

CMPE333 - Microprocessor Systems Laboratory

Lab Report #3 : HCS12 Input and Output Ports

Shamsah Al Ali - 100037059

Fatima Alzaabi - 100036968

Submitted on: Thursday, 17th, February 2015

# Table of Contents

***Table of Contents \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2***

***List of Figures and Tables*** ***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 3***

***Summary*** ***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 4***

***Introduction*** ***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 5***

***Tasks and Results Analysis*** ***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 6***

***Assignment Questions\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 9***

***Conclusion\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_10***

# List of Figures and Tables

**Figure 1** Output Through LEDs Assembly Code 6

**Figure 2:** LED display on board 7

**Figure 3** Input Through DIP Switches Assembly Code 7

Figure 4- reading DIP switches and writing to LEDs code 8

Figure 5- Delay Subroutine code 10

Figure 6- Assignment Question 1 11

Figure 7- Assignment Question 2 12

Figure 8- Assignment Question3 14

# Summary

This report is the result of conducting the third laboratory experiment on the HC input and Output Ports which consists of 4 main tasks. The first task looks at how to derive output lines to and examine its effects on the dragon12+ board. The second task works on reading input lines on the board and how to modify them to get specified values. The third task presents the procedure of reading DIP switches and writing them to LEDs. Finally, the last task works on using subroutine to implement delay.

**1. Introduction**

One of the most important features of the micro controller is its ability to receive and deliver input and output from and to hardware components of a device through its I/O ports. Micro controllers initially reserve the memory locations $0000-$03FF for control registers that hold the I/O features and its controls.

Parallel ports are a type of I/O ports that are able to perform operations as input or as outputs, but when mapped with the hardware, the Dragon12+, each port is linked to a certain component on the board. In this lab, ports B, J, P and H will be looked at. Port B is used for the 8 LEDs and 7-segments display. Port J is responsible for the serial clock. Port P is used for the common cathode signals for 7-segment display while Port H is responsible for the 8-position DIP switches as well as the 4 push buttons on the Dragon12+ board.

Another important feature of the I/O ports is the data direction registers which are specified for each port. These registers control whether this port will be an input or an output. For example, DDRA = 0x00 will set port A as an input. Having DDRA = 0xFF will set port A as an output.

This laboratory experiment will look at the I/O features which are the DIP switches and the LEDs of the M68HCS12 microcontroller and how to configure an assembly code to enable input and output from these features respectively.

Aim

This experiment's aim is to introduce and familiarize students with the read and write methods to fetch data from and to the input and output ports respectively.

Objectives

* Learn how micro controller I/O ports work and configure them.
* Read and write data from the I/O ports.
* Analyze how the DIP switches function on the hardware.
* Execute I/O programming in HCS12 Language.

**2. Tasks, Results and Analysis**

**Task 1: Derive Output Lines**

This task required to assemble the following code, download it to the micro controller and run it on the board to verify that the value in accumulator A is displayed through the LEDs on the board. Setting Port B to output mode will not allow the LEDs to display the values on the board, this is because PTJ1 pin controls the LEDs. To disable it, the PTJ1 pin must be programmed as output and then set its value to 0x00 to allow the LEDs to show data. Recall that port B maps the usage of both the LEDs and 7-segments display. In this laboratory, the output needs to be only viewed through the LEDs so the 7-segments display has to be disabled. In order to do that, Port P will have to be disabled since it maps to the common cathode of the 7-segments display.

ORG $4000 ;Flash ROM address for Dragon12+ Entry:

LDAA #$FF  
STAA DDRB ;Make PORTB output  
;PTJ1 controls the LEDs connected to PORTB LDAA #$FF  
STAA DDRJ ;Make PORTJ output  
STAA DDRP ;Make PORTP output  
LDAA #$00  
STAA PTJ ;Turn off PTJ1 to allow the LEDs to show data  
LDAA #$0F  
STAA PTP ; Disable the 7-segment display

;-------Switch on LEDs connected to PORTB based on the Acc A value

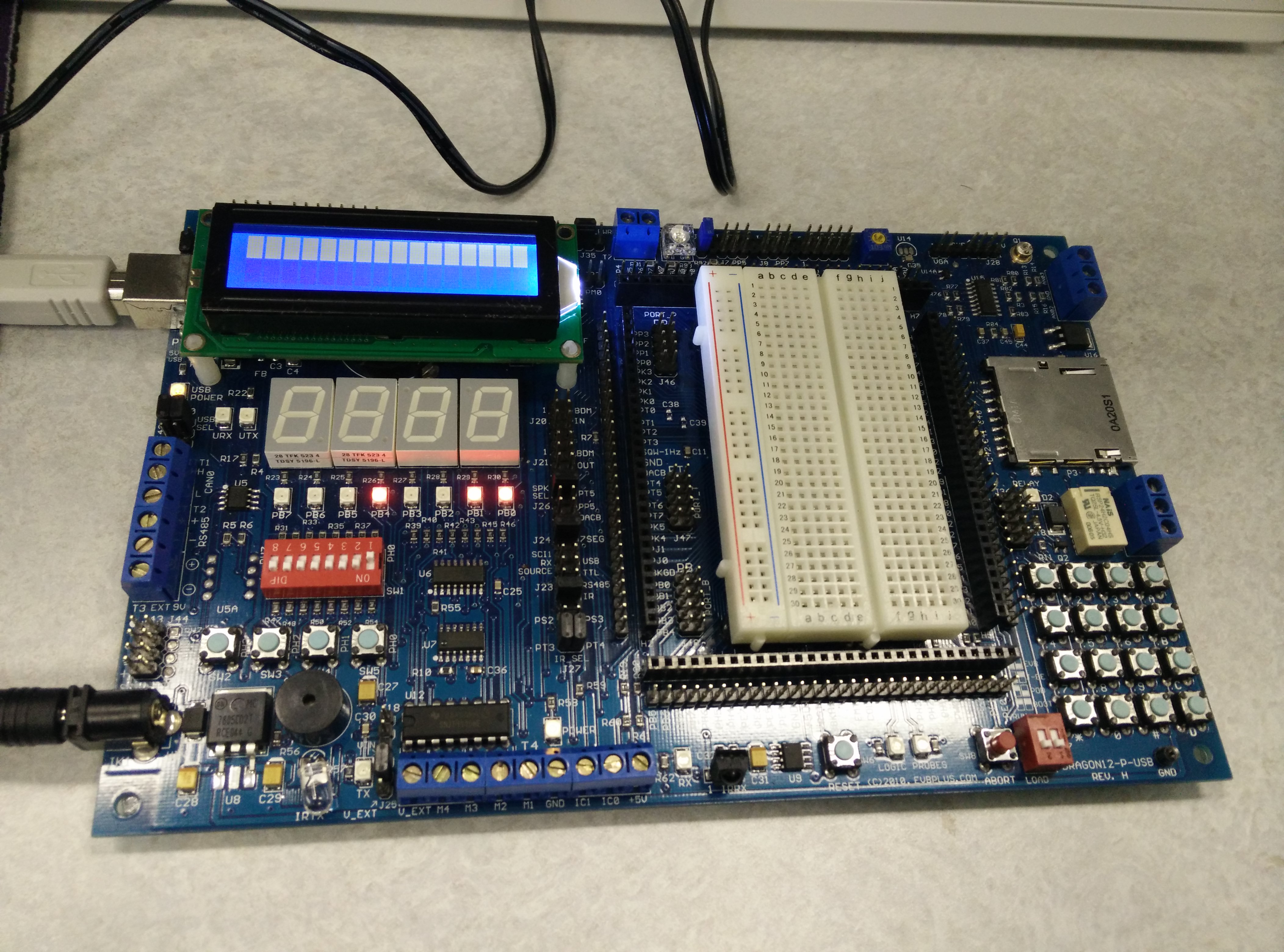
LDAA #$55  
STAA PORTB ;Store A into PORTB

BRA Entry

**Figure 1** Output Through LEDs Assembly Code

The Accumulator A holds the value of 13. Equivalently on the board, the LEDs showed the values (00010011) where each 4 consecutive bits represent a digit. The results shown on the board reflects the value in accumulator A as shown in the figure below.

**Figure 2:** LED display on board



### **Task 2: Reading Input Lines**

For this task, it was required to assemble the following code and verify whether the input values from the board are similar to the values in accumulator A.

ORG $4000 ;Flash ROM address for Dragon12+

Entry:

LDAA #$0

STAA DDRH

LDAA PTH

BRA Entry

**Figure 3** Input Through DIP Switches Assembly Code

This code will allow the program to take its input from the DIP switches (Hence enable Port H to input mode) and then store the value from the DIP switches into accumulator A. The DIP switches value representation is similar to the LEDs where each 4 consecutive bits represent a single digit. For example, if the value 10 is needed to be stored, only the fourth switch will be flipped on, so the switches board will look similar to (0001000).

Afterwards, the code was modified to read bit 4 of PTH using the following line:

ANDA #PTH\_PTH4

The input value on the board was changed to 20 and the result in Accumulator A was found to be 10, this is because the op code ANDA is a logical and operation, this operation takes the value found from the DIP switches and the bit 4 of PTH and does a logical AND operation on it.

0 0 1 1 0 0 0 0 (20)

0 0 0 1 0 0 0 0 (10)

——————-

0 0 0 1 0 0 0 0 (10)

* 1. **Task3: Reading Dip Switches and Writing Them to LEDs**

In this task, we need to read to states of the DIP switches and reflect them immediately to the LEDs continuously (read from switches and write to the LEDs.) In order to do that, we combine task1 and task2 into the following code:

INCLUDE 'derivative.inc'

XDEF Entry, \_Startup, main

XREF \_\_SEG\_END\_SSTACK ; symbol defined by the linker for the end of the stack

MY\_EXTENDED\_RAM: SECTION

Counter ds.w 1

FiboRes ds.w 1

MyCode: SECTION

main:

 \_Startup:

ORG $4000 ;Flash ROM address for Dragon12+ Entry:

LDAA #$FF ; load the immediate value $FF to accumulator A

STAA DDRB ;Make PORTB output ; load the value of accumulator A into DDRB to Make PORTB output

;PTJ1 controls the LEDs connected to PORTB

 LDAA #$FF ; load the immediate value $FF to accumulator A

STAA DDRJ ; load the value of accumulator A into DDRJ to Make PORTJ output

 STAA DDRP ; load the value of accumulator A into DDRP to Make PORTP output

LDAA #$00 ; load the immediate value $00 to accumulator A

STAA PTJ ; load the value of accumulator A to Turn off PTJ1 to allow the LEDs to show data

 STAA DDRH ; load the value of accumulator A into DDRH to assign it as an input

LDAA #$0F ; load the immediate value $0F to accumulator A

 STAA PTP ; load the value of accumulator A to Disable the 7-segment display

;-------Switch on LEDs connected to PORTB based on the Acc A value

LDAA PTH **; load the value of the switches into accumulator A**

STAA PORTB **;Store the value of Accumulator A into PORTB**

;-------------------

BRA Entry

Figure 4- reading DIP switches and writing to LEDs code

Firstly, we load $FF to register A which in turn store it in DDRB, DDRJ, and DDRP, so that we make these three ports serve as an output. Then, we load $00 into register A and store it in PRJ1 to allow the LEDs show the data, after that we disable the 7-segment display.

Finally, we set Port H as input by loading $00 to register A and storing it to DDRH, so that we can read the input of the DIP switches and show the output on the LEDs. The BRA instruction is used here to enable us to repeat the read and write processes continuously.

* 1. **Task4: Using Subroutine to Implement Delay**

In this task, we were asked to modify task 3 in a way that we introduce a delay between reading the switches and writing to the LEDs using the following delay subroutine:

ORG $4000 ;Flash ROM address for Dragon12+

Entry:

LDAA #$FF ; load the immediate value $FF to accumulator A

STAA DDRB ;Make PORTB output ; load the value of accumulator A into DDRB to Make PORTB output

;PTJ1 controls the LEDs connected to PORTB

 LDAA #$FF ; load the immediate value $FF to accumulator A

 STAA DDRJ ; load the value of accumulator A into DDRJ to Make PORTJ output

STAA DDRP ; load the value of accumulator A into DDRP to Make PORTP output

LDAA #$00 ; load the immediate value $00 to accumulator A

 STAA PTJ ; load the value of accumulator A to Turn off PTJ1 to allow the LEDs to show data

STAA DDRH ; load the value of accumulator A into DDRH to assign it as an input

LDAA #$0F ; load the immediate value $0F to accumulator A

STAA PTP ; load the value of accumulator A to Disable the 7-segment display

 ;-------Switch on LEDs connected to PORTB based on the Acc A value

 LDAA PTH **; load the value of the switches into accumulator A**

JSR DELAY; jump to subroutine DELAY

 STAA PORTB **;Store the value of Accumulator A into PORTB** ;-------------------

BRA Entry

DELAY ; Delay function

PSHA ;Save the value of Reg A on Stack

LDAA #10 ; load the immediate value 10 to accumulator A

STAA R3 ; load the value of accumulator A into R3

L3 LDAA #100; load the immediate value 100 to accumulator A

STAA R2; load the value of accumulator A into R2

L2 LDAA #240; load the immediate value 240 to accumulator A

STAA R1; load the value of accumulator A into R1

L1 NOP ;1 Intruction Clk Cycle

NOP ;1

NOP ;1

 DEC R1 ;4

BNE L1 ;3

DEC R2 ;Total Instr.Clk=10

BNE L2

DEC R3

BNE L3

;-------------- PULA ;Restore Reg A

RTS

Figure 5- Delay Subroutine code

As the code get executed on the board it was clear that a delay was created whenever we set one of the dip switches on in order to see the output is set on or off corresponding output the LED.

1. Assignment Questions

**1) Create a program that implements a binary counter. Use the delay subroutine given in Task-4 to display the counting sequence on the LEDs connected to PORTB.**

Entry:

LDAA #$FF

STAA DDRB ;Make PORTB output

;PTJ1 controls the LEDs connected to PORTB

LDAA #$FF

STAA DDRJ ;Make PORTJ output

STAA DDRP ;Make PORTP output

LDAA #$00

STAA PTJ ;Turn off PTJ1 to allow the LEDs to show data

STAA DDRH

LDAA #$0F

STAA PTP ; Disable the 7-segment display

;-------Switch on LEDs connected to PORTB based on the Acc A value

LDAA PTH JSR DELAY STAA PORTB ;Store A into PORTB

LOOP INCA

BRA DELAY STAA PORTB BRA LOOP

Figure 6- Assignment Question 1

A label is created in order to implement the binary counter. A binary counter can be constructed from J-K flip-flops by taking the output of one cell to the clock input of the next. This produces a binary number equal to the number of cycles of the input clock signal. first it increment A then goes through the delay branch and then store A to Port B. every time it goes through the loop A is incremented to know how many times the code ran.

**2) Create a program that flash all the LED’s with a delay of 0.1 sec in between.**

Entry:

LDAA #$FF

STAA DDRB ;Make PORTB output

;PTJ1 controls the LEDs connected to PORTB

LDAA #$FF

STAA DDRJ ;Make PORTJ output

STAA DDRP ;Make PORTP output

LDAA #$00

STAA PTJ ;Turn off PTJ1 to allow the LEDs to show data

STAA DDRH

LDAA #$01

STAA PTP ; Disable the 7-segment display

;-------Switch on LEDs connected to PORTB based on the Acc A value

LDAA PTH JSR DELAY STAA PORTB ;Store A into PORTB

LP LSLA

JSR DELAY STAA PORTB BRA LP

Figure 7- Assignment Question 2

To write the code for this question we need to use the code of task4 in addition we need to modify it. We used the LSL command in order for us to move to the next LED as each led represents a multiple of 2 number from 1- 128. A loop was required as well to keep the movement between the LEDs each time we multiply A with 2.

**3) Re-examine the toggle program in assignment question 2 which flashes the LEDs of PORTB with 0.1 sec delay. Now, modify that program to get the byte of data from PTH switches and give it to R3 register of the DELAY loop. Run the program to show how you can set the time delay size using the PTH switches.**

Entry:

LDAA #$FF

STAA DDRB ;Make PORTB output

;PTJ1 controls the LEDs connected to PORTB

LDAA #$FF

STAA DDRJ ;Make PORTJ output

STAA DDRP ;Make PORTP output

LDAA #$00

STAA PTJ ;Turn off PTJ1 to allow the LEDs to show data

STAA DDRH

LDAA #$01

STAA PTP ; Disable the 7-segment display

;-------Switch on LEDs connected to PORTB based on the Acc A value

LDAA PTH JSR DELAY STAA PORTB ;Store A into PORTB

LP LDAA PTH LDAB PTH JSR DELAY STAA PORTB BRA LP

here the loop loads the accumulator A and B with the PTH switch.

Figure 8- Assignment Question3

1. Analysis and Interruption

Lab three illustrates the use of the ports of the microcontroller board. The student determined the port PORTB control the LEDs in the first task and PTJ should be turned off with the use of Accumulator A. In task two the Dip switches are used as inputs in the and it was controlled using PTH. Combining the codes of task 1 and 2 together to use the dipswitches as inputs and the LEDs as outputs. The last task included a subroutine of a delay in order to use it in other codes and it showed how the delay was created between the LEDs.

1. Conclusion and Recommendations

The main objective of this lab is to determine, understand how the Dragon12 board works. This experiment illustrates the various ports of the microcontroller hardware, basic techniques for example, using the LEDs lights as output and the Dip switches as inputs. Delays were introduced and implemented ,for instance the delay was added each program in order to see the effect on the output. Each of the four tasks included a particular code in order to notice the benefit of the usage of each port on the board.