NATIONAL UNIVERSITY OF SINGAPORE

SCHOOL OF COMPUTING

MIDTERM ASSESSMENT
AY2019/20 Semester 1

CS2100 — COMPUTER ORGANISATION

October 2019 Time Allowed: **1 hour**

INSTRUCTIONS

1. This question paper contains **SEVENTEEN (17)** questions (including the bonus question and a check question) and comprises **TEN (10)** printed pages.
2. Page 9 is a blank page. Page 10 contains the MIPS reference data.
3. You are provided with:
    a) An **Answer Sheet**, comprising **TWO (2)** printed page
    b) An **OCR form**
4. Write your **Name**, **Student Number** and **Tutorial Group Number** on the Answer Sheet with a **PEN**.
5. Shade **and write** your Student Number on the OCR form.
6. Answer Question 1 to 12 by shading on the OCR Form with **at least 2B pencil**.
7. Answer Question 13 to 17 within the space provided on the Answer Sheet. You may write in pencil.
8. Submit only the Answer Sheet and the OCR form at the end of the test.
9. Maximum score is **40 marks**.
10. This is a **CLOSED BOOK** test. However, a single-sheet double-sided A4 reference sheet is allowed.
11. Calculators and computing devices such as laptops and PDAs are not allowed.

——— END OF INSTRUCTIONS ———

Aim and Methods in Question Setting 😊:

- **[Easy questions]** Try to ascertain basic understanding (~10-20% of the questions)
- **[Medium questions]** Check whether you can extrapolate from basic info, spot the correct approach (wrong approach will cost a lot of time), needs more than just "paper" understanding. [50-60% of the questions]
- **[Hard questions]** Need in-depth understanding of a topic to avoid "traps", link two to three topics in unfamiliar context. [~10-20% of the questions]

Questions 1 – 5: Each multiple-choice-question has only one correct answer. Write your answers in the boxes on the **Answer Sheet**. Two marks are awarded for each correct answer and no penalty for wrong answer.

1. What is 12012_3 in base 5?

   A. 30_5

   B. 140_5

   C. 1030_5

   D. 341022_5

   E. None of the above

ANS: C.

[Easy question. Remember to convert to base-10 before converting to base-5.]
[Stats: A:1.11% | B:3.32% | **C:81.55 %** | D:1.11% | E:12.55%]

2. Given the following C program:

```
char value = 0;
while (value >= 0 )
        value++;   //value = value + 1
printf("%d\n", value);
```

We know that the **char** data type uses **8-bit 2s complement** representation on this system. What is the value printed by this code fragment?

   A. -1

   B. -127

   C. -128

   D. 128

   E. No output as the code goes into infinite loop.

ANS: C.

[Easy question. Need to extrapolate from "int" case and understand 8-bit 2S complement range]

[A:2.21% | B:11.81% **| C:76.01 %** | D:2.95% | E:7.01%]

3. Given $t1 = 0x89 AB CD EF, what is the content of $t1 at the end of the following MIPS code?

```
andi $t2, $t1, 31
xor $t1, $t1, $t2
```

A. `0x00 00 00 1F`

B. `0x89 AB CD 00`

C. `0x89 AB CD 70`

D. `0x89 AB CD E0`

E. None of the above

ANS: D

[A:6.27% | B:4.8% | C:1.85 % | D:59.41% | E:26.2%]

31 = 32-1 = 11111_2, so the andi get the last 5-bit of $1 ➔ 01111_2. xor with "1" = flip; with "0" = no change. So, [First 28-bit no change, flip last 4-bit].

[A:6.27% | B:4.8% | C:1.85 % | **D:59.41%** | E:26.2%] (it seems that I failed to predict the "common wrong-answer" and many chose "E" because they get totally different result…?)

4. Given the following partially encoded MIPS program:

Instruction Address	Instruction & Encoding
…	…
`ABCD1208`	…
`ABCD120C`	`beq $11, $8, loop`
`ABCD1210`	…
`ABCD1214`	`j loop [Encoded: 0x0A F3 44 80]`
…	…

What is the immediate value encoded for the beq instruction? (All answers in decimal).

A. -16

B. -4

C. 4

D. 16

E. None of the above

ANS: B.

[Easy-Medium Question.] Quite happy that I managed to find a clean way to combine branch and jump in the same question ☺. A good check on whether you understand the "direct addressing" of jump versus the "relative addressing of branch"

Note the working below is in "reverse order", i.e. you should start from the last row and work towards the first row:

OPCODE	**Target Address'**
J	ABCD1200 ~~1010~~ 1011 1100 1101 0001 0010 0000 ~~0000~~
000010	1011 1100 1101 0001 0010 0000 00
0000 1010 1111 0011 0100 0100 1000 0000	

"Loop" label at ABCD1200, i.e. (ABCD1200 – ABCD1210 (BEQ+1)) / 4 = -10_{16} / 4 = -4_{10}

[A:17.34% | **B:35.06%** | C:15.5 % | D:2.21% | E:28.78%] (The options are all common errors: "forgot that the branch counts in words", "forgot that the branch can go backward (i.e. negative immediate").

5. On a **MIPS pipeline processor with NO early branching** but with **delayed branching**, how many **nop** instruction(s) need to be inserted after the branch instruction for the program below? You can assume that the compiler **did the best possible job to move instruction(s) into the delay slots**.

```
and $t2, $t1, $t7              //A (added for explanation)
sub $t5, $t4, $t3              //B
add $t7, $t2, $t6              //C
beq $t5, $zero, exit           //D
add $s0, $s0, 4                //E
...
```

A. 0

B. 1

C. 2

D. 3

E. None of the above

ANS: B.

[Hard question] Need to extrapolate delayed branching to a new setting (no early branch). Since the delayed branching slot = the distance between the "fetch" and the "resolution" of the branch, we have **3 slots** in this case (branch resolved in "W", 3 stages away from "F").

Next, need to understand the dependency that stop instructions from moving into delayed slots. Result of "instruction B" is used directly by the branch, so cannot be moved. Both "A" and "C" are safe to move.

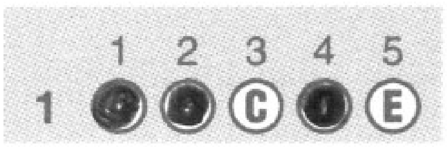
The final trap is to understand that "E" is **control dependent** on the branch, i.e. it is NOT part of the delayed branch slots. Instructions in the delay branch slots are **executed regardless of branch outcome,** but "E" **is only executed if branch "D" failed**.

Hence, we have 1 nop + 2 useful instructions in the 3 delayed slots.

[A:29.15% | **B:39.11%** | C:16.97 % | D:14.39% | E:0%] (It seems that the last trap is the hardest to avoid (quite a number of students chose "A" probably because of the wrong understanding that "E" is in the delayed slot).

Questions 6 – 10: Each **multiple response question** has **four** options. **Shade ALL** options that you think are applicable to the question. **[0-1 correct option = 0 mark, 2 correct options = 1 mark, 3 correct options = 1.5 mark, 4 correct options = 2 marks.]**

Example:

1. Which number is a **prime number?** A. 2 B. 3 C. 4 D. 5	Correct Answer: (A, B and D) 

I understand that MRQs take more time to evaluate, so q6 to 10 are actually simpler than most of the MCQs 😊.

6. Which of the following statement(s) is/are **TRUE** for Sign-and-Magnitude representation? Note: Same range means it can represent the same set of numbers (positive and negative).

A. With the same number of bits, it has the same range as 1s-complement representation.

B. With the same number of bits, it has the same range as 2s-complement representation.

C. With the same number of bits and a suitable bias, it has the same range as the bias representation.

D. It has the same ability to transform subtraction into addition as the complement representation.

ANS: A only
[Easy? question]. The key point is that S-&-M representation has two zeros (+ve and -ve zeroes), so it will represent ONE less number than any representation with only one zero representation.
C is incorrect: there is no need to try to find the bias, since bias system has only 1 zero, S-&-M will never match the range exactly.
D is incorrect: Can use any simple example as counter-example. e.g. 4-bit S&M, -2 + 2 ➔ 1010 + 0010 ➔ 1100 ➔ -4 (WRONG!)

[A:81.92% | B:87.08% | C:62.36 % | D:67.16%] (Note: these are the percentage of students correctly evaluating each of the options, e.g. there are only 67.16% student correctly determine that "D" is false).

7. Suppose register $t1 contains some random value, which of the following code fragment can set register $t1 to be all 1s (i.e. 32 1's) **for sure** (i.e. regardless of what's the original value in $t1)?

A.	`ori $t1, $t1, 0xFFFF`
	`lui $t1, 0xFFFF`

B.	`nor $t2, $t1, $zero`
	`xor $t1, $t1, $t2`

C.	`xor $t1, $t1, $zero`
D.	`addi $t1, $zero, -1`

ANS: B and D only

[Medium] (A) is a common misunderstanding (lui **clears the lower 16-bit** as explained in lecture). (B) exploit two facts: i) nor with '0's give you the negated bits, so xor between a bit and its negated value ➔ always 1. (D) -1 = 0xffff ffff in 2s complement.

[A:34.32% | B:75.28% | C:93.73 % | D:58.3%]

8. If an **unknown R-Type MIPS instruction** `xyz $2, $3, 5` is encoded as `0x00 03 11 43`, which of the following encodings is for the **same operation** but with other parameters (i.e. different registers and / or immediate value)?

A. **0x00031141**
B. **0x00131145**
C. **0x00231143**
D. **0x08030143**

ANS: None

[Easy? question] No need to look up the instruction in data sheet. Just know that it is a R-type ➔ Opcode = 0, Funct code must be the same. Split the original into the fields, we get:

Opcode	**RS**	**RT**	**RD**	**Shift**	**Funct**
000000	**00000**	**00011**	**00010**	**00101**	**000011**

Only values can change are RT, RD and Shift if it is the same instruction.

[A:77.86% | B:77.12% | C:6.64 % | D:89.67%] ((C) is "tricky" as most missed the fact that if shift amount is not zero, this is not an instruction that uses RS)

9. Suppose a **non-pipeline single-cycle MIPS processor** has the following timing parameters:

Fetch	Decode	Execute	Memory	Write Back
150ps	100ps	120ps	150ps	80ps

Which of the following statement is true?

A. An R-Type instruction takes about 450ps to finish execution.

B. The SW instruction takes about 520ps to finish execution.

C. Conditional branch takes about the same time as the R-Type instruction to finish execution.

D. The clock cycle time for this processor should be 150ps.

ANS: A, B only

[Easy question] All options are evaluated properly by majority of the class. (A) and (B) are not intended to "trick" you. I tried to phrased it carefully to avoid the misinterpretation of "cycle time", i.e (cycle time is the same for ALL instruction under single-cycle, but individual type of instruction can finish execution earlier and waste the rest of the time as there is no way for us to push in another instruction).

[A:76.38% | B:70.48% | C:70.48 % | D:62.73%]

10. Suppose Mr.Mipro **stores the value '1' into register $t1**, select all correct translations for the following C-like code fragment. Note that the variable name is the same as the register name for simplicity.

```
        if (t3 >= t4) {
             ......
        }
end:     //this is outside of the if statement
```

A.	`slt $t2, $t3, $t4`
	`beq $t2, $t1, end`

B.	`slt $t2, $t3, $t4`
	`bne $t2, $t1, end`

C.	`slt $t2, $t4, $t3`
	`beq $t2, $t1, end`

D.	`slt $t2, $t4, $t3`
	`bne $t2, $t1, end`

ANS: A only

[Easy question] Just to catch those who use “standard translation” of BLT without understanding why 😊
[A:85.61% | B:88.56% | C:91.88 % | D:76.01%]

---

11. (**Bonus 1 mark**) [Uncle Soo dont know how to count] Which lecture was mis-numbered this semester? Give the lecture number that somehow correspond to two different set of notes.

A. Lecture 8
B. Lecture 9
C. Lecture 10
D. Lecture 11
E. Lecture 12

ANS: B. We have "L09 – MIPS Encoding" and "L09 – Processor – Datapath"
[Give away 😊] Just to turn one of my embarrassment into source of free mark.
[A:6.27% | B:66.79% | C:14.39 % | D:4.8% | E:1.85%]

12. [Magic Number] Please shade option **A** for question 12.

Questions 13 – 17: Write your answer in the space provided on the **Answer Sheet**. You do not need to show workings, unless otherwise stated.

13. Ms. Wiser says:

If two **positive** floating point values **f1** and **f2** represented in IEEE754 32-bit format as **B1** and **B2** respectively , we can tell the relative magnitude (i.e. which number is bigger / smaller) by **treating their representation B1 and B2 as** 2s-complement integer and check:

```
if (B1 < B2)     //compare the representations as integers
    f1 is smaller
else
    f2 is smaller or equal
```

Is she right? If yes, briefly explain why this works; otherwise, give a counter example (no need to explain).

Ans:

13.

She is [✓ right] **/ wrong** (circle the correct answer). [4]

Explanation / Counter Example:

1. **Exponents is the more significant part of the integer representation.**
2. **Since bias system is used, larger exponent = larger absolution value.**
3. **The mantissa is normalized to 1.xxxxx, i.e. exponents determine the magnitude.**

I intend this to be (one of the) hardest question in this paper. Looks very innocent, but involve some deeper understanding of the data representation topic. Note that point (2) is important, bias system maintain the relative position value, allow us to conclude that if the binary rep of a value is larger in bias-system, the original value is also larger. This is NOT true for complement, S&M system.

Marking notes:

- If you just circle "right" without any attempt of explanation (i.e. empty explanation), you get zero mark.
- If you fail to point out (or miss the case) of why "-ve exponent" (not negative floating point value) works, 1 mark will be deducted.

[Average: 1.34 / 4]

14. Translate the C-code statement below with the **least number of instruction**. You can assume the variable mappings on the right are already performed for you (i.e. the indicated registers already stores the corresponding information at the start of your code). Make sure the end result is available in the relevant memory location(s).

C-Like Statement	**Variable Mappings**
//Assume ptr and y initialized **x = *ptr + y;**	**x → $s0 | &x → $s1** **y → $s2 | &y → $s3** **ptr → $s4 | &ptr → $s5**

ANS:

Operation	Parameter
lw	**$t0, 0($s4)**
add	**$s0, $t0, $s2**
sw	**$s0, 0($s1)**

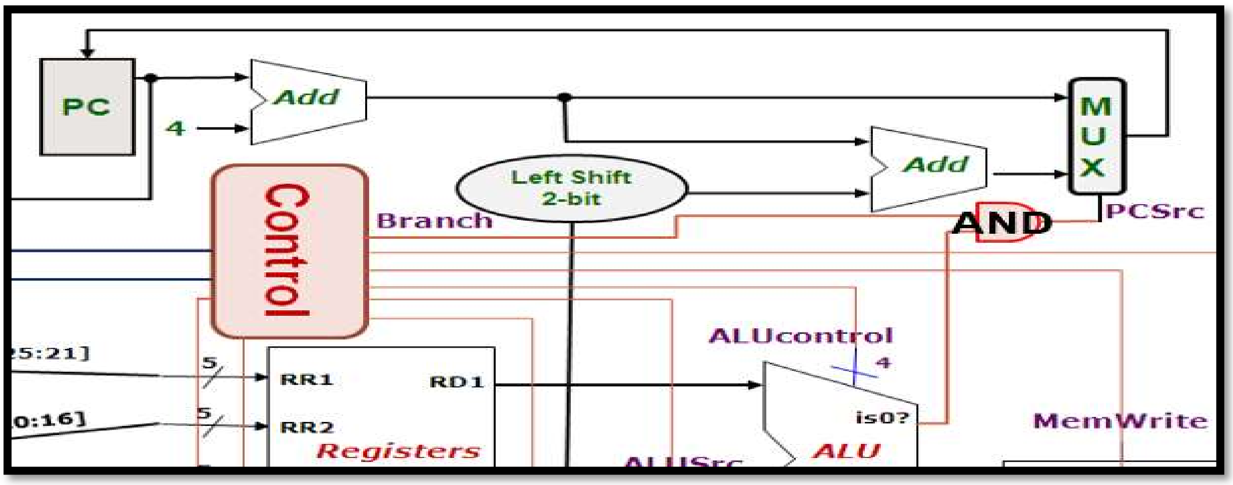
[Medium Question] Check your understanding of C (pointer derefencing) and memory loading in MIPS.

Marking notes:

- If you miss the "lw" entirely, -2 mark
- "lw $t0, 0(**$t5**)", -1 mark
- "sw" is work 1 mark

[Average = 2.15 / 4]

15. Suppose the "Branch" Signal on a particular MIPS processor chip is fused to a "1" signal, i.e. it is permanently set to '1'. Answer the following questions.



a. Give a **R-type** instruction that works normally under this processor. Give the content of the register used if applicable.
b. Describe the effect of executing "`addi $t1, $zero, 0`" under this processor.

ANS:
[Medium Question] Check understanding of datapath and signal. The "Branch" signal will only take effect IF the "isZero" signal is also True (i.e. the result calculated by ALU is a zero, the actual computation doesn't matter).

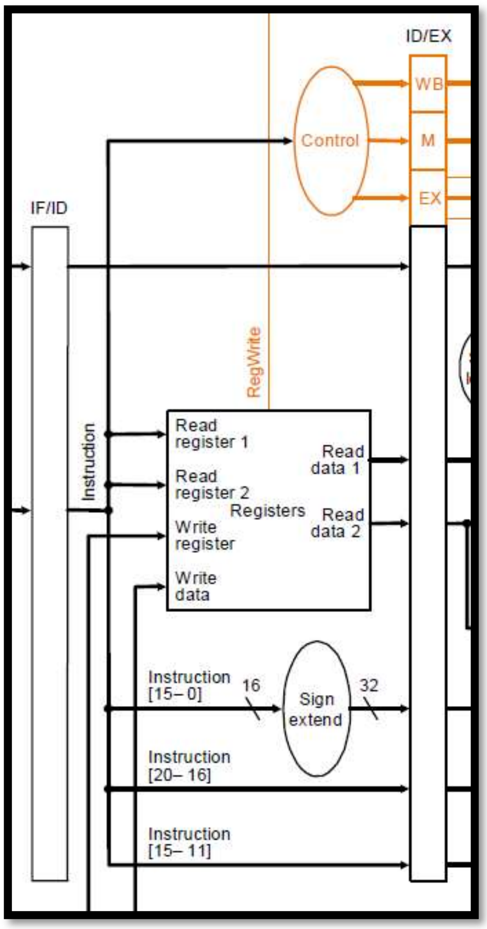
15a.	R-Type Instruction: _**Anything with non-zero result, e.g. add $t0, $zero, $t1 ($t1 is non zero)** ____	[2]

a. Any **R-Type instruction** with NON-zero result is ok. If your instruction requires the register to contain specific value(s), you MUST specified, otherwise -1 mark.
The "**cleverest**" answer is "nor, $t1, $zero, $zero" as you get a non-zero result WITHOUT any special value in any register 😊.

15b.	Effect (write in point form): - **$t1 = 0** - **Since immediate is zero, the BTA = (PC+4) ➔ will execute the next instruction normal** - **i.e. it would seems the processor is working properly.**	[2]

b. Need to point out the computation result is a zero (and should point out that it is stored properly in $t1). Next, must point out that the next instruction is (PC+4).

[Average: 2.42 / 4]



16. The diagram on the left shows the **ID stage** of a MIPS pipeline processor. Suppose we are able to get components with the following latency:

- Control Unit: Takes **120ps** to generate control signals.
- Register File: Takes **100ps** to finish one round of access (read or write).

a. Give the **latency upperbound (i.e. slowest speed)** for the **sign extend** unit.

b. What if the sign extend unit we find exceed the upper bound in (a)? **Use one sentence to describe the implication.**

ANS:

[Hard question] Requires several non-obvious understandings. The use of Register file in pipeline processor (two rounds of access in a cycle, Write followed by Read). Next is the understanding that the **stage time** can **affect overall cycle time IF the stage time happen to be the slowest already**. This is because Cycle Time = Max(stage time among FDEMW stages).

16a.	**Latency <= _200ps_ (2 x 100ps for "Write then Read")** **120ps = 1 pt**	[2]

The obvious answer is "120ps" as the Register File and the Control Unit are two parallel paths and we should choose the longest. This is reasonable (so get 1 mark) but miss the "Write then read" nature of register file.

16b.		[2]

- **Stage time increased as it was 200ps previously**
- **Overall cycle time _may_ increase if ID stage took the most time before.**

As mentioned, need to be quite clear about the conclusion. If it is written as “stage time increased so cycle time MUST increase”, then you may get a “-1”.

[Average: 1.32 / 4]

17. Given a MIPS pipeline processor with **data forwarding, early branching** but no branch prediction nor delay branching, indicate the data forwarding path utilized by instruction 2 to 4 in the program below:

```
lw   $t1, 12($s0)
add  $t4, $t1, $s1    //instruction 2
sll  $t5, $t1, 2      //instruction 3
bne  $t4, $t5, exit   //instruction 4
```

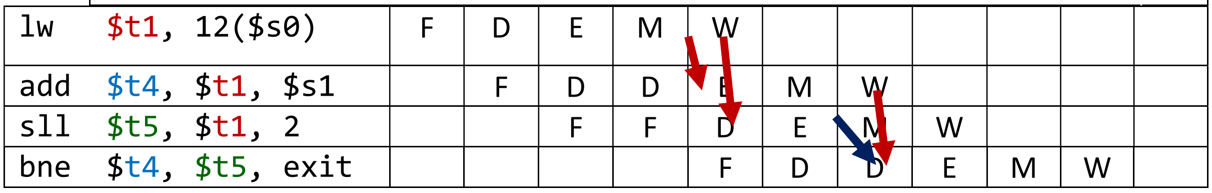
Refer to the answer sheet, use "1" to indicate "utilized", "0" to indicate "not utilized" in the corresponding column for each instruction. **Unfilled columns will be considered as wrong**. You only need to focus on the data forwarding paths given.

Ans:

Test whether you understand the actual path taken by data-forwarding. Inst 2, 3 are [easy-medium], Inst 4 is intended to be hard.

17. **(1 = "Utilized by the instruction"; 0 = "Not Utilized")** [4]

Instruction	**E/M → ALU**	**M/W → ALU**	**Register File (Write then Read)**
2	**0**	**1 ($t1)**	**0**
3	**0**	**0**	**1 ($t1)**
4	**0**	**0**	**1 ($t4)**



lw $t1, 12($s0)	F	D	E	M	W						
add $t4, $t1, $s1		F	D	D	E	M	W				
sll $t5, $t1, 2			F	F	D	E	M	W			
bne $t4, $t5, exit					F	D	D	E	M	W	

Note: The E/M to D stage forwarding for bne is a separate forwarding path (i.e. not asked by the question). Marking: +1 for any correct answer amount instruction 2, 3 and 4. 1 mark for an instruction with 1 additional wrong ‘1’ (only ONCE, i.e. if you have multiple instruction with 1 extra ‘1’ == overall 1 mark only. [Average: 1.89 / 4]

**~~~ End of Paper ~~~**