Khalifa University of Science, Technology and Research

Electronic Engineering Department

ELCE333: Microprocessor Systems Laboratory

Laboratory Experiment No. 4

HCS12 Input and Output Ports

Laboratory Partners

Name: Afra Bin Fares, ID#:100033139

Name: Alya Humaid AlAlili ,ID#100037087

Name:Anoud Alshamsi ,ID#100035514

Laboratory Instructor:

Mohammed Ali Saif Al Zaabi

Mahmoud Khonji

Spring 2015

Table of Contents

Abstract

The Lab experiment four mainly focuses on writing programs with C language and introduce HCS12 ports. Also, this experiment is divide five tasks and most of the tasks their idea is taken from the previous lab .The first task is about reading DIP switches and writing them to LEDs, whereas the second task is about reading from Key Pad and writing into LEDs. Next, the third task explains how to write to LCD. After that, we have the fourth task we need to write into the LEDs as well as the LCD, when they take the input from the DIP switches. Finally, the last task is about explaining how we can interface the LCD with Key Pad and push buttons.

1. **Introduction:**

The M68HCS12 has a particularly rich, fully integrated, suite of I/O capabilities, including parallel and serial I/O, analog input, and timer functions. Many of the I/O pins have shared or dual purpose functions. The parallel I/O ports in the port integration module have bidirectional capability. That is, they operate in either output or input mode.

Ports A and B

These ports are available as general-purpose I/O ports in single- chip mode only. When in this mode, a data direction register for each, DDRA and DDRB control the direction, input or output, of each bit in each register. Any of the memory addressing modes may be used when reading from or writing to these registers.

Port E

Port E is a register in which the bits have a variety of functions in the expanded modes. In single-chip mode, bits 2 – 7 may be input or output as controlled by the data direction register DDRE. Bits 0 and 1 are associated with the interrupt pins.

Port J

Port J is a two-bit, general-purpose register with interrupt capabilities

Port S

The serial communications interface (SCI) shares two of the four bidirectional Port S pins. If the SCI is enabled, bit-0 is assigned to received-data (RXD) and is configured as an input. Bit-1 is assigned to transmitted-data (TXD) and configured as an output.

Data Direction Register

Each bit in the bidirectional data registers may be programmed to be either input or output. When the CPU is reset, all registers are placed in the input mode, and must set bits in a Data Direction Register – DDR to change input bits to be outputs.

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.

2. Design, Results and Analysis

In this part of the report the tasks performed in the experiment will be discussed by explaining the steps of these tasks, listing, explaining and analyzing the results obtained from this experiment. The experiment is basically divided into five tasks. All five tasks will be performed using c language written in the code warrior software, and the Dragon Plus Trainer Board.

2.1. TASK-1: Reading DIP Switches and Writing them to LEDs

In this part of the experiment, using task3, lab 3 information, a C program was written that takes the inputs from the DIP switches and outputs their status in the LEDs.
The following code was used for this task to make the DIP Switches read the input and then write them to the LED:

```
#include <hidef.h>      /* common defines and macros */
#include "derivative.h"      /* derivative-specific definitions */

void main(void) {

  DDRB=0xFF;                    //MAKE PORTB OUTPUT by assigning DDRB to FF
  DDRJ=0xFF;                   // Make PORTJ output in order to make the leds functional
  PTJ=0x00;                    // Turn off PTJ1 to allow the LEDs to show data
  DDRP=0xFF;                  // Make PORTP output
  PTP=0x0F;                  //TURN OFF 7SEG LED
  DDRH=0x00;           // assign it as an input - switches
 for(;;) {
    PORTB=PTH;   // send the value of the switches into port B which is the led
 }
}
```

Observations: the task was successfully done were it was observed on the Dragon Plus Trainer Board that the values appeared on the LEDs matches the value inserted on the switches.

2.2. TASK-2: Reading Key Pad and Writing it to LEDs

In this part of the experiment, using the code provided in labscript 4,a C program was written in order to take the inputs from the Key Pad and outputs their binary equivalent status in the LEDs.

The following code was used for this task:

```
#include <hidef.h>      /* common defines and macros */
#include "derivative.h"      /* derivative-specific definitions */
int key_pad(); // defining function key_pad
void main(void) {
  DDRA=0x0F;                                //MAKE PORTA INPUT to read the key_pad
  PUCR=0X01;       //turning onthe pull up resistors for the pins associated with the core of PORTA
  DDRB=0xFF;                 //MAKE PORTB OUTPUT by assigning DDRB to FF
  DDRJ=0xFF;                      // Make PORTJ output in order to make the leds functional
  PTJ=0x00;                    // Turn off PTJ1 to allow the LEDs to show data
  DDRP=0xFF;                   // Make PORTP output
  PTP=0x0F;                      //TURN OFF 7SEG LED
 for(;;) {
   PORTB=key_pad();   // send the value of the keypad into port B which is the led
 }
}
int key_pad(void) //function key_pan algorithm
{
int X;
while(1)
{
PORTA = 0XFE;       //enable first column of the keypad ( associated with 1,4,7,E)
```

```
X = PORTA;
if (X == 0xEE)return 0x01;
if (X == 0xDE)return 0x04;
if (X == 0xBE)return 0x07;
if (X == 0x7E)return 0x0E;
PORTA = 0XFD;              //enable 2nd column of the keypad ( associated with 2,5,8,0)
X = PORTA;
if (X == 0xED)return 0x02;
if (X == 0xDD)return 0x05;
if (X == 0xBD)return 0x08;
if (X == 0x7D)return 0x00;
PORTA = 0XFB;              //enable 3rd column of the keypad ( associated with 3,6,9,F)
X = PORTA;
if (X == 0xEB)return 0x03;
if (X == 0xDB)return 0x06;
if (X == 0xBB)return 0x09;
if (X == 0x7B)return 0x0F;
PORTA = 0XF7;              //enable 4th column of the keypad ( associated with A,B,C,D)
X = PORTA;
if (X == 0xE7)return 0x0A;
if (X == 0xD7)return 0x0B;
if (X == 0xB7)return 0x0C;
if (X == 0x77)return 0x0D;
}
}
```

Observations: the task was successfully done were it was observed on the Dragon Plus Trainer Board that the values appeared on the LCD display equivalent to the values inserted on the key pad.

2.3. Task-3: Writing to LCD

In this part of the experiment, a C language program was written using the LCD function calls (provided in files lcd.h and lcd.c) to print the following text message "ELCE 333 Lab" on the first line of Dragon 12 LCD and "Microprocessor" on the second line.

These two messages were displayed for 0.5 second and cleared for another 0.5 second and so on as a flashing text. The flashing text was done alternatively so that when the first line appears, the second disappear and vice versa.

The following code was used for this task:

```
#include <hidef.h>      /* common defines and macros */
#include "derivative.h"      /* derivative-specific definitions */
#include "lcd.h"    // include lcd class to call the functions

void main(void) {
LCD_Init(); // Initialize the LCD Controller
  for(;;) {
LCDWriteLine(1,"ELCE 333 Lab"); // Print the ASCII string(ELCE 333 Lab )on the 1st line of LCD display.
delay (500); // delay in 500 msec –equals 0.5 sec
LCD_clear_line(1); //Clear line 1
delay (500); // delay in 500 msec –equals 0.5 sec
LCDWriteLine(2,"Microprocessor");
delay (500); // delay in 500 msec –equals 0.5 sec
LCD_clear_line(2); ////Clear line 2
delay (500); // delay in 500 msec –equals 0.5 sec
  }
}
```

Observations: the task was successfully done were it was observed on the Dragon Plus Trainer Board that the LCD functions as it is required to.

2.4. Task-4: Reading DIP Switches and Writing them to LED's and LCD

In this part of the experiment, It was required to Re-examine the program of Task-1 and modify it to display the DIP Switches value in the LEDs as well as the LCD.
The following code was used for this task to make the DIP Switches read the input and then write them to the LED and LCD:

```
#include <hidef.h>      /* common defines and macros */
#include "derivative.h"      /* derivative-specific definitions */
#include "lcd.h"    // include lcd class to call the functions

void main(void) {
  DDRB=0xFF;                    //MAKE PORTB OUTPUT by assigning DDRB to FF
  DDRJ=0xFF;                     // Make PORTJ output in order to make the leds functional
  PTJ=0x00;                    // Turn off PTJ1 to allow the LEDs to show data
  DDRP=0xFF;                  // Make PORTP output
  PTP=0x0F;                  //TURN OFF 7SEG LED
  DDRH=0x00;                // assign it as an input - switches

LCD_Init(); // Initialize the LCD Controller
 for(;;) {
LCDWriteLine(1,""); // initiate the 1st line of LCD display  to be written at
LCDWriteInt(PTH);  //print the value of the DIP switch into the LCD display
PORTB=PTH; // send the value of the switches into port B which is the led
delay (1000) ; // delay in 1000 msec
LCD_clear_line(1); //Clear line 1
 }
}
```

Observations: the task was successfully done were it was observed on the Dragon Plus Trainer Board that the values appeared on the LEDs matches the value inserted on the switches and the values appeared on the LCD display equivalent to the values inserted on the switches.

2.5. Task-5: Interfacing LCD with Key Pad and Push Buttons

In this part of the experiment, it was required to write a C language program that prompts the user to enter a number. Where the number should be displayed as is on the LCD display second line if the switch is not on and should be divided by 2 if the switch is on.

```
#include <hidef.h>      /* common defines and macros */
#include "derivative.h"      /* derivative-specific definitions */
#include "lcd.h"    // include lcd class to call the functions
int key_pad(); // defining function key_pad
int a;
void main(void) {
int a;
  DDRA=0x0F;                                    //MAKE PORTA INPUT to read the key_pad
  PUCR=0X01;       //turning onthe pull up resistors for the pins associated with the core of
PORTA
  DDRB=0xFF;                 //MAKE PORTB OUTPUT by assigning DDRB to FF
  DDRJ=0xFF;                          // Make PORTJ output in order to make the leds functional
  PTJ=0x00;                  // Turn off PTJ1 to allow the LEDs to show data
  DDRP=0xFF;                  // Make PORTP output
  PTP=0x0F;                    //TURN OFF 7SEG LED
DDRH=0x00;                    // assign it as an input – switches
LCD_Init(); // Initialize the LCD Controller
 for(;;) {
LCDWriteLine(1, "Enter Number");// Print the ASCII string(Enter Number)on the 1st line of LCD
display
LCDWriteLine(2,""); // initiate the 2nd line of LCD display  to be written at
PORTB = PTH_PTH5;//read the switch
 if (PTH!=0){ // check if the switch is high
a = key_pad()/2; //divide the value of the key pad by 2 and enter it as an integer into a
   LCDWriteInt(a); // print the value of a in the first line of LCS
 } else
LCDWriteInt(key_pad()); // print the value of the key pad as it is
delay(1000); // delay
LCD_clear_line(2);// clear the first line } }
int key_pad(void) {//function key_pan algorithm
int X;
while(1){
PORTA = 0XFE;       //enable first column of the keypad ( associated with 1,4,7,E)
X = PORTA;
if (X == 0xEE)return 0x01;
if (X == 0xDE)return 0x04;
if (X == 0xBE)return 0x07;
if (X == 0x7E)return 0x0E;
PORTA = 0XFD;               //enable 2nd column of the keypad ( associated with 2,5,8,0)
X = PORTA;
if (X == 0xED)return 0x02;
if (X == 0xDD)return 0x05;
if (X == 0xBD)return 0x08;
if (X == 0x7D)return 0x00;
PORTA = 0XFB;              //enable 3rd column of the keypad ( associated with 3,6,9,F)
X = PORTA;
if (X == 0xEB)return 0x03;
if (X == 0xDB)return 0x06;
if (X == 0xBB)return 0x09;
if (X == 0x7B)return 0x0F;
PORTA = 0XF7;             //enable 4th column of the keypad ( associated with A,B,C,D)
X = PORTA;
if (X == 0xE7)return 0x0A;
if (X == 0xD7)return 0x0B;
if (X == 0xB7)return 0x0C;
if (X == 0x77)return 0x0D;
}}
```

Observations: in this task team failed to use the push button however they overcame this problem by replacing the push button by a switch key and the task was successfully done as observed on the Dragon Plus Trainer Board and the LCD functions as it is required to.

3. Conclusion and Recommendations

In conclusion, we learned the main concept of this laboratory experiment which is how to interface with the HCS12 board using C language programs. From the laboratory tasks , we learned how to use DIP switches and Key Pad for writing them into LEDs , how to display whatever line we want on it, how to write from the DIP switches and Key Pad and display into the LCD.Finally, we end up this experiment with grateful result .

4.Assignment Questions

1)

```
#include <hidef.h>      /* common defines and macros */
#include "derivative.h"
#include "lcd.h"

     /* derivative-specific definitions */

void main(void) {

  DDRB=0xFF;                  //MAKE PORTB OUTPUT by assigning DDRB to FF
  DDRJ=0xFF;                    // Make PORTJ output in order to make the leds functional
  PTJ=0x00;                   // Turn off PTJ1 to allow the LEDs to show data
  DDRP=0x0F;                    // Make PORTP output
  PTP=0x0F;                     //TURN OFF 7SEG LED
DDRH=0x00;                // assign it as an input - switches
    PORTB =1;



 for(;;)

 {
 while(1)   {


  if(PTH==0xFF) {
   delay(500);

 PORTB =PORTB>>1;
      delay(500);
     if(PORTB==0x80)
     PORTB=1;
      delay(500);

 PORTB =PORTB>>1;




   }
 else    {

       delay(500);

 PORTB =PORTB<<1;
      delay(500);
      if(PORTB==1)
     PORTB=0x80;
       delay(500);

 PORTB =PORTB<<1;

 }
 }
 }

}
```

2)

```
#include <hidef.h>      /* common defines and macros */
#include "derivative.h"      /* derivative-specific definitions */
#include "lcd.h"


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;
}
}


    void main(void) {



int num1, num2, result;
char op;
DDRH = 0x00;
PUCR = 0xFF;
//EnableInterrupts;
LCD_Init () ;
for(;;) {

if (PTH_PTH4 == 0 && PTH_PTH5 ==0 && PTH_PTH6 ==0 && PTH_PTH7 == 0) num1= 0;
if (PTH_PTH4 == 1 && PTH_PTH5 ==0 && PTH_PTH6 ==0 && PTH_PTH7 == 0) num1= 1;
if (PTH_PTH4 == 0 && PTH_PTH5 ==1 && PTH_PTH6 ==0 && PTH_PTH7 == 0) num1= 2;
if (PTH_PTH4 == 1 && PTH_PTH5 ==1 && PTH_PTH6 ==0 && PTH_PTH7 == 0) num1= 3;
if (PTH_PTH4 == 0 && PTH_PTH5 ==0 && PTH_PTH6 ==1 && PTH_PTH7 == 0) num1= 4;
if (PTH_PTH4 == 1 && PTH_PTH5 ==0 && PTH_PTH6 ==1 && PTH_PTH7 == 0) num1= 5;
if (PTH_PTH4 == 0 && PTH_PTH5 ==1 && PTH_PTH6 ==1 && PTH_PTH7 == 0) num1= 6;
if (PTH_PTH4 == 1 && PTH_PTH5 ==1 && PTH_PTH6 ==1 && PTH_PTH7 == 0) num1= 7;
if (PTH_PTH4 == 0 && PTH_PTH5 ==0 && PTH_PTH6 ==0 && PTH_PTH7 == 1) num1= 8;
if (PTH_PTH4 == 1 && PTH_PTH5 ==0 && PTH_PTH6 ==0 && PTH_PTH7 == 1) num1= 9;
if (PTH_PTH4 == 0 && PTH_PTH5 ==1 && PTH_PTH6 ==0 && PTH_PTH7 == 1) num1= 10;
if (PTH_PTH4 == 1 && PTH_PTH5 ==1 && PTH_PTH6 ==0 && PTH_PTH7 == 1) num1= 11;
if (PTH_PTH4 == 0 && PTH_PTH5 ==0 && PTH_PTH6 ==1 && PTH_PTH7 == 1) num1= 12;
if (PTH_PTH4 == 1 && PTH_PTH5 ==0 && PTH_PTH6 ==1 && PTH_PTH7 == 1) num1= 13;
if (PTH_PTH4 == 0 && PTH_PTH5 ==1 && PTH_PTH6 ==1 && PTH_PTH7 == 1) num1= 14;
if (PTH_PTH4 == 1 && PTH_PTH5 ==1 && PTH_PTH6 ==1 && PTH_PTH7 == 1) num1= 15;
```

```
if (PTH_PTH4 == 1 && PTH_PTH5 ==1 && PTH_PTH6 ==1 && PTH_PTH7 == 1) num1= 15;
if (PTH_PTH3 == 1) {op= '/';
 result=num1/num2; }
if (PTH_PTH2 ==1) {op= '*'; result= num1*num2;}
if (PTH_PTH1 ==1) {op= '-'; result= num1-num2;}
if (PTH_PTH0 ==1) {op= '/'; result= num1+num2;}

LCDWriteLine(1, "No.1=");
LCDWriteInt(num1);
delay(500);
LCDWriteLine(1, "No.2=");
num2= key_pad();

LCDWriteInt(num2);
LCDWriteLine(2, "op=");
LCDWriteChar(op);
LCDWriteLine(2, "res=");
LCDWriteInt(result);


   _FEED_COP(); /* feeds the dog */
}
   }
```