ENGINE

Teaching online electronics, microcontrollers and programming in Higher Education

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

Practice leaflet: Module_2-5 Keypad 4x4

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 Where other published and unpublished source materials have been used, these have been acknowledged.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 Keypad 4x4.

Chapter 1: **Overview**

Table 1. Overview

Title / short summary	**5. Keypad 4x4**
Expected learning outcomes	• The student will be able to connect a keypad 4x4 on the PIC18F4550 • The student will be able to modify the keypad driver for CCS C Compiler • The student will be able to read ASCII characters from a keypad 4x4 • The student will be able to design a simple calculator with a keypad 4x4 and a LCD 16x2 • The student will be able to load and animate a microcontroller program in the Proteus Design Suite
Keywords	Keypad 4x4, matrix, rows, columns
Duration	The duration of the module_2-5 is 3 hours • Presentation of the module_2-5 by the teacher, 30 minutes • 1st activity, keypad 4x4 and LEDs, 30 minutes • 2nd activity, keypad 4x4 and LCD 16x2, 45 minutes • 3rd activity, simple calculator, 75 minutes
Involved	**The teacher:** Presents the slides associated with the module_2-5 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_2-5 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, Module_2-3 and Module_2-4
Educational content	• CCS C Compiler manual (C Compiler Reference Manual) • MICROCHIP, PIC18F2455/2550/4455/4550 Data Sheet • Module_2-5 slides • Module_2-5 Evaluation leaflet • Module_2-5 Open project leaflet • Module_2-5 Programs, Schematic Proteus (Compressed folder)
Tips	***Tip1.*** *The #define row0 PIN_ etc statements in keypad.h should be checked and possibly modified.* ***Tip2.*** *The #define LCD_DB4 PIN_B4 etc statements in flex_lcd.c should be checked and possibly modified.*

Chapter 2: **Activities**

2.1 Activity 1. Keypad 4x4 and LEDs

The purpose of this activity is for microcontroller to read an ASCII character from a keypad 4x4, and display its code in 8 LEDs.

Table 2. ASCII code for digits

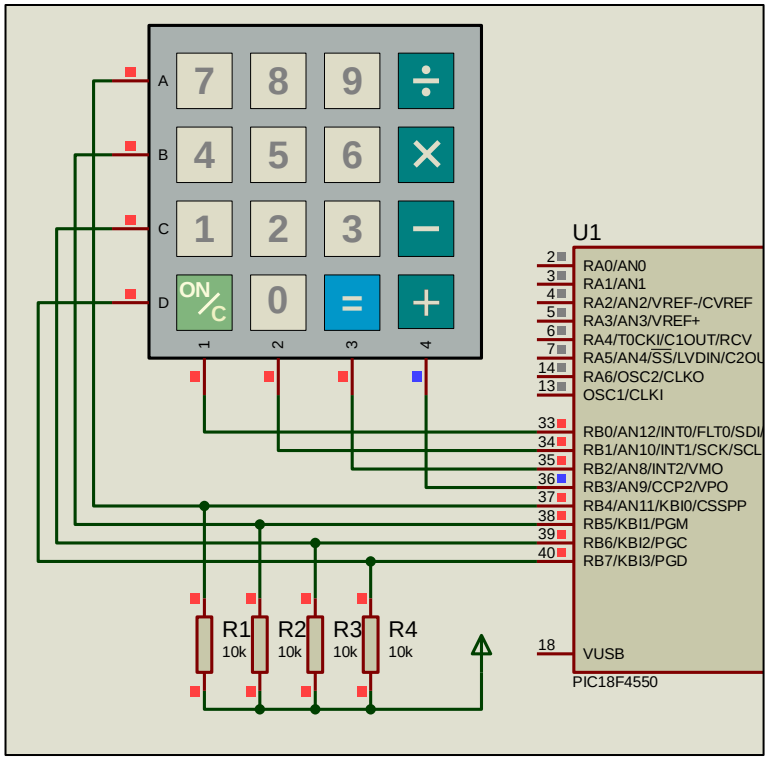
Digit	ASCII code (hex)	ASCII code (binary)
0	0x30	0011 0000
1	0x31	0011 0001
2	0x32	0011 0010
3	0x33	0011 0011
4	0x34	0011 0100
5	0x35	0011 0101
6	0x36	0011 0110
7	0x37	0011 0111
8	0x38	0011 1000
9	0x39	0011 1001

Table 3. ASCII code for characters

Character	ASCII code (hex)	ASCII code (binary)
/	0x2F	0010 1111
X	0x58	0101 1000
-	0x2D	0010 1101
+	0x2B	0010 1011
=	0x3D	0011 1101
C	0x43	0100 0011

Table 4. 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 (a). Keypad and LEDs*

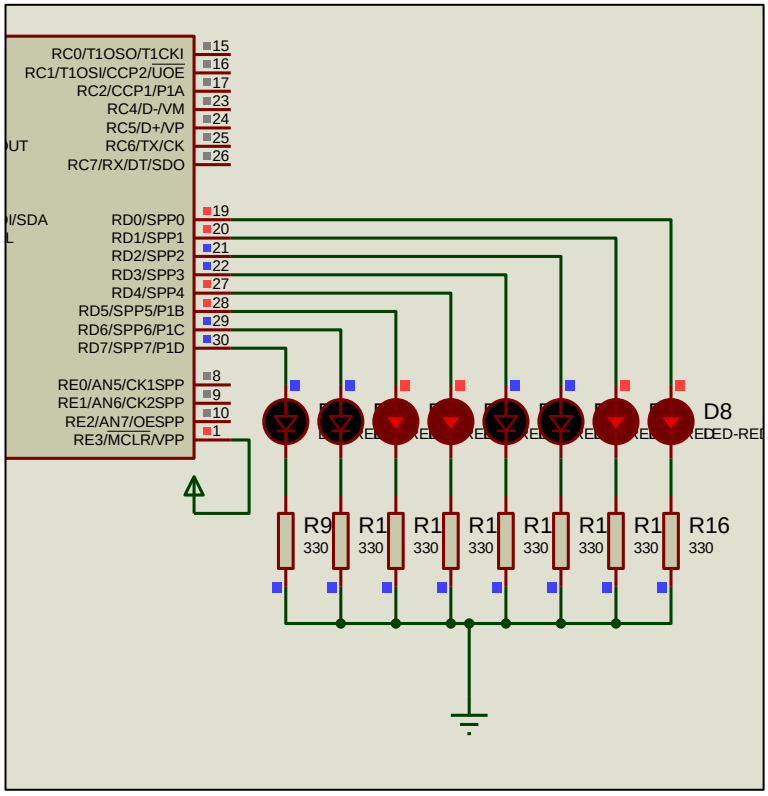


Figure 1 (b). Keypad and LEDs

Step 2
(5 minutes)

Write in CCS C 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 <keypad.h>    // The h file of the keypad
driver
                        // should be in the same
folder where we will save our program.
                        // The #define row0 PIN_B4
etc statements in keypad.h
                        // should be checked and
possibly modified.
                        //     These     statements
determine the pins of the microcontroller
                        // that are connected to the
keypad 4x4.

#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

//initialization routine
void init(void);

// ********* main program ***********************
void main(){
   char k;       //variable for storing the ASCII
code of the key pressed
   init();      //initialization routine
   kbd_init();   //initialization routine for the
keypad 4x4

   while(TRUE){
      k=kbd_getc(); //Keypad reading.
                    //If  a  key  is  pressed,  the
variable k gets the ASCII code of the key.
                    //If  no  key  is  pressed,  the
function kbd_getc() returns the value 0.
      If(k!=0){     //If a key is pressed
         PORTD=k;    //The ASCII code of the key
pressed is transferred to PORTD
      }
   }
}

//initialization routine
void init (void){
   set_tris_d(0x00);  //PORTD is defined as output
   PORTD=0x00;          //PORTD takes the initial
value of 0x00
}
```

Step 3 (4 minutes)	Compile the program in C in order to create the program in the microcontroller machine code (hex file).

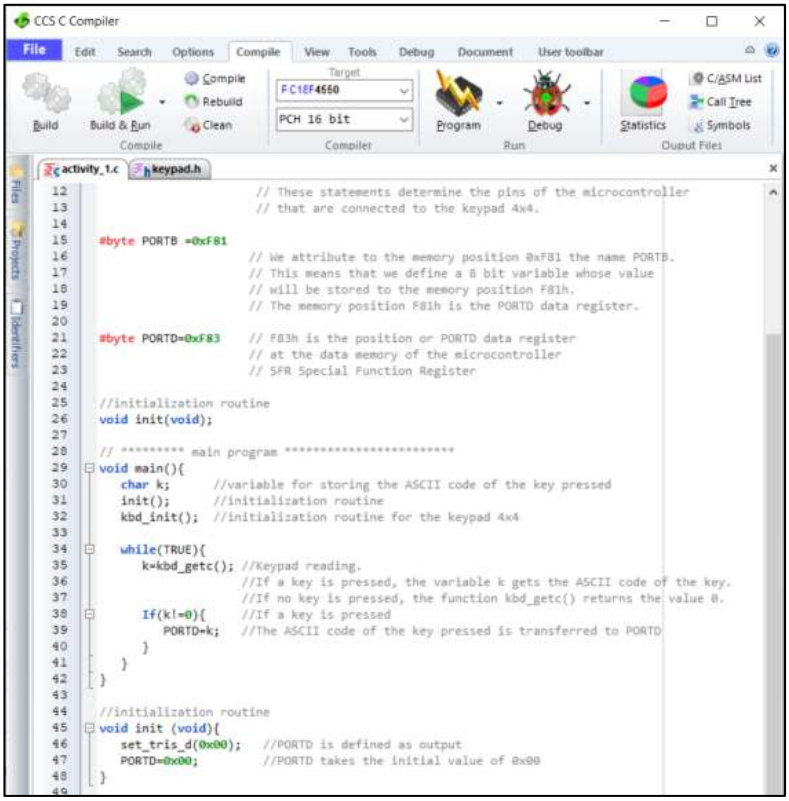


Figure 2. CCS C Compiler, translation to machine code (hex file)

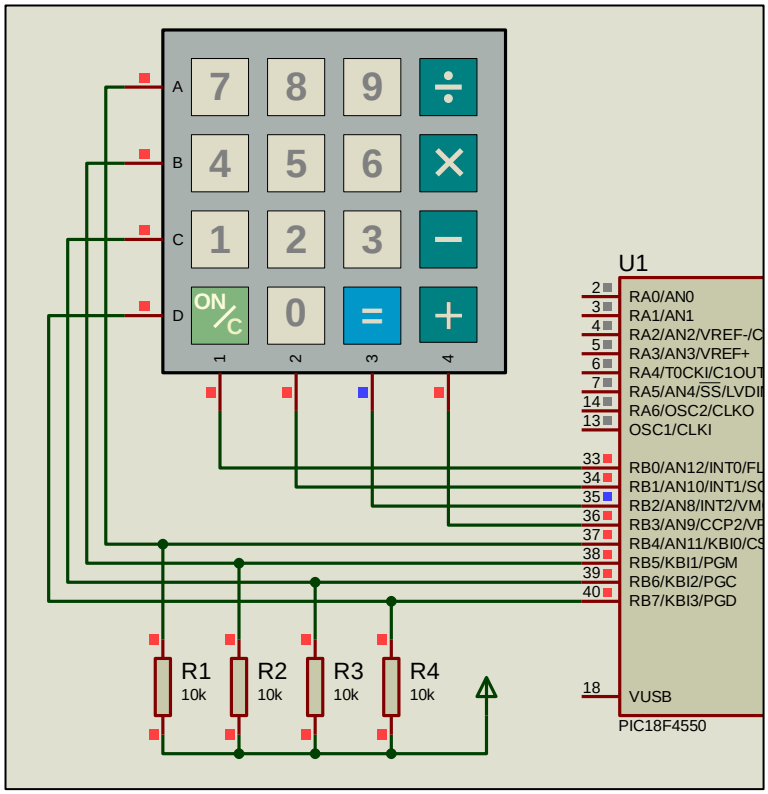
Step 4 (1 minutes)	Load to the microcontroller the hex file (program in machine code) that was created from the CCS C Compiler.
Step 5 (5 minute)	Run the simulation and check the correct operation of the circuit.
Step 6 (5 minutes)	Suggested modifications and discussion: • modify the code so that PORTD does not display the ASCII code but the numeric value of the key ***Tip**. Notice the values of the first 4 bits of the ASCII codes of digits 0, 1, 2,… 9.*

2.2 Activity 2. Keypad 4x4 and LCD 16x2

The purpose of this activity is for microcontroller to read an ASCII character from a keypad 4x4, and display it in a LCD 16x2.

- If ‘C’ key is pressed, then the screen clears.

Table 5. 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). Keypad 4x4 and LCD 16x2*

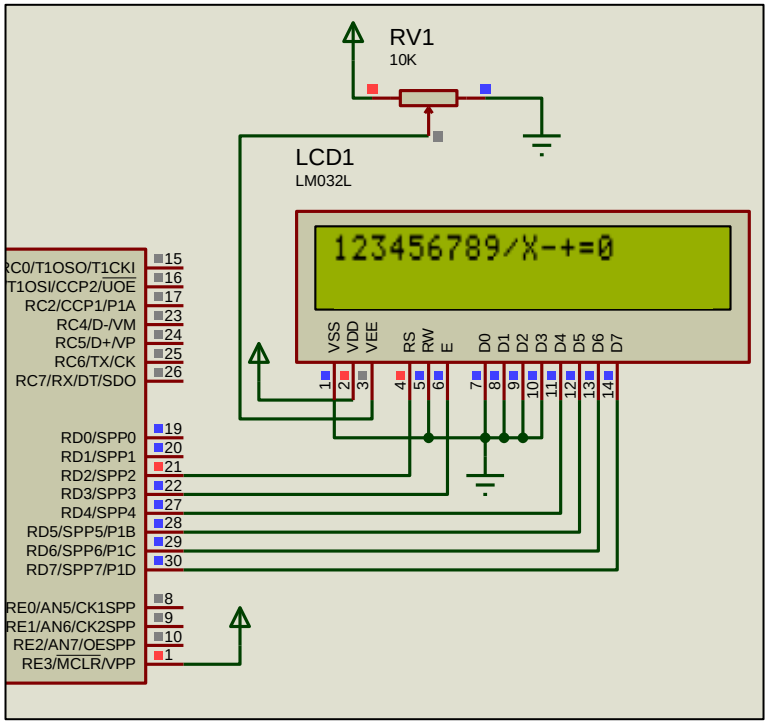


Figure 3 (b). Keypad 4x4 and LCD 16x2

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.

#include <keypad.h>   // The h file of the keypad
driver
                      // should be in the same
folder where we will save our program.
```

```
                        // The #define row0 PIN_B4
etc statements in keypad.h
                        // should be checked and
possibly modified.
                        //     These     statements
determine the pins of the microcontroller
                        // that are connected to the
keypad 4x4.

#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

//initialization routine
void init(void);

// ********* main program ************************
void main(){
   char k;       //variable for storing the ASCII
code of the key pressed
   init();      //initialization routine
   kbd_init();   //initialization routine for the
keypad 4x4
   lcd_init();   //initialization routine for the
LCD 16x2
   while(TRUE){
      k=kbd_getc();    //keypad reading

      //If a key is pressed (k! = 0) and if the
key pressed is not 'C',
      //the character will appear on the LCD
      if (k!=0 && k!='C'){
         printf(lcd_putc,"%c",k);
      }

      //If  'C'  key  is  pressedm  then  the  screen
clears
      else if(k=='C'){
         printf(lcd_putc,"\f");
      }
   }
}

// initialization routine
void init(void){
   set_tris_d(0x00);       //PORTD  is  defined  as
output
}
```

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 keypad 4x4 is connected to PORTC?

2.3 Activity 3. Simple calculator

This activity uses a 4x4 keypad and a LCD 16x2, with the aim of making the PIC18F4550 a simple calculator that can perform 4 basic operations between 2 single-digit numbers.

The PIC18F4550:

- reads the first number
- reads which mathematical operation will be performed (+, -, X, /)
- reads the second number
- displays the result on the LCD
- waits for 'C' to be pressed to start the process from the beginning

Table 6. 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. **Step 4.** The animation is activated. **Step 5.** Modification and discussion.

Step 1
(15 minutes)

Draw the circuit of the picture in the Proteus Design Suite.

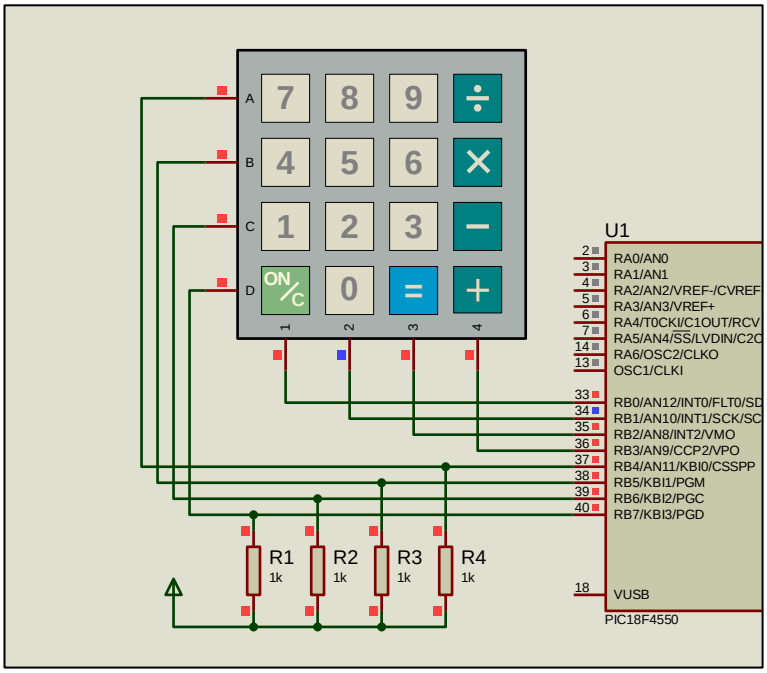


Figure 4 (a). Keypad 4x4 and LCD 16x2

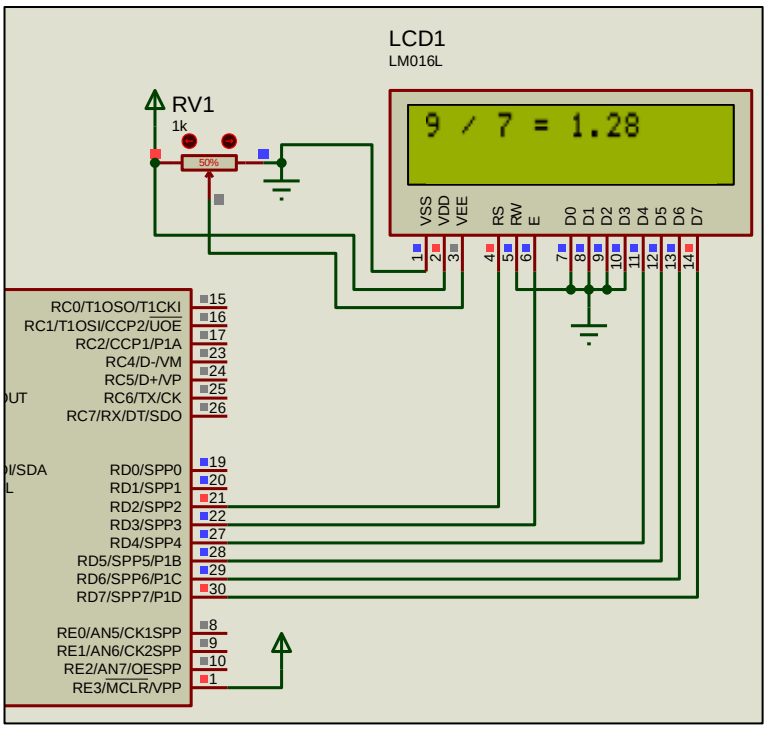


Figure 4 (b). Keypad 4x4 and LCD 16x2

Step 2 (35 minutes)	Write in CCS C 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.

#include <keypad.h>   // The h file of the keypad
driver
                      // should be in the same
folder where we will save our program.
                      // The #define row0 PIN_B4
etc statements in keypad.h
                      //  should be checked and
possibly modified.
                      //     These     statements
determine the pins of the microcontroller
                      // that are connected to the
keypad 4x4.

#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

char key;            //variable to save keypad's
characters
int num1=10;      //variable for the first number
int num2=10;      //variable for the second number
```

```
char operation;   //variable for the operation
float result;     //variable for the result
boolean flag=0;    //flag raised when the divider
is 0

//this function converts keypad's character to
integer
int convert_to_number(char c);

// ********* main program ***********************
void main() {
   set_tris_d(0x00);       //PORTD is defined as
output - LCD 16x2
   kbd_init();            //initialization routine
for the keypad 4x4
   lcd_init();            //initialization routine
for the LCD 16x2
   lcd_putc("\f");      //clear the screen

   while(TRUE){
      //read the first number
      do{
         //wait until a key is pressed
         key=kbd_getc();
         if(key!=0){
            //call the "convert_to_number"
            num1=convert_to_number(key);
         }
      }
      while(num1>9);
      //print the first number
      printf(lcd_putc,"%d",num1);

      //read the operation
      do{
         //wait until a key is pressed
         operation=kbd_getc();
      }
      while(operation!='X'  &&  operation!='/'  &&
operation!='+' && operation!='-');

      //print the operation
      if(operation=='+'){
         printf(lcd_putc," + ");
      }
      else if(operation=='-'){
        printf(lcd_putc," - ");
      }
      if(operation=='X'){
         printf(lcd_putc," * ");
      }
      else if(operation=='/'){
         printf(lcd_putc," / ");
      }

      //read the second number
      do{
         //wait until a key is pressed
         key=kbd_getc();
         if(key!=0){
            //call the "convert_to_number"
            num2=convert_to_number(key);
```

```
            }
         }
         while(num2>9);
         //print the second number
         printf(lcd_putc,"%d = ",num2);

         //calculate the result
         if(operation=='+'){
             result=num1+num2;
          }
         else if(operation=='-'){
            if(num1>num2){
               result=num1-num2;
            }
            else{
               result=num2-num1;
               printf(lcd_putc,"-");
            }
         }
         else if(operation=='X'){
           result=num1*num2;
         }
         else if(operation=='/'){
            if(num2==0){
               flag=1;
            }
            else{
               result=(float)(num1)/num2;
            }
         }

         if(flag==1){ //divider = 0
            flag=0;
            printf(lcd_putc,"\nUndefined");
         }
         else{
            //print the result
            printf(lcd_putc,"%f",result);
         }

         //wait until the "C" is pressed
         do{
            key=kbd_getc();
         }
         while(key!='C');

         //clear the LCD
         lcd_putc("\f");
         num1=num2=10;
      }
   }

   //this function converts keypad's character to
   integer
   int convert_to_number(char c){
     if(c=='0'){
       return 0;
     }
     else if(c=='1'){
       return 1;
     }
     else if(c=='2'){
```

	`` return 2; } else if(c=='3'){ return 3; } else if(c=='4'){ return 4; } else if(c=='5'){ return 5; } else if(c=='6'){ return 6; } else if(c=='7'){ return 7; } else if(c=='8'){ return 8; } else if(c=='9'){ return 9; } else if(c=='/'){ return 10; } else if(c=='X'){ return 11; } else if(c=='-'){ return 12; } else if(c=='+'){ return 13; } else if(c=='='){ return 14; } else if(c=='C'){ return 15; } else{ return 16; } } ```
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.
Step 5 (10 minutes)	Suggested modifications and discussion:

	• can other operations be added, such as power, or square root? ***Tip***. *#include <math.h>, pow(), sqrt()*

Chapter 3: **Recapitulation**

- ☞ The schematic of the circuits was drawn with Proteus Design Suite
- ☞ A keypad 4x4 (and a LCD 16x2) were used to implement applications such as a simple calculator.
- ☞ 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

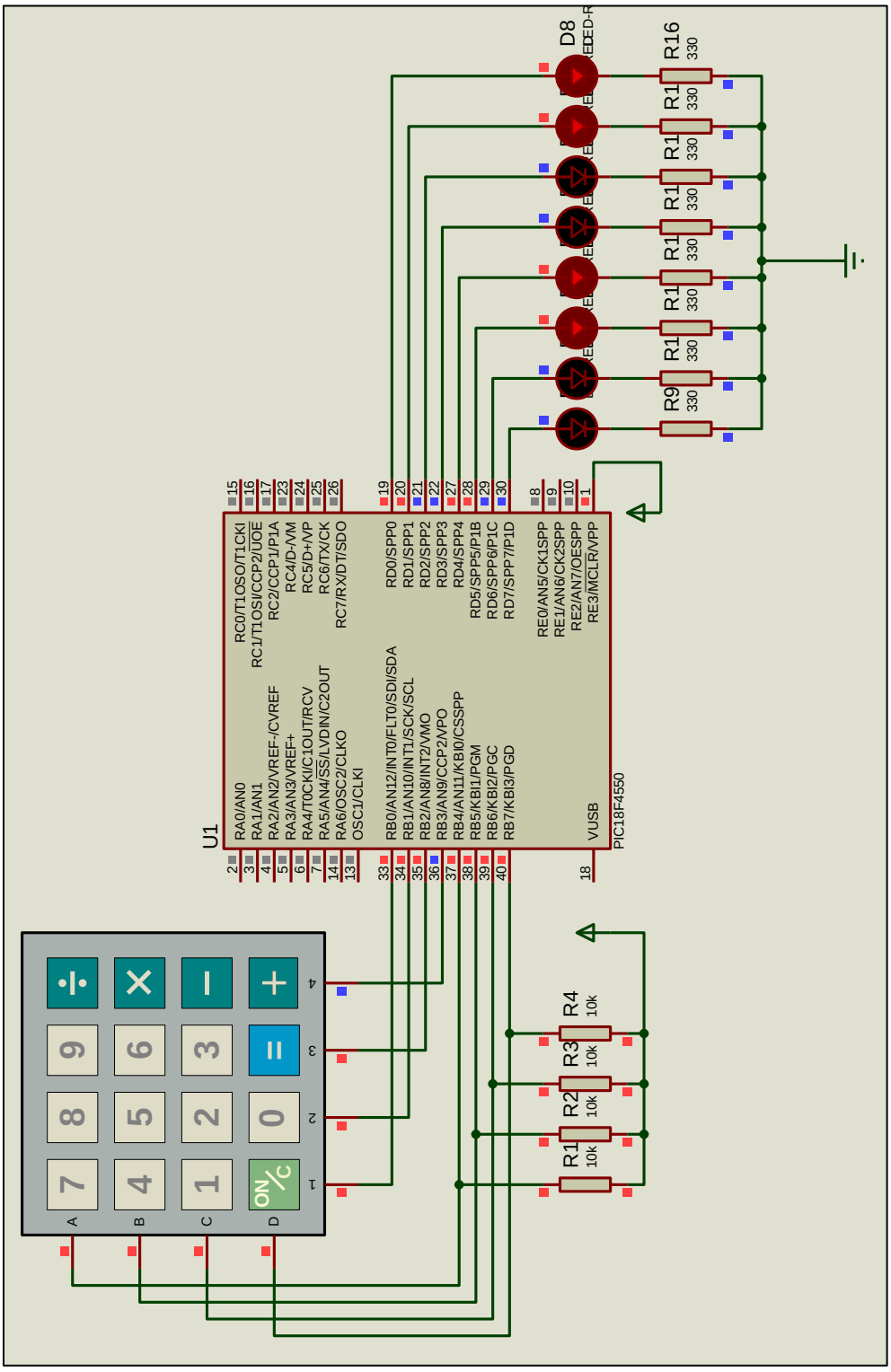


Figure 1. Keypad and LEDs

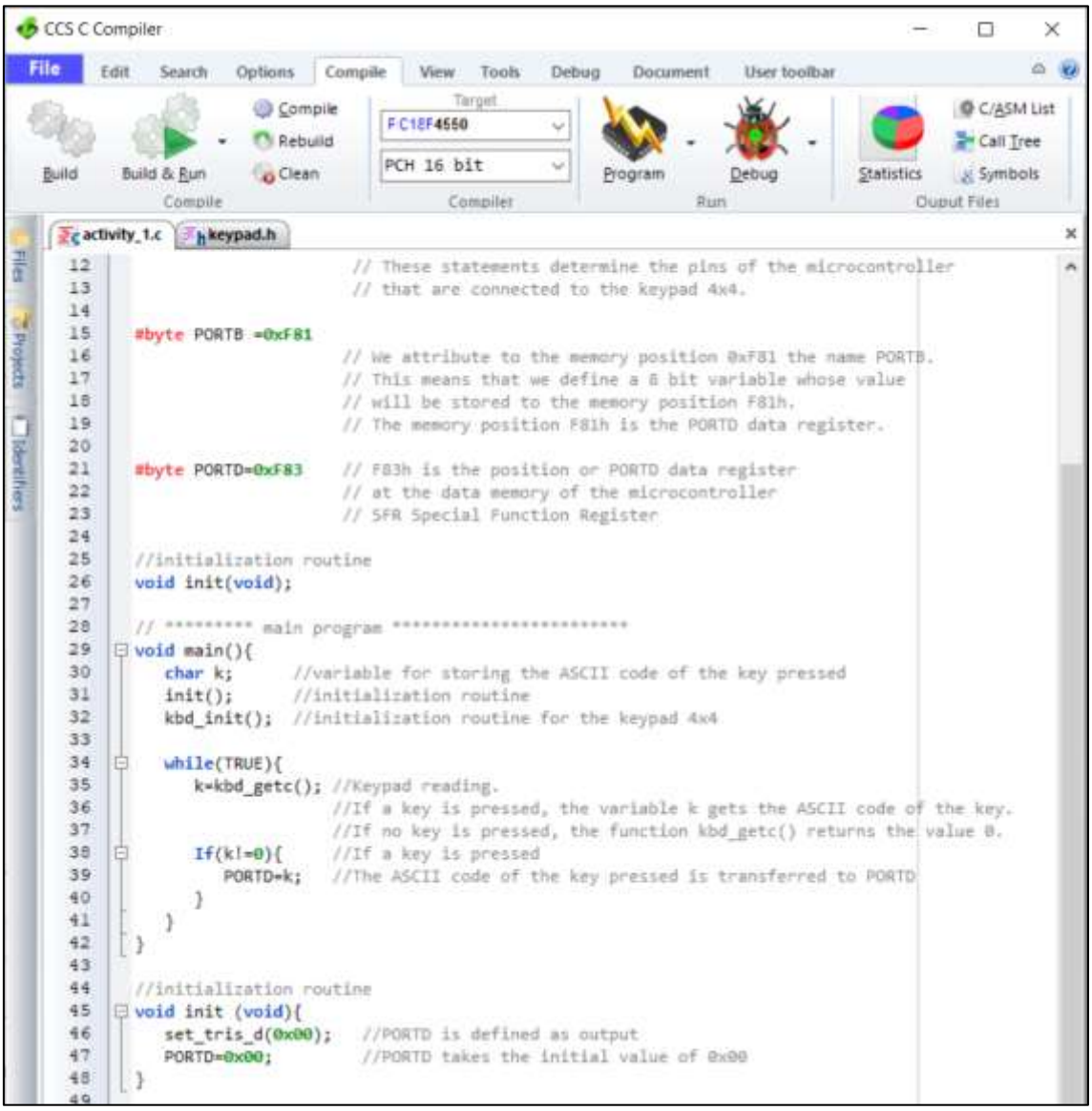


Figure 2. CCS C Compiler, translation to machine code (hex file)

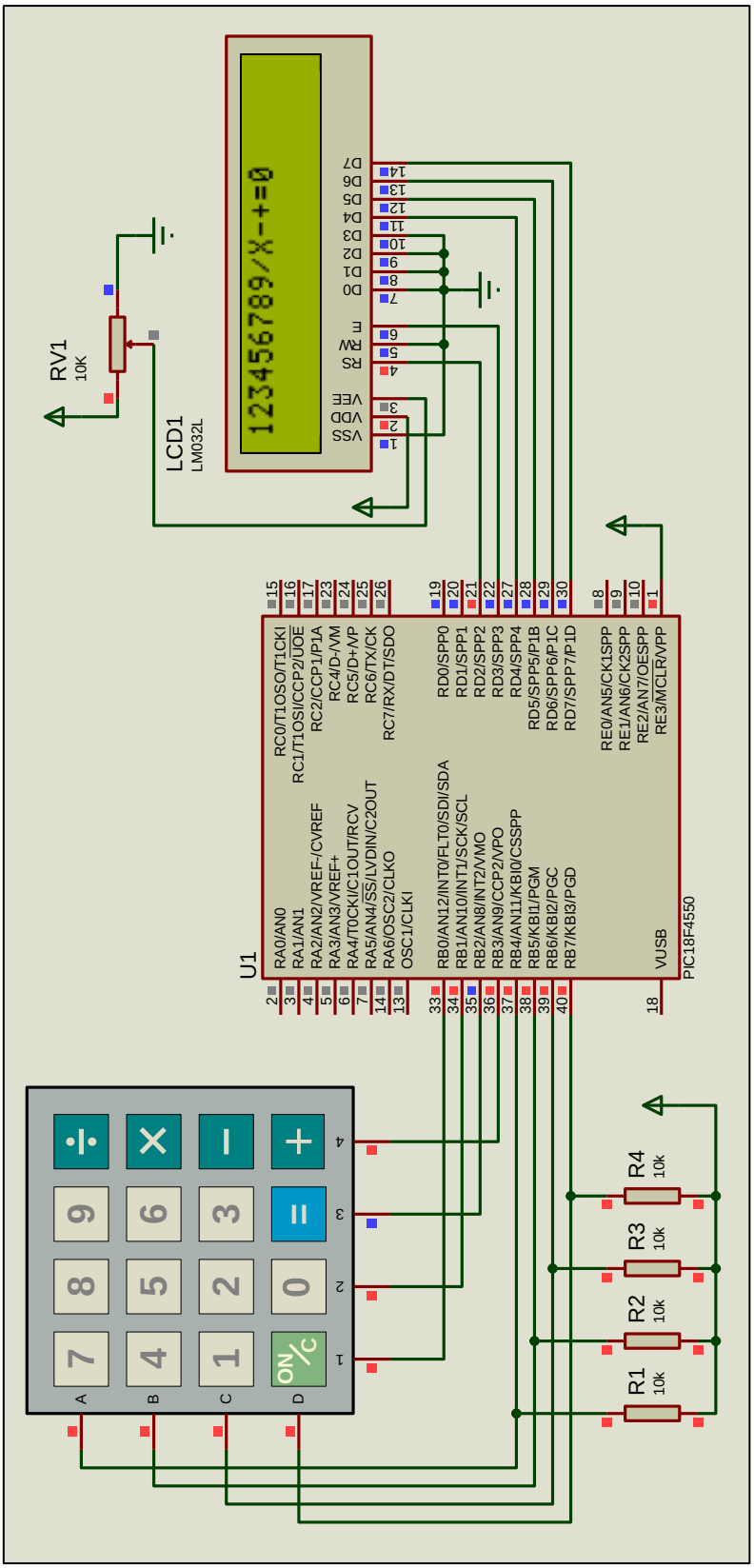


Figure 3. Keypad 4x4 and LCD 16x2

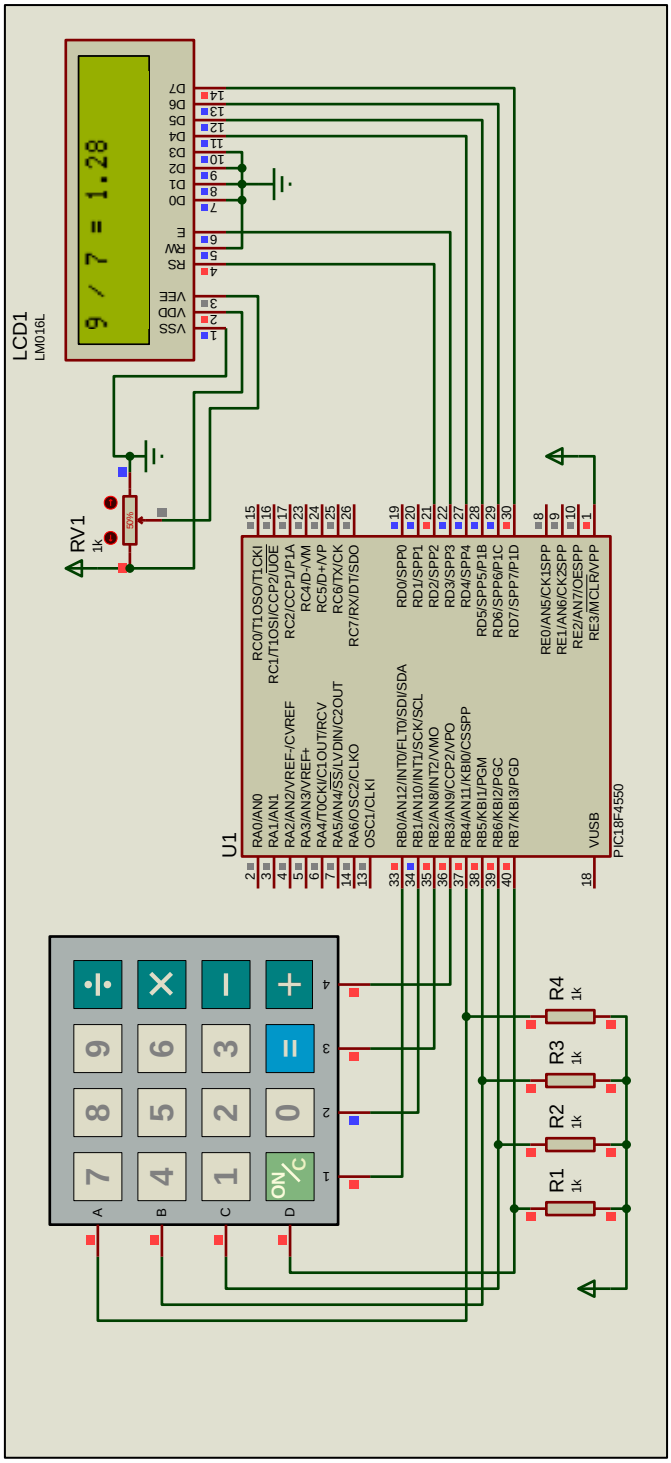


Figure 4. Simple calculator