Mapúa University
School of EECE

COE121L/C1 – MICROPROCESSOR SYSTEMS
4th Quarter SY 2016-2017

4-DIGIT KEY CODE SECURITY SYSTEM

By

Dacanay, Jim Alfred M.
De Castro, Mary Josanne A.
Gozar, Lea Adelle A.
Salas, Joyce Ann O.

Engr. Isagani Villamor
Professor

ACKNOWLEDGEMENT

We would like to express our gratitude to our professor, Engr. Isagani Villamor for giving us the opportunity to apply everything that we have learned from microprocessor systems laboratory through perform our design experiment entitled “4 Digit Keycode Security System”.

This design would allow us to test and apply our skills and knowledge in creating an assembly code using the CodeWarrior Development Software and the HCS12C microcontroller kit.

We would also like to thank our family and friends for being our support in doing this project. Through them, we can do this design experiment with enough inspiration such that we can finish our tasks for this project with ease.

TABLE OF CONTENTS

ABSTRACT

Our design experiment is a security system. It aims to utilize the HCS12 microcontroller to develop an algorithm that will authenticate the password inputted by the user. The algorithm will be programmed using the CodeWarrior Development Software. For our security system design, the user will input a password of four characters using the push buttons available in the microcontroller. The algorithm will then store the inputted password and compare it with the default password provided in the system and provide an output depending on whether the input of the user matches with the default password or not. If the input is correct, LEDs will light up. If the input is not correct, no LEDs will light up and the program will end.

Keywords: I/O Ports, Password, Keycode, Compare, Store

OBJECTIVES

- Write assembly programs for the microcontroller;
- Design and assemble a program in the CodeWarior integrated development environment using the skills they have acquired during the whole term;
- Simulate the correct program using the trainer kit;
- Download the program to the microcontroller and run it while the CodeWarrior hardware;
- Test and run program on a student learning kit;
- Retrieve information from an input device and output to an output device.

REQUIREMENTS

- The group should be able to enter and simulate assembly language program;
- The program should receive an input combination via 8 push buttons but only accept 4 inputs.
- The program will light the LEDs 1, 2, 3 and 4 if the input password is correct otherwise it will not light the LEDs and the program will end.
- The program will utilize a “start” option.

MATERIALS

- 1 pc Freescale HCS12C Family Student Learning it
- CodeWarrior Development Software

DISCUSSION

This project design was just a complicated pattern of the 4th laboratory experiment we had taken entitled “Digital Input and Output” in our Microprocessors Laboratory. The same software was used namely the CodeWarrior Development Software and the Freescale HCS12C Family Student Learning Kit. The code used for this design project was altered to match the needed requirements proposed.

I/O Ports

It was discussed in the 4th laboratory experiment that all microcontrollers have I/O ports. These I/O ports can be used to receive information from an input device or provide information through an output device. These ports can be made to be an input alone or an output alone, but they can also be used as input and output at the same time.

In this design experiment, these I/O ports are used for the input and output devices. The input devices are the push buttons while the output devices are Light Emitting Diodes (LEDs). These ports are connected to the push buttons are programmed to be solely input, and the ports are connected to the LEDs are programmed to be solely output.

Array

Array is used to store a group of elements. The elements stored in the array must be of the same data type such as an integer or a string. The elements stored in the array can be easily sorted or searched. In assembly language, the data definition derivatives to the assembler are used for allocating storage for variables. The variable stored could be initialized with some specific value and this initialized value could be specified in binary, decimal or hexadecimal form.

HCS12 Digital Input and Output Ports

HCS12 microcontroller has a couple of ports that can be used as either a digital input, output or both. The PAD0 to PAD7 are A/D channel bits that can be used for general inputs or outputs considering that they aren't being used for analog input. The PTAD or port AD I/O register can be used for both digital inputs and outputs. The PTIAD or port AD input register can only be used as input. Lastly the PORTA can be used as a digital input only read register.

PROCEDURE

1. Launch the CodeWarrior IDE.
2. Create New Project.
3. In the New Project Wizard
a. Enter a Project Name.
b. Enter a Location for the project and click OK.
c. Select MC99S12C32 as the derivative you want to use and click Next.
d. Check the language support you want (Assembly) and click Next.
e. Relocatable Assembly.
f. Full Chip Simulation and P&E Multilink/Cyclone Pro and click Finish.
4. Switch to the Full Chip simulation for debugging by clicking the pull-down arrow and selecting Full Chip Simulation. When using the HCS12 hardware, choose P&E Multilink/Cyclone Pro.
5. Create listing files by opening the simulator settings panel by clicking the icon shown above.

a. Click on + next Target in the Target Settings.
b. Highlight Assembler for HCS12 and in that panel click on Options.
   i. Check Generate a listing file.
   ii. Check Object File Format. Choose ELF/DWARF 2.0 Object File Format
   iii. Click OK.
c. Click OK.
6. Open the sources folder.
7. Double click on main.asm and enter the following code after the comment:

```
;CONNECTIONS:
;Push Button 1 - PORTAD0_BIT0 - PIN 18
;Push Button 2 - PORTAD0_BIT1 - PIN 20
;Push Button 3 - PORTAD0_BIT2 - PIN 22
;Push Button 4 - PORTAD0_BIT3 - PIN 24
;Push Button 5 - PTT_BIT6 - PIN 58
;Push Button 6 - PTT_BIT7 - PIN 60
;Push Button 7 - PORTAD0_BIT6 - 12
;Push Button 8 - PORTAD0_BIT7 - 10
;LED 1 - PIN 55
;LED 2 - PIN 53
;LED 3 - PIN 51
;LED 4 - PIN 49


;**************************************
; Define the entry point for the main program
 XDEF Entry, main
 XREF __SEG_END_SSTACK ;note double underbar
;*****************************
; Include files
 include mc9s12c32.inc
; Based on CPU DB MC9S12C32, version 2.87.286
Data: SECTION
BIT0 EQU %00000001
BIT1 EQU %00000010
BIT2 EQU %00000100
BIT3 EQU %00001000
BIT4 EQU %00010000
BIT5 EQU %00100000
BIT6 EQU %01000000
BIT7 EQU %10000000


PASSWORD1:     DS.B 1
PASSWORD2:     DS.B 1
PASSWORD3:     DS.B 1
PASSWORD4:     DS.B 1
;*****************************
; Code Section
MyCode: SECTION
Entry:
main:
;*****************************
; Initialize stack pointer register
 lds #__SEG_END_SSTACK
;*****************************
   bset ATDDIEN,BIT0|BIT1|BIT2|BIT3|BIT4|BIT5|BIT6|BIT7 ;set ATDDIEN bits
   bclr DDRAD, BIT0|BIT1|BIT2|BIT3|BIT4|BIT5|BIT6|BIT7
```

```
;clear DDRAD bit
   bset DDRA,
BIT0|BIT1|BIT2|BIT3|BIT4|BIT5|BI
T6|BIT7
;set DDRA bit
   bset PTT, BIT0|BIT6|BIT7
   bclr DDRT, BIT5|BIT6|BIT7

   bset
DDRB,BIT0|BIT1|BIT2|BIT3|BIT4

main_loop:
   ldaa 0
   ldab #0
   staa PASSWORD1
   staa PASSWORD2
   staa PASSWORD3
   staa PASSWORD4

;Choose
Password/////////////////////////////
///////////////////////////////////////
///////////////////////////////////////
///////////////////////////////////////
///////////////
choosepass:
   bclr
PORTA,BIT0|BIT1|BIT2|BIT3|BIT4|
BIT5
   bset PORTB,BIT0
   bclr PORTB,BIT1
   bclr PORTB,BIT2
   bclr PORTB,BIT3

;First Digit
firstnum:
   brset PORTAD0,BIT0,two
   ldaa #1
   staa PASSWORD1
   bset PORTA,BIT0
   call shortDelay ;//debounce
   lbra secondnum

two:
   brset PORTAD0,BIT1,three
   ldaa #2
   staa PASSWORD1
   bset PORTA,BIT0
   call shortDelay
   lbra secondnum

three:
   brset PORTAD0,BIT2,four
   ldaa #3
   staa PASSWORD1
   bset PORTA,BIT0
   call shortDelay
   lbra secondnum

four:
   brset PORTAD0,BIT3,five
   ldaa #4
   staa PASSWORD1
   bset PORTA,BIT0
   call shortDelay
   lbra secondnum
firstnum1:
   bra firstnum
five:
   brset PTT,BIT6,six
   ldaa #5
   staa PASSWORD1
   bset PORTA,BIT0
   call shortDelay
   lbra secondnum

six:
   brset PTT,BIT7,seven
   ldaa #6
   staa PASSWORD1
   bset PORTA,BIT0
   call shortDelay
   lbra secondnum

seven:
   brset PORTAD0,BIT6,eight
   ldaa #7
   staa PASSWORD1
```

```
  bset PORTA,BIT0
  call shortDelay
  lbra secondnum

eight:
  brset PORTAD0,BIT7,firstnum1
  ldaa #8
  staa PASSWORD1
  bset PORTA,BIT0
  call shortDelay
  lbra secondnum


;Second
Digit//////////////////////////////////
///////////////////////////////////////
///////////////////////////////////////
///////////////////////////////


secondnum:

  brset PORTAD0,BIT0,two1
  ldaa #1
  staa PASSWORD2
  bset PORTA,BIT1
  call shortDelay
  lbra thirdnum

two1:
  brset PORTAD0,BIT1,three1
  ldaa #2
  staa PASSWORD2
  bset PORTA,BIT1
  call shortDelay
  lbra thirdnum

three1:
  brset PORTAD0,BIT2,four1
  ldaa #3
  staa PASSWORD2
  bset PORTA,BIT1
  call shortDelay
  lbra thirdnum
```

```

four1:
  brset PORTAD0,BIT3,five1
  ldaa #4
  staa PASSWORD2
  bset PORTA,BIT1
  call shortDelay
  lbra thirdnum
secondnum1:
  bra secondnum
five1:
  brset PTT,BIT6,six1
  ldaa #5
  staa PASSWORD2
  bset PORTA,BIT1
  call shortDelay
  lbra thirdnum

six1:
  brset PTT,BIT7,seven1
  ldaa #6
  staa PASSWORD2
  bset PORTA,BIT0
  call shortDelay
  lbra thirdnum

seven1:
  brset PORTAD0,BIT6,eight1
  ldaa #7
  staa PASSWORD2
  bset PORTA,BIT1
  call shortDelay
  lbra thirdnum

eight1:
  brset
PORTAD0,BIT7,secondnum1
  ldaa #8
  staa PASSWORD2
  bset PORTA,BIT1
  call shortDelay
  lbra thirdnum

;Third
```

```
Digit//////////////////////////////////
///////////////////////////////////////
///////////////////////////////////////
///////////////////////////////////////
//////

thirdnum:

   brset PORTAD0,BIT0,two2
   ldaa #1
   staa PASSWORD3
   bset PORTA,BIT2
   call shortDelay
   lbra fourthnum

two2:
   brset PORTAD0,BIT1,three2
   ldaa #2
   staa PASSWORD3
   bset PORTA,BIT2
   call shortDelay
   lbra fourthnum

three2:
   brset PORTAD0,BIT2,four2
   ldaa #3
   staa PASSWORD3
   bset PORTA,BIT2
   call shortDelay
   lbra fourthnum

four2:
   brset PORTAD0,BIT3,five2
   ldaa #4
   staa PASSWORD3
   bset PORTA,BIT2
   call shortDelay
   lbra fourthnum
thirdnum1:
   bra thirdnum
five2:
   brset PTT,BIT6,six2
   ldaa #5
   staa PASSWORD3
   bset PORTA,BIT2
   call shortDelay
   lbra fourthnum

six2:
   brset PTT,BIT7,seven2
   ldaa #6
   staa PASSWORD3
   bset PORTA,BIT2
   call shortDelay
   lbra fourthnum

seven2:
   brset PORTAD0,BIT6,eight2
   ldaa #7
   staa PASSWORD3
   bset PORTA,BIT2
   call shortDelay
   lbra fourthnum

eight2:
   brset PORTAD0,BIT7,thirdnum1
   ldaa #8
   staa PASSWORD3
   bset PORTA,BIT2
   call shortDelay
   lbra fourthnum

;Fourth
Digit//////////////////////////////////
///////////////////////////////////////
///////////////////////////////////////
///////////////////////////////////////
////////

fourthnum:

   brset PORTAD0,BIT0,two3
   ldaa #1
   staa PASSWORD4
   bset PORTA,BIT3
   call shortDelay
   lbra enterpass
```

```
two3:
   brset PORTAD0,BIT1,three3
   ldaa #2
   staa PASSWORD4
   bset PORTA,BIT3
   call shortDelay
   bra enterpass

three3:
   brset PORTAD0,BIT2,four3
   ldaa #3
   staa PASSWORD4
   bset PORTA,BIT3
   call shortDelay
   lbra enterpass

four3:
   brset PORTAD0,BIT3,five3
   ldaa #4
   staa PASSWORD4
   bset PORTA,BIT3
   call shortDelay
   lbra enterpass
fourthnum1:
   bra fourthnum
five3:
   brset PTT,BIT6,six3
   ldaa #5
   staa PASSWORD4
   bset PORTA,BIT3
   call shortDelay
   lbra enterpass

six3:
   brset PTT,BIT7,seven3
   ldaa #6
   staa PASSWORD4
   bset PORTA,BIT3
   call shortDelay
   lbra enterpass

seven3:
   brset PORTAD0,BIT6,eight3
   ldaa #7
   staa PASSWORD4
   bset PORTA,BIT3
   call shortDelay
   lbra enterpass

eight3:
   brset PORTAD0,BIT7,fourthnum1
   ldaa #8
   staa PASSWORD4
   bset PORTA,BIT3
   call shortDelay
   lbra enterpass

;Enterpass///////////////////////////
///////////////////////////////////////
///////////////////////////////////////
///////////////////////////////////////
/////

enterpass:

   call Delay

   bclr
PORTA,BIT0|BIT1|BIT2|BIT3|BIT4|
BIT5
;
   bclr PORTB,BIT0
   bset PORTB,BIT1
   bclr PORTB,BIT2
   bclr PORTB,BIT3

firstPass:
   brset PORTAD0,BIT0,two4
   ldaa #1
   cmpa PASSWORD1
   lbne incorrectPass
   bset PORTA,BIT0
   call shortDelay
   lbra secondPass

two4:
   brset PORTAD0,BIT1,three4
   ldaa #2
   cmpa PASSWORD1
   lbne incorrectPass
   bset PORTA,BIT0
   call shortDelay
   lbra secondPass

three4:
   brset PORTAD0,BIT2,four4
   ldaa #3
   cmpa PASSWORD1
   lbne incorrectPass
   bset PORTA,BIT0
   call shortDelay
   lbra secondPass

four4:
   brset PORTAD0,BIT3,five4
   ldaa #4
   cmpa PASSWORD1
   lbne incorrectPass
   bset PORTA,BIT0
   call shortDelay
   lbra secondPass
firstPass1:
   bra firstPass
five4:
   brset PTT,BIT6,six4
   ldaa #5
   cmpa PASSWORD1
   lbne incorrectPass
   bset PORTA,BIT0
   call shortDelay
   lbra secondPass

six4:
   brset PTT,BIT7,seven4
   ldaa #6
   cmpa PASSWORD1
   lbne incorrectPass
   bset PORTA,BIT0
   call shortDelay
   lbra secondPass

seven4:
   brset PORTAD0,BIT6,eight4
   ldaa #7
   cmpa PASSWORD1
   lbne incorrectPass
   bset PORTA,BIT0
   call shortDelay
   lbra secondPass

eight4:
   brset PORTAD0,BIT7,firstPass1
   ldaa #8
   cmpa PASSWORD1
   lbne incorrectPass
   bset PORTA,BIT0
   call shortDelay
   lbra secondPass


;//////////////////////////////////////
///////////////////////////////////////
///////////////////////////////////////
///////////////////////////
secondPass:

   brset PORTAD0,BIT0,two5
   ldaa #1
   cmpa PASSWORD2
   lbne incorrectPass
   bset PORTA,BIT1
   call shortDelay
   lbra thirdPass

two5:
   brset PORTAD0,BIT1,three5
   ldaa #2
   cmpa PASSWORD2
   lbne incorrectPass
   bset PORTA,BIT1
   call shortDelay
   lbra thirdPass

three5:
   brset PORTAD0,BIT2,four5
   ldaa #3
   cmpa PASSWORD2
   lbne incorrectPass
   bset PORTA,BIT1
   call shortDelay
   lbra thirdPass

four5:
   brset PORTAD0,BIT3,five5
   ldaa #4
   cmpa PASSWORD2
   lbne incorrectPass
   bset PORTA,BIT1
   call shortDelay
   lbra thirdPass
secondPass1:
   bra secondPass
five5:
   brset PTT,BIT6,six5
   ldaa #5
   cmpa PASSWORD2
   Lbne incorrectPass
   bset PORTA,BIT1
   call shortDelay
   lbra thirdPass

six5:
   brset PTT,BIT7,seven5
   ldaa #6
   cmpa PASSWORD2
   lbne incorrectPass
   bset PORTA,BIT0
   call shortDelay
   lbra thirdPass

seven5:
   brset PORTAD0,BIT6,eight5
   ldaa #7
   cmpa PASSWORD2
   lbne incorrectPass
   bset PORTA,BIT1
   call shortDelay
   lbra thirdPass

eight5:
   brset
PORTAD0,BIT7,secondPass1
   ldaa #8
   cmpa PASSWORD2
   lbne incorrectPass
   bset PORTA,BIT1
   call shortDelay
   lbra thirdPass

;//////////////////////////////////////
///////////////////////////////////////
///////////////////////////////////////
///////////////////////////////////////
//

thirdPass:

   brset PORTAD0,BIT0,two6
   ldaa #1
   cmpa PASSWORD3
   lbne incorrectPass
   bset PORTA,BIT2
   call shortDelay
   Lbra fourthPass

two6:
   brset PORTAD0,BIT1,three6
   ldaa #2
   cmpa PASSWORD3
   lbne incorrectPass
   bset PORTA,BIT2
   call shortDelay
   lbra fourthPass

three6:
   brset PORTAD0,BIT2,four6
   ldaa #3
   cmpa PASSWORD3
   lbne incorrectPass
   bset PORTA,BIT2
   call shortDelay
   lbra fourthPass

four6:
   brset PORTAD0,BIT3,five6
   ldaa #4
   cmpa PASSWORD3
   lbne incorrectPass
   bset PORTA,BIT2
   call shortDelay
   lbra fourthPass
thirdPass1:
   bra thirdPass
five6:
   brset PTT,BIT6,six6
   ldaa #5
   cmpa PASSWORD3
   lbne incorrectPass
   bset PORTA,BIT2
   call shortDelay
   lbra fourthPass

six6:
   brset PTT,BIT7,seven6
   ldaa #6
   cmpa PASSWORD3
   lbne incorrectPass
   bset PORTA,BIT2
   call shortDelay
   lbra fourthPass

seven6:
   brset PORTAD0,BIT6,eight6
   ldaa #7
   cmpa PASSWORD3
   lbne incorrectPass
   bset PORTA,BIT2
   call shortDelay
   lbra fourthPass

eight6:
   brset PORTAD0,BIT7,thirdPass1
   ldaa #8
   cmpa PASSWORD3
   lbne incorrectPass
   bset PORTA,BIT2
   call shortDelay
   lbra fourthPass
```

```
;//////////////////////////////////////
///////////////////////////////////////
///////////////////////////////////////
///////////////////////////////////////
////

fourthPass:

   brset PORTAD0,BIT0,two7
   ldaa #1
   cmpa PASSWORD4
   lbne incorrectPass
   bset PORTA,BIT3
   call shortDelay
   lbra accessgranted

two7:
   brset PORTAD0,BIT1,three7
   ldaa #2
   cmpa PASSWORD4
   lbne incorrectPass
   bset PORTA,BIT3
   call shortDelay
   lbra accessgranted

three7:
   brset PORTAD0,BIT2,four7
   ldaa #3
   cmpa PASSWORD4
   lbne incorrectPass
   bset PORTA,BIT3
   call shortDelay
   lbra accessgranted

four7:
   brset PORTAD0,BIT3,five7
   ldaa #4
   cmpa PASSWORD4
   lbne incorrectPass
   bset PORTA,BIT3
   call shortDelay
   lbra accessgranted
fourthPass1:
   bra fourthPass
five7:
   brset PTT,BIT6,six7
   ldaa #5
   cmpa PASSWORD4
   lbne incorrectPass
   bset PORTA,BIT3
   call shortDelay
   lbra accessgranted

six7:
   brset PTT,BIT7,seven7
   ldaa #6
   cmpa PASSWORD4
   lbne incorrectPass
   bset PORTA,BIT3
   call shortDelay
   lbra accessgranted

seven7:
   brset PORTAD0,BIT6,eight7
   ldaa #7
   cmpa PASSWORD4
   lbne incorrectPass
   bset PORTA,BIT3
   call shortDelay
   lbra accessgranted

eight7:
   brset PORTAD0,BIT7,fourthPass1
   ldaa #8
   cmpa PASSWORD4
   lbne incorrectPass
   bset PORTA,BIT3
   call shortDelay
   lbra accessgranted

;///////////////////////////////////////
////////////////////////////////////////
////////////////////////////////////////
////////////////////////////////////////

incorrectPass:
   ;bset PORTA,BIT4
```

```
   bclr PORTB,BIT0
   bclr PORTB,BIT1
   bclr PORTB,BIT2
   bclr PORTB,BIT3

   incb
   cmpb #3
   beq passLimit
   call Delay
   lbra enterpass

passLimit:
   ;bset PORTA,BIT4
   bset PORTB,LED1
   bset PORTB,LED2
   bset PORTB,LED3
   bset PORTB,LED4

   call Delay

   bclr PORTB,LED1
   bclr PORTB,LED2
   bclr PORTB,LED3
   bclr PORTB,LED4

   call Delay

   bset PORTB,LED1
   bset PORTB,LED2
   bset PORTB,LED3
   bset PORTB,LED4

   call Delay

   bclr PORTB,LED1
   bclr PORTB,LED2
   bclr PORTB,LED3
   bclr PORTB,LED4

   call Delay

   bset PORTB,LED1
   bset PORTB,LED2
   bset PORTB,LED3
   bset PORTB,LED4

   call Delay


   bclr PORTB,LED1
   bclr PORTB,LED2
   bclr PORTB,LED3
   bclr PORTB,LED4
   lbra endless_loop

endless_loop:
   bra endless_loop

accessgranted:
   ;bset PORTA,BIT5
   bset PORTB,LED1
   bset PORTB,LED2
   bset PORTB,LED3
   bset,PORTB,LED4
   brset PORTAD0,BIT0,cont  ;Reset
Program - SAME
PASSWORD///////////////////////////
/////////////////////////
   lbra enterpass
cont:
   brset
PORTAD0,BIT1,accessgranted
;Reset Program - Enter New
Password///////////////////////////
   bclr
PORTA,BIT0|BIT1|BIT2|BIT3|BIT4|
BIT5
   call shortDelay
   lbra main_loop


;DELAYS//////////////////////////////
////////////////////////////////////////
////////////////////////////////////////
//////////////////////////////////
shortDelay:
   ldx $1
```

```
shortDelay2:
   nop
   dex
   cpx #0
   bhi shortDelay2
   rtc

 Delay:
      ldy $0002

    Outer:
      dey
      beq All_done
      ldx $FFFF

    Inner:
      Dex
      Bne Inner
      Bra Outer

      All_done:
      rtc
```

QUESTIONS WITH ANSWERS

1. How must the data direction control bit in the case register be initialized so that the bit is an output? An input?

· At reset, control bit should be ‘0’ to be initialized as an input. At set, control bit should be ‘1’ to indicate that the register is used as an output

2. Why do microcontroller’s I/O ports operate as input ports when it is reset, even though they may be connected to output hardware?

· It’s safest to connect them to the input because if it is first connected to the input hardware and it is an output port, the devices may be damaged.

3. What was used as an indicator while setting the default password?

. The LED 1 was used as an indicator while setting the default password.

4. What was used as an indicator when the password to be compared is to be input into the system?

. The LED 2 was used as an indicator for this step.

5. What is the output when the correct password is entered?

· All of the LEDs will light up.

6. What will be the output when the wrong password is entered?

· The LEDs will blink after 3 attempts.

7. Summarize what the algorithm for this design experiment does.

· The main point of this algorithm is to store and compare input data.

DISCUSSION

In this design project, we were able to apply some of the things we learned from the Microprocessors Laboratory. Particularly, experiment 4 which is the Digital Input and Output. Our goal is to produce a working program that can compare an input password using four push buttons to a default password that can be set on the microprocessor. Using the CodeWarrior software, we were able to write a code for this program and compile it for errors. Errors happened during the writing and compiling of the code but through successive debugging, we were able to remove these errors.

Some instructions used for this design project are based from the instructions from experiment 4. The PTAD or port AD I/O register was used for the push buttons and LEDs since this port can be used as an input port or an output port. The PTAD is being used for determining which push buttons are pressed. The push buttons are used to input the desired password, as well as the password to be compared to the set default password. After inputting a four-digit password, the program will start to compare the input password to the default password. For storing the password, the PORTA is being used. The default password to be set is stored in Bit 0 to Bit 3, Bit 0 being the first digit and Bit 3 being the last. The comparing was done by using the cmpa instruction. As for counting the attempts, the cmpb instruction is being used since the register B is the counter.

The LED 1 was used as an indicator when the default password to be set by the user can already be inputted into the system; while the LED 2 was used as an indicator that the user can now enter a password that would be compared to the default password that has already been set. For the system to remember the attempts it has accumulated, the incb is being used. When the two passwords match, the LEDs will light up. If the password did not match after 3 attempts, the LEDs will blink. After the comparison has been made, the push button 1 can be pressed to reset the program without having to set another default password. When it is desired to set a new default password, the push button 2 can be pressed.

REMARKS AND CONCLUSION

For this experiment, we are tasked to write an assembly program of our own design for the microcontroller using CodeWarrior Development Software and the Freescale HCS12C Family Learning Kit. We were able to design a project and create a functional program that can function as a security system.

For our security system design, the user will input a password of four characters using the push buttons available in the microcontroller. The algorithm will then store the inputted password and compare it with the default password provided by the user. If the inputted value matches with the default password set by the user, then the microcontroller will light the LEDs. If the values do not match even after 3 attempts, then the LEDs will blink.

In programming language, an array is used to store elements so that they are easy to be searched and sorted. Storing data can be different for each programming language. In this experiment the data stored which is the inputted password was compared to the default password.

Our program used I/O ports. These ports are what we used to retrieve input information from an input device and output data to an output device. The input devices that we utilized are the push buttons in the microcontroller, and the output device is the Light Emitting Diode. The input device and output device was connected to the PTAD which can be used either an input port or output port.