2019 EXAMINATIONS

Lancaster University

Part I (First Year)

SCHOOL OF COMPUTING AND COMMUNICATIONS

SCC.150 Digital Systems (1 hour & 30 Minutes)

- **Answer any <u>THREE</u> out of the four questions.**
- **Use a <u>separate</u> answer book for <u>each</u> question.**

QUESTION 1 (25 marks)

1.a i. Convert the decimal number 333_{10} to its corresponding unsigned binary number. **[1 mark]**

ii. Convert the unsigned 8-bit binary number 10111110_2 to its corresponding decimal number. **[1 mark]**

iii. Convert the decimal number 300_{10} to its corresponding hexadecimal number. **[1 mark]**

iv. Convert the following decimal numbers to 8-bit 2's complement: 50_{10}, 44_{10}. Then add them, giving the answer also in 8-bit 2's complement. **[2 marks]**

v. Convert the following decimal numbers to 8-bit 2's complement: -60_{10} , -55_{10}. Then add them, giving the answer also in 8-bit 2's complement. **[2 marks]**

1.b i. Specify the decimal number -128_2 in an 8-bit binary sign and magnitude representation. **[2 marks]**

ii. Specify the hexadecimal ACE_{16} in an 8-bit binary sign and magnitude representation. **[2 marks]**

iii. Convert the decimal number $-14/32_{10}$ to the 32-bit IEEE 754 floating point and express your answer in hexadecimal. (Reminder: the 32 bits are used as follows: Bit 1: sign of mantissa, bits 2-9: 8-bits of exponent in excess 127, bits 10-32: 23 bits for magnitude of mantissa.) **[6 marks]**

1.c

i. Using Boolean algebra identities please simplify the following expressions:
   a. F = AB + ABC + ABCD + ABCDE + ABCDEF
   b. F = (A+C) (AD + AD') + AC + C

[6 marks]

ii. Produce the desired inverted NAND expression using De Morgan's law on the following sum of products:
   a. F = A'B' + CDE + A' + B'

[2 marks]

[Total 25 Marks]

QUESTION 2 (25 marks)

2.a The MIPS ISA defines an exception mechanism.

i. Describe what is an exception in the MIPS assembly. Give an example of an exception.

[2 marks]

ii. Describe the two types of exceptions in the MIPS architecture.

[4 marks]

2.b Convert the instruction below to its MIPS machine language representation (32 bits).

sltiu $s1, $t0, 10

i. Give the full 32-bit representation of the instruction in binary.

[5 marks]

ii. Convert the binary representation obtained in question (i) to hexadecimal.

[1 mark]

Set less than unsigned immediate

`sltiu rt, rs, imm`

0xb	rs	rt	imm
6	5	5	16

Set register `rt` to 1 if register `rs` is less than the sign-extended immediate, and to 0 otherwise.

Name	Register number
$zero	0
$v0–$v1	2–3
$a0–$a3	4–7
$t0–$t7	8–15
$s0–$s7	16–23
$t8–$t9	24–25
$gp	28
$sp	29
$fp	30
$ra	31

2.c Register $t0 contains a 32-bit integer. Write a MIPS program that will load the upper 16-bits of the register into register $t1 and the lower 16-bits to register $t2.

[5 marks]

Question 2 continues on the next page…

2.d The following MIPS program contains two programming mistakes. The code is well-formed, and all instructions are valid MIPS instructions. Can you identify the mistakes and discuss which actions much be taken to correct them?

```
.data
str: .word 0x00000001 0x00000002 0x00000003

.text
j main

sum:
sll $a1, $a1, 2
li $v0, 0
loop: addi $a1, $a1, -4
add $t0, $a0, $a1
lw $s0, 0($t0)
add $v0, $v0, $s0
bnez $a1, loop
j end

main:
la $a0, str
li $a1, 3
jal sum

end: nop
```

[8 marks]

[Total 25 Marks]

QUESTION 3 (25 marks)

3.a Function pointers are used commonly in C programs.

i. What is a function pointer?

[3 marks]

ii. Define a variable f that is a pointer to a function accepting an int and a float argument and the function returns an integer.

[2 marks]

3.b Describe what is a struct and a union in the C language. What is the size in bytes of the structs student1 and student2 and the union student3 defined below in a 32-bit processor architecture? Describe in detail your calculations.

```
struct student1 {
   char *name;
   char initial;
   int id;
};
struct student2 {
   char name[16];
   char initial;
   int id;
};
union student3 {
   char *name;
   char initial;
   int id;
};
```

[8 marks]

Question 3 continues on the next page…

3.c Memory is organized in regions called Text, Data, Stack and Heap. The program below defines the variables a and b and a pointer c. In which memory region does the content of the variables a and b reside? In which memory region is the pointer c pointing to after the malloc?

```
#include <stdio.h>
#include <stdlib.h>
int a = 5;

int func1(int x){
   int b=5;
   char *c = (char*) malloc (x+1);
}
main {
   func1(a);
   return 0;
}
```

[6 marks]

3.d The following program defines and prints the two variables x and y. It produces the given output. Explain why the character 'A' appears in the output of variable x.

```
Program:
#include <stdio.h>
main () {
   char x[6] = "12345\0";
   char y[6] = "67890\0";
   y[7]='A';
   printf("X: %s\n",x);
   printf("Y: %s\n",y);
}

Program output:
X: 1A345
Y: 67890
```

[6 marks]

[Total 25 Marks]

QUESTION 4 (25 marks)

4.a

i. Describe the two alternative ways in which the control unit's fetch-decode-execute cycle may be implemented and briefly state their advantages and disadvantages for each.

[5 marks]

ii. State the two I/O challenges when considering a von Neumann architecture.

[2 marks]

4.b Register spilling is a common technique used in MIPS programs when invoking a procedure.

i. Explain when it is necessary to spill registers when calling procedures.

[2 marks]

ii. Discuss whether register spilling is required for registers $ra and $v0.

[2 marks]

iii. Below is a procedure which spills registers on the stack and performs a calculation. Complete the code of the procedure (moving the computation result from $s0 into an appropriate register, restoring registers, returning from the procedure).

```
sum:
    addi $sp,$sp,-8      #grow stack by 2 words
    sw $s1,4($sp)        #save register $s2
    sw $s0,0($sp)        #save register $s1
    add $t0,$a0,$a1      #$t0 contains g+h
    add $s1,$a2,$a3      #$s1 contains i+j
    sub $s0,$s1,$t0      #f = $s1 - $t0
    << YOUR CODE >>
```

[6 marks]

Question 4 continues on the next page…

4.c A function **`mystrncmp`** is required which accepts as input two c strings, s1 and s2, terminated with a NULL character ('\0') and an integer length variable n. The **`mystrncmp`** compares up to the first n characters of the two strings and returns 0 if the two strings match or 1 otherwise. Characters that appear beyond the NULL character are not compared and the result depends only on the characters before the NULL character. Provide an implementation of **`mystrncmp`**.

```
int mystrncmp(char *s1, char *s2, int n) {
<<YOUR CODE>>
}
```

[8 marks]

[Total 25 Marks]

---END OF PAPER---