CSE 331 MIDTERM – FALL 2020

1. **(Computers and Performance) (25min)**

a. (5pts) What is the CPI assuming the percentages of instruction types in the program?

Instr. Type	# of cycles	percentage in the program
Class A	3 cycles	38%
Class B	5 cycles	42%
Class C	2 cycles	20%

b. (8pts) Assume that we can implement all Class B instructions using two Class C instructio
What is the new CPI?

c. (12pts) Compare the execution times of part a and b. Show computations to find which c
faster and by how much?

2. **(MIPS Assembly) (35min)**

```
        .data
words:  .word 0x000012A4, 0x300A2B3, 0xDEDE0ADA, 0xF213A7CD,
        0x01020A0C, 0x0DAA7328, 0x0BABA17F, 0x01204501, 0x10014004

        .text
        .globl main
main:
        la $s0, words
        addi $t1, $s0, 12
        lw $s1, 0($t1)
        lw $s2, 8($t1)
        ori $s3, $zero, 4

cilek:  beq $s3, $zero, portakal
        ara $s1, $s1, 4
        sll $s2, $s2, 2
        addi $s3, $s3, -1
        j cilek

portakal: or $s3, $s1, $s2
        sw $s3, -12($t1)
        sw $s2, -8($t1)
        sw $s1, -4($t1)
```

a. (15pts) Assume words label has address 0x10010000 in hex format. Write down the contents of addresses starting from 0x10010000 to 0x1001001C after the program is executed.

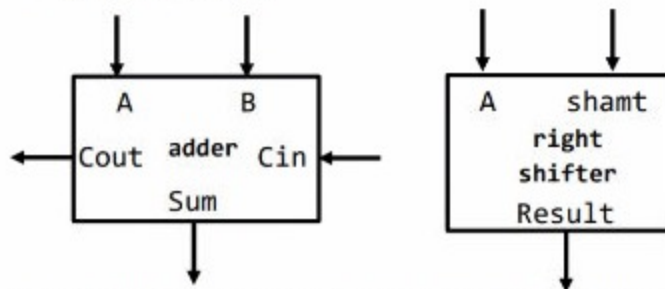b. (15pts) Convert below instructions to machine code in hexadecimal.
```
        ori $s3, $zero, 4
        beq $s3, $zero, portakal
        j cilek
```

3. (Computer Arithmetic) (20min)

(20pts) Assume that you have the following digital circuit blocks in hand:
- One 32-bit adder
- One 32-bit right shifter

Design the ALU that can perform the operations given in the following table. You can use logic gates, MUXes, Decoders, etc. in any size. Do not use any unnecessary components:

ALU Op	ALU Result
000	Or
001	Add
010	Sub
011	Set On Less Than Output
100	Shift Right Arithmetic
101	Shift Right Logical
110	Shift Left Logical
111	And

CSE 331 MIDTERM – FALL 2020

4. **(Single Cycle CPU) (20min)**

a. (15pts) Insert a new I-type instruction named **addm** that does the following operation:

```
addm $rs, $rt, imm16
MEM[0x100100F0] <= $rs + $rt + SignExtend(imm16)
```

Perform minimum number of modifications and all previous instructions must continue to work. Do not insert any unnecessary component to the datapath. Do not insert any adders or ALUs to the datapath. Do not increase the number of inputs for the ALU or Adders in the datapath. Show your solution drawing on the cheat sheet front side and send that page after scanning.

b. (10pts) Assume the opcode of the instruction is **001101**. Then, show the Logic which generates an **addm** signal which becomes 1 when **addm** comes.