ENGINE

Teaching online electronics, microcontrollers and programming in Higher Education

Output 2: Online Course for Microcontrollers: syllabus, open educational resources

Practice leaflet: Module_2-4 LCD 16x2

Lead Partner: International Hellenic University (IHU)

Authors: Theodosios Sapounidis [IHU], Aristotelis Kazakopoulos [IHU], Aggelos Giakoumis [IHU], Sokratis Tselegkaridis [IHU]

Erasmus+

GRANT NUMBER: 2020-1-PL01-KA226-HE-095653

Declaration

This report has been prepared in the context of the ENGINE project. Where other published and unpublished source materials have been used, these have been acknowledged.

Copyright

Funding Disclaimer

This project has been funded with support from the European Commission. This report reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Table of Contents

Executive summary

In this Module we will use PIC18F4550 with a Liquid Crystal Display (LCD).

Chapter 1: **Overview**

Table 1. Overview

Title / short summary	**4. Liquid Crystal Display (LCD) 16x2**
Expected learning outcomes	• The student will be able to write messages on the LCD • The student will be able to send variable values to the LCD • The student will be able to design a simple alarm system with messages on the LCD • The student will be able to load and animate a microcontroller program in the Proteus Design Suite
Keywords	LCD 16x2, inputs/outputs
Duration	The duration of the module_2-4 is 3 hours • Presentation of the module_2-4 by the teacher, 30 minutes • 1st activity, write a message on the LCD, 30 minutes • 2nd activity, count the 1s of PORTD and show the result on the LCD, 45 minutes • 3rd activity, simple alarm system, 75 minutes
Involved	**The teacher:** Presents the slides associated with the module_2-4 and answers question **The students:**

	Draw circuits in Proteus Schematic, write programs in C language, load programs to a microcontroller and run the simulation using the Proteus Design Suite
Assignment	At the end of the Module_2-4 will be given: • Open Project
Educational tools and equipment	• **Material:** PC • **Software:** CCS C compiler, Proteus Design Suite
Prerequisites / pre-existing knowledge	• The student must be familiarized with the Proteus Design Suite (link1) • The student must be completed Module_2-1, Module_2-2 and Module_2-3
Educational content	• CCS C Compiler manual (C Compiler Reference Manual) • MICROCHIP, PIC18F2455/2550/4455/4550 Data Sheet • Module_2-4 slides • Module_2-4 Evaluation leaflet • Module_2-4 Open project leaflet • Module_2-4 Programs, Schematic Proteus (Compressed folder)
Tips	***Tip1.*** *Carefully check the LCD driver for proper pin connection* ***Tip2.*** *The microcontroller reads 1 when the magnetic reed switches are closed, while when they are open it reads 0*

Chapter 2: **Activities**

2.1 Activity 1. Display a message

The purpose of this activity is to display a message on the LCD 16x2.

Table 2. Activity 1

Activity 1st (30 minutes)	**Step 1.** The circuit is drawn in the Proteus Design Suite. **Step 2.** The program in C language is written. **Step 3.** The program is compiled with the use of CCS C compiler to the microcontroller machine code. **Step 4.** The machine code is loaded to the microcontroller. **Step 5.** The animation is activated. **Step 6.** Modifications and discussion.
Step 1 (10 minutes)	Draw the circuit of the picture in the Proteus Design Suite.

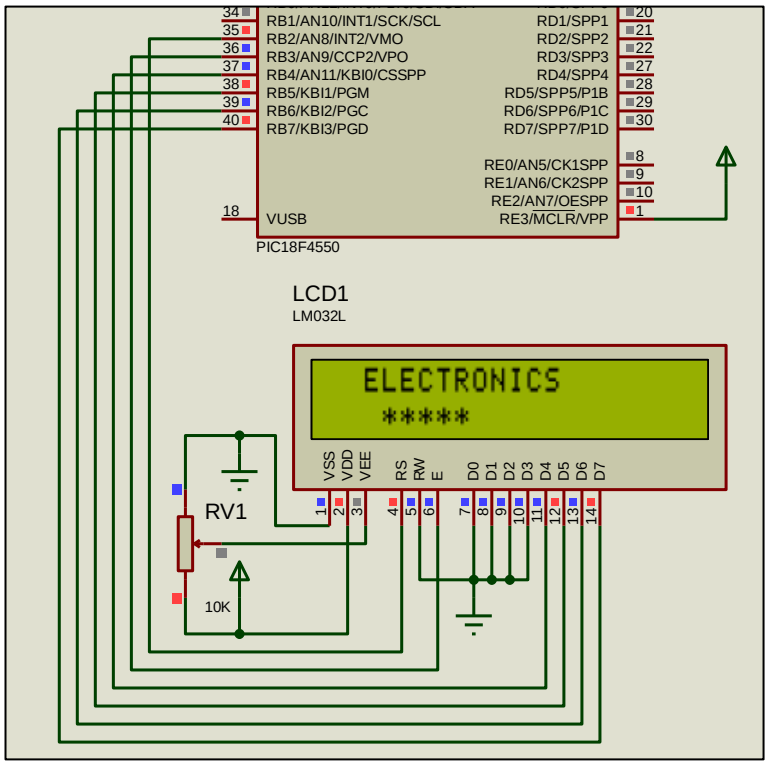
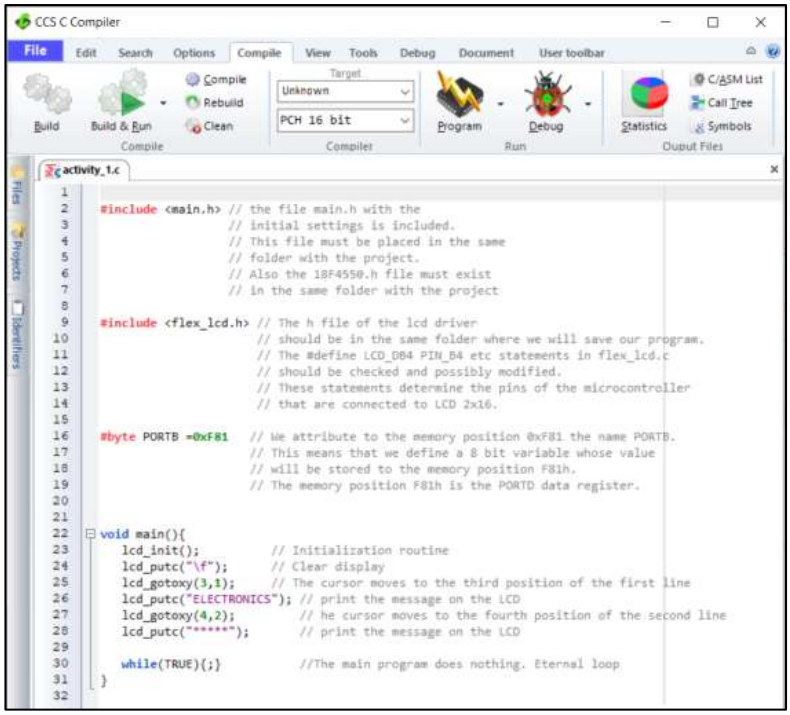


Figure 1. Message on the LCD 16x2

Step 2
(5 minutes)

Write in CCS Compiler the program in C language

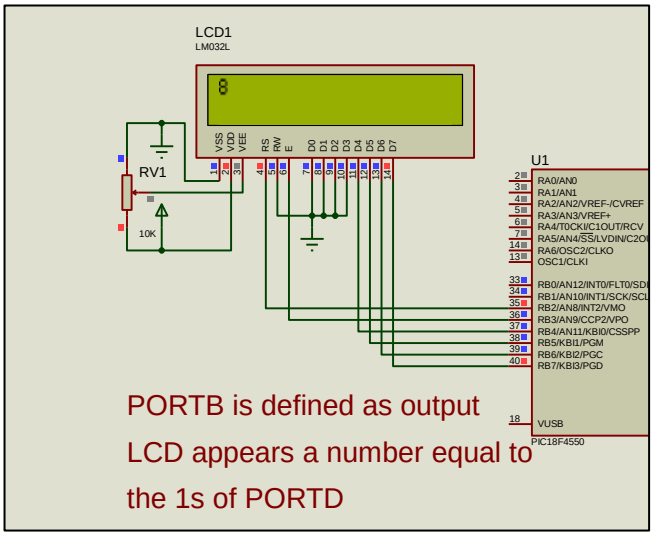
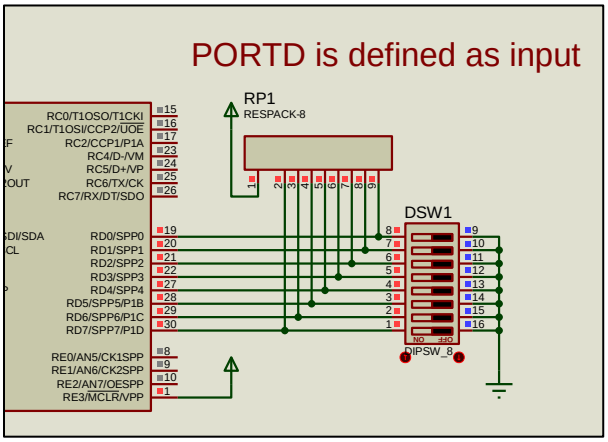
```
#include <main.h> // the file main.h with the
                  // initial settings is included.
                  // This file must be placed in
the same
                  // folder with the project.
                  // Also the 18F4550.h file must
exist
                  // in the same folder with the
project

#include <flex_lcd.h> // The h file of the lcd
driver
                      // should be in the same
folder where we will save our program.
                      //   The   #define   LCD_DB4
PIN_B4 etc statements in flex_lcd.c
                      // should be checked and
possibly modified.
                      //     These     statements
determine the pins of the microcontroller
                      // that are connected to LCD
16x2.

#byte PORTB =0xF81   // We attribute to the memory
position 0xF81 the name PORTB.
                     // This means that we define
a 8 bit variable whose value
                     // will be stored to the
memory position F81h.
                     // The memory position F81h
is the PORTD data register.


void main(){
   lcd_init();          // Initialization routine
   lcd_putc("\f");      // Clear display
   lcd_gotoxy(3,1);     // The cursor moves to the
third position of the first line
   lcd_putc("ELECTRONICS"); // print the message
on the LCD
   lcd_gotoxy(4,2);         // he cursor moves to
the fourth position of the second line
   lcd_putc("*****");        // print the message
on the LCD

   while(TRUE){;}          //The main program does
nothing. Eternal loop
}
```

Step 3 (4 minutes)	Compile the program in C in order to create the program in the microcontroller machine code (hex file).  *Figure 2. CCS C Compiler, translation to machine code (hex file)*
Step 4 (1 minutes)	Load to the microcontroller the hex file (program in machine code) that was created from the CCS Compiler.
Step 5 (5 minute)	Run the simulation and check the correct operation of the circuit.
Step 6 (5 minutes)	Suggested modifications and discussion: • modify the code and circuit accordingly so that the LCD displays the message: !!!Module_2-4!!! **********

2.2 Activity 2. Counting the 1s of PORTD

The purpose of this activity is to count the 1s of PORTD and show the result on the LCD 16x2.

Table 3. Activity 2

Activity 2nd (45 minutes)	**Step 1.** The circuit is drawn in the Proteus Design Suite. **Step 2.** The program in C language is written. **Step 3.** The program is compiled with the use of CCS C compiler to the microcontroller machine code. The machine code is loaded to the flash memory of the microcontroller. **Step 4.** The animation is activated. **Step 5.** Modifications and discussion.
Step 1 (15 minutes)	Draw the circuit of the picture at the Proteus Design Suite.  *Figure 3(a). Inputs and LCD 16x2*  *Figure 3(b). Inputs and LCD 16x2*

Step 2 (15 minutes)	Write in CCS Compiler the program in C language

```
#include <main.h> // the file main.h with the
                  // initial settings is included.
                  // This file must be placed in
the same
                  // folder with the project.
                  // Also the 18F4550.h file must
exist
                  // in the same folder with the
project

#include <flex_lcd.h> // The h file of the lcd
driver
                       // should be in the same
folder where we will save our program.
                       //   The   #define   LCD_DB4
PIN_B4 etc statements in flex_lcd.c
                       // should be checked and
possibly modified.
                       //     These     statements
determine the pins of the microcontroller
                       // that are connected to LCD
16x2.

#byte PORTB =0xF81
                      // We attribute to the memory
position 0xF81 the name PORTB.
                      // This means that we define
a 8 bit variable whose value
                      // will be stored to the
memory position F81h.
                      // The memory position F81h
is the PORTD data register.

#byte PORTD=0xF83     // F83h is the position or
PORTD data register
                      // at the data memory of the
microcontroller
                      //   SFR   Special   Function
Register


// ********* main program ************************

void main(){                    // Opening bracket
of main
   set_tris_b(0x00);    // PORTB is set as output
port
                        //     (PORTB     Direction
Register = 0000 0000)

   set_tris_d(0xff);  // PORTD is set as input port
                       // (PORTD Direction Register
= 1111 1111)

   lcd_init();                  // Initialization
routine
```

	<pre>PORTB=0b00000000; // PORTB takes the initial value of 0000 0000
	int i=0; // Integer varable used in the for() { }
	int a; // Integer varable a
	while(TRUE) { // Endless loop(condition always true)
	delay_ms(500); // Wait for 0.5 second
	a=0;
	for (i=0; i<=7; i++){
	a = a + bit_test(PORTD,i);
	// The function bit_test(PORTD,i) checks the bit i of PORTD data register
	// If the bit equals to 1 then the function returns the value 1.
	// If the bit is 0, then the function returns the value 0.
	// Variable a equals the multitude of 1s of PORTD
	} // Closing bracket of for() { }
	
	lcd_gotoxy(1,1); // The cursor moves to the first position of the first line
	printf(lcd_putc,"%d",a); // The value of variable a is shown on the LCD
	} // Closing bracket of while
	} // Closing bracket of main</pre>

Step 3 (5 minutes)	Use the CCS C Compiler to translate the programm from C language to the microcontroller machine code. Load to the microcontroller the hex file (machine code) that was created from the CCS Compiler.
Step 4 (5 minutes)	Run the simulation and check the correct operation of the circuit.
Step 5 (5 minutes)	Suggested modifications and discussion: • what do we need to change in the circuit and in the code so that the LCD 16x2 is connected to the PORTC?

2.3 Activity 3. Simple home alarm system

The purpose of the activity is for the PIC18F4550 to function as a simple alarm system. The microcontroller uses a 16x2 liquid crystal display as output device. Instead of magnetic reed switches, dip switches are used.

- Alarm system function:

- in the protection area there are 5 switches/sensors and they are grouped in 2 zones
- Zone A has 2 switches/sensors
- Zone B has 3 switches/sensors
- each zone can be activated independently of the other using 2 switches
- two LEDs indicate the activation of each zone
- the alarm activates a buzzer
- the buzzer is ON until the alarm zone is deactivated

- Microcontroller's inputs:

Table 4. Pins' description

PIN	Description
RB0	Activation of Zone A
RB1	Activation of Zone B
RB3	Switch/sensor 1 – Zone A
RB4	Switch/sensor 2 – Zone A
RB5	Switch/sensor 3 – Zone B
RB6	Switch/sensor 4 – Zone B
RB7	Switch/sensor 5 – Zone B

- All inputs are activated with "0"
- This alarm system could be used to protect the area in which the user is inside

Table 5. Activity 3

Activity 3rd (75 minutes)	**Step 1.** The circuit is drawn at the Proteus Design Suite. **Step 2.** The program in C language is written. **Step 3.** The program is compiled with the use of CCS C compiler to the microcontroller machine code (the hex.file is created). The program in machine code is loaded to the microcontroller. **Step4.** The animation is activated.

Step 1
(25 minutes)

Draw the circuit of the picture in the Proteus Design Suite

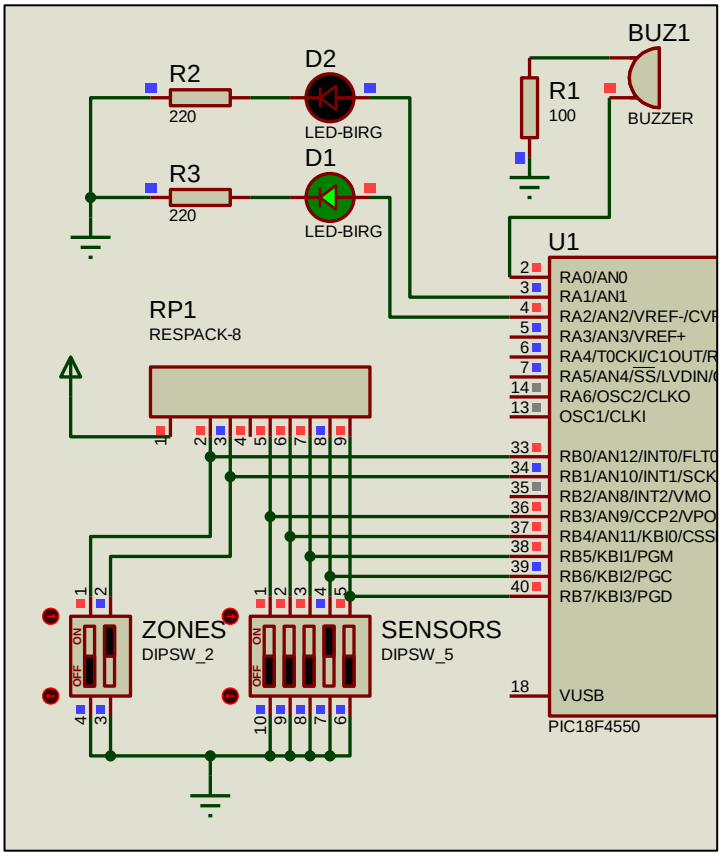


Figure 4(a). Alarm system

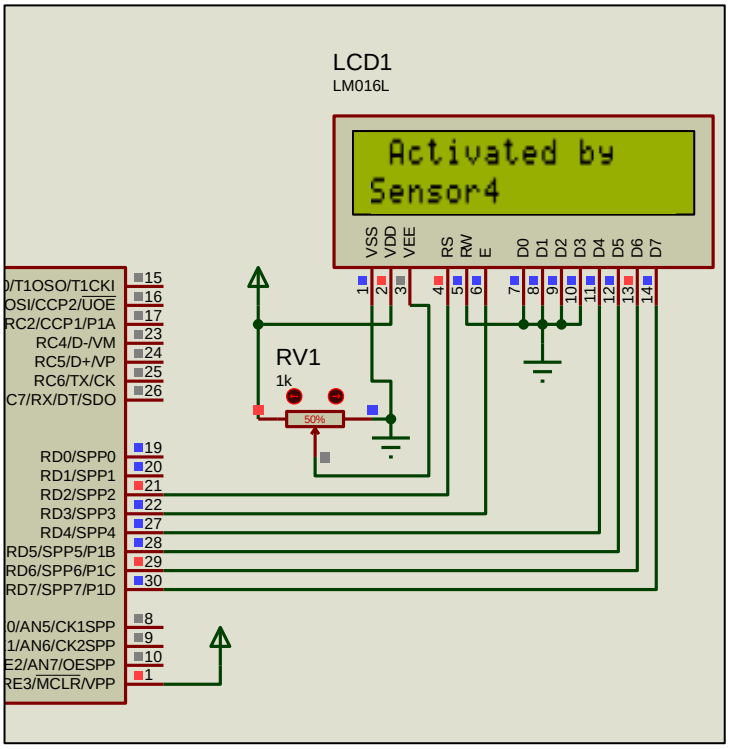


Figure 4(b). Alarm system

Step 2 (35 minutes)	Write in CCS Compiler the program in C language

```
#include <main.h> // the file main.h with the
                  // initial settings is included.
                  // This file must be placed in
the same
                  // folder with the project.
                  // Also the 18F4550.h file must
exist
                  // in the same folder with the
project

#include <flex_lcd.h> // The h file of the lcd
driver
                       // should be in the same
folder where we will save our program.
                       //   The   #define   LCD_DB4
PIN_B4 etc statements in flex_lcd.c
                       // should be checked and
possibly modified.
                       //     These     statements
determine the pins of the microcontroller
                       // that are connected to LCD
16x2.

#byte PORTB =0xF81
                      // We attribute to the memory
position 0xF81 the name PORTB.
                      // This means that we define
a 8 bit variable whose value
                      // will be stored to the
memory position F81h.
                      // The memory position F81h
is the PORTD data register.

#byte PORTD=0xF83     // F83h is the position or
PORTD data register
                      // at the data memory of the
microcontroller
                      //   SFR   Special   Function
Register


#byte PORTA=0xF80     // F80h is the position or
PORTA data register
                      // at the data memory of the
microcontroller
                      //   SFR   Special   Function
Register


boolean zone1; //flag raised when Zone 1 is
activated
boolean zone2; //flag raised when Zone 2 is
activated
boolean alarm; //flag raised when the alarm goes
off

// ********* main program ************************
```

```
void main() {
   set_tris_d(0x00);      //PORTD is defined as
output to drive the LCD
   set_tris_b(0xFF);   //PORTB is defined as input
for sensors and control switches
   set_tris_a(0x00);      //PORTA is defined as
output to set on/off the buzzer and LEDs

   output_low(PIN_A0);  //buzzer activations
   output_low(PIN_A1);  //LED1 (zone1) is off
   output_low(PIN_A2);  //LED2 (zone2) is off
   lcd_init();          //Initialization routine
   lcd_putc("\f");      //Clear display
   printf(lcd_putc," Alarm is off"); //display the
message

   while(TRUE) {

      //check for zoneA activation
      if(input(PIN_B0)==0){
         zone1=1; //zoneA is activated
      }
      else{
         zone1=0; //zoneA is de-activated
         alarm=0; //alarm turned off
      }
      //check for zoneB activation
      if(input(PIN_B1)==0){
         zone2=1; //zoneB is activated
      }
      else{
         zone2=0; //zoneB is de-activated
         alarm=0; //alarm turned off
      }

      //print a message on the LCD
      if(zone1==0 && zone2==0){
         lcd_putc("\f");
         printf(lcd_putc," Alarm is off");
         delay_ms(150);
      }
      else{
         lcd_putc("\f");
         printf(lcd_putc," Alarm is on");
         delay_ms(150);
      }


      if(alarm==0){
         if(zone1==1){
            output_high(PIN_A1); //LED for zoneA
is ON
            //check the sensors
            if(input(PIN_B3)==0){
               alarm=1;
               lcd_putc("\f");
               printf(lcd_putc,"        Activated
by\nSensor1");
            }
            else if(input(PIN_B4)==0){
               alarm=1;
               lcd_putc("\f");
```

```
                  printf(lcd_putc,"         Activated
by\nSensor2");
               }

               if(alarm==1){
                  output_high(PIN_A0); //buzzer is on
                  while(input(PIN_B0)==0){;}   //wait
to turn off the ZoneA
               }
            }
            else{
               output_low(PIN_A1); //LED for zoneA is
OFF
               output_low(PIN_A0); //buzzer is OFF
            }

            if(zone2==1){
               output_high(PIN_A2);//LED for zoneB is
ON
               //check the sensors
               if(input(PIN_B5)==0){
                  alarm=1;
                  lcd_putc("\f");
                  printf(lcd_putc,"         Activated
by\nSensor3");
               }
               else if(input(PIN_B6)==0){
                  alarm=1;
                  lcd_putc("\f");
                  printf(lcd_putc,"         Activated
by\nSensor4");
               }
               else if(input(PIN_B7)==0){
                  alarm=1;
                  lcd_putc("\f");
                  printf(lcd_putc,"         Activated
by\nSensor5");
               }

               if(alarm==1){
                  output_high(PIN_A0); //buzzer is on
                  while(input(PIN_B1)==0){;}   //wait
to turn off the ZoneB
               }
            }
            else{
               output_low(PIN_A2); //LED for zoneB is
OFF
               output_low(PIN_A0); //buzzer is OFF
            }
         }
      }
   }
```

Step 3 (5 minutes)	Compile the program in order to create the hex.file (program in machine code). Load the program (hex.file) to the microcontroller.

Step 4 (10 minutes)	Run the simulation and check the correct operation of the circuit.

Chapter 3: **Recapitulation**

- ☞ The schematic of the circuits was drawn with Proteus Design Suite
- ☞ The schematic of the circuit was drawn with Proteus Design Suite.
- ☞ A LCD 16x2 were used to implement applications such as a simple alarm system.
- ☞ The programs in C was written in CCS C compiler.
- ☞ The programs in C was compiled to the microcontroller machine code (hex file).
- ☞ The machine code was “loaded” to the microcontroller and the animation was activated.

References

CCS C Compiler Manual. Ccsinfo.com. (2021). Retrieved from https://www.ccsinfo.com/downloads/ccs_c_manual.pdf.

PIC18F2455/2550/4455/4550 Data Sheet. Ww1.microchip.com. (2006). Retrieved from https://ww1.microchip.com/downloads/en/devicedoc/39632c.pdf.

Proteus Tutorial : Getting Started with Proteus PCB Design (Version 8.6). Youtube.com. (2017). Retrieved from https://www.youtube.com/watch?v=GYAHwYUUs34.

Simple LED Circuits. Electronics Hub. (2017). Retrieved from https://www.electronicshub.org/simple-led-circuits/.

Appendix. Figures with high resolution

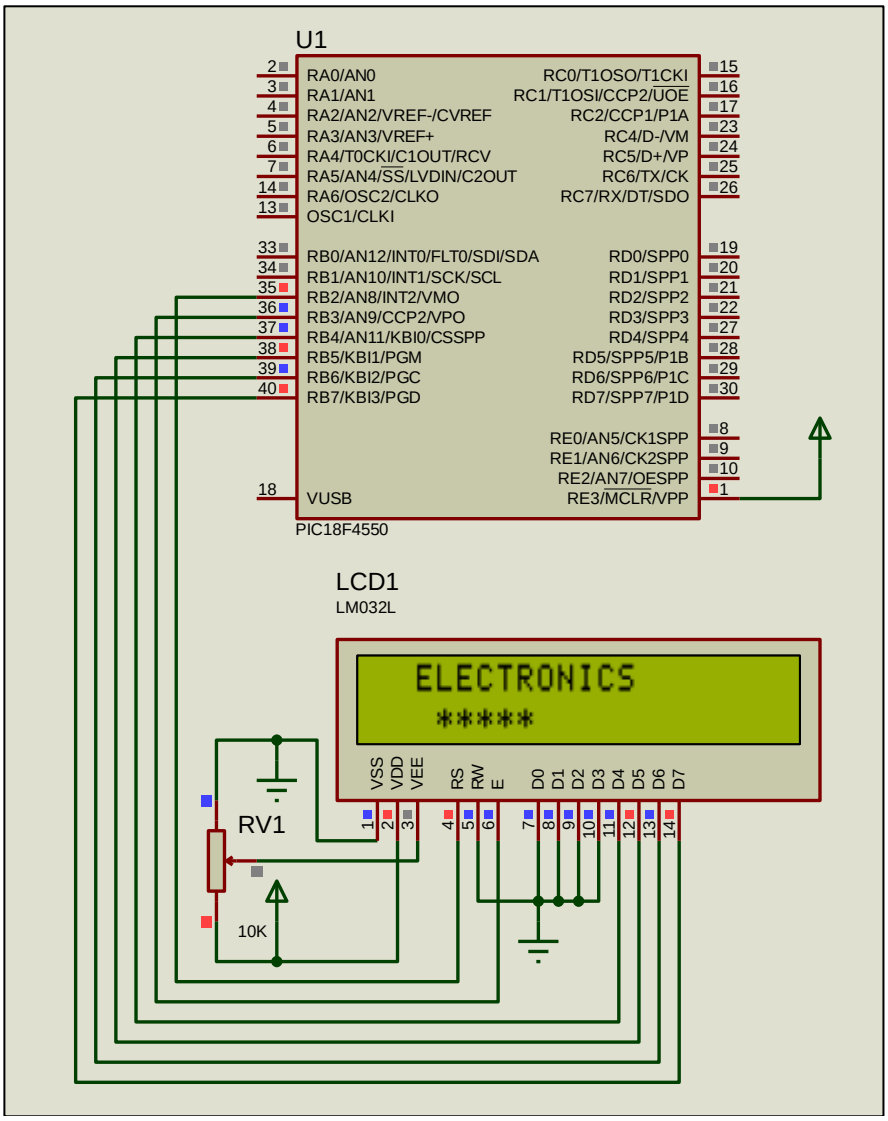


Figure 1. Message on the LCD 16x2

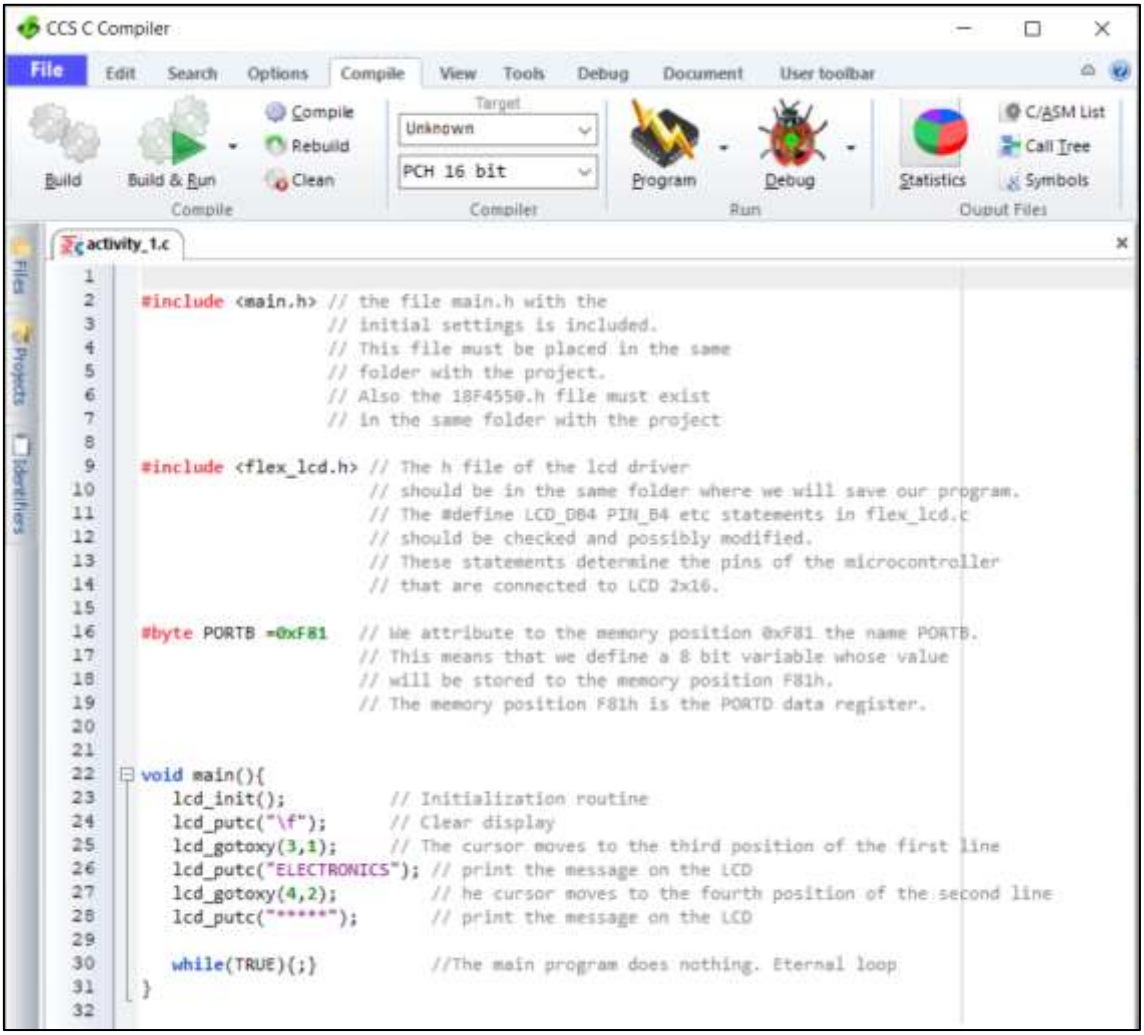


Figure 2. CCS C Compiler, translation to machine code (hex file)

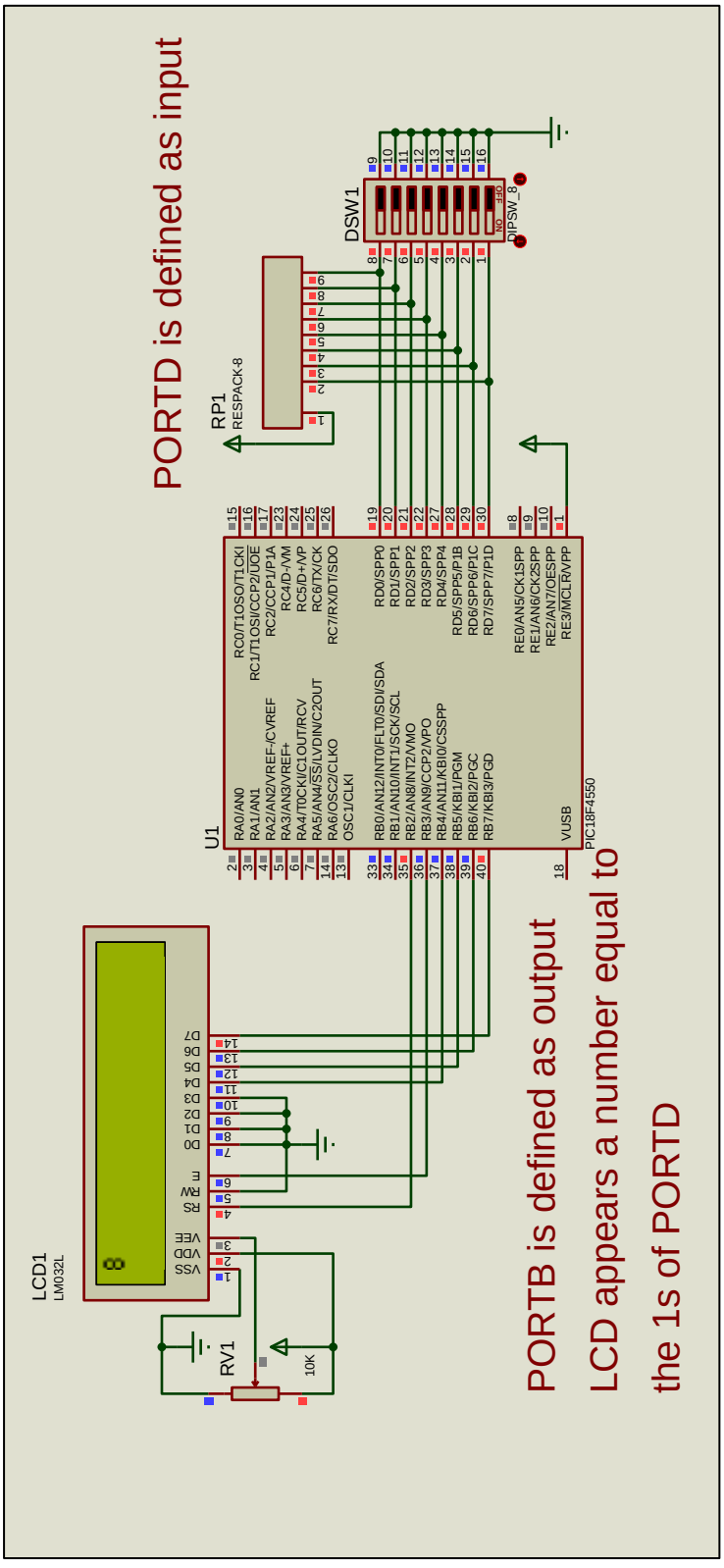


Figure 3. Inputs and LCD 16x2

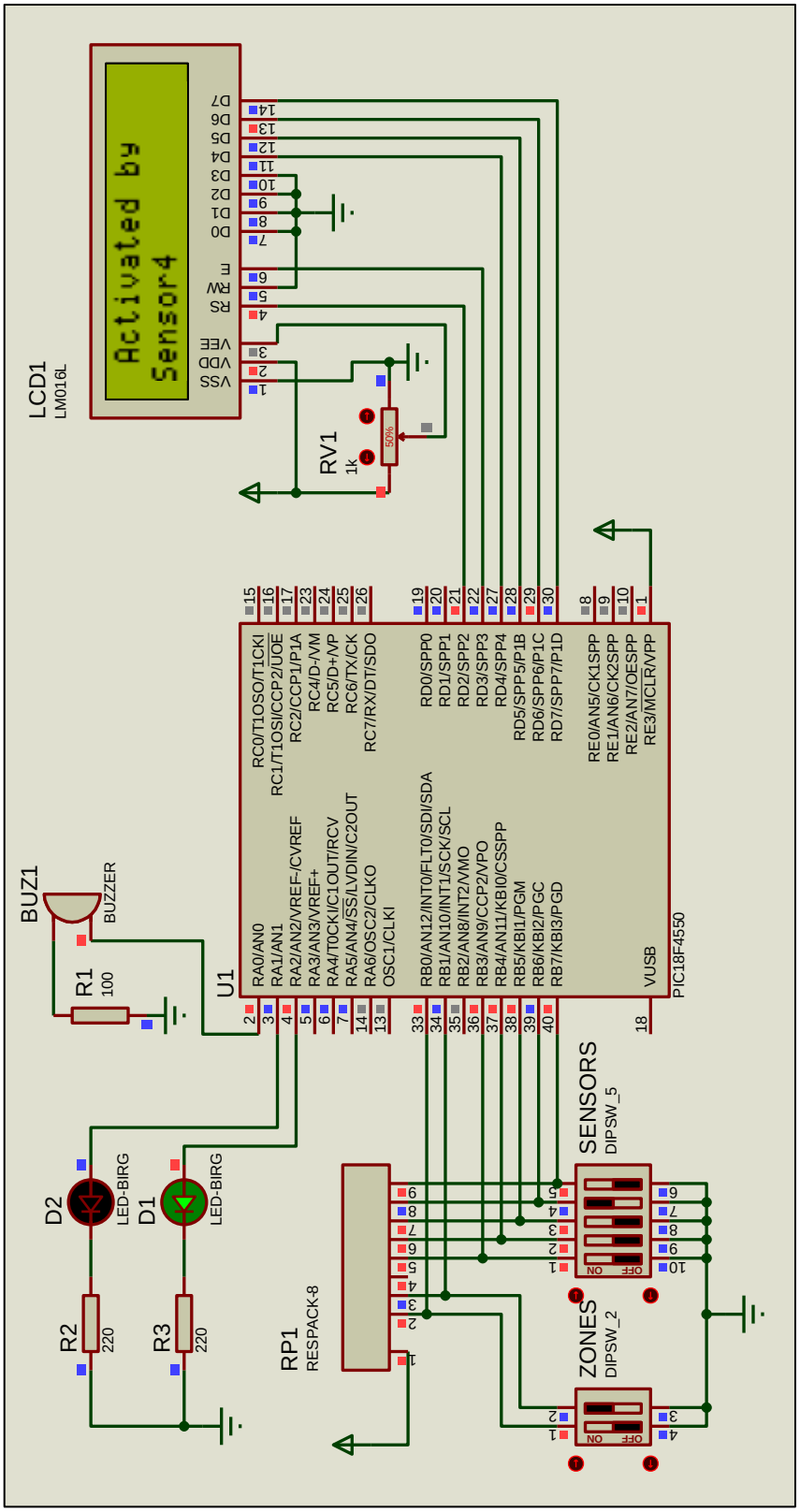


Figure 4. Alarm system