Министерство Образования и Исследований Республики Молдова
Технический Университет Молдовы
Факультет Вычислительной Техники, Информатики и Микроэлектроники
Департамент Программной Инженерии и Автоматики

Лабораторная работа №1

по предмету «Интернет вещи»

Выполнил: ст. гр. TI-196
Н. Шарафудинов
Проверил: А. Бырназ

Кишинев 2022

Тема: Взаимодействие с пользователем

Цель: Реализация приложений для взаимодействия с пользователем, простая Button-Led и классическая Keypad-LCD с использованием библиотеки STDIO для облегчения классического взаимодействия с функциями типа printf(..) и scanf(...)

Задание:

1. разработать приложение на базе микроконтроллера, которое изменяет состояние светодиода при обнаружении нажатия на кнопку

2. разработать приложение в базе микроконтроллера, которое будет получать команды от терминала через последовательный интерфейс для установки состояния светодиода.

- ledOn() для включения и ledOff() для выключения, система должна отвечать текстовыми сообщениями о подтверждении команды
- использовать библиотеку STDIO для обмена текстом через терминал

3. разработать приложение на базе микроконтроллера для обнаружения кода с клавиатуры 4x4, для проверки кода и отображения сообщения на ЖК-дисплее.

- для действительного кода загорается зеленый светодиод, для недействительного кода - красный светодиод.
- используйте STDIO для сканирования клавиатуры и отображения на ЖК-дисплее.

Ход работы:

Выполнение работы происходило на платформе Autodesk Tinkercad, так же использовался видеоматериал, прикрепленный к лабораторной работе на платформе ELSE. Для работы с командами Arduino был использована документация с сайта https://www.arduino.cc/reference/en/

Задание 1

Разработать приложение на базе микроконтроллера, которое изменяет состояние светодиода при обнаружении нажатия на кнопку

для определение куда будет подключена кнопка использовалась команда pinMode() – она необходима для настройки указанного контакта на вход или выход.

Команда имеет следующую структуру

- pinMode (pin, mode) где
    - **pin**: номер вывода Arduino для установки режима.
    - **mode**: INPUT, OUTPUT или INPUT_PULLUP.
        - INPUT – Если ваш контакт настроен как ВХОД и вы считываете данные с переключателя, когда переключатель находится в разомкнутом состоянии,

входной контакт будет «плавающим», что приведет к непредсказуемым результатам. Чтобы обеспечить правильное считывание, когда переключатель разомкнут, необходимо использовать подтягивающий или подтягивающий резистор.

- OUTPUT – выводы, настроенные как OUTPUT с помощью pinMode(), считаются в состоянии с низким импедансом. Это означает, что они могут обеспечить значительное количество тока для других цепей.
- INPUT_PULLUP – Микроконтроллер ATmega на Arduino имеет внутренние подтягивающие резисторы, к которым вы можете получить доступ.
- LED_BUILTIN – Большинство плат Arduino имеют контакт, подключенный к встроенному светодиоду последовательно с резистором. Константа LED_BUILTIN — это номер вывода, к которому подключен встроенный светодиод. На большинстве плат этот светодиод подключен к цифровому контакту 13.

При выполнении, работы вывод (светодиод) был определен с помощью константы LED_BUILTIN, а также из-за использования внутреннего резистора платы для корректной работы входа использовалась константа INPUT_PULLUP.

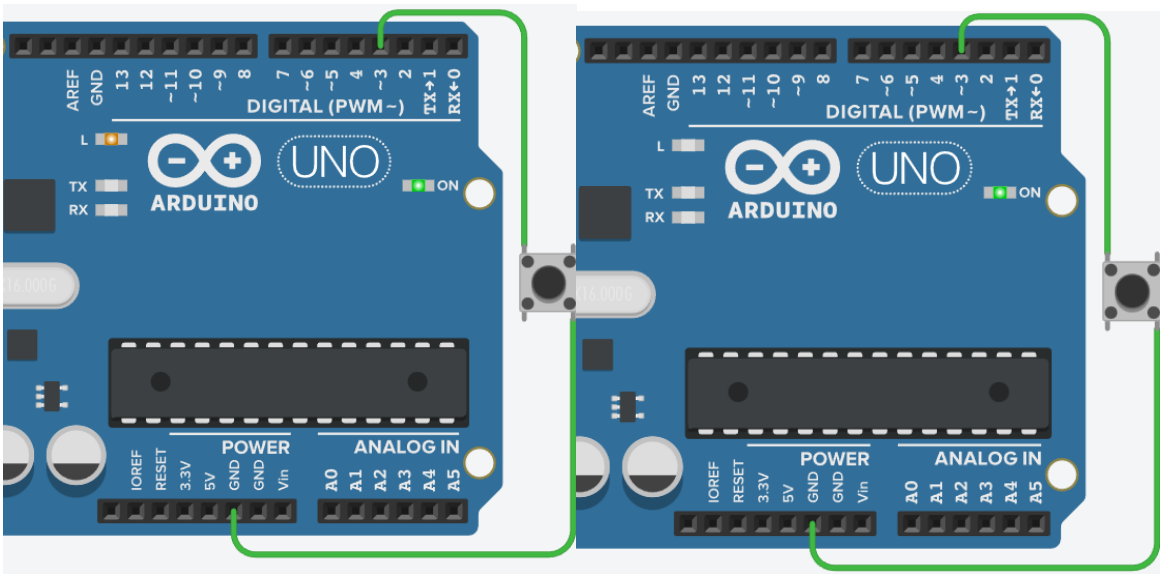


Рисунок 1. Схема 1-го задания

Основную работу выполняет функция ledOnOff() которая принимает состояние диода и сообщение которое будет выведено в консоль.

```
void ledOnOff(bool state, String
message){
  if((millis() - timing > 300)){
    Serial.println(message);
    timing = millis();
    digitalWrite(ledPin, state);
  }
}
```

Так же в этой части кода используется такая функция, как millis() - возвращает количество миллисекунд, прошедших с момента запуска текущей программы на плате Arduino. Это число переполнится (вернется к нулю) примерно через 50 дней.

Подробнее о реализации смотрите в приложение 1.

Задание 2

Разработать приложение в базе микроконтроллера, которое будет получать команды от терминала через последовательный интерфейс для установки состояния светодиода. Для выполнения этого задания была использована библиотека <stdio.h> так же были переопределены потоки ввода и вывода для того, чтобы использовать их для работы с терминалом.

Были реализованы следующие команды для взаимодействия пользователя с диодом через консоль:

- “on” – включить диод;
- “off” – выключить диод;
- “blink” – мерцание диода;
- “toggle” – переключить текущее состояние на обратное.

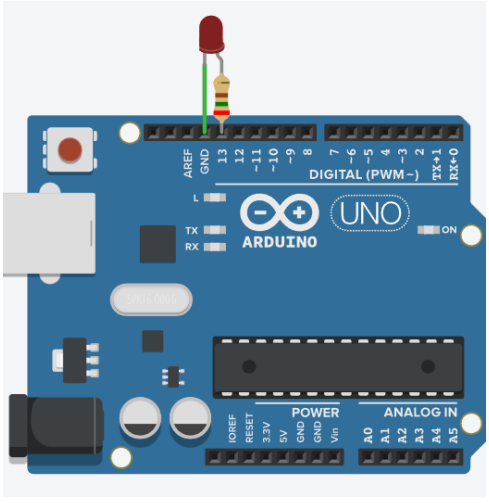


Рисунок 2. Схема 2-го задания

```
Enter [on/off/toggle/blink] Current led status: on

Enter [on/off/toggle/blink] Current led status: off

Enter [on/off/toggle/blink] Led toggled to: on

Enter [on/off/toggle/blink] BLINK
```

Рисунок 3. Вывод в консоль

Подробнее о реализации смотрите в приложение 2.

Задание 3

Разработать приложение на базе микроконтроллера для обнаружения кода с клавиатуры 4x4, для проверки кода и отображения сообщения на ЖК-дисплее. Для работы с ЖК-дисплеем и клавиатурой были использованы 2 библиотеки: <Keypad.h>, <Adafruit_LiquidCrystal.h>, помимо уже ранее упомянутой библиотеки, <stdio.h>

Для работы с клавиатурой была изначально создана матрица символов

```
const byte ROWS = 4;
const byte COLS = 4;
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};
byte rowPins[ROWS] = {11,10, 9, 8};
byte colPins[COLS] = {7, 6, 5, 4};
Keypad keypad = Keypad(    makeKeymap(keys), rowPins,
                           colPins, ROWS, COLS );
```

Так же в данном пункте были переопределены потоки ввода вывода, в качестве ввода используется клавиатура, а в качестве вывода используется ЖК-дисплей.

Для изменения потоков ввода и вывода использовалась функция fdev_setup_stream. **fdev_setup_stream** – предоставленный пользователем буфер как поток stdio. Этот макрос принимает предоставленный пользователем буферный поток и устанавливает его как поток, допустимый для операций stdio, подобно тому, который был получен динамически из fdevopen(). Буфер для настройки должен иметь тип FILE. Аргументы put и get идентичны тем, которые необходимо передать в fdevopen(). Аргумент rwflag может принимать одно из значений _FDEV_SETUP_READ, _FDEV_SETUP_WRITE или _FDEV_SETUP_RW для чтения, записи или чтения.

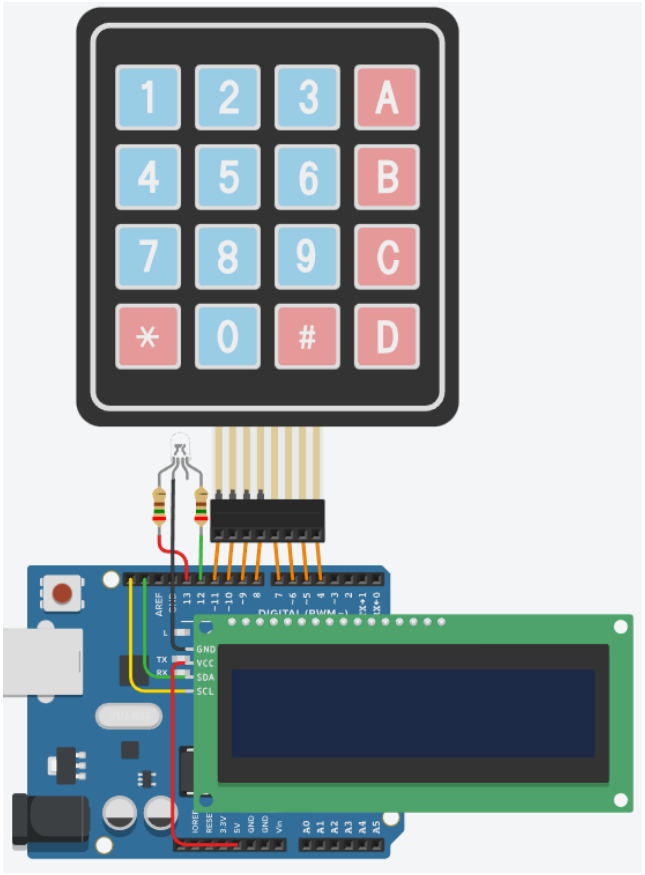
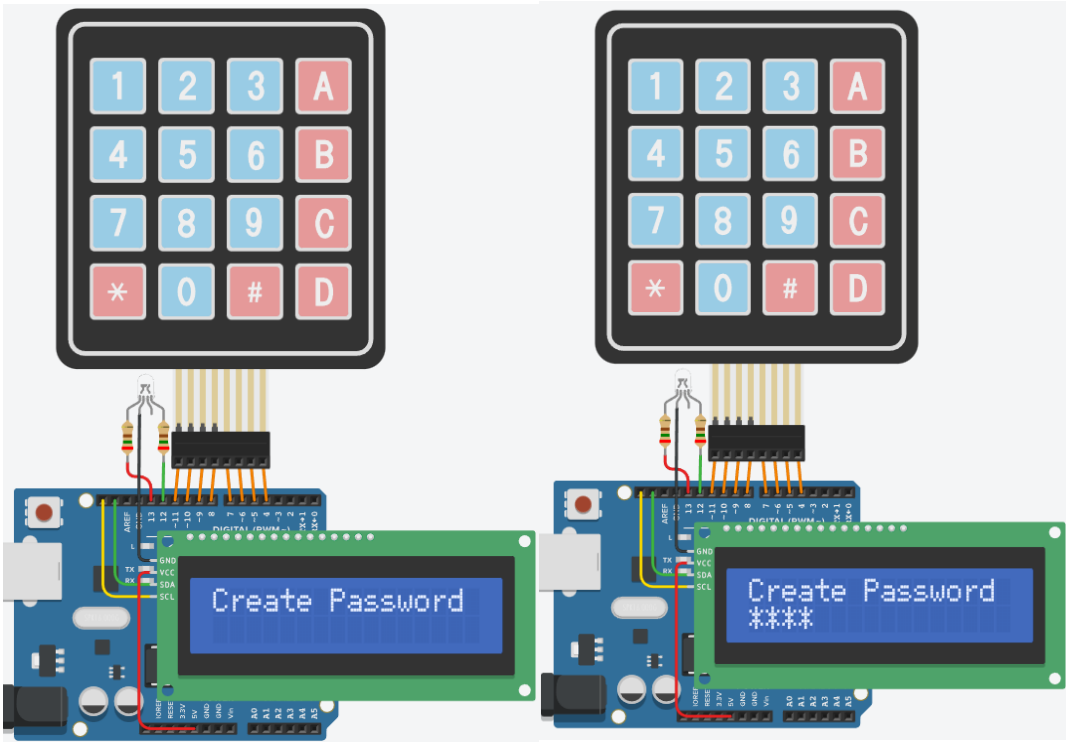


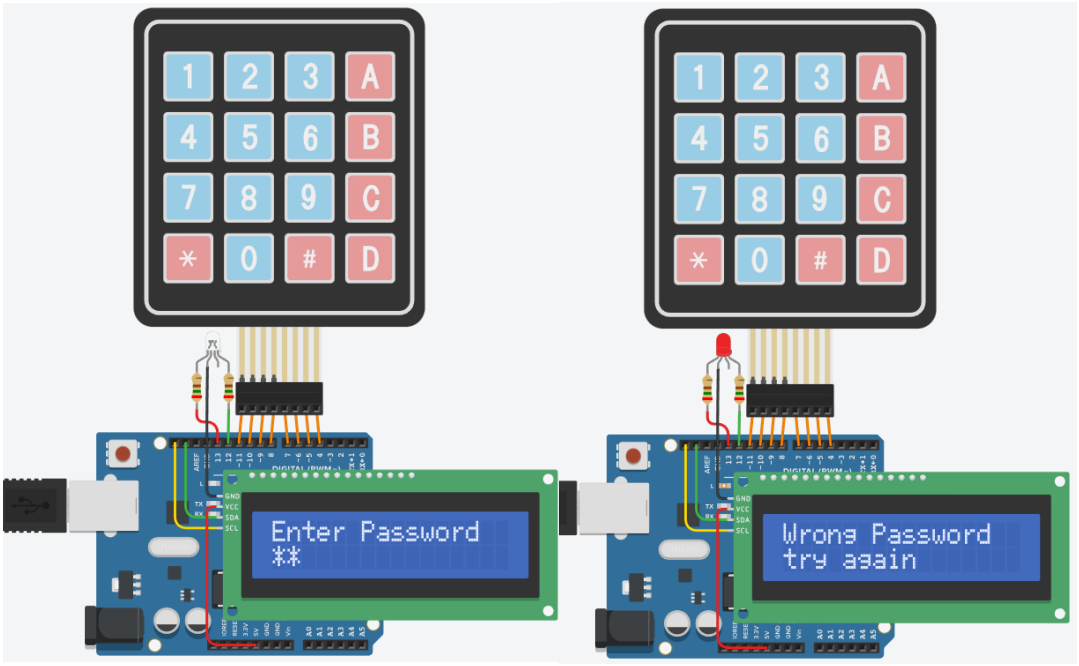
Рисунок 4. Схема 3-го пункта

Результат работы схемы 3:

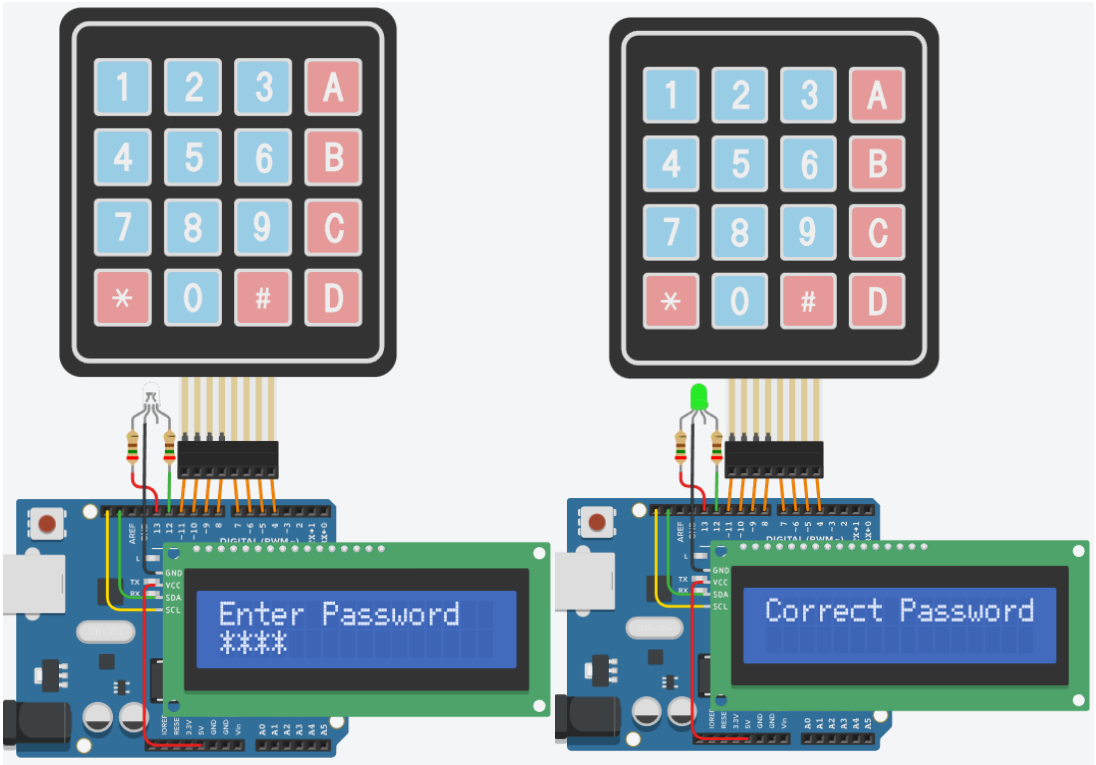
Создание пароля: (пусть пароль 123A) клавиша ввода «*»



Ввод пароля: (ввод неверного пароля)



Ввод пароля: (ввод верного пароля)



Подробнее о реализации смотрите в приложение 3.

Вывод

В данной лабораторной работе были рассмотрены способы взаимодействия пользователя с приложение на базе микроконтроллера с помощью нажатия на кнопку, зажигающую диод, с помощью написания пользователем команд в консоль для управления диодом, а также с помощью клавиатуры и ЖК-экрана с использованием библиотеки STDIO для облегчения классического взаимодействия с функциями типа printf(..) и scanf(...).

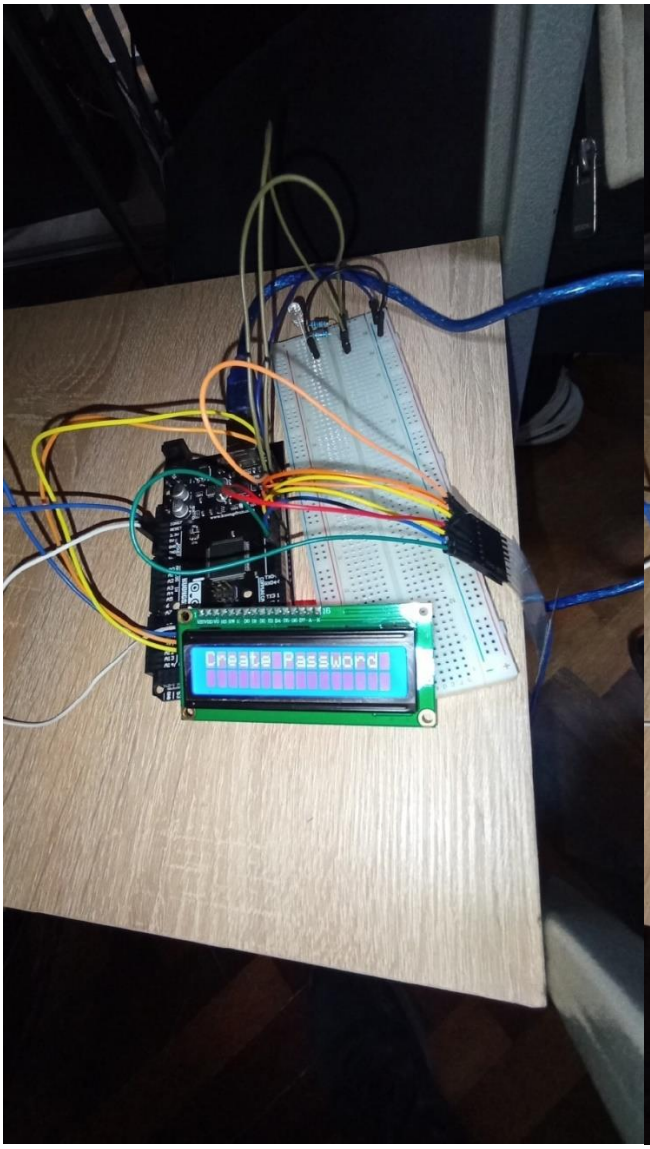
В ходе работы были рассмотрены разные способы взаимодействия с приложение на базе микроконтроллера, что показало, что с микроконтроллерами можно взаимодействовать очень гибко, обезличив ввод и вывод стало в разы проще использовать их, не обращая внимание на то какой должен быть описан ввод или вывод.

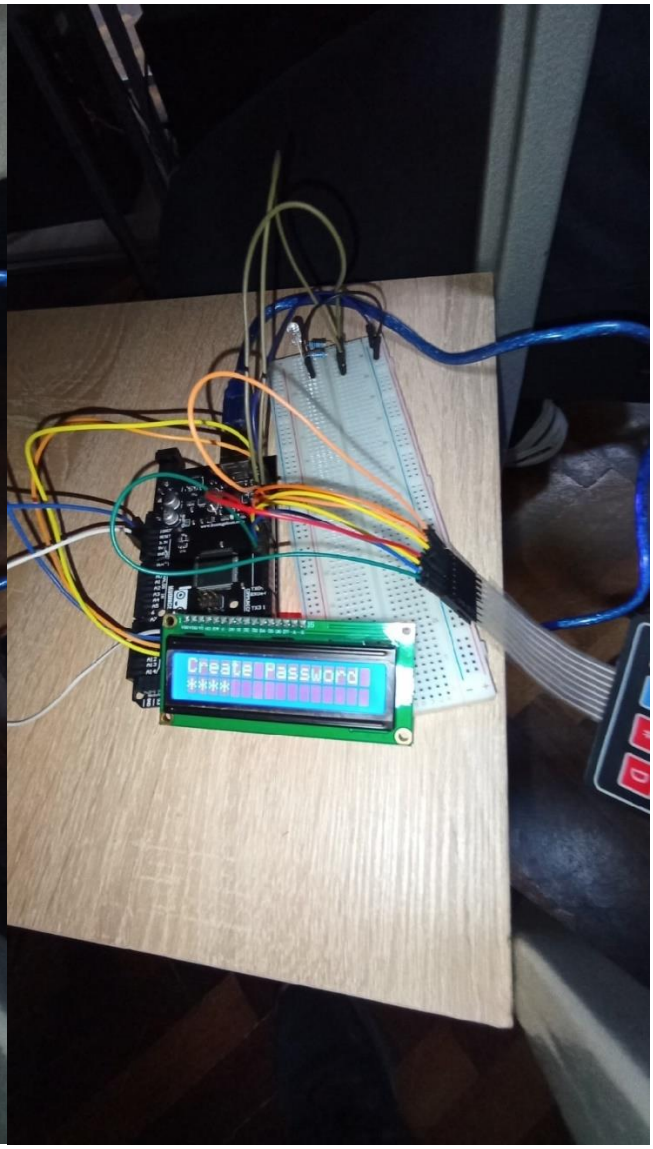
Так же стало понятно, что на платформе Autodesk Tinkercad можно эмулировать все эти цепи что позволяет с удобством пользоваться как схемами, так и помогает с отсутствием личного набора Arduino.

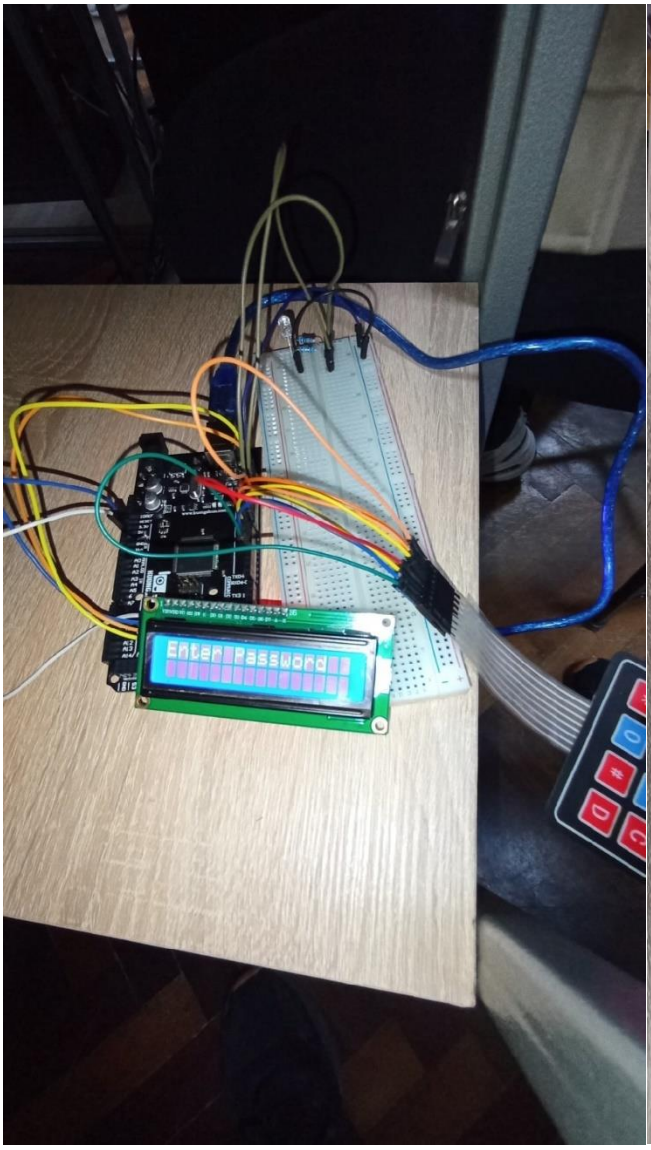
Ссылки на схемы в Tinkercad:

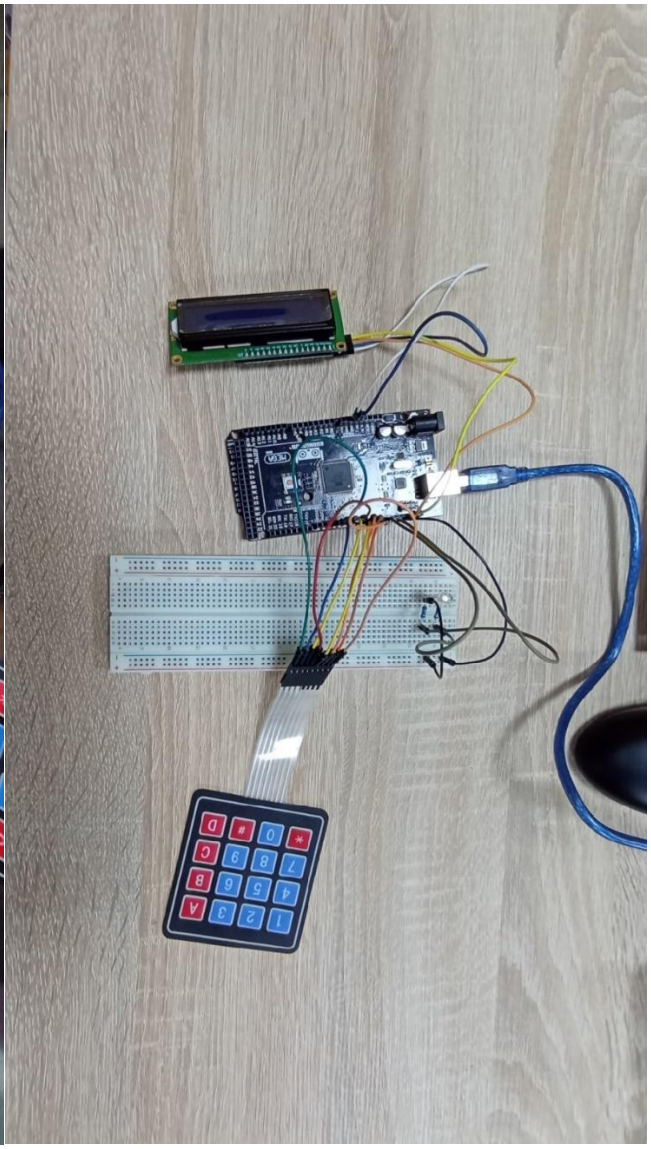
- Задание 1: https://www.tinkercad.com/things/hnVdxHgI8mH?sharecode=2x3TL19JSB3VpN8kgMhWpenDIWJlQnMCj1wX3OQWAMI
- Задание 2: https://www.tinkercad.com/things/fyOoOaa2f4n?sharecode=ji5Iu2XTLJaQOTtXoxjgGOawgUsM1REwQJHu0F3dz7o
- Задание 3: https://www.tinkercad.com/things/3RHOVrVYfOm?sharecode=TfffLyQltqRrtB3RkjWb_XgNJi3QuaHfTtfXJQ7MIAg

Собранная схема в живую






EnterPassword



Библиография:

1. Документация Arduino - https://www.arduino.cc/reference/en/
2. Стандартные средства ввода/вывода - https://www.nongnu.org/avr-libc/user-manual/group__avr__stdio.html
3. Документация библиотеки Keypad.h - https://playground.arduino.cc/Code/Keypad/
4. Документация библиотеки Adafruit_LiquidCrystal.h - https://www.arduino.cc/reference/en/libraries/liquidcrystal/
5. Учебные материалы предоставленные универом - https://else.fcim.utm.md/course/view.php?id=429#section-4

Приложение 1

```
unsigned long timing;

int ledPin = LED_BUILTIN;
int button = 3;

char ledOn[] = "led on";
char ledOff[] = "led off";

void setup()
{
  pinMode(ledPin, OUTPUT);
  pinMode(button, INPUT_PULLUP);
  Serial.begin(9600);
}

void loop()
{
  bool pushed = !digitalRead(button);

  if(pushed && !digitalRead(ledPin)){
    ledOnOff(true, "led on");
  }

  if(pushed && digitalRead(ledPin)){
    ledOnOff(false, "led off");
  }
}

void ledOnOff(bool state, String message){
  if((millis() - timing > 300)){
    Serial.println(message);
    timing = millis();
    digitalWrite(ledPin, state);
  }
}
```

Приложение 2

```
#include <stdio.h>
#define MSG_LEN 7
unsigned long timing = 0;
const long interval = 500;

int ledState = LOW;
int ledPin = LED_BUILTIN;

char onMsg [MSG_LEN]= "on";
char offMsg [MSG_LEN]= "off";
char blinkMsg [MSG_LEN]= "blink";
char toggleMsg [MSG_LEN]= "toggle";

static FILE my_stream = {0};

int my_putChar( char ch, FILE * f){
  return Serial.write( ch );
}

int my_GetChar(FILE * f){
  while (!Serial.available());
  return Serial.read();
}

void setup()
{
  pinMode(ledPin, OUTPUT);
  Serial.begin(9600);

  fdev_setup_stream(&my_stream, my_putChar, my_GetChar, _FDEV_SETUP_RW);

  stdin = &my_stream ;
  stdout = &my_stream ;
}

void loop()
{
  char msg[MSG_LEN] = {0};
  printf("\nEnter [on/off/toggle/blink] ");

  scanf("%s",msg);

  if(strcmp(msg, onMsg) == 0){
```

```
    ledOnOff(true, msg);
  }

  if(strcmp(msg, offMsg) == 0){
    ledOnOff(false, msg);
  }

  if(strcmp(msg, toggleMsg) == 0){
    ledToggle();
  }

  if(strcmp(msg, blinkMsg) == 0){
    ledBlink();
  }
}

void ledOnOff(bool state, char message[]){
    printf("Current led status: %s\n", message);
    digitalWrite(ledPin, state);
}

void ledBlink(bool state, char message[]){
  printf("Current led status: %s\n", message);
  digitalWrite(ledPin, state);
}

void ledToggle(void){
  if(digitalRead(ledPin)){
    printf("Led toggled to: off\n");
    digitalWrite(ledPin, LOW);
  }
  else{
    printf("Led toggled to: on\n");
    digitalWrite(ledPin, HIGH);
  }
}

void ledBlink(void){
  printf("BLINK\n");
  while(true){
    unsigned long currentMillis = millis();

    if (currentMillis - timing >= interval) {
      timing = currentMillis;
```

```
      if (ledState == LOW) {
        ledState = HIGH;
      } else {
        ledState = LOW;
      }
      digitalWrite(ledPin, ledState);
    }
  }
}
```

Приложение 3

```
// C++ code
#include <Keypad.h>
#include <stdio.h>
#include <Adafruit_LiquidCrystal.h>
#define MAX_PASS_LEN 10

int rLed = 13;
int gLed = 12;

int oBut = 3;
int yBut = 4;
int gBut = 5;
int bBut = 6;

char pass[MAX_PASS_LEN];
char currPass[MAX_PASS_LEN];

unsigned long timing;

Adafruit_LiquidCrystal lcd(0);

const byte ROWS = 4; // 4 строки
const byte COLS = 4; // 4 столбца
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};
byte rowPins[ROWS] = {11,10, 9, 8};
byte colPins[COLS] = {7, 6, 5, 4};
Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );
int posMainPass = 0, posCurrPass = 0;
int keyOut = '*';

static FILE my_stream = {0};

int lcdPutChar( char ch, FILE * f){
  if(ch != '\0')lcd.print(ch);
}

int keypadGetChar(FILE * f){
  char key;
```

```
  do{ key = keypad.getKey();}
  while(key == 0);
  return key;
}

void setup()
{
  Serial.begin(9600);
  lcd.begin(16, 2);

  pinMode(rLed, OUTPUT);
  pinMode(gLed, OUTPUT);

  pinMode(oBut, INPUT_PULLUP);
  pinMode(yBut, INPUT_PULLUP);
  pinMode(gBut, INPUT_PULLUP);
  pinMode(bBut, INPUT_PULLUP);

  fdev_setup_stream(&my_stream, lcdPutChar,
                    keypadGetChar, _FDEV_SETUP_RW);

  stdin = &my_stream ;
  stdout = &my_stream ;
}

void loop()
{
  //create password
  if(!pass[0]){
    lcd.clear();
    printf("Create Password");
    enterPassword(pass);
  }

  lcd.clear();
  printf("Enter Password");
  enterPassword(currPass);

  if(compare())
    blink(gLed, "Correct Password");
  else
    blink(rLed, "Wrong Password");
}
void enterPassword(char pass[]){
```

```
  int pos = 0;
  while(true){
    char key;
    scanf("%c", &key);
    if(key == keyOut){
      for(int i = 0; i < realLength(); i++)
       printf("%s", pass[i]);
      break;
    }

    pass[pos] = key;
    lcd.setCursor(pos, 1);
    printf("*");
    pos++;

  }
}

bool compare(void){
  for(int i = 0; i < realLength(); i++){
    if(currPass[i] != pass[i])
      return false;
  }
  return true;
}

void blink(int led, char msg[]){
  lcd.clear();
  lcd.setCursor(0, 0);
  printf(msg);

  if(led == 13){
    lcd.setCursor(0, 1);
    printf("try again");
    digitalWrite(led, HIGH);
    delay(500);
    digitalWrite(led, LOW);
    loop();
  }
  else{
       while(true)
      digitalWrite(led, HIGH);
  }
}
```

```
int realLength(void){
  int count = 0;
  while(pass[count++]){}
  return count;
}
```