ENEE 350
TEST 1–Fall 2004-10/14/04
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):

(a) Suppose that a processor executes instructions each of which is 16-bits long. How many different instructions can the instruction repertoire of this processor include if you assume that each instruction must be identified by a 4-bit opcode?

16 instructions

(b) Suppose that only 7 instructions in the instruction repertoire of this processor require a 12-bit address, and all of its other instructions will execute operations on operands stored in a pair of registers (A and B) as in the **VESP** processor. Describe an instruction format that allows this processor to have 16 such instructions in addition to the 7 instructions that require a 12-bit address.

Use the first bit to select between those instructions that require a 12-bit address and those that execute their operations on a pair of registers. For example, when the first bit is 0, the next three bits can be used to code the 7 instructions that require a 12-bit address,and when the same bit is 1, the next four bits can be used to code the 16 instructions that operate on register operands. Note that in the latter case, the 12-bit address field is not needed.

Type-1 instructions (those that require a 12-bit address)

0 000, 0 001, 0 010, 0 011, 0 100, 0 101, 0 110 + 12 bit address

Type-2 instructions (those that operate on registers)

1 0000, 1 0001, 1 0010, 1 0011, ..., 1 1111

Problem 2 (20 points):

The computation engine of a given processor executes four shift instructions on a 4-bit operand,

A =[a_0,a_1,a_2,a_3]

whose instruction format is given below:

Instruction	Op-code	Shift direction	Number of shifts
SHL1	00	Left	1
SHL2	01	Left	2
SHR1	10	Right	1
SHR2	11	Right	2

Design a combinational circuit that can carry out these shift instructions in the computation engine of this processor. (*Note: You are required to describe only a combinational circuit that performs these shift instructions, not the entire computation engine of the processor.*)

This problem can be solved in a number of ways. The most straightforward is to use four 4x1 multiplexers.

Multiplexer-1 Data inputs	*Multiplexer-2 Data inputs*	*Multiplexer-3 Data inputs*	*Multiplexer-4 Data inputs*
a_1 a_2 - -	*a_2 a_3 a_0 -*	*a_3 - a_1 a_0*	*- - a_2 a_1*
data output-1	*data output-2*	*data-output-3*	*data-output-4*

The two select inputs of all the multiplexers are fed with the two opcode-bits given in the table in the problem.

Note that the - entries can be specified further depending on whether or not shifts are arithmetical or logical. For example, for arithmetic shifts,

a_1 a_2 a_0 a_0 a_2 a_3 a_0 a_0 a_3 0 a_1 a_0 0 0 a_2 a_1

Problem 3 (25 points): A useful instruction to include in a processor's repertoire is a move instruction that can index memory locations with a variable index displacement. The instruction format below describes one such instruction:

op-code	direction of move	source or destination address	index displacement
1111	1 bit	13 bits	6 bits

If the *direction of move* bit is 0 then the 13-bit address points to the destination of the operand and the address of the operand in the memory is found in the index register IX. If this bit is 1 then the 13-bit address points to the source of the operand and its destination address is found in the index register IX. In both cases, the address in the index register is replaced by IX + *index displacement*, where the index displacement is a 6-bit signed number in 2's complement notation.

Assuming that this instruction has already been identified (decoded), describe its execution using an instruction register(IR), index register (IX), a memory address register (MAR), and a temporary register (TEMP). (You may assume that the memory (MEMORY) holds 24-bit instructions and data.)

The execution of this instruction can be described by the following C-code:

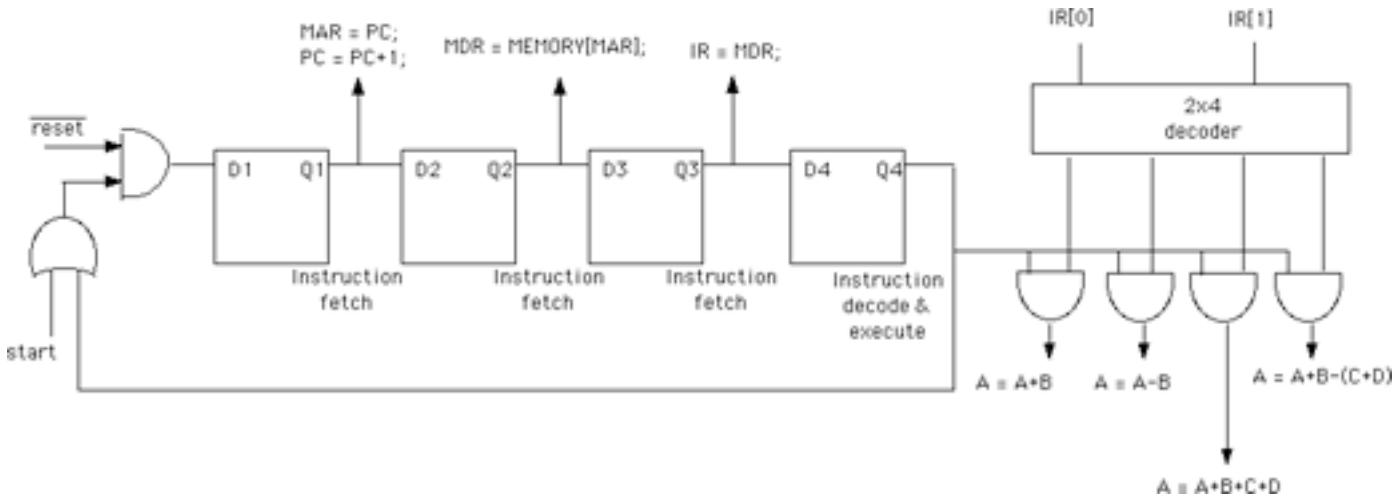
```
if(IR[4]==0)
{MAR = IX;
 MDR = MEMORY[MAR];
 MAR = IR &((6 >> IR)&0x1FFF)
 MEMORY[MAR] = MDR;
}
else
{MAR = IR &((6 >> IR)&0x1FFF);
 MDR = MEMORY[MAR];
 MAR = IX;
 MEMORY[MAR] = MDR;
}
 IX = IX + IR & 0x3F;
```

Problem 4 (20 points): A given processor has four operand registers A, B, C, and D 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: A = A + B; break;
case 1: A = A - B; breakl
case 2: A = A+ B+ C + D; break;
case 3: A = A + B - (C + D); 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.

See the figure on the next page.


MAR = PC;
PC = PC+1;
MDR = MEMORY[MAR];
IR = MDR;
IR[0]
IR[1]
2x4
decoder
reset
start
D1
Q1
D2
Q2
D3
Q3
D4
Q4
Instruction
fetch
Instruction
fetch
Instruction
fetch
Instruction
decode &
execute
A = A+B
A = A-B
A = A+B-(C+D)
A = A+B+C+D

Problem 5 (15 points): Write a vesp program to change the signs of the three numbers stored in memory locations 120,121,122.

```
Load 1st number into A, complement+inc it, store it back into location 120.
Load 2nd number into A, complement+inc it, store it back into location 121.
Load 3rd number into A, complement+inc it, store it back into location 122.

3000  move 120 into A
0078
1000  complement it
8000  increment it
3078  move A into 120
0000
3000  move 121 into A
0079
1000  complement it
8000  increment it
3079  move A into 121
0000
3000  move 122 into A
007A
1000  complement it
8000  increment it
307A  move A into 122
0000
7000  halt
```