EE 4363 / CSci 4203 Midterm Exam 1 – Spring, 2006
(closed book, closed notes, no calculators, no electronic devices)
Be sure to clearly show how you obtain your answers to each question.

1. The IEEE single-precision floating-point format is composed of a 1-bit sign, an 8-bit biased exponent with a bias of 127 and a 23-bit fraction. Consider the following two operands X and Y in this format, which are specified in hex:

```
X = C30C8000    Y = C30CA000
```

(a) (20 points) Determine the decimal value of each of the operands.

(b) (10 points) Determine the IEEE single-precision floating-point representation (specified in hex) for the difference `X - Y`. (Perform the subtraction in decimal and then convert the result into the IEEE format.)

2. The following machine language VeSPA program is placed into the file v.out. (Note that some instruction encodings are provided on the following page.)

```
58 C0 00 01
58 80 01 00
10 46 10 00
28 02 00 00
F8 00 00 00
```

(a) (20 points) Give the corresponding assembly language program.

(b) (20 points) Give the 32-bit contents (specified in hex) of the PC and registers r0 through r3, inclusive, and the values of the condition code bits C, V, Z and N after each instruction has been executed by behavioral.v.

3. On a certain computer, instructions in Class A have a CPI of 2, instructions in Class B have a CPI of 3 and instructions in Class C have a CPI of 5. Consider the following two code sequences: Code sequence 1 contains 4 instructions in Class A, 2 instructions in Class B and 2 instructions in Class C. Code sequence 2 contains 5 instructions in Class A, 2 instructions in Class B and 3 instructions in Class C.

(a) (10 points) Determine the number of instructions executed in each sequence.

(b) (10 points) Determine the number of CPU clock cycles required for each sequence.

(c) (10 points) Determine the CPI for each sequence.

SUB* – Subtraction

*This instruction sets the condition code bits.

Assembly code notation

a) SUB `rdst`, `rs1`, `rs2`

b) SUB `rdst`, `rs1`, #immed16

Instruction encoding

a)

31 ··· 27	26 ··· 22	21 ··· 17	16	15 ··· 11	10 ··· 0
00010	`rdst`	`rs1`	0	`rs2`	000 0000 0000

b)

31 ··· 27	26 ··· 22	21 ··· 17	16	15 ··· 0
00010	`rdst`	`rs1`	1	immed16

HLT – Halt

Assembly code notation

HLT

Instruction encoding

31 ··· 27	26 ··· 0
11111	000 0000 0000 0000 0000 0000 0000

LDI – Load immediate

Assembly code notation

LDI `rdst`, #value

Instruction encoding

31 ··· 27	26 ··· 22	21 ··· 0
01011	`rdst`	immed22

NOT – Bit-wise logical complement

Assembly code notation

a) NOT `rdst`, `rs1`

Instruction encoding

31 ··· 27	26 ··· 22	21 ··· 17	16 ··· 0
00101	`rdst`	`rs1`	0 0000 0000 0000 0000

XOR – Bit-wise logical exclusive-OR

Assembly code notation

a) XOR `rdst`, `rs1`, `rs2`

b) XOR `rdst`, `rs1`, #immed16

Instruction encoding

a)

31 ··· 27	26 ··· 22	21 ··· 17	16	15 ··· 11	10 ··· 0
00110	`rdst`	`rs1`	0	`rs2`	000 0000 0000

b)

31 ··· 27	26 ··· 22	21 ··· 17	16	15 ··· 0
00110	`rdst`	`rs1`	1	immed16