ENEE350
Sample Midterm-Fall 2008
CLOSED BOOK AND NOTES
EXAM PERIOD 75 MINUTES

Instructions:

- Points for each problem are indicated right after the problem. The total score is 100 points.
- Use the space provided below each problem. if you need more space, please ask a proctor.
- Write your name and student id on the cover sheet.
- Promptly hand in your test to a proctor when the test is over.

NAME: _____

STUDENT ID: _____

Problem 1 (20 points): Consider the following Vesp 2.0 program. Specify the values in memory locations 0, 10 and 32 (addresses are expressed in decimal) after the execution of the program is completed, and explain your answer in each case.

```
100     2000    0020
102     2001    0021
104     1000    0000
106     8000
107     300A    0000
109     3020    0001
10B     0000    D000
10D     7000
```

Note: All addresses and operand values in the program are expressed in hexadecimal.

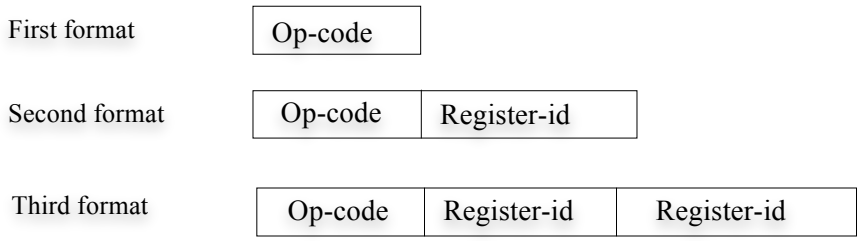
Problem 2 (20 points): Write a Vesp 2.0 program to find the maximum of a set of numbers stored in memory locations 0x100 through 0x200.

Problem 3 (20 points): A given processor has four operand registers *W*, *X*, *Y*, and *Z* in addition to a program counter (PC), an instruction register (IR), memory address register, and memory data register, MDR. The following sequence of steps describes the instruction fetch, decode, and execute steps of this CPU:

```
while (reset == 0)
{Step 1: MAR = PC; PC = PC +1;
 Step 2: MDR = MEMORY[MAR];
 Step 3: IR = MDR;
 Step 4: switch(IR[0:1])
   {case 0: W = W << X; break;
    case 1: W = W+X; Y = Y+Z; break;
    case 2: W = W+ X+ Y + Z; break;
    case 3: W = W + X-Y-Z; break;
   }
}
```

Give a control (sequential) circuit that performs the instruction fetch-decode-execute sequence of this processor, clearly indicating the events that occur at each state of the circuit.

Problem 4: (15 points.) The instruction repertoire of a processor consists of three different formats for its instructions as shown below:



Suppose that 12 instructions use the third format, 14 instructions use the second format, and 50 instructions use the first format. If the processor has a scratchpad of 16 registers:

(a) How many bits are needed at the minimum to code all the instructions?
(b) Give a coding scheme that codes all the instructions using the number of bits you specified in part (a).

Problem 5: (25 points.)

The ABUS, BBUS, OBUS, destination, and branch condition selection logic of a micro-programmed processor with a datapath that is identical to vesp 2.0M is shown below.

ABUS Select	ABUS	BBUS Select	BBUS	OBUS Select	OBUS	Dest. Select.	Destination
000	0	000	MDR	0000	ABUS	0000	A
001	A	001	IR	0001	BBUS	0001	PC
010	PC	010	B	0010	ABUS + BBUS	0010	IR
011	MAR	011	1	0011	~ABUS	0011	IX
100	1	100	IX	0100	ABUS + ~BBUS + 1	0100	MDR
101	N/A	101	N/A	0101	ABUS & BBUS	0101	MAR
110	N/A	110	N/A	0110	ABUS | BBUS	0110	B
111	N/A	111	N/A	0111	ABUS << 1	0111	RET SET
				1000	ABUS >> 1	1000	C
				1001	ABUS + 4	1001	Z
				1010	BBUS + 4	1010	S
				1011	ABUS - 4	1011	F
				1100	BBUS - 4	1100	read
				1101	N/A	1101	write
				1110	N/A	1110	reset
				1111	N/A	1111	N/A
Condition Select	Condition						
0000	A>0						
0001	A=0						
0010	A<0						
0011	1						

Using the information in the table, and in binary notation,

(a) Write a microprogram to perform the computation:

A = 5; B = 13; B = 2A + 3B; MEMORY[8] = B;

(b) Assuming that the following microprogram has been stored in location 0x100, and A = 2, B = 3, PC = 5, MAR = 4; MDR = 6; IR = 4; IX = 7, explain what it does.

Location	Microcode
0x100	0100010001100110
0x101	0000110010000000
0x102	0000110001100100
0x103	0001001100010111

Instruction Repertoire of Vesp 2.0

```
/* A:     16 bits (Implicit) It refers to location 0 in  VESP's memory
/* B:     16 bits (Implicit) It refers to location 1 in  VESP's memory
/* IX:    16 bits (Implicit) It refers to location 2 in  VESP's memory
/* IR:    16 bits
/* MAR:   12 bits
/* PC:    12 bits

/* ADD: Opcode: 0000  ----- A = A+B                          HexCode: 0
/* CMP: Opcode: 0001  ----- A = ~A                           HexCode: 1
/* LDA: Opcode: 0010  ----- M[IR[4:15] ] = M[MAR+1]          HexCode: 2
/* MOV: Opcode: 0011  ----- M[IR[4:15] ] = M[M[MAR+1][3:15]] HexCode: 3
/* JMP  Opcode: 0100  ----- PC = IR[4:15]                    HexCode: 4
/* JEZ: Opcode: 0101  ----- If (A = 0)  PC = IR[4:15]        HexCode: 5
/* JPS: Opcode: 0110  ----- If (A > 0)  PC = IR[4:15]        HexCode: 6
/* HLT: Opcode: 0111  ----- reset = 1                        HexCode: 7
/* INC: Opcode: 1000  ----- A = A + 1                        HexCode: 8
/* DEC: Opcode: 1001  ----- A = A - 1                        HexCode: 9
/* AND: Opcode: 1010  ----- A = A ^ B                        HexCode: A
/* IOR: Opcode: 1011  ----- A = A | B                        HexCode: B
/* SHL: Opcode: 1100  ----- A << 1                           HexCode: C
/* SHR: Opcode: 1101  ----- A >> 1                           HexCode: D
/* MXF  Opcode: 1110  ----- M[IR[3:15]] = M[IX]; IX = IX + 1;HexCode: E
/* MXT  Opcode: 1111  ----- M[IX] = M[IR[3:15]]; IX = IX + 1;HexCode: F
```