ENGINE

Teaching online electronics, microcontrollers and programming in Higher Education

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

Practice leaflet: Module_1-4 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 Where other published and unpublished source materials have been used, these have been acknowledged.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 4

Chapter 1: Overview 5

Chapter 2: Activities 8

2.1 Activity 1. Basics with LCD display 16x2 8

2.2 Activity 2. LCD 16x2 and push-buttons 12

2.3 Activity 3. LCD 16x2 and ADC 15

Chapter 3: Recapitulation 19

References 20

Executive summary

In this Module we will use a Liquid crystal display (16 columns, 2 rows).

Chapter 1: **Overview**

Table 1. Overview

Title / short summary	**LCD display 16x2:** **Liquid crystal display (16 columns, 2 rows)**
Expected learning outcomes	Students completing the course will be able to: • Recognize basic Arduino Uno functions and programming structures • Use a library to communicate with the LCD 16x2 • Design and implement simple circuits with LCD 16x2
Keywords	Liquid crystal display 16x2
Duration	The duration of the module_1-4 is 3 hours • Module_1-4 slides - 30 minutes • 1st activity: Basics with LCD 16x2 - 50 minutes • 2rd activity: LCD 16x2 and push-buttons - 50 minutes • 3nd activity: LCD 16x2 and ADC - 50 minutes

Involved	**The students**: • Take part in activities • Complete code • Answer questionnaires **The teachers**: • Show the presentation of the module • Answer questions • Point out the tips • Encourage participation and discussion
Assignment	The module_1-4 includes: • 2 Open Projects
Educational tools and equipment	• Material: PC • Software: browser, Tinkercad
Prerequisites / pre-existing knowledge	• Students should have knowledge of wiring electronic components in breadboard (link1) • Students should have basic programming knowledge in C language (link2) • Students should be familiar with the Tinkercad environment (link3, tutorial video) • Students should have studied the educational material (slides) of Module_1-1, Module_1-2, Module_1-3, and Module_1-4
Educational content	Accompanying material: • Module_1-4 slides • Module_1-4 Evaluation leaflet • Module_1-4 Open Projects

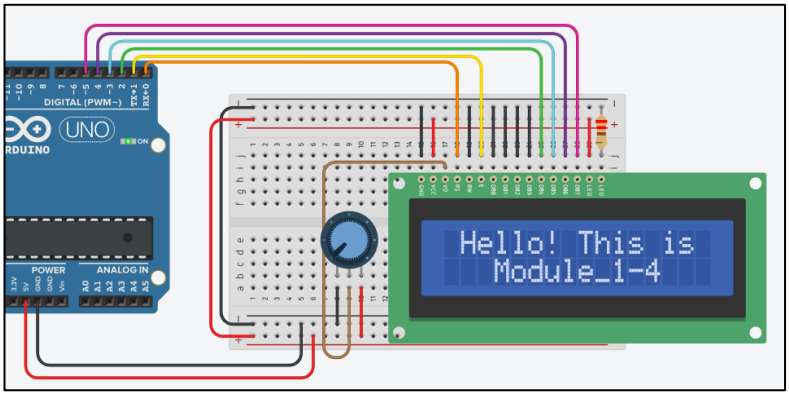
Tips	***Tip1***. *Adjusting the contrast on the LCD is done via a potentiometer* ***Tip2***. *The current in the LCD backlight must be limited by means of a resistor* ***Tip3***. *"lcd.setCursor()" starts counting (column or row) from 0, not from 1*

Chapter 2: **Activities**

2.1 Activity 1. Basics with LCD display 16x2

This activity uses a liquid crystal display 16x2.

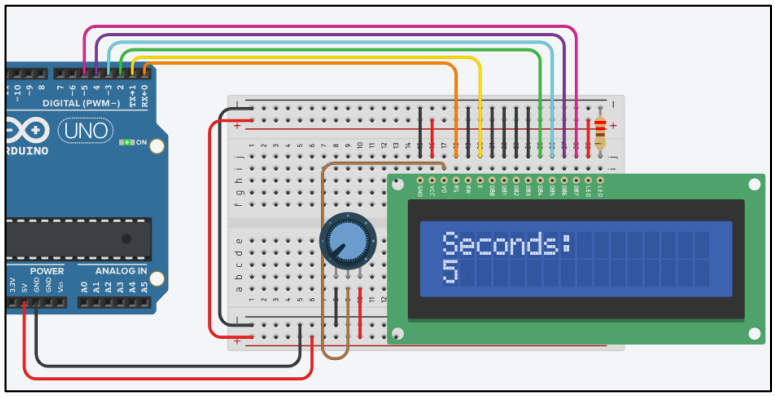
Table 2. Activity 1

Activity 1a (22 minutes)	In this part the aim is for the Arduino Uno to display the message "Hello! This is Module_1-4 "on the LCD. **Step 1**. Draw the circuit in Tinkercad **Step 2**. Write the microcontroller code **Step 3**. Simulate the circuit and test it
Step 1 (10 minutes)	Draw the next circuit in Tinkercad.  *Figure 1. LCD display "hello"*

Step 2
(10 minutes)

Study the code and write it on the microcontroller:

```
/* Hello! This is Module_1-4

Circuit Connections:
** LCD
        Ground      => Gnd
        Power       => Vcc
        Contrast    => Potentiometer
        RS          => PIN_0
        RW          => Gnd
        E           => PIN_1
        DB0         => Gnd
        DB1         => Gnd
        DB2         => Gnd
        DB3         => Gnd
        DB4         => PIN_2
        DB5         => PIN_3
        DB6         => PIN_4
        DB7         => PIN_5
        LED Anode   => Vcc
        LED Cathode => Resistor 220Ω => Gnd
** Potentiometer
        Terminal 1  => Gnd
        Wiper       => LCD_Contrast
        Terminal 2  => Vcc
*/

//include the library
#include <LiquidCrystal.h>

#define RS 0             //give the name "RS " to
PIN_0
#define EN 1       //give the name "EN " to PIN_1
#define DB4 2      //give the name "DB4 " to PIN_2
#define DB5 3      //give the name "DB5 " to PIN_3
#define DB6 4      //give the name "DB6 " to PIN_4
#define DB7 5      //give the name "DB7 " to PIN_5

//configure the library with Arduino Uno - LCD
interface
LiquidCrystal lcd(RS, EN, DB4, DB5, DB6, DB7);

//The setup() function initializes and sets the
initial values
//It will only run once after each power up or
reset
void setup() {
  //configure the LCD's columns and rows
  lcd.begin(16, 2);
  //print a message
  lcd.print(" Hello! This is");
  //go to: first column, second row
  lcd.setCursor(0,1);
  //print a message
  lcd.print("   Module_1-4");
}

//loops consecutively
```

	`` void loop() { ; //do nothing } ``` ***Tip***. *In the loop() we do not need to do anything as the operation of the application has already been achieved.*
Step 3 (2 minutes)	Run the simulation and check the correct operation of the circuit
Activity 1b (28 minutes)	In this part the aim is for the Arduino Uno to count seconds. The display takes place on a liquid crystal display 16x2. **Step 1**. Draw the circuit in Tinkercad **Step 2**. Write the microcontroller code **Step 3**. Simulate the circuit and test it **Step 4**. Modifications and discussion
Step 1 (10 minutes)	Draw the next circuit in Tinkercad.  *Figure 2. Display seconds on the LCD*

Step 2
(10 minutes)

Study the code and write it on the microcontroller. The 2 missing lines must be completed:

```
/* Counting seconds

Circuit Connections:
** LCD
        Ground      => Gnd
        Power       => Vcc
        Contrast    => Potentiometer
        RS          => PIN_0
        RW          => Gnd
        E           => PIN_1
        DB0         => Gnd
        DB1         => Gnd
        DB2         => Gnd
        DB3         => Gnd
        DB4         => PIN_2
        DB5         => PIN_3
        DB6         => PIN_4
        DB7         => PIN_5
        LED Anode   => Vcc
        LED Cathode => Resistor 220Ω => Gnd
** Potentiometer
        Terminal 1  => Gnd
        Wiper       => LCD_Contrast
        Terminal 2  => Vcc
*/

//include the library
#include <LiquidCrystal.h>

#define RS 0             //give the name "RS " to
PIN_0
#define EN 1       //give the name "EN " to PIN_1
#define DB4 2      //give the name "DB4 " to PIN_2
#define DB5 3      //give the name "DB5 " to PIN_3
#define DB6 4      //give the name "DB6 " to PIN_4
#define DB7 5      //give the name "DB7 " to PIN_5

//configure the library with Arduino Uno - LCD
interface
LiquidCrystal lcd(RS, EN, DB4, DB5, DB6, DB7);

//The setup() function initializes and sets the
initial values
//It will only run once after each power up or
reset
void setup() {
  //configure the LCD's columns and rows
  lcd.begin(16, 2);
  //print a message
  lcd.print("Seconds:");
}

//loops consecutively
void loop() {
  //go to: first column, second row
  lcd.setCursor(0,1);
```

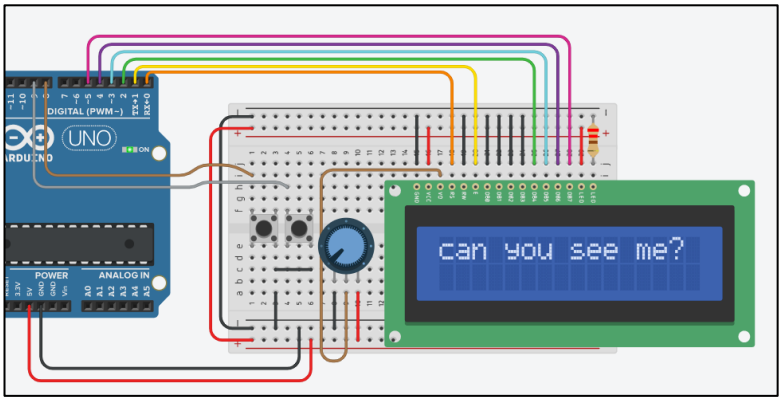
	`//print a message` `lcd.print(millis() / 1000);` `//millis() return the number of milliseconds` `//passed since the program began running` `}`
Step 3 (3 minutes)	Run the simulation and check the correct operation of the circuit
Step 4 (5 minutes)	Suggested modifications and discussion: • Try writing a message on the LCD larger than 16 characters. What will happen?

2.2 Activity 2. LCD 16x2 and push-buttons

In this activity the Arduino Uno reads 2 push-buttons. One push-button hides the text written on a liquid crystal display, while the other push-button reappears the text.

Table 3. Activity 2

Activity 2 (50 minutes)	The text on the LCD is not erased. The text is displayed or not by the display() and noDisplay() functions. **Step 1**. Draw the circuit in Tinkercad **Step 2**. Write the microcontroller code **Step 3**. Simulate the circuit and test it **Step 4**. Modifications and discussion

Step 1 (15 minutes)	Draw the next circuit in Tinkercad.  *Figure 3. LCD and display()/noDisplay()*

Step 2 (15 minutes)	Study the code and write it on the microcontroller. The 2 missing lines must be completed:

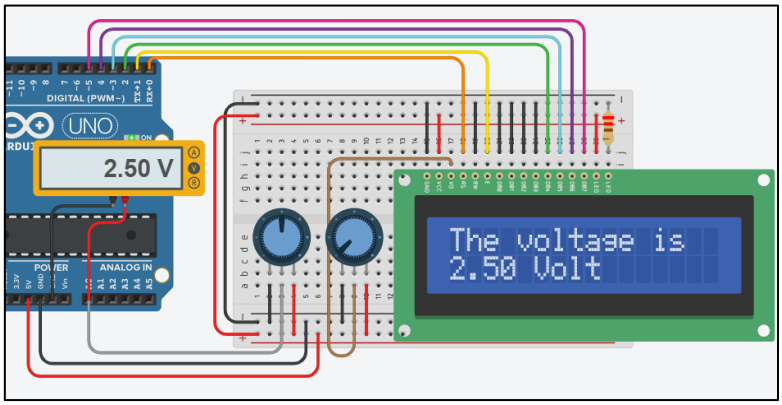
```
/* LCD and display / noDisplay()

Circuit Connections:
** LCD
         Ground      => Gnd
         Power       => Vcc
         Contrast    => Potentiometer
         RS          => PIN_0
         RW          => Gnd
         E           => PIN_1
         DB0         => Gnd
         DB1         => Gnd
         DB2         => Gnd
         DB3         => Gnd
         DB4         => PIN_2
         DB5         => PIN_3
         DB6         => PIN_4
         DB7         => PIN_5
         LED Anode   => Vcc
         LED Cathode => Resistor 220Ω => Gnd
** PIN_8 (built-in pullup) => Push-button1   =>
Gnd
** PIN_9 (built-in pullup)      =>   Push-button2
  => Gnd
*/

//include the library
#include <LiquidCrystal.h>

#define RS 0       //give the name "RS " to PIN_0
#define EN 1       //give the name "EN " to PIN_1
#define DB4 2      //give the name "DB4 " to PIN_2
#define DB5 3      //give the name "DB5 " to PIN_3
#define DB6 4      //give the name "DB6 " to PIN_4
#define DB7 5      //give the name "DB7 " to PIN_5

#define pb1 8     //give the name "pb1" to PIN_8
#define pb2 9     //give the name "pb1" to PIN_9

//configure the library with Arduino Uno - LCD
interface
=>

//The setup() function initializes and sets the
initial values
//It will only run once after each power up or
reset
void setup() {
  //Configure the PIN_8 to behave as input with
pull-up resistor
  pinMode(pb1, INPUT_PULLUP);
  //Configure the PIN_9 to behave as input with
pull-up resistor
  pinMode(pb2, INPUT_PULLUP);
  //configure the LCD's columns and rows
=>
```

	`//print a message` `lcd.print("can you see me?");` `}` `//loops consecutively` `void loop() {` `//check the push-button1` `if(digitalRead(pb1)==false){//pb1 is pressed` `delay(25); //wait for debouncing` `while(digitalRead(pb1)==false){;} //until pb1 is released` `lcd.noDisplay(); //the text disappeared` `}` `//check the push-button2` `if(digitalRead(pb2)==false){ //pb2 is pressed` `delay(25); //wait for debouncing` `while(digitalRead(pb2)==false){;} //until pb2 is released` `lcd.display(); //the text appeared` `}` `}`
Step 3 (5 minutes)	Run the simulation and check the correct operation of the circuit
Step 4 (15 minutes)	Suggested modifications and discussion: • Could the application work instead of two with one push-button? • Add a switch. When the switch is open, the text on the LCD can be hidden by the corresponding push-button. When the switch is closed, the text on the LCD will be displayed whether a push-button is pressed or not. Write the appropriate code and run the simulation

2.3 Activity 3. LCD 16x2 and ADC

This activity uses the Arduino Uno's built-in analog-to-digital converter. A liquid crystal display is used as the output device.

Table 4. Activity 3

Activity 3 (50 minutes)	The Arduino Uno: • Reads the analog voltage of a potentiometer • Converts adc_value into a voltage • A voltmeter has been added to the circuit to check the voltage of the potentiometer **Step 1**. Draw the circuit in Tinkercad **Step 2**. Write the microcontroller code **Step 3**. Simulate the circuit and test it **Step 4**. Modifications and discussion
Step 1 (10 minutes)	Draw the next circuit in Tinkercad.  *Figure 4. Arduino Uno as Voltmeter*

Step 2 (15 minutes)	Study the code and write it on the microcontroller:

```
/* Voltmeter

Circuit Connections:
** LCD
        Ground      => Gnd
        Power       => Vcc
        Contrast    => Potentiometer
        RS          => PIN_0
        RW          => Gnd
        E           => PIN_1
        DB0         => Gnd
        DB1         => Gnd
        DB2         => Gnd
        DB3         => Gnd
        DB4         => PIN_2
        DB5         => PIN_3
        DB6         => PIN_4
        DB7         => PIN_5
        LED Anode   => Vcc
        LED Cathode => Resistor 220Ω => Gnd
** Potentiometer1
        Terminal 1  => Gnd
        Wiper       => LCD_Contrast
        Terminal 2  => Vcc
** Potentiometer2
        Terminal 1  => Gnd
        Wiper       => PIN_A0
        Terminal 2  => Vcc
*/

//include the library
#include <LiquidCrystal.h>

#define RS 0       //give the name "RS " to PIN_0
#define EN 1       //give the name "EN " to PIN_1
#define DB4 2      //give the name "DB4 " to PIN_2
#define DB5 3      //give the name "DB5 " to PIN_3
#define DB6 4      //give the name "DB6 " to PIN_4
#define DB7 5      //give the name "DB7 " to PIN_5
#define pot_pin A0 //give the name "pot_pin" to
PIN_A0

//configure the library with Arduino Uno - LCD
interface
LiquidCrystal lcd(RS, EN, DB4, DB5, DB6, DB7);

//variable to save data from ADC
int adc_value; //number range 0~1023
//variable to calculate the analog voltage
float voltage;

//The setup() function initializes and sets the
initial values
//It will only run once after each power up or
reset
void setup() {
  //configure the LCD's columns and rows
```

	`` lcd.begin(16, 2); //print a message lcd.print("The voltage is"); } //loops consecutively void loop() { //read value from ADC adc_value = analogRead(pot_pin); //calculate the analog voltage voltage=(float)adc_value*5/1024; //go to: first column, second row lcd.setCursor(0,1); //print a message lcd.print(voltage); lcd.print(" Volt"); //wait for 0.5s delay(500); } ```
Step 3 (5 minutes)	Run the simulation and check the correct operation of the circuit
Step 4 (20 minutes)	Suggested modifications and discussion: • Do the voltmeter readings always match the value on the LCD? • What should be changed in order to read the values of potentiometer2 from PIN_A5? Replace the potentiometer2 with the temperature sensor TMP36 and turn the Arduino Uno into a thermometer. The temperature in degrees Celsius measured by the sensor is $T = (V_{sensor} - 0.5) * 100$

Chapter 3: **Recapitulation**

The circuits were designed and simulated with Tinkercad.

Basic Arduino Uno programming functions were used, such as:

- millis()
- lcd.begin()
- lcd.print()
- lcd.setCursor()
- lcd.display()
- lcd.noDisplay

Through the activities were utilized

- Arduino Uno pins for driving a LCD 16x2

References

Breadboard \ Wiring. Retrieved from http://wiring.org.co/learning/tutorials/breadboard/

Brown, R. (2020). *Active vs. Passive buzzer: the differences*. Retrieved from https://nerdytechy.com/active-vs-passive-buzzer/

Learn C - Free Interactive C Tutorial. Retrieved from https://www.learn-c.org/

Learn how to use Tinkercad | Tinkercad. Retrieved from https://www.tinkercad.com/learn/circuits