Introduction to Computer Architecture – Sample Questions

1. CPI (10 pts, 5 pts each): Processor A's clock frequency is 2.5 GHz.

   a. The number of cycles for processing an instruction on Processor A is 2 cycles except for the following cases:
      - Load: 5 cycles
      - Multiply / Divide: 4 cycles
      - Taken branch / Jump: 6 cycles
      - Not taken Branch: 2 cycles

      On average, 20% of the executed branch instructions are taken branches. Program B executes the instructions in the given ratio in the table. **What is the average CPI** when processor A executes program B?

Arithmetic	Logic	Branch	Load	Store	Multiply	Divide	Jump	Shift
20%	15%	20%	10%	10%	5%	10%	5%	5%

   b. If program B executes 10 billion (10^{10}) instructions, **how long does it take to complete program B** on processor A? Ignore any additional delay except for the CPU time.

2. There are two types of CPUs, CPU ***A*** and CPU ***B***. CPU ***A***'s clock frequency is 4GHz and CPU ***B***'s clock frequency is 2GHz. Other than the difference in the clock frequency, they have the same specifications (i.e., same CPI). You want to build a processor chip that has multiple CPU cores. Due to the limited budget, you need to choose one of the following options: 1) Two CPU ***A***s, 2) One CPU ***A*** + Two CPU ***B***s

   The following are the programs that need to be executed on the processor.

Program	P1	P2	P3	P4	P5
Number of cycles	55 billion	40 billion	25 billion	20 billion	10 billion

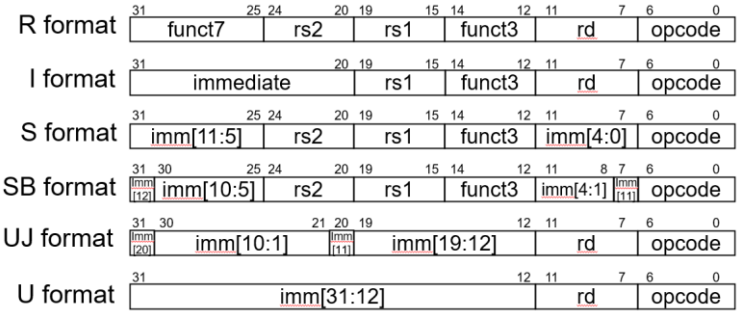
   Which option would be faster, and by how much? (10 pts, 5 pts each) The following are the detailed conditions:

   - Each program is executed once.
   - All programs can be executed any of CPU A or CPU B
   - Once a program is started on a CPU, the program completes execution on that CPU without interruption.
   - All programs are independent. They can be executed in parallel (i.e., simultaneously) on different CPUs.
   - Write the performance difference in fraction: e.g., A (or B) is faster by $^{x}/_{y}$ times.

3. Find the results of the following RISC-V Program Execution. **Write the values of the destination register after executing each instruction** one at a time. Assume you can freely read / write from any memory address. Write the values in hexadecimal**.** (10 pts, 1pts each)

`addi x1, x0, 0x100`	`x1:`
`slli x2, x1, 2`	`x2:`
`addi x3, x2, -75`	`x3:`
`sll x4, x2, x3`	`x4:`
`slt x5, x4, x0`	`x5:`
`lui x6, 0x10000`	`x6:`
`lui x7, 0xABCDE`	`x7:`
`sw x7, 0(x6)`	
`lw x8, 0(x6)`	`x8:`
`lh x9, 2(x6)`	`x9:`
`lbu x10, 3(x6)`	`x10:`

Instruction	Opcode	Funct3	Funct7
add	0110011	000	0000000
sub	0110011	000	0100000
sll	0110011	001	0000000
slt	0110011	010	0000000
lw	0000011	010	-
sw	0100011	010	-
lui	0110111	-	-
jalr	1100111	000	-
addi	0010011	000	-
slti	0010011	010	-
slli	0010011	001	-



4. Binary representation (10 pts, 5 pts each). Please use the following opcode/funct table and the RISC-V binary encoding formats. The codes are written in binary (base 2)

   a. Convert the following RISC-V assembly instruction **into the machine code**. Write the 32-bit machine code in hexadecimal number.

      `sw x3, 0x693(x10)`

   b. Convert the following machine code written in hexadecimal number **into the RISC-V assembly instruct**ion.

      0x407209B3

5. The single-cycle CPU we studied in class can be easily extended to support I-format arithmetic instructions. When the CPU is processing the following RISC-V instruction loaded from memory address 0x1000, **write the values at the indicated datapath locations** in the figure (you need to write a total of 5 values and write in hexadecimal). (10 pts, 2pts each)

addi x1, x2, -48

Currently, the register file has the following values

x1:	0xBAD	x3:	0xFEED	x15:	0xFACE	x17:	0xDEAD
x2:	0xDECAF	x4:	0xBAAD	x16:	0xCAFE	X18:	0xBEEF