NANYANG TECHNOLOGICAL UNIVERSITY

SEMESTER 1 EXAMINATION 2019-2020

CE3001/CZ3001 – ADVANCED COMPUTER ARCHITECTURE

Nov/Dec 2019 Time Allowed: 2 hours

INSTRUCTIONS

1. This paper contains 4 questions and comprises 7 pages.
2. Answer **ALL** questions.
3. This is a closed-book examination.
4. All questions carry equal marks.

---

1. (a) Listing Q1 shows a code segment that is intended to be executed in a 5-stage pipelined MIPS processor. No data forwarding is allowed but write-back and register-read operations of different instructions can be performed in the same clock cycle. Note that program counter is updated with the branch target address at the decode stage. Let the initial values be `$s5=0x0000` and `$s6=0x0036` (`bgtz`: *branch if greater than 0*).

Listing Q1

```
I1   loop:  lw    $t1, 0($s5)
I2          addi  $t5, $t1, #0x0004
I3          sw    $t5, 0($s5)
I4          addi  $s5, $s5, #0x0008
I5          addi  $s6, $s6, #0xFFFF
I6          bgtz  $s6, loop
     finish
```

(i) Identify the data dependencies in the code segment in Listing Q1 and calculate the steady state CPI needed for the execution of the code if data forwarding is NOT allowed. Calculate the number of loop iterations.

(6 marks)

Note: Question No. 1 continues on Page 2

(ii) Listing Q1 is now intended to be executed in a two-way superscalar processor. One way in the superscalar processor is reserved for Load/Store instructions and the other way for the rest of the instructions. Find the CPI of the code segment for the two-way superscalar architecture after necessary reordering of the instructions for reducing the number of stall cycles to the minimum.

(5 marks)

(iii) Perform unrolling of the loop in Listing Q1 by a factor of 4 and do the necessary reordering of instructions for reducing the number of stall cycles to the minimum. Find the CPI achieved by such loop unrolling and instruction reordering. You are allowed to use new temporary registers to get rid of hazards.

(8 marks)

(b) An engineer in company S improves the performance of a machine by considering an enhancement E that is applied to 60% of the original instructions, and speeds up those instructions by a factor of 4. The management of company S has concerns on the complexity and cost effectiveness of E and suggests that the engineer should consider an alternative enhancement E'. Enhancement E', if applied only to some (as yet unknown) fraction of the original instructions, would speed them up by a factor of 2. Determine what percentage of all the instructions should be optimized using enhancement E' in order to achieve the same overall speedup as obtained using enhancement E.

(6 marks)

2. (a) Figure Q2a shows the block diagram of a single-cycle datapath for the implementation of R-format and I-format instructions including arithmetic and logic instructions, load, store and conditional branch instructions. Propagation delays of the datapath components at V=2.5 volt are given in Table Q2.

Table Q2

PCin→ PCout	ALU	Adder (for branch)	Shifter	Mux	AND gate	REG (R/W)	Sign-extend
100ps	250ps	200ps	50ps	10ps	5ps	200ps	50ps

PC++	Control logic	D-mem (R/W)	I-mem (R)
200ps	50ps	500ps	500ps

Note: Question No. 2 continues on Page 3

PC++ stands for the delay for PC increment. I-MEM (R) stands for the time for reading instruction memory, and D-MEM (R/W) stands for the time required for read or write operation on data memory. REG (R/W) stands for the time required for read or write operation on register file.

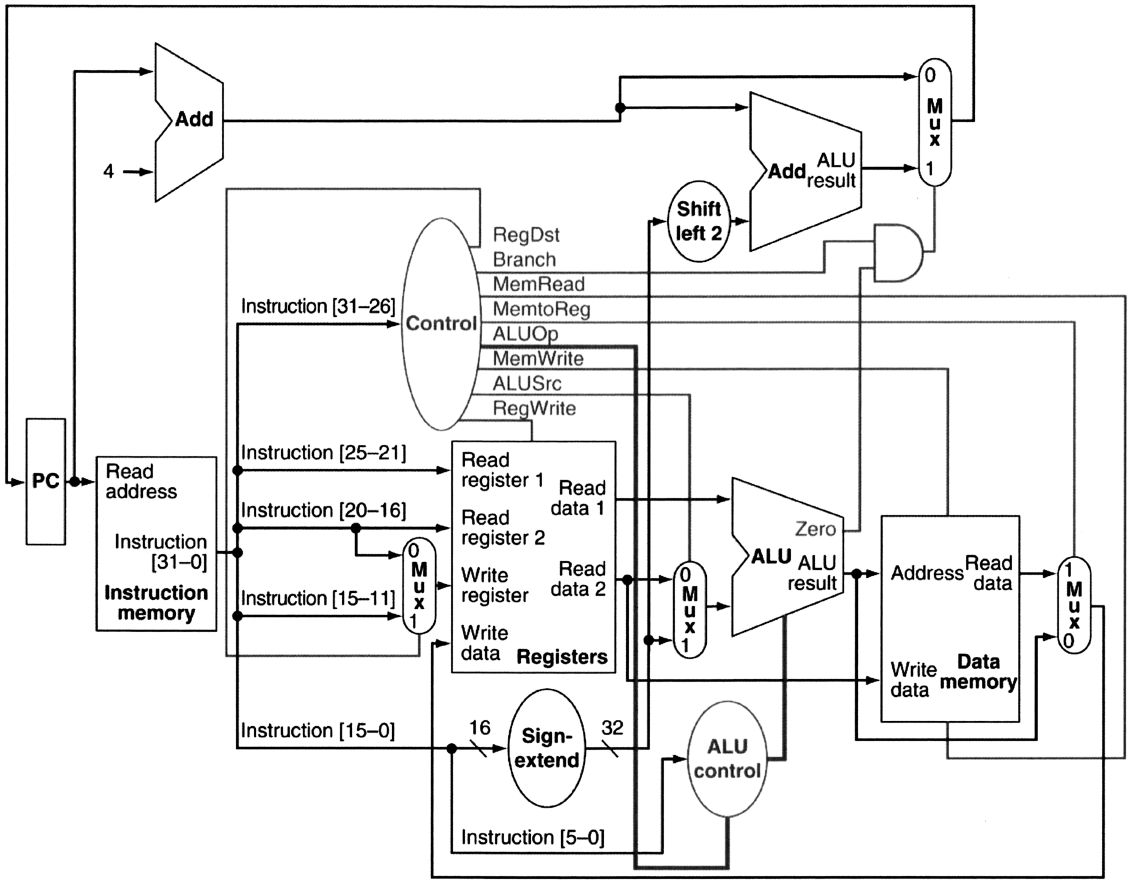

Figure Q2a

(i) Explain the operations of SW and BEQ instructions, respectively, by providing details on the operations resulting in the maximal latency. Use examples if needed.

(8 marks)

(ii) Find the maximum possible frequency at which the given datapath can function if the operating voltage V=2.5 volt.

(3 marks)

(iii) Find the maximum possible operating frequency of the given datapath at the operating voltage V=1 volt.

(3 marks)

Note: Question No. 2 continues on Page 4

(b) State the two addressing modes used for conditional and unconditional branch instructions, respectively. Determine the minimum and maximum possible addresses to which the two (conditional and unconditional branch) instructions can branch, respectively, assuming the PC values of the two branch instructions are both 0x20006F0C.

(6 marks)

(c) Assume a branch predictor uses the 2-bit prediction scheme as shown in Figure Q2b. Consider a branch with the following sequence of actual outcome: 'T T T T T T T T, N N T N T T', where T indicates the branch is taken and N indicates the branch is not taken. What is the prediction accuracy for the last 6 occurrences of this branch if the 2-bit branch predictor is applied?

(5 marks)

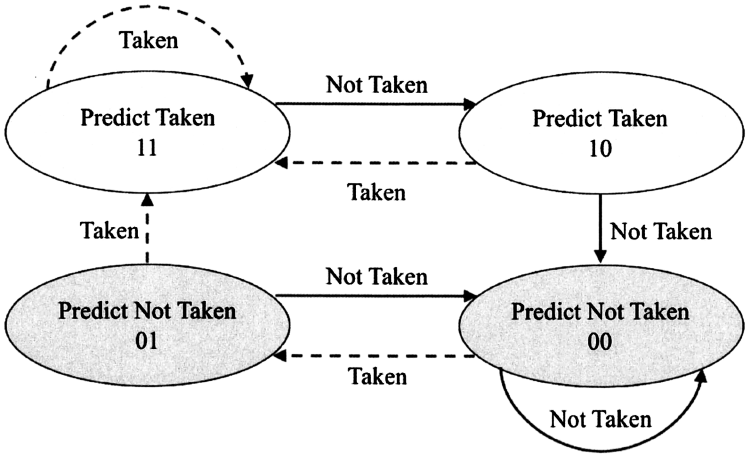

Figure Q2b

3 (a) Name the four types of processor architectures in the processor taxonomy according to Flynn's classification. Which one(s) among them provide(s) better support to the data-level parallelism?

(6 marks)

(b) Tables Q3a and Q3b show the cache miss rates and average memory access time for different cache sizes and block sizes of a 1-level cache. Assume a cache hit takes 1 clock cycle and the cache miss penalties (for memory access) with different block sizes are shown in Table Q3b.

(i) Compute the missing entries X, Y and Z in Tables Q3a and Q3b.

(6 marks)

Note: Question No. 3 continues on Page 5

Table Q3a: Cache Miss Rate v.s. Cache Size and Block Size

Block Size (Bytes)	Cache Size			
	4KB	16KB	64KB	256KB
16	8.57%	3.94%	2.04%	1.09%
64	7.00%	2.64%	X%	0.51%
256	9.51%	3.29%	1.15%	0.49%

Table Q3b: Average Memory Access Time v.s. Cache Size and Block Size

Block Size (Bytes)	Miss Penalty	Cache Size			
		4KB	16KB	64KB	256KB
16	82	8.03	4.23	2.67	1.89
64	88	7.16	Y	1.93	Z
256	112	11.65	4.69	2.29	1.55

(ii) Given a fixed cache size, the miss rate first decreases when the block size increases, but then increases when the block size increases further in most cases. Explain this phenomenon shortly by referring to the reasons for cache misses.

(6 marks)

(iii) Between the cache miss rate and the average memory access time, are they consistently changing among different cache sizes when the block size is increased from 16 to 64 to 256 Bytes? Which one plays a more important role in maximizing the cache performance? Please elaborate on your answer shortly.

(7 marks)

4 (a) State the main differences between the typical architecture designs of a CPU and a GPU.

(6 marks)

(b) The code snippet in Figure Q4a shows a C program that uses a CUDA kernel `saxpy()` to compute the SAXPY (Single precision A.X plus Y) operation:

$$\mathbf{Y} = A\mathbf{X} + \mathbf{Y}$$

where `A` is a scalar, **`X`** and **`Y`** are vectors, each consisting of `N` floating-point numbers.

Note: Question No. 4 continues on Page 6

```
Line
1   __global__
2   void saxpy(int n, float a, float *x, float *y){
3     int i = blockIdx.x * blockDim.x + threadIdx.x;
4     if (i<n)
5        y[i] = a*x[i] + y[i];
6     }
7
8   int main(void){
9     int N = 1000000; // N = 1 million
10    float A = 3.0;
11    X = (float *)malloc(N*sizeof(float));
12    Y = (float *)malloc(N*sizeof(float));
13    // get values of vectors X and Y
:          :
:          :
n     saxpy<<<.....>>>(N, A, d_X, d_Y);
:          :
:     return 0
n+k   }
```

Figure Q4a

(i) Complete the code shown in Line *n* if the number of threads per block is set as 256.

(4 marks)

(ii) Describe (no program code is needed) what other operations are needed to be performed in the `main()` function in order for the kernel `saxpy()` to execute correctly and return the values of vector **Y**.

(7 marks)

(c) Figure Q4b shows a CUDA kernel that computes the dot product of two vectors **A** and **B** and outputs a scalar value `C`:

$$C = \mathbf{A} \bullet \mathbf{B} = \sum_{i=1}^{N} a_i b_i = a_1 b_1 + a_2 b_2 + \cdots + a_N b_N$$

(i) If `N` = 256, explain how the kernel should be launched, in terms of the number of block(s) and thread(s).

(4 marks)

Note: Question No. 4 continues on Page 7

(ii) What is the potential problem of the code in Figure Q4b and how should it be fixed?

(4 marks)

```
Line
1   __global__
2   void dotprod(int *a, int *b, int *c ) {
3    __shared__ int temp[256];
4    int i = blockIdx.x * blockDim.x + threadIdx.x;
5    temp[i] = a[i]*b[i];
6
7    // Thread 0 sums the pairwise products
8    if (i == 0) {
9      int sum = 0;
10     for (int j = 0; j < N; j++)
11       sum += temp[j];
12     *c = sum;
13     }
14  }
```

Figure Q4b

END OF PAPER

CE3001 ADVANCED COMPUTER ARCHITECTURE
CZ3001 ADVANCED COMPUTER ARCHITECTURE

Please read the following instructions carefully:

1. **Please do not turn over the question paper until you are told to do so. Disciplinary action may be taken against you if you do so.**

2. You are not allowed to leave the examination hall unless accompanied by an invigilator. You may raise your hand if you need to communicate with the invigilator.

3. Please write your Matriculation Number on the front of the answer book.

4. Please indicate clearly in the answer book (at the appropriate place) if you are continuing the answer to a question elsewhere in the book.