ECE 5362 Homework 4
Due 10:20am Oct 9 (Carmen PDF Submission)

1. (24 pts) The following question is related to the OSIAC machine. For each of the following operations, either list the control lines to make **all** the register transfers happen in a ***single*** clock cycle (i.e., a single state), or explain why it cannot be done. You should make no special assumption about the contents of the registers. (2 pts each)

    a. [T1]-[T5]→Q
    b. [T2]+[Q] →Q
    c. [T1]+1 → Q
    d. One's complement of [Q]→Q
    e. Two's complement of [SP]→Q
    f. Two's complement of [AC]→X
    g. [T4]→T2, [T3]→X
    h. Exchange the contents of T1 and AC
    i. [T1]-1→ Q
    j. [T1]→Q, [T2]→T4
    k. [IR]→AC
    l. [[MAR]] →MDR, [T1]+1→Q, [AC]→T5

2. (25 points) For the OSIAC machine, consider the following program. DATA = 0x17 is the memory address of a 16-bit data word: 0x00CD. Assume the program starts from memory address 0, so you can get the values of addresses FUNC, LOOP, and END. Follow the OSIAC description to hand assemble the program, i.e., turn the program into OSIAC machine code in hexadecimal. Use 0 for any "don't cares". You need to give the memory contents, in hexadecimal, from address 0x0 to 0x17, as in an OSIAC test program. If you cannot figure out the content of any address in the between, you may use 0xFFFF for it. Note that CLRC is to clear the C condition, as introduced in class. The purpose of this problem is to get your familiar with OSIAC instruction formatting and test program.

```
        MOVE SP, -(X)
        SUB #20, SP
        ADD (SP), AC
        EXG -2(X), AC
        NEG (X)+
        MOVE #DATA, X
        CLR AC
        JSR FUNC
        HALT
FUNC:   MOVE DATA, AC
        INC AC
LOOP:   DEC X
        BPL LOOP
        JMP END
        CLRC
END:    RTS
```

3. (51 pts) For the following OSIAC test program that can be used as sample test cases for your MP2 and MP3, fill in the highlighted blanks in its output file "summary.1". The format of summary.1 was introduced in class. Here "*memory write:: FFFF->M(F)*" means a content of 0xFFFF is written to memory address 0xF in the below instruction (i.e. instruction 1). You should first turn the instructions into assembly instructions and then analyze what they do to change the contents of the registers, memory contents, and condition codes CVZN. The purpose of this problem is to get you familiar with the summary file that is important to your future MP debugging. (34 blanks, 1.5 pts each)

FF00 AC
000F X
0012 SP
0000 PC
0000 CVZN
0411
4104
0421
4204
0432
4031
0442
FFFF
4042
FFFE
4050
000F
0450
0010
0000
0001
00FF
0002

Summary.1:

```
memory write:: FFFF->M(   F)
  At end of instruction 1
    ac     x    sp    pc    cvzn
  FF00     F    12     1    ____
 memory write:: ____->M( ___)
  At end of instruction 2
    ac     x    sp    pc    cvzn
  ____     F    12     2    ____
 memory write:: ____->M(   F)
  At end of instruction 3
    ac     x    sp    pc    cvzn
  ____   ___    12     3    0000
 memory write:: ____->M( ___)
  At end of instruction 4
    ac     x    sp    pc    cvzn
  ____   ___   ___     4    0000
```

```
memory write:: ____->M( ___)
 At end of instruction 5
   ac     x    sp    pc    cvzn
 ____    ___   ___     5    ____
memory write:: ____->M( ___)
 At end of instruction 6
   ac     x    sp    pc    cvzn
 ____    ___    11     6    0001
memory write:: ____->M( ___)
 At end of instruction 7
   ac     x    sp    pc    cvzn
 ____    10    11     8    0001
memory write:: ____->M( ___)
 At end of instruction 8
   ac     x    sp    pc    cvzn
 ____    10    11     A    0001
memory write:: ____->M( ___)
 At end of instruction 9
   ac     x    sp    pc    cvzn
 FFFF    10    11     C    0001
memory write:: ____->M( ___)
 At end of instruction 10
   ac     x    sp    pc    cvzn
 FFFF    10    11     E    ____
```