EE 308 – Homework 2

Due Feb. 1, 2012

1. Write an instruction sequence to take the unsigned 8-bit number in memory location $1000, divide it by two, and put the result into memory location $1100. Also, take the unsigned 8-bit number at memory location $1100, divide it by two, and put the result into memory location $1000. If $1000 orignally had a 17, and $1100 originally had a 22, then after the instructions are executed, $1000 should have am 8 and $1100 should have a 12.

2. Problem E1.18 (Page 37 of text).

3. Problem E1.19 (Page 37 of text).

4. Consider Program 1 from Lab 2:

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

        org     prog    ; Set initial program counter value
        ldy     #2345   ; Immediate (IMM) addressing mode
        ldab    #123
        aby             ; Inherent (INH) addressing mode
        sty     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?

5. Consider Program 2 from Lab 2:

```
prog:     equ    $2000       ; Starting address for program
data:     equ    $1000       ; Starting address for data
count:    equ    10          ; 10 elements in the table

          org    prog
          ldaa   #count      ; ACCA keeps count of numbers left in table
          ldx    #table      ; X points to table of data
          ldy    #0          ; Y holds sum; initialize to 0
```

```
repeat:   ldab   1,X+        ; get data from table int B; X points to next element
          aby                ; Compute 16-bit sum
          deca               ; Decrement counter
          bne    repeat      ; If not done, continue with next element
          sty    result      ; Save sum
          swi

          org    data        ; Put data starting at this location

; Initialize data in table
table:    dc.b   $44,$AB,$74,$61,$C2,$54,$61,$62,$F2,$13
result:   ds.w   1
```

(a) Hand assemble the program. Indicate the addressing mode for each of the instructions

6. Write an instruction sequence to set the lower four bits of the number at address $0049 to 0, and leave the upper four bits unchanged.

7. Problem E2.17 (Page 86 of the text).

8. Problem E2.19.

9. Consider the following program fragment:

```
          ldy         #50000
loop1:    ldaa        #250
loop2:    dbne        a,loop2
          dbne        y,loop1
          swi
```

   (a) Hand assemble the program. (Add an **org** assembler directive to put the program in memory starting at address 0x2000.)

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

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

10. 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	E5	3B	7C	10	04	86	80	B7	10	25	3B	FC	10	26	F3
10E0	10	D4	A5	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 #21
```

   (b)
```
ldx $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+
```