Teaching online electronics, microcontrollers and programming in Higher Education

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

Practice leaflet: Module_2-7 PushButton

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 Where other published and unpublished source materials have been used, these have been acknowledged.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 push-button.

Chapter 1: **Overview**

Table 1. Overview

Title / short summary	**7. Push-button**
Expected learning outcomes	• The student will be able to connect a push-button on the microcontroller • The student will be able to handle a push-button with a function • The student will be able to design simple circuits with a push-button • The student will be able to load and animate a microcontroller program in the Proteus Design Suite
Keywords	Push-button, inputs/outputs
Duration	The duration of the module_2-7 is 3 hours • Presentation of the module_2-7 by the teacher, 30 minutes • 1st activity, use a push-button, 30 minutes • 2nd activity, read a push-button, 45 minutes • 3rd activity, read a push-button with function, 75 minutes
Involved	**The teacher:** Presents the slides associated with the module_2-7 and answers question **The students:**

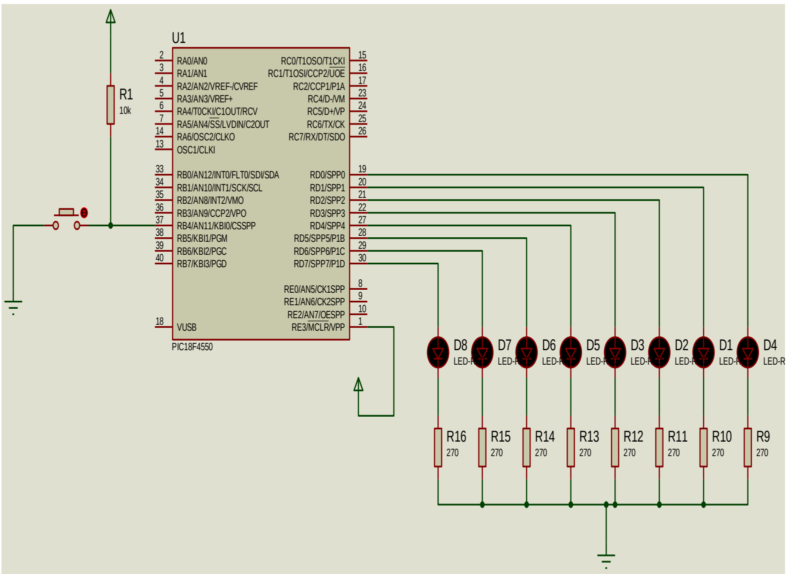
	Draw circuits in Proteus Schematic, write programs in C language, load programs to a microcontroller and run the simulation using the Proteus Design Suite
Assignment	At the end of the Module_2-7 will be given: • Open Project
Educational tools and equipment	• **Material:** PC • **Software:** CCS C compiler, 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 and Module_2-2
Educational content	• CCS C Compiler manual (C Compiler Reference Manual) • MICROCHIP, PIC18F2455/2550/4455/4550 Data Sheet • Module_2-7 slides • Module_2-7 Evaluation leaflet • Module_2-7 Open project leaflet • Module_2-7 Programs, Schematic Proteus (Compressed folder)
Tips	***Tip.*** *Push-button connected with pullup or pulldown resistor?*

Chapter 2: **Activities**

2.1 Activity 1. Increase a counter using push-button

The purpose of this activity is that every time we press a push-button which is connected to the RB4, the value of the PORTD increases by 1. The result of the increase is shown in 8 LEDs which are connected to the PORTD.

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 1 (10 minutes)	Draw the circuit of the picture in the Proteus Design Suite.  *Figure 1. Connections*

Step 2 (10 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


#byte PORTB =0xF81
/* We attribute to the memory position 0xF81 the
name PORTB. This means that we define an 8-bit
variable whose value will be stored to the memory
position F81h.*/
#byte PORTD =0xF83
// The position F83h is the PORTD data register.


void main()
{
   set_tris_d(0x00);//PortD becomes output
   set_tris_b(0xFF);//PortB becomes input
   PORTD=0x00;
   //PortD is given initial value 00000000
   while(TRUE) {
       while(input(PIN_B4) == 1) {
       // Wait until the button is pressed
                                    }
// In the wait state, no command is executed
// When the button is pressed the loop is exited
// and the next command is executed

      delay_ms(50);
      // 50ms delay to avoid bounce effect
      while(input(PIN_B4) == 0) {
      //Wait until the button is released
                                }
// In the wait state, no command is executed
// When the button is released the loop is exited
// and the next command is executed

      delay_ms(50);
      // 50ms delay to avoid bounce effect
      PORTD=PORTD+1;
      //Increment the value of PORTD
      }

}
```

<table>
<tr>
<td>Step 3
(4 minutes)</td>
<td>

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

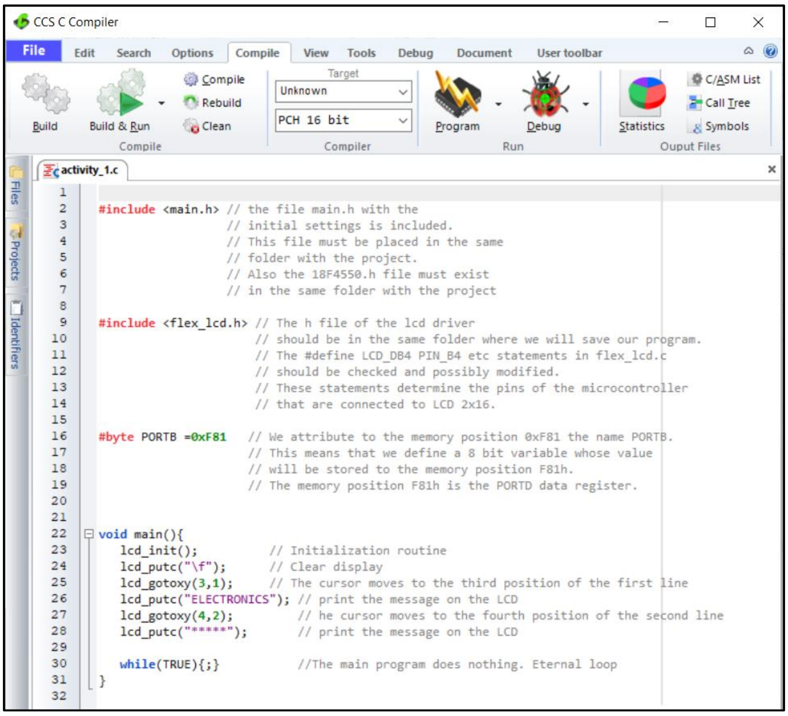


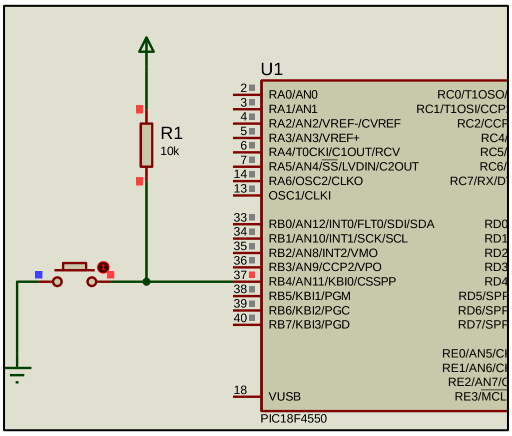
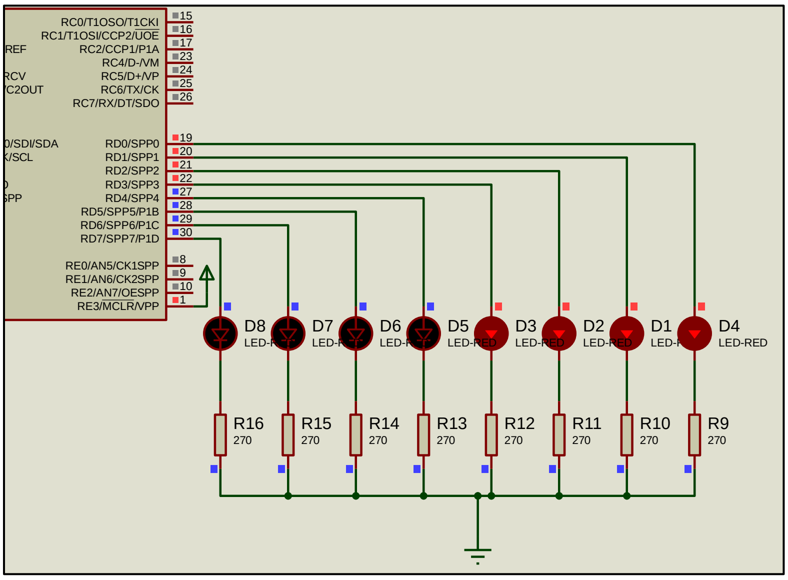
Figure 2. CCS C Compiler, translation to machine code (hex file)

</td>
</tr>
<tr>
<td>Step 4
(1 minutes)</td>
<td>Load to the microcontroller the hex file (program in machine code) that was created from the CCS Compiler.</td>
</tr>
<tr>
<td>Step 5
(5 minute)</td>
<td>Run the simulation and check the correct operation of the circuit.</td>
</tr>
</table>

2.2 Activity 2. Reading a Push Button Toggle

In this activity we want every time a push-button is pressed and released the state of each pin of PORTD changes, i.e. 1's become 0's and 0's become 1's. PORTD will be given the initial value 00001111.

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 1 (10 minutes)	Draw the circuit of the picture at the Proteus Design Suite.  *Figure 3(a). Push-button and LEDs*  *Figure 3(b). Push-button and LEDs*

Step 2
(25 minutes)
	Complete in CCS C Compiler the program
	
	`#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`
	
	`#byte PORTB =0xF81`
	`/* We attribute to the memory position 0xF81 the name PORTB. This means that we define an 8-bit variable whose value will be stored to the memory position F81h.*/`
	`#byte PORTD =0xF83`
	`// The position F83h is the PORTD data register.`
	
	`void main()`
	`{`
	`   set_tris_d(0x00);//PortD becomes output`
	`   set_tris_b(0xFF);//PortB becomes input`
	`   PORTD=0x0F;`
	`   //PortD is given initial value 00001111`
	`   while(TRUE) {`
	
	`   // Wait until the button is pressed`
	`   // 50ms delay to avoid bounce effect`
	
	`   // Wait until the button is released`
	`   // 50ms delay to avoid bounce effect`
	
	`      PORTD=PORTD^0b11111111;`
	`   //Invert PORTD bits via XOR logic gate`
	`      }`
	`}`

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.

2.3 Activity 3. Reading a Push Button with function

In this activity we want every time a push-button is pressed and released the state of each pin of PORTD changes, i.e. 1's become 0's and 0's become 1's. PORTD will be given the initial value 00001111. A function will be used to read the push button.

Table 4. 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. **Step5.** Modifications and discussion.
Step 1 (10 minutes)	Draw the circuit of the Picture 3 in the Proteus Design Suite
Step 2 (30 minutes)	Complete in CCS C Compiler the program

```
#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


#byte PORTB =0xF81
/* We attribute to the memory position 0xF81 the
name PORTB. This means that we define an 8-bit
variable whose value will be stored to the memory
position F81h.*/
#byte PORTD =0xF83
// The position F83h is the PORTD data register.

void push_button(void);
//Declaration of push-button function

void main()
{
   //PortD becomes output
   //PortB becomes input
   PORTD=0x0F;
   //PortD is given initial value 00001111
   while(TRUE) {
```

	`//The push-button function is called`
	`//switch states in PORTD bits`
	`}`
	`}`
	
	`//Define push_button function`
	`//The function definition is written after the main program`
	`//With this function the microcontroller waits`
	`//a full button press (press+release)`
	`//and then go to the next command.`
	
	`void push_button(void) {`
	`// Wait until the button is pressed`
	`// In the wait state, no command is executed`
	`// When the button is pressed the loop is exited`
	`// and the next command is executed`
	`// 50ms delay to avoid bounce effect`
	`//Wait until the button is released`
	`// In the wait state, no command is executed`
	`// When the button is released the loop is exited`
	`// and the next command is executed`
	`// 50ms delay to avoid bounce effect`
	`}`

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 (20 minutes)	Suggested modifications and discussion: • modify the hardware and the code so that the push-button been connected in RD0

Chapter 3: **Recapitulation**

☞ The schematic of the circuits was drawn with Proteus Design Suite

☞ 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

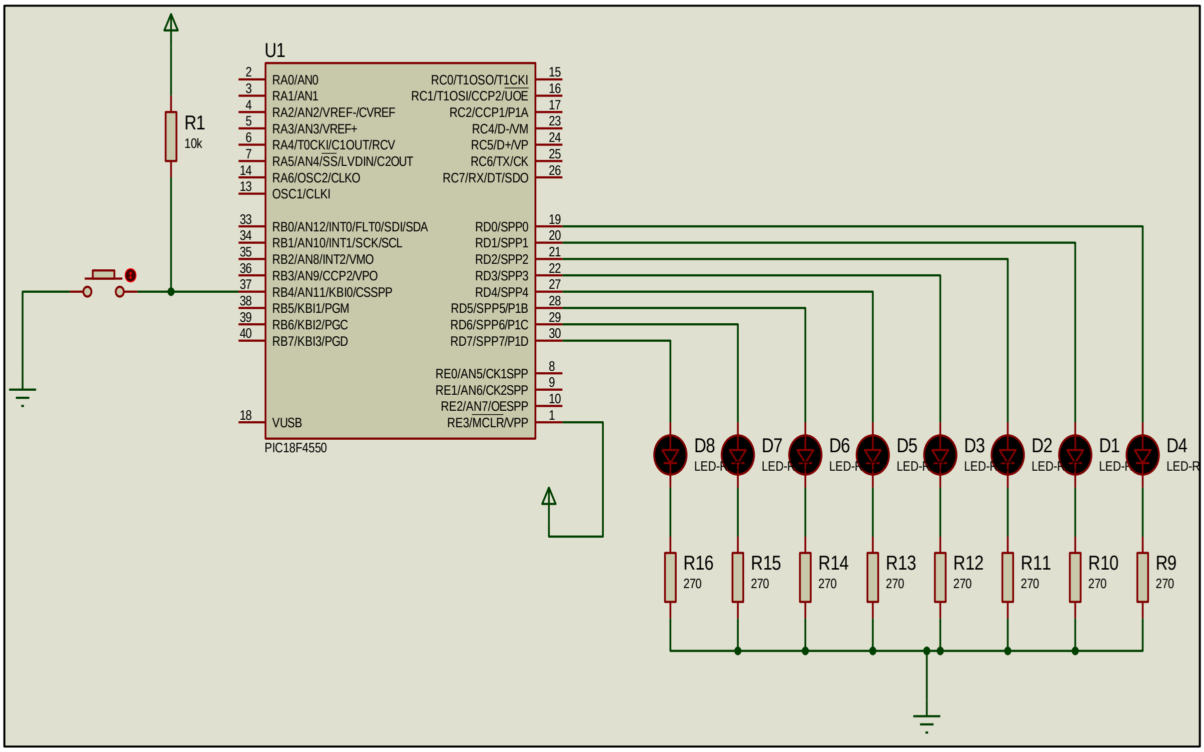


Figure 1. Connections

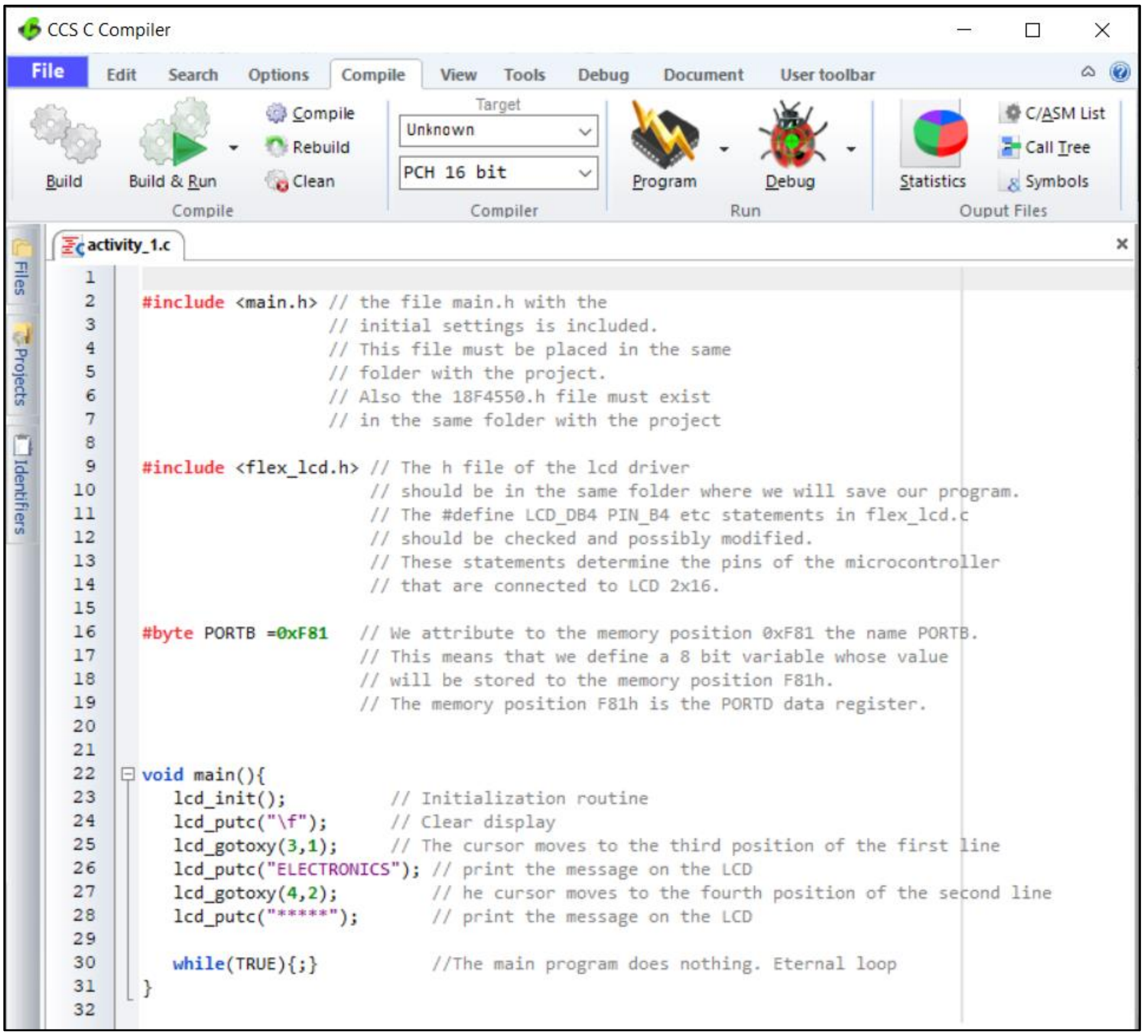


Figure 2. CCS C Compiler, translation to machine code (hex file)

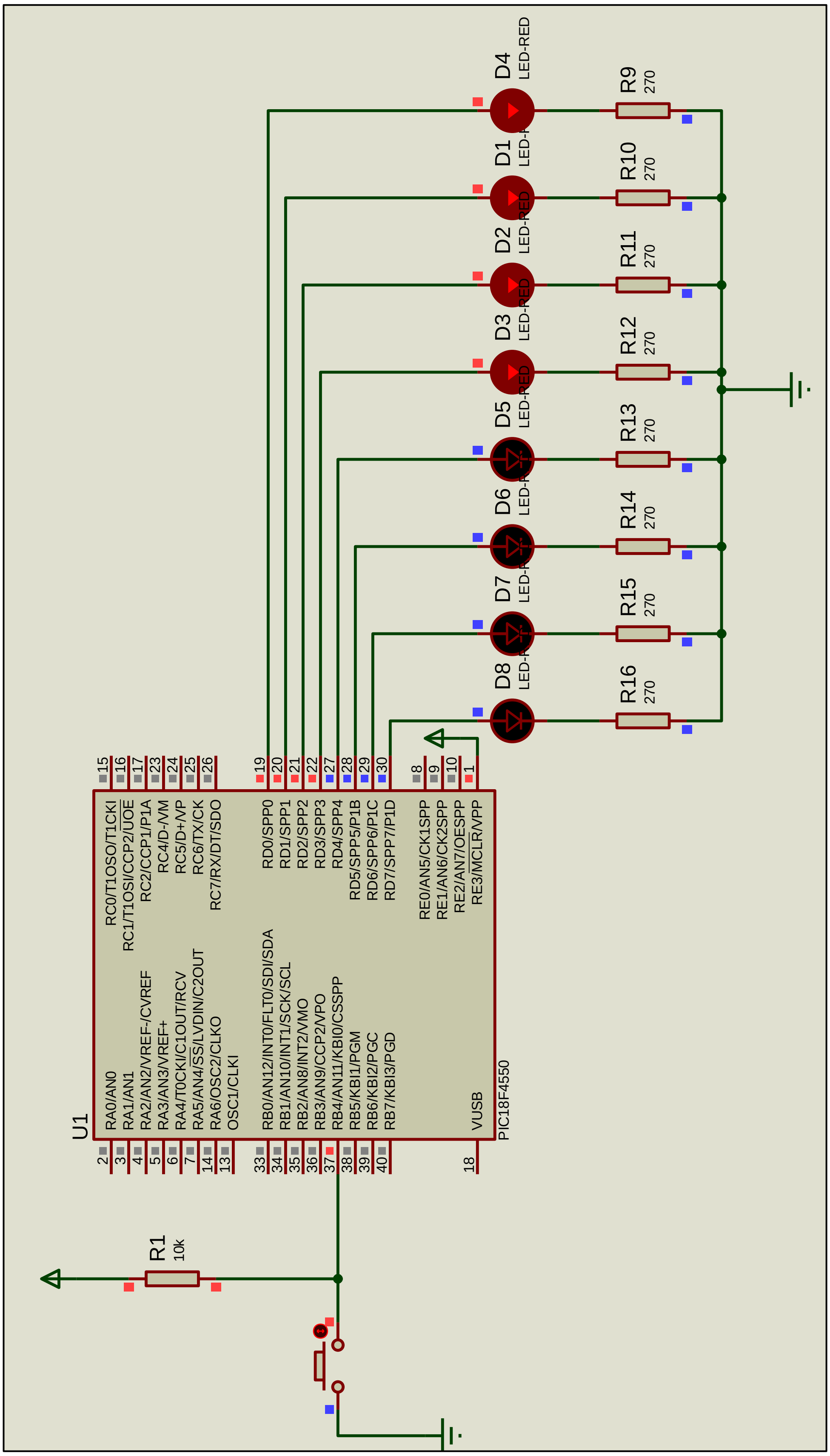


Figure 3. Push-button and LEDs