

**Khalifa University of Science, Technology and Research**

**Electronic Engineering Department**

**ELCE332 Microprocessor Systems laboratory**

**Laboratory Experiment 4**

**HCS12 Input and Output Ports**

**Lab Partners**

Hamad Eisa Alhazami 100036985

Jasim Alhammadi 100035510

**Date Experiment Performed**: 18-2-2015

**Date Lab Report Submitted**: 25-2-2015

**Lab Instructor: Mr. Mohammed Alzaabi/ Mr. Mahmoud Khounji**

**Spring 2015**

# Summary

The purpose of this report is to discuss the results obtained after conducting the five main tasks in the laboratory session. This experiment was very similar to the one preceding it; the main difference was that the tasks in this lab were implemented using C programming in addition to using other components, such as the LCD screen and the Key Pad. The first task explained how to use LEDs and the 8 bits in accumulator A to derive output lines. The second task focused on taking the input from the Key Pad and displaying the output on the LEDs. The third task was about displaying simple messages with time delays on the LCD screen embedded in the Dragon12 Plus Trainer Board. The forth task required taking the input from DIP switches and displaying the output on both the LEDs and the LCD screen. The fifth and final task was a challenging one, the Key Pad was used as input, and the LCD screen would then show the results corresponding to the conditions of the algorithm.

Contents

[Summary 2](#_Toc412633312)

[1. Introduction 4](#_Toc412633313)

[1.1. Aim 4](#_Toc412633314)

[1.2. Objectives 4](#_Toc412633315)

[2. Design and Result 5](#_Toc412633316)

[2.1. Task 1 : Reading DIP switches and writing them to LEDs 5](#_Toc412633317)

[2.2. Task 2: Reading keypad and writing to LEDs 6](#_Toc412633318)

[2.3. Task 3: Writing to LCD 8](#_Toc412633319)

[2.4. Task 4: Reading DIP switches and writing them to LEDs and LCD 9](#_Toc412633320)

[2.5. Task 5: Interfacing LCD with keypad and pushes button 10](#_Toc412633321)

[3. Analysis and interpretation 12](#_Toc412633322)

[4. Assignment Questions 13](#_Toc412633323)

[5. Conclusions and Recommendations 15](#_Toc412633324)

# Introduction

The HCS12 microcontroller consists of various components; in addition to that, it includes a fully integrated suite of I/O capabilities. These capabilities are a parallel and serial I/O, analog input and timer functions. Dual purpose functions are shared by the different I/O pins in the microcontroller; those functions are done by using a set of 1024 control registers which are located at memory location $0000-$03FF.

Similarly to the previous lab, the tasks in this experiment mainly focused on specific components in the microcontroller which are parallel ports, Light Emitting Diodes (LED), push buttons and DIP switches. The main difference between this experiment and the one preceding it, is that the tasks in this lab also focused on other components, such as the Key Pad and the LCD screen.

Parallel ports serve as general purpose I/O ports; they function as either input or output ports. There are nine parallel ports with the names A, B, E, AD, H, J, K, P and T, out of these nine parallel points, this experiment concentrated on the LEDs controlled by ports B, J and P in addition to the DIP switches, the port H controlled push buttons, Port K’s LCD screen and Port A’s Key Pad.

## Aim

To introduce the students to the read and write data from the input and output ports and how delays can be implemented using loops.

## Objectives

1. Learn how a C program can access I/O registers
2. Develop simple programs for an embedded system.
3. Use the CodeWarrior Integrated Development Environment for the development of HCS12 microcontroller C programs.
4. Writing simple C language programs for interfacing DIP switches and Key Pad with LCD.
5. To assemble, download and run a C program using CodeWarrior C compiler and Dragon12 Plus Trainer board.
6. Download, run, and test code on a Dragon12+ Board.

# Design and Result

# Task 1 : Reading DIP switches and writing them to LEDs

In the first task, it is required to write a C program that takes the input from the DIP switches and shows the output as a binary number using the LEDs. Using code-warrior, the program is written and run on the Dragon 12 board. The code is shown below.

**#include <hidef.h> /\* common defines and macros \*/**

**#include "derivative.h" /\* derivative-specific definitions \*/**

**void main() {**

**EnableInterrupts;**

**DDRB = 0xFF;**

**DDRP = 0xFF;**

**DDRJ = 0xFF;**

**PTJ = 0x00;**

**PTP = 0x0F;**

**DDRH = 0x00;**

**for(;;) {**

**PORTB= PTH;**

**\_FEED\_COP(); /\* feeds the dog \*/**

**} /\* loop forever \*/**

**/\* please make sure that you never leave main \*/**

**}**

The program works by taking the value from the register PTH (for the DIP switches), and storing it in register PORTB (for the LEDs). In order to let the LEDs show the data, the values of registers DDRB, DDRJ, and DDRP are set to the value $FF in order to set them as outputs. Also, the value of PTJ is set to the value $00 to display by the LEDs, and the value of PTP is set to the value $0F to disable the 7-segments display. The register DDRH value is set to the value $00 in order to set it as the input.

The program is run on Dragon12 Plus Trainer Board. Several inputs are tested on the board using the DIP switches. For every input, the LEDs showed the correct output in binary. For example, when the input is set to value $04, the LEDs showed the correct output for it (0000 0100).

## Task 2: Reading keypad and writing to LEDs

In the second task, it is required to write a C program that takes the input from the keypad and shows the output as a binary number in the LEDs. Using code-warrior, the program is written and run on the Dragon12 Plus Trainer board. The code is shown below.

**#include <hidef.h> /\* common defines and macros \*/**

**#include "derivative.h" /\* derivative-specific definitions \*/**

**int key\_pad();**

**void main() {**

**EnableInterrupts;**

**DDRB = 0xFF;**

**DDRP = 0xFF;**

**DDRJ = 0xFF;**

**PTJ = 0x00;**

**PTP = 0x0F;**

**DDRH = 0x00;**

**DDRA=0x0F;**

**PUCR=0x01;**

**for(;;) {**

**PORTB=key\_pad();**

**\_FEED\_COP(); /\* feeds the dog \*/**

**} /\* loop forever \*/**

**}**

The program works by taking the value from the function key\_pad (), and storing its value in register PORTB (for the LEDs). The same configuration for registers DDRB, DDRJ, DDRP, PTJ, PTP, and DDRH are set. In addition, in order to configure the keypad, the register DDRA value is set to $0F to configure PORTA where (PA0-PA3) are output columns and (PA4-PA7) are input rows. Moreover, PUCR value is set to the value $01 to turn on the pull up resistors associated with port A. The code for the keypad is shown on the next page.

The program is run on Dragon12 Plus Trainer Board. Several inputs are tested on the board using the keypad. For every input, the LEDs showed the correct output in binary. For example, when the input is set to value $03, the LEDs showed the correct output for it (0000 0011).

Key\_pad() function code:

**int key\_pad(void)**

**{**

**int X;**

**while(1)**

**{**

**PORTA = 0XFE;**

**X = PORTA;**

**if (X == 0xEE)return 0x01;**

**if (X == 0xDE)return 0x04;**

**if (X == 0xBE)return 0x07;**

**if (X == 0x7E)return 0x0E;**

**PORTA = 0XFD;**

**X = PORTA;**

**if (X == 0xED)return 0x02;**

**if (X == 0xDD)return 0x05;**

**if (X == 0xBD)return 0x08;**

**if (X == 0x7D)return 0x00;**

**PORTA = 0XFB;**

**X = PORTA;**

**if (X == 0xEB)return 0x03;**

**if (X == 0xDB)return 0x06;**

**if (X == 0xBB)return 0x09;**

**if (X == 0x7B)return 0x0F;**

**PORTA = 0XF7;**

**X = PORTA;**

**if (X == 0xE7)return 0x0A;**

**if (X == 0xD7)return 0x0B;**

**if (X == 0xB7)return 0x0C;**

**if (X == 0x77)return 0x0D;**

**}**

**}**

## Task 3: Writing to LCD

In the third task, it is required to write a C program using the LCD functions that write the message “Microprocessor” on the first line, and “ELCE 333 Lab” message on the second line. Each message should appear for a half of a second then disappears to show the next message in a loop. Also, each message should be shown separately in which when the first message appears the second message disappears. Using code-warrior, the program is written and run on the Dragon12 Plus Trainer Board. The code is shown below.

**#include <hidef.h> /\* common defines and macros \*/**

**#include "derivative.h" /\* derivative-specific definitions \*/**

**#include "lcd.h" /\* LCD driver file \*/**

**void main() {**

**void LCD\_Init();**

**EnableInterrupts;**

**LCD\_Init();**

**while(1) {**

**LCDWriteLine(1, "ELCE 333 Lab");**

**delay(250);**

**delay(250);**

**LCD\_clear\_line(1);**

**LCDWriteLine(2, "Microprocessor");**

**delay(250);**

**delay(250);**

**LCD\_clear\_line(2);**

**} /\* loop forever \*/**

**}**

The program works by taking the two strings “Microprocessor” and “ELCE 333 Lab” and their line number using the function LCDWriteLine (Line number, message). In order to clear the line, the function LCD\_clearline (Line number) is used. Finally, the delay is set by the function delay (millisecond). The program is written inside a loop to let the messages appear consequently.

The program is run on Dragon12 Plus Trainer Board. As observed, the two messages appeared and disappeared consequently with a delay between them.

## Task 4: Reading DIP switches and writing them to LEDs and LCD

In the fourth task, it is required to write a C program that takes input from the DIP switches and show the output as a binary number on the LEDs and the LCD screen on the Dragon12 Plus Trainer Board. The code from the first task is used and modified. The code is shown below.

**#include <hidef.h> /\* common defines and macros**

**#include "derivative.h" /\* derivative-specific definitions \*/**

**#include "lcd.h"**

**void main() {**

**void LCD\_Init();**

**EnableInterrupts;**

**DDRB = 0xFF;**

**DDRP = 0xFF;**

**DDRJ = 0xFF;**

**PTJ = 0x00;**

**PTP = 0x0F;**

**DDRH = 0x00;**

**LCD\_Init();**

**while(1) {**

**//PORTB = PTH;**

**int num = PTH;**

**LCDWriteInt(num);**

**delay(250);**

**delay(250);**

**LCDWriteLine(1," ");**

**delay(250);**

**delay(250);**

**} /\* loop forever \*/**

**}**

The program works by taking the value from the PTH register , and storing it in register PORTB (for the LEDs) and the function LCDWriteInt (number) . The same configuration in task 1 is done for registers DDRB, DDRJ, DDRP, PTJ, PTP, and DDRH. In order to write the number and clear it in case the number is changed, the function LCD\_WriteLine (Line number, message) is used.

The program is run on Dragon12 Plus Trainer Board. Several inputs are tested on the board using the DIP switches. For every input, the LEDs and LCD showed the correct output in binary. For example, when the input is set to value $04, the LEDs and the LCD showed (0000 0100) and (4).

## Task 5: Interfacing LCD with keypad and pushes button

In the fifth task, it is required to write a C program that asks the user to input a number using the Key Pad and displays it on the LCD screen. The message should appear in the first line and the number on the second line. There will be a switch, if it is turned on, the number should be divided by 2 and displayed on the screen. Otherwise, the original number should appear. Using code-warrior, the program is written and run on the Dragon12 Plus Trainer Board. The code is shown below.

**#include <hidef.h> /\* common defines and macros \*/**

**#include "derivative.h" /\* derivative-specific definitions \*/**

**#include "lcd.h"**

**int key\_pad();**

**void main() {**

**void LCD\_Init();**

**int num;**

**float num2;**

**EnableInterrupts;**

**DDRB = 0xFF;**

**DDRP = 0xFF;**

**DDRJ = 0xFF;**

**PTJ = 0x00;**

**PTP = 0x0F;**

**DDRH = 0x00;**

**DDRA=0x0F;**

**PUCR=0x01;**

**LCD\_Init();**

**while(1) {**

**if (PTH == 0){**

**LCDWriteLine(1," \*Enter number\* ");**

**LCDWriteLine(2," Num = ");**

**num = key\_pad();**

**LCDWriteInt( num);**

**} else {**

**LCDWriteLine(1," \*Enter number\* ");**

**LCDWriteLine(2," Num/2 = ");**

**Num2 = key\_pad()/2;**

**LCDWriteFloat( num2);**

**}**

**} /\* loop forever \*/**

**}**

The program works by taking the input from the function key\_pad() , and storing its value in the function LCDWriteInt (number) . The same configuration in tasks 1 and 2 is done for registers DDRB, DDRJ, DDRP, PTJ, PTP, DDRH, DDRA, and PUCR. There will be two if-else conditions by checking the condition of the switch if it is set on or off. If the switch is off (PTH =0), the LCD will display the default value of the input. If the switch is on, the input value will be divided on 2 and displayed on the screen.

The program is run on Dragon12 Plus Trainer Board. Several inputs are tested on the board using the DIP switches. For every input, the LEDs and LCD showed the correct output in LCD. For example, when the input is set to value $04, the LCD showed the value (4) when the switch is off and (2) when the switch is on.

# Analysis and interpretation

The lab focused mainly on using the C language (HLL) for programming the Dragon12 Plus Trainer Board. The first task is the same as the first task in Lab 3 but instead of using assembly language, C language is used. The difference is that C language is easier to deal with in comparison with the assembly language. For example, in order to set the value of the registers in assembly, accumulators are used as an intermediate. While in C, registers values can be set directly. The second task is similar to the first task but instead of using the DIP switches, the Key Pad is used by including the provided function in the program. The difference is in the mechanism used inside both of them. In the third task, the LCD functions are used by using the provided header and main files that contain the definitions for the functions. The delay function is used twice to introduce the delay for a period of 0.5 seconds to avoid memory overflow. In the fourth task, DIP switch input is used for both LCD and LEDs to show the difference in reading the data from them. In the last task, the value is displayed as an integer. Thus, when the switch was on, only even values were displayed correctly.

# Assignment Questions

Q1) Create a C language program that checks the state of PH2 (while all DIP switches at their high position) and flashes all the eight LEDs of Port B from left to right if PH2 is high, and flashes them from right to left (like they do when the board is first powered on) if PH2 was low. Hint: Use the keyword >> to shift right or << to shift left.

**void main(void) {**

**EnableInterrupts;**

**DDRB=0XFF ;**

**DDRJ=0XFF ;**

**DDRP=0XFF ;**

**PTJ=0X00 ;**

**PTP=0X0F ;**

**DDRH=0X00 ;**

**Short int i =1; ;**

**for(;;) { ;**

**if ((PTH==0) && (i!=0)) ;**

**{**

**if (PTH\_PTH2==1)**

**{ For (i=128; i!=0; i=i>>1)**

**{ PORTB=i;**

**delay(500) ;**

**}}**

**else**

**{**

**For (i=1;i!=0; i=i<<1)**

**{PORTB=i;**

**}}**

**}**

**\_FEED\_COP(); /\* feeds the dog \*/**

**}}**

2) The keypad function code will be used in this program but it will not be mentioned in the code to save space. (To check keypad function code, look to Figure 3)

**int first\_num, sec\_num ;**

**int key\_pad(void) ;**

**float add=0, sub=0,mul=0,div=0;**

**void main(void) { EnableInterrupts;**

**LCD\_Init() ;**

**DDRA=0X0F ;**

**PUCR=0X01 ;**

**DDRH=0X00 ;**

**for (;;) {**

**first\_num=PTH >> 4 ;**

**sec\_num=key\_pad() ;**

**LCDWriteLine(1 , "N1: ") ;**

**LCDWriteInt(first\_num);**

**LCDWriteLine(1 , ", N2: ");**

**LCDWriteInt(sec\_num);**

**LCDWriteLine(2 , "OP: ");**

**If ((PTH & 15)== 1){**

**add = first\_num + sec\_num;**

**LCDWriteLine(2 , " + , Res. = ");**

**LCDWriteLine(add) ;**

**}If ((PTH & 15)== 2){**

**sub = first\_num - sec\_num;**

**LCDWriteLine(2 , " - , Res. = ");**

**LCDWriteLine(sub) ;**

**}If ((PTH & 15)== 4){**

**mul = first\_num \* sec\_num;**

**LCDWriteLine(2 , " \* , Res. = ");**

**LCDWriteLine(mul) ;**

**If ((PTH & 15)==8){**

**If (sec\_num==0) continue;**

**div = first\_num / sec\_num;**

**LCDWriteLine(2 , " / , Res. = ");**

**LCDWriteLine(div) ;**

**} \_FEED\_COP() ; /\* feeds the dog \*/**

**}}**

# Conclusions and Recommendations

To conclude, the tasks focused on introducing the basic techniques of using DIP switches and the Key Pad as inputs and LEDs and the LCD screen embedded in the Dragon 12 Plus Trainer board as outputs using C programming. On the contrary to assembly language, complex operations can be simplified to look friendlier to humans with a high language like C. In addition to that, operations take fewer lines in C when compared to assembly.