less than 0, the next instruction is taken from memory location c. SIC has no registers and no instructions other than sbn.

A. Identify what the following code does and write the equivalent code in C (The ".+1" in the code means "The address after this one") (3p).

```
start: sbn temp, temp, .+1
       sbn temp, a, .+1
       sbn b, b, .+1
       sbn b, temp, .+1
```

B. Write a SIC program to add a and b, leaving the result in a and leaving b unmodified. (5p)

Task 6 – Multiprocessing (4p + 4p BONUS)

Ray tracing is a technique in computer graphics that tries to render a scene using physics of light. To calculate the color of each pixel in a scene, several rays (representing rays of light) are projected from the pixel to the scene (These are called *primary rays*) and their interaction with the scene objects and light sources is used to calculate the color of that pixel. For each primary ray we do the following:

- Test the ray against all the objects in the scene (this is called *intersection test*). We then pick the closest hit point.
- For each hit point we create two types of *secondary rays*: *shadow* rays and *reflection* ray. The reflection ray represents how the ray will be reflected from the hit point. The shadow rays represent the direction from the hit point to each light source. So, for each hit point we will have <u>one reflection ray</u> and as many <u>shadow rays as light sources</u>.
- We then perform another set of intersection tests for these newly created rays.
- This process can continue recursively, but for this task we ignore that. So, the <u>reflection rays and shadow rays do not create new secondary rays</u>.

Consider the following scene:

- Two light sources.
- 10000 primitive objects which are all reflective.
- Screen resolution of full HD (1920x1080).

Also assume that:

- 70% of primary rays, 80% of reflection rays and 40% shadow rays generate an intersection.
- For each pixel 4 primary rays are fired.
- We test each ray against all the objects.

a) **BONUS**: What are the total number of ray intersection tests that we need to perform in order to render the scene? (2p)

Now consider the following systems (non-intersection code runs on one core and cannot be parallelized):

	System	Frequency (GHz)	No. Cores	Percentage of non-intersection code	No. intersection tests per 100 CPU cycles for each core
A	CPU renderer	4.5	16	20%	1
B	CPU renderer using AVX	3.4	16	20%	16
C	GPU renderer	0.5	2048	40%	1

If you could not solve part (a) then use 10^{12} as the number of total rays required for the following parts, otherwise use your result from part A.

b) How many ray intersection tests can each of these systems perform in one second? (3p)

c) What is the framerate for each of these systems? (1p)

d) **BONUS**: If we try to create a system that uses both systems B and C (runs both on CPU and GPU), what do <u>you</u> think will be the most important factor that affects the performance? Explain. (1p)

Task - Cache (13p)

We are going to microbenchmark the cache hierarchy of a computer with the following two codes. The array `data` contains 32-bit unsigned integer values. For simplicity, we consider that accesses to the array `latency` bypass all caches (i.e., `latency` is not cached). `timer()` returns a timestamp in cycles. The cache hierarchy has two levels. L1 is a 4KB set associative cache.

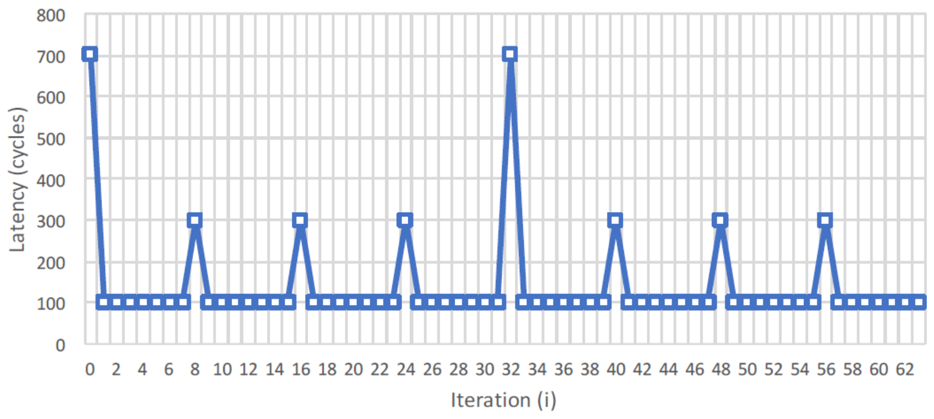
Code (1)

```
j = 0;
for (i=0; i<size; i+=stride){
    start = timer();
    d = data[i];
    stop = timer();
    latency[j++] = stop - start;
}
```

Code (2)

```
for (i=0; i<size1; i+=stride1){
    d = data[i];
}
j = 0;
for (i=0; i<size2; i+=stride2){
    start = timer();
    d = data[i];
    stop = timer();
    latency[j++] = stop - start;
}
```

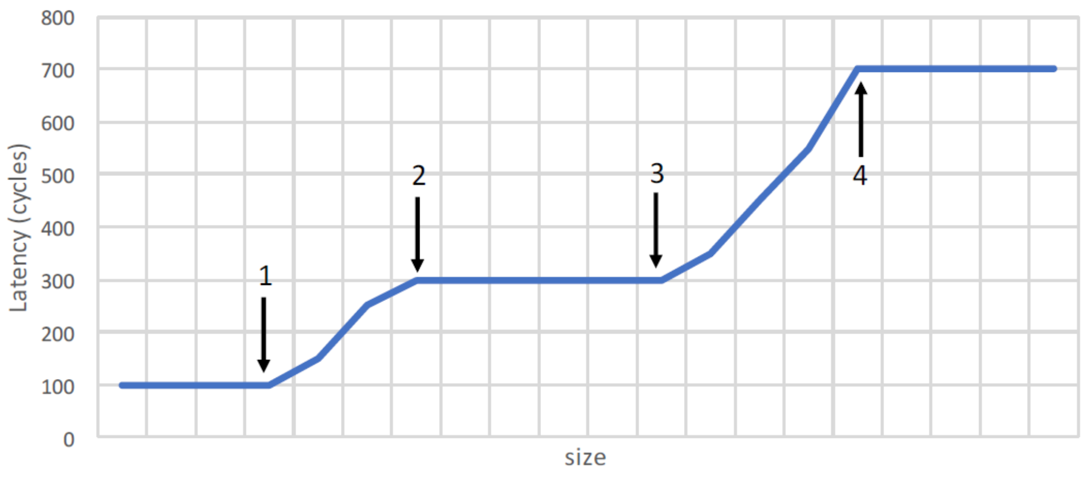
A. When we run code (1), we obtain the latency values in the following chart for the first 64 reads to the array `data` (in the first 64 iterations of the loop) with `stride` equal to 1. What are the cache block sizes in L1 and L2? (4p)



B. Using code (2) with `stride1 = stride2 = 32`, `size1 = 1056`, and `size2 = 1024`, we observe `latency[0] = 300` cycles. However, if `size1 = 1024`, `latency[0] = 100` cycles. What is the maximum number of ways in L1? (4p)

C. Now we carry out two consecutive runs of code (1) for different values of `size` (so for each `size` we run it twice and we test many different sizes). In the first run, `stride` is equal to 1. In the second run, `stride` is equal to 16. We ignore the latency results of the first run and average the latency results of the second run. We obtain the following graph. What do the following parts of the graph represent? (5p)

   a. Before arrow 1.
   b. Between arrow 1 and arrow 2.
   c. Between arrow 2 and arrow 3.
   d. between arrow 3 and arrow 4.
   e. After arrow 4.



Answer Sheet For Task 1

Write Your exam registration code here:

	A	B	C	D	E
Task 1-A					
Task 1-B					
Task 1-C					
Task 1-D					
Task 1-E					
Task 1-F					