Lab 2 – Part 2
Assembly Language Programming and 9S12 Ports

Introduction and Objectives

The purpose of this laboratory is to write a few assembly language programs and test them on your MC9S12.
You will learn how to create delays with software loops, and how to do simple I/O.

The Dragon12 Plus has several ways to display data. It has an LCD display, four seven-segment LED displays, and eight individual LEDs. Programming the LCD display is rather complicated, and will not be discussed at this time. The easiest display to start with is the individual LEDs. You will write some programs to display patterns on the LEDs.

You can display patterns on the LEDs by writing to the I/O port at address 0x0001 (called PORTB). Because of the way the LEDs are connected, it is necessary to do some other setup of I/O ports as well. The first five instructions in the program below set up the MC9S12 hardware so that you can write patterns to the LEDs. (For now, we will give you the code you need to set up the I/O system properly. In later labs, you will learn how to do the setup yourself.) The following program will flash the LEDs:

Program 1 Demo program for week 2 of Lab 2.

```
PORTB   equ $_____      ; Port B data register
DDRB    equ $_____      ; Port B direction register
PTP     equ $_____      ; Port P data register
DDRP    equ $_____      ; Port P direction register
PTJ     equ $_____      ; Port J data register
DDRJ    equ $_____      ; Port J direction register

        bset    DDRP,#$0F    ; Make PPO-PP3 outputs
        bset    PTP,#$0F     ; Turn off seven-sed LEDs
        bset    DDRJ,#$02    ; Make PJ1 output
        bclr    PTJ,#$02     ; Turn on individual LEDs
        bset    DDRB,#$FF    ; Activate control lines for LEDs
        movb #$55,PORTB      ; Turn on every other LED
loop:   com     PORTB        ; Toggle LEDs
        bra loop             ; Repeat
```

1. Prelab

Make sure you have the programs written and clearly thought out before you come to the lab. You should put all your code starting at memory location 0x2000. You are encouraged to bring the programs in on a flash drive.

Prelab Part 1. Complete Program 1 by adding the necessary addresses and assembler directives.

Prelab Part 2. Write a program which will start with all the LEDs off, then increment the LEDs, implementing a binary counter.

Prelab Part 3. Write a program to determine the largest and smallest 16-bit number in memory locations 0x8000 to 0x80ff. Store the maximum in address 0x1000, and the minimum in address 0x1002. Treat the numbers as signed.

Prelab Part 4. Write a program which puts the exclusive OR of the eight-bit numbers from memory locations 0x8000 through 0x8FFF and display the result on the LEDs. (This operation is often used to generate a check sum to verify data transmission. The sending computer generates and transmits the check sum along with the data. The receiving computer calculates the check sum for the received data and compares it with the check sum sent by the sending computer. If the two values do not match, then there was an error in the transmission.)

2. The Lab

Make sure you have all your programs written from the prelab section.

2.1 Test the the Following During Lab

1. Consider the program developed in Prelab Part 1.

    (a) Test your program on the MC9S12. Trace through the loop to see what is happening. Note: The program should cause the LEDs to flash on and off.

    (b) Run your program by typing g 2000. When you do this, the LEDs flash so quickly that it looks like they are all lit. Add some code to create a 100ms delay between the last two instructions of the program. When you do this, you will be able to see the lights flash.

2. Consider the program developed in Prelab Part 2. Run your program to verify that it works. Make sure you use a delay so you can see the LEDs incrementing.

3. Consider the program developed in Prelab Part 3. Run your program and verify that it works.

4. Consider the program developed in Prelab Part 4. Run your program and verify that it works.