**COMP4300 Spring 2021 Homework 1  
Due 11:59pm, Feb 12, 2021**

**Each problem worth 20 points**

1. For a PDP-8, generate assembly code to multiply the number in hex address 0x200 by 4, and store the result in address 0x201. The program should start in address 0x100. You can assume the number in 0x200 is positive and less than 0x100. You will need to consult the PDP-8 programming card for mnemonics and instruction formats. Note in particular that the PDP-8 has no multiply instruction. Be sure to give the address of each instruction.

Address assembly

0x100 SWAB - this swaps from mode A to B since we are referencing an address instead of a number with the muy step

0x101 mv MQ, 4

0x102 MUY 0x200

0x103 mv 0x201, MQ - most significant bit is located in MQ after muy

2. From the following assembly language code for a 32-bit MIPS processor, generate the binary machine language for this code fragment. Consult the Internet, for example:

http://max.cs.kzoo.edu/cs230/Resources/MIPS/MachineXL/InstructionFormats.html

for the bit patterns for opcodes and register numbers (5 points/instruction).

Start: LW R1,40(R2)

LW R3,1000(R0)

ADDU R1, R2, R3

J Start:

(where Start is at 32-bit address 0x00001000)

\*\* ORDER IS BIG-ENDIAN \*\*

Tell what bits go in each memory address. Remember that an address holds 8 bits.

OpCode base address data register immediate

(35) 100011 00010 00001 0000000000101000

(35) 100011 00000 00011 0000001111101000

OpCode 1st reg operand 2nd reg operand reg destination function

(0) 000000 00010 00001 00011 (33) 0000000000100001

OpCode Address (value is calculated by divided address by 4, so 00001000 in decimal is 4096 and 4096/4 = 1024)

(2) 000010 00000000000000010000000000

Address Bits stored in address

0x00001000 100011 00

0x00001001 010 00001

0x00001002 00000000

0x00001003 00101000

0x00001004 100011 00

0x00001005 000 00011

0x00001006 00000011

0x00001007 11101000

0x00001008 000000 00

0x00001009 010 00001

0x0000100A 00000000

0x0000100B 00100001

0x0000100C 000010 00

0x0000100D 00000000

0x0000100F 00000100

0x00010000 00000000

3. Suppose a given optimization to the ALU speeds up execution for the system as a whole by a factor of 1.75. After optimization, ALU operations take up 1/6 of the total execution time. What fraction of execution time BEFORE OPTIMIZATION was taken up by ALU operations? What was the speedup factor to ALU operations due to the optimization?

where x is the fraction of execution before optimization

x = 52%

= 1.287 which is the speedup factor to the ALU operations due to optimization

4. For a particular computer, the CPI for certain types of instructions is as follows:

ALU operations, 2 cycles, make up 25% of dynamic (run-time) instruction count

Load/store operations, 10 cycles, 30% of dynamic instruction count

Control flow, 3 cycles, 20% of dynamic instruction count

All other instructions, 1 cycle

What is the average CPI?

where in the fourth set of parathesis, the 0.25 indicates that all other instructions take up the remaining 25%

Suppose there is an optimization in which the CPI of load/store is reduced to 5, but cycle time is lengthened by 20%. What is the speedup due to this optimization?

5. Suppose for the problem in question 4, Load/store operations were made to take 1 cycle, without lengthening the cycle time. What would be the speedup due to that optimization?