EE 308 – Homework 2

Due Feb. 1, 2010

1. Write an instruction sequence to take the 8-bit number an memory location $1000, negate it, and put the result into memory location $1100. Also, take the 8-bit number at memory location $1100, negate it, and put the result into memory location $1000. If $1000 orignally had a -17, and $1100 originally had a +23, then after the instructions are executed, $100 should have a - 23 and $1100 should have a +17.

2. Write an instruction sequence which adds the eight-bit number at address $1100 to the the 8-bit number at locations $1101 and stores the 16-bit result into addresses $1000 and $1001. Treat the values stored in $1100 and $1101 as signed numbers. This should work so that (+127) + (+127) = (+254), and (-127) + (-127) = (-254) – that is, convert the eight-bit numbers to 16-bit numbers before adding them. (Hint: use the `SEX` instruction.)

3. Consider the following program from Lab 2:

```
prog:   equ     $2000   ; Starting address from program
data:   equ     $1000   ; Starting address for data

        org     prog    ; Set initial program counter value
        ldx     #1234   ; Immediate (IMM) addressing mode
        ldab    #235
        abx             ; Inherent (INH) addressing mode
        stx     result  ; Extend (EXT) addressing mode
        swi

        org     data    ; Put data starting at this location
result: ds.w    1       ; Reserve one word (two bytes) for results
```

   (a) Hand-assemble the program. That is, figure out what the op codes of the instructions are, and where they will be located in memory.

   (b) How many cycles will it take the MC9S12 to execute this program. (Do not include the `swi` instruction.)

   (c) How long will it take an MC9S12 with a 24 MHz E clock to execute this program?

   (d) Determine the state of the N, Z, V and C bits after each instruction has been executed. (Assume that, when the program starts, all these bits are zero.)

   (e) What will be the contents of addresses $1000 and $1001 after the program executes?

4. Write an instruction sequence to set the upper four bits of the number at address $0049 to 1, and leave the lower four bits unchanged.

5. Write an instruction sequence to clear all the odd bits and toggle all the even bits of the 8-bit number at address $0048. If the contents of $0048 were 01101011 before the instruction, it would be 00010100 after the instruction.

6. Consider the following program fragment:

```
          ldy         #5000
loop1:    ldx         #5000
loop2:    dbne        x,loop2
          dbne        y,loop1
          swi
```

   (a) How many instruction cycles will it take the MC9S12 to execute the following program? (Do not consider the `swi` instruction.)

   (b) How many seconds will this take the MC9S12 with an 24 Mhz E-clock? (You should give the answer to the nearest microsecond.)

7. An MC9S12 has the following data in its memory:

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
10D0	10	23	3B	7C	10	04	86	80	B7	10	25	3B	FC	10	18	F3
10E0	10	F5	FD	10	18	86	40	B7	10	23	3B	FC	10	12	DD	02
10F0	86	CE	A2	53	1A	2F	A3	10	03	86	40	B7	10	25	3B	86

Determine the contents of the A and X register after executing the following code fragments. (Before the first instruction, the X register has $0000.) List the values in hexadecimal. Also, indicate what addressing mode is used, and what the effective address of the instruction is. (Assume that the first instruction is at address $2000, and that the instructions that follow are in subsequent locations – i.e., the instruction of (a) takes two bytes, so the first instruction of (b) is at address $2002.)

   (a)
```
ldaa #43
```

   (b)
```
ldaa $10E7
```

   (c)
```
ldx $10E0
ldaa -2,X
```

   (d)
```
ldx #$10E0
ldaa -2,X
```

   (e)
```
ldx #$10E0
ldaa 2,+X
```

   (f)
```
ldx #$10E0
ldaa 2,X+
```