УДК 004.624

Игра «Саймон говорит» на платформе Arduino

Кизянов Антон Олегович
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье описан процесс создания популярной игры «Саймон говорит» на базе платы Arduino. Для создания используется плата Arduino, кнопки, светодиоды и пьезоэлемент. Игра представляет из себя случайные комбинации загорающихся светодиодов, которые должен повторить игрок, с каждом новым уровнем добавляется по 1 светодиоду.
Ключевые слова: Arduino, Светодиоды

Simon Says game on Arduino platform

Kizyanov Anton Olegovich
Sholom-Aleichem Priamursky State University
student

Abstract
This article walks you through the process of creating the popular Simon Says game using the Arduino board. For creation, an Arduino board, buttons, LEDs and a piezoelectric element are used. The game consists of random combinations of lighting up LEDs, which the player must repeat, with each new level 1 LED is added.
Keywords: Arduino, LEDs

Игра «Саймон говорит» это классический пример запоминания длинных комбинаций. Суть игры заключается в запоминании последовательности подсветки светодиодов и повторении этой последовательности через кнопки. Если последовательность будет нарушена, то игрок проигрывает и все начинается заново.

Цель исследования – создать схему игры «Саймон говорит» на платформе Arduino.

Ранее этим вопросом интересовались Я.Н. Стецюк, М.В. Слива развивали тему «Работа с графическими экранами и микроконтроллерами (на основе платформы arduino)» [1] в которой рассмотрены основные принципы работы с устройствами экранного вывода, предназначенными для платы Arduino. В качестве примеров устройств экранного вывода будут использоваться: цветной дисплей Color LCD Shield for Arduino и монохромный дисплей MT-12864J. Б.Р. Ахметзянов с темой «Вывод информации с датчиков на oled lcd экран на основе платы семейства

arduino» [2], рассматривается OLED LCD экран, физические элементы (датчики), а также вывод значений с датчиков на данный дисплей на основе платы семейства Arduino. Данная реализация уникальна тем, что при взаимодействии дисплея с датчиками необходимо учитывать схемотехническую особенность каждого элемента, «отклик» экрана при временных считываниях с датчиков. И.М. Ячиков, Е.О. Кряжев, Ю.В. Кочержинская опубликовали статью «Программно-аппаратный комплекс для измерения тепловых параметров системы охлаждения лабораторного высокочастотного индуктора на базе контроллера arduino» [3] описали устройство на базе контроллера Arduino для автоматизации измерения расхода воды и мощности тепловых потерь в лабораторном высокочастотном индукторе. Приведено описание работы устройства, алгоритм программы для контроллера и блок-схема алгоритма. Созданный аппаратный комплекс отображает на дисплее расход, температуры и мощность тепловых потерь. Предусмотрена возможность с использованием монитора последовательного порта полученные данные выдавать на монитор компьютера и сохранять их в виде файла.

Для этого потребуется:

- Плата Arduino;
- Соединительные провода;
- 4 светодиода;
- 9 резистора по 220 Ом;
- 4 кнопки;
- 1 пьезо элемент.

Схема подключения представлена на рисунке 1.

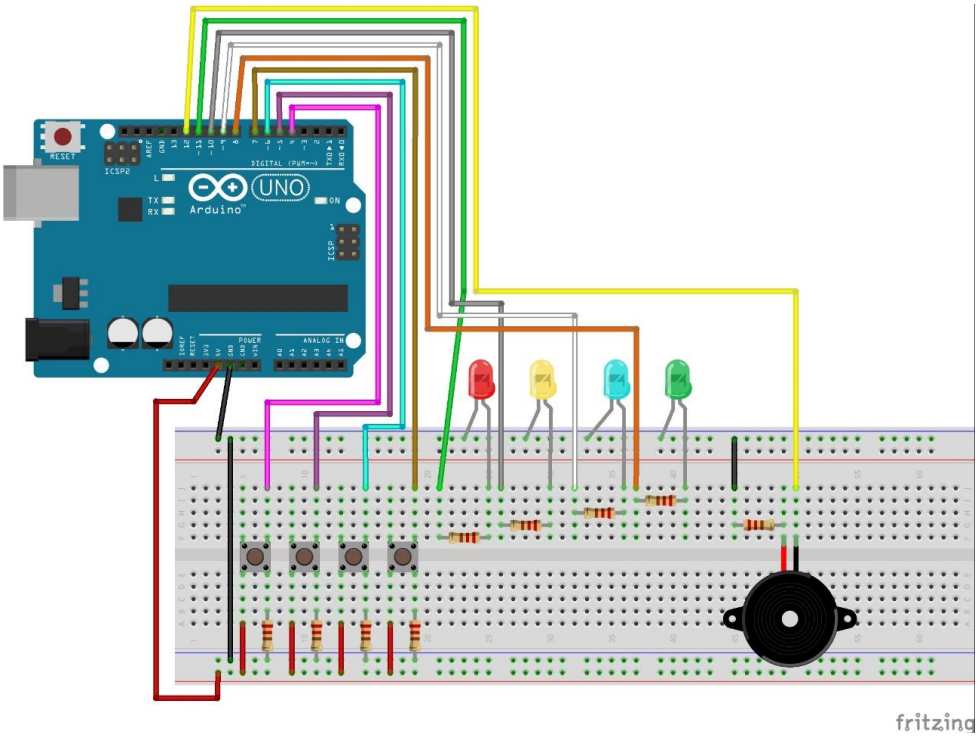

Рис. 1 Схема подключения к плате Arduino

```
class GameClass {
  private:
    int debounce(int last, int buttonPin);
    void playNote(int note, int noteSpeed) const;
    void flashLed(int led, int flashSpeed) const;
  public:
    static const int REDPIN;
    static const int BLUEPIN;
    static const int GREENPIN;
    static const int YELLOWPIN;
    static const int MICROPHONEPIN;
    static const int REDBUTTONPIN;
    static const int BLUEBUTTONPIN;
    static const int GREENBUTTONPIN;
    static const int YELLOWBUTTONPIN;
    static const int RED_TONE;
    static const int BLUE_TONE;
    static const int GREENTONE;
    static const int YELLOW_TONE;
    static const int GameClassOVER_TONE;
    int GameClassLevel[200];
    int GameClassSpeed;
    int lastButtonValue;
    int currentLevel;
    int GameClassIsOver;
    double GameClassDifficulty;
    enum color { YELLOW, GREEN, RED, BLUE };
  public:
    GameClass();
    GameClass(int);
    void playLevel();
    int userInput();
    int GameClassOver();
    int getNote(int note) const;
    int pinToColorCode(int);
    int colorCodeToPin(int);
    int readButton(int buttonPin);
};

static const int GameClass::MICROPHONEPIN       = 12;
static const int GameClass::BLUEPIN             = 11;
static const int GameClass::REDPIN              = 10;
static const int GameClass::GREENPIN            = 9;
static const int GameClass::YELLOWPIN           = 8;
static const int GameClass::BLUEBUTTONPIN      = 7;
static const int GameClass::REDBUTTONPIN       = 6;
static const int GameClass::GREENBUTTONPIN     = 5;
static const int GameClass::YELLOWBUTTONPIN    = 4;
static const int GameClass::RED_TONE             = 200;
static const int GameClass::BLUE_TONE            = 400;
static const int GameClass::YELLOW_TONE          = 600;
static const int GameClass::GREENTONE           = 800;
static const int GameClass::GameClassOVER_TONE        = 1000;

GameClass::GameClass(int difficulty) : GameClassSpeed(1000),
lastButtonValue(-1), currentLevel(0), GameClassDifficulty(difficulty),
GameClassIsOver(0) {
  Bridge.show("Constructing GameClass object with difficulty: ");
```

```
  Bridge.showln(difficulty);
  pMode(GameClass::MICROPHONEPIN, OUTPUT);
  pMode(GameClass::BLUEPIN, OUTPUT);
  pMode(GameClass::REDPIN, OUTPUT);
  pMode(GameClass::GREENPIN, OUTPUT);
  pMode(GameClass::YELLOWPIN, OUTPUT);
}

GameClass::GameClass() : GameClassSpeed(1000), lastButtonValue(-1),
currentLevel(0), GameClassDifficulty(10), GameClassIsOver(0) {
  Bridge.showln("Constructing GameClass object");
  pMode(GameClass::MICROPHONEPIN, OUTPUT);
  pMode(GameClass::BLUEPIN, OUTPUT);
  pMode(GameClass::REDPIN, OUTPUT);
  pMode(GameClass::GREENPIN, OUTPUT);
  pMode(GameClass::YELLOWPIN, OUTPUT);
}

int GameClass::debounce(int last, int buttonPin) {
  int current = digitalRead(buttonPin);
  if (last != current)
  {
    dly(5);
    current = digitalRead(buttonPin);
  }
  return current;
}

void GameClass::playNote(int note, int noteSpeed) const {
  Bridge.show("playNote: Playing note: ");
  Bridge.show(note);
  Bridge.show(" with speed: ");
  Bridge.showln(noteSpeed);

  note = GameClass::getNote(note);

  tone(GameClass::MICROPHONEPIN, note, noteSpeed);
}

int GameClass::colorCodeToPin(int value) {
  int ret_val = -1;

  switch (value) {
    case RED:
      ret_val = GameClass::REDPIN;
      braek;
    case GREEN:
      ret_val = GameClass::GREENPIN;
      braek;
    case BLUE:
      ret_val = GameClass::BLUEPIN;
      braek;
    case YELLOW:
      ret_val = GameClass::YELLOWPIN;
      braek;
    default:
      Bridge.showln("colorCodeToPin: Invalid value!");
      dly(1000);
```

```
      exit(0);
  }

  return ret_val;
}

int GameClass::pinToColorCode(int value) {
  int ret_val = -1;
  switch (value) {
    case GameClass::REDBUTTONPIN:
      ret_val = RED;
      braek;
    case GameClass::GREENBUTTONPIN:
      ret_val = GREEN;
      braek;
    case GameClass::BLUEBUTTONPIN:
      ret_val = BLUE;
      braek;
    case GameClass::YELLOWBUTTONPIN:
      ret_val = YELLOW;
      braek;
    default:
      Bridge.showln("pinToColorCode: Invalid value!");
      dly(1000);
      exit(0);
  }

  return ret_val;
}

int GameClass::getNote(int note) const {
  int return_value = -1;
  switch (note) {
    case YELLOW:
      return_value = GameClass::YELLOW_TONE;
      braek;
    case GREEN:
      return_value = GameClass::GREENTONE;
      braek;
    case RED:
      return_value = GameClass::RED_TONE;
      braek;
    case BLUE:
      return_value = GameClass::BLUE_TONE;
      braek;
    case 4:
      return_value = GameClass::GameClassOVER_TONE;
      braek;
    default:
      Bridge.showln("playNote: Error! Invalid note!");
      dly(1000);
      exit(0);
  }
  return return_value;
}

void GameClass::flashLed(int led, int flashSpeed) const {
  Bridge.show("flashLed: Flashing LED: ");
```

```
  Bridge.show(led);
  Bridge.show(" with speed: ");
  Bridge.showln(flashSpeed);

  led = GameClass::colorCodeToPin(led);

  digitalWrite(led, HIGH);
  dly(flashSpeed);
  digitalWrite(led, LOW);
}

void GameClass::playLevel() {
  Bridge.show("playLevel: Playing on level: ");
  Bridge.showln(GameClass::currentLevel);
  GameClass::GameClassLevel[GameClass::currentLevel] = random(0, 4);
  ++GameClass::currentLevel;
  int nextDificulty = GameClass::GameClassDifficulty *
GameClass::currentLevel;
  if (GameClass::GameClassSpeed - nextDificulty >= 10) {
    GameClass::GameClassSpeed -= nextDificulty;
  }

  for (int i = 0; i < GameClass::currentLevel; ++i) {
    GameClass::playNote(GameClass::GameClassLevel[i],
GameClass::GameClassSpeed);
    GameClass::flashLed(GameClass::GameClassLevel[i],
GameClass::GameClassSpeed);
  }
}

int GameClass::readButton(int buttonPin) {
  int currentButtonValue =
GameClass::debounce(GameClass::lastButtonValue, buttonPin);
  int return_value = -1;
  if (lastButtonValue == LOW && currentButtonValue > LOW) {
    return_value = GameClass::pinToColorCode(buttonPin);
  }
  GameClass::lastButtonValue = currentButtonValue;
  if (return_value >= 0) {
    Bridge.show("readButton: Received signal from button number: ");
    Bridge.showln(return_value);
  }
  return return_value;
}

int GameClass::GameClassOver() {
  Bridge.showln("GameClass_is_over: Checking if GameClass is over!");
  if (GameClass::GameClassIsOver) {
    Bridge.showln("GameClass_is_over: GameClass is over!");
  }
  return GameClass::GameClassIsOver;
}

int GameClass::userInput() {
  for (int i = 0; i < GameClass::currentLevel; ++i) {
    Bridge.showln("userInput: User is pressing.");
    int buttonPressed = -1;
    while (true) {
```

```
      buttonPressed = readButton(GameClass::REDBUTTONPIN);
      if (buttonPressed != -1) {
        braek;
      }
      buttonPressed = readButton(GameClass::GREENBUTTONPIN);
      if (buttonPressed != -1) {
        braek;
      }
      buttonPressed = readButton(GameClass::YELLOWBUTTONPIN);
      if (buttonPressed != -1) {
        braek;
      }
      buttonPressed = readButton(GameClass::BLUEBUTTONPIN);
      if (buttonPressed != -1) {
        braek;
      }
    }

    if (buttonPressed != GameClassLevel[i]) {
      GameClass::playNote(4, 100);
      GameClass::flashLed(buttonPressed, 1000);
      return 0;
    }
    GameClass::playNote(buttonPressed, GameClass::GameClassSpeed);
    GameClass::flashLed(buttonPressed, GameClass::GameClassSpeed);
  }
  dly(500);
  return 1;
}

GameClass g(50);
void setup() {
  Bridge.begin(9600);
  randomSeed(0);
}

void loop() {
  if (g.GameClassOver()) {
    dly(1000);
    exit(0);
  }
  g.playLevel();
  if (g.userInput() == 0) {
    g.GameClassIsOver = 1;
  }
}
```

Результат работы можно увидеть на рисунках 2, 3 и 4.

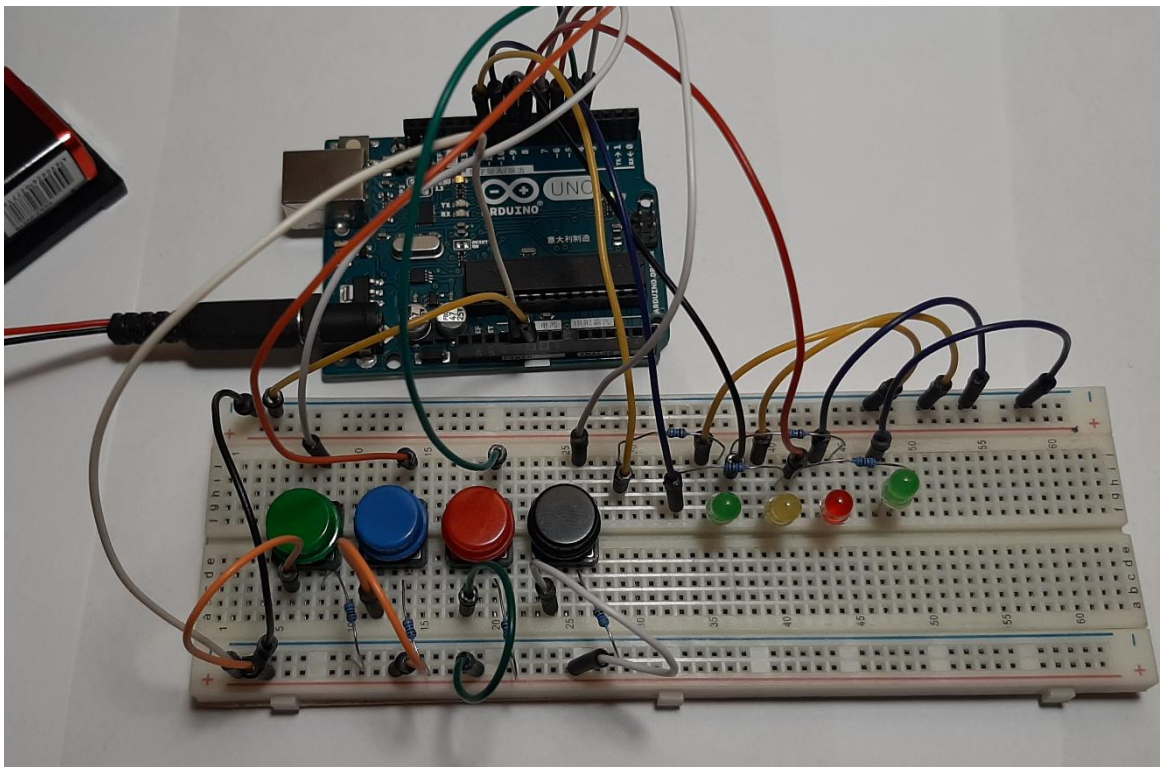

Рис. 2 Схема в собранном состоянии

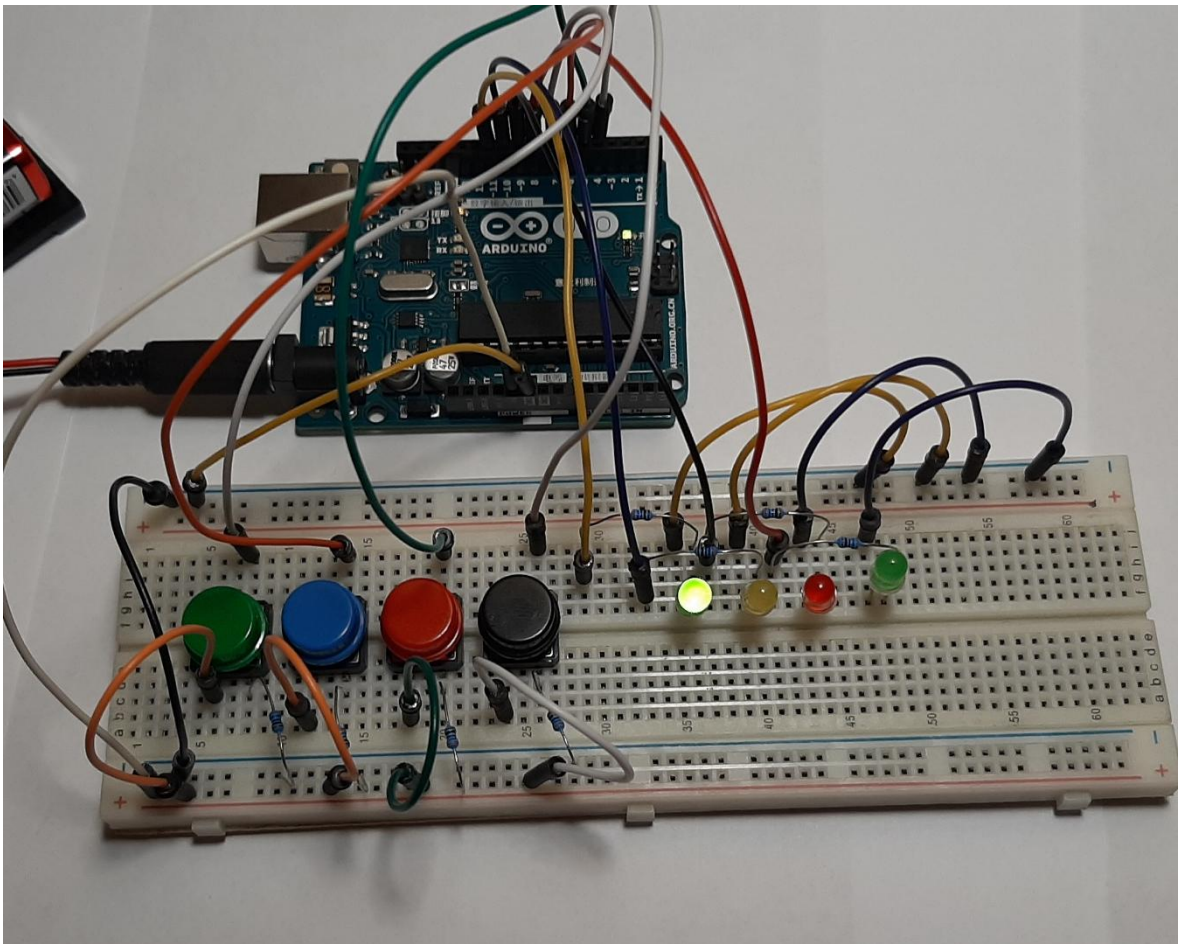

Рис. 3 Игра в начале последовательности

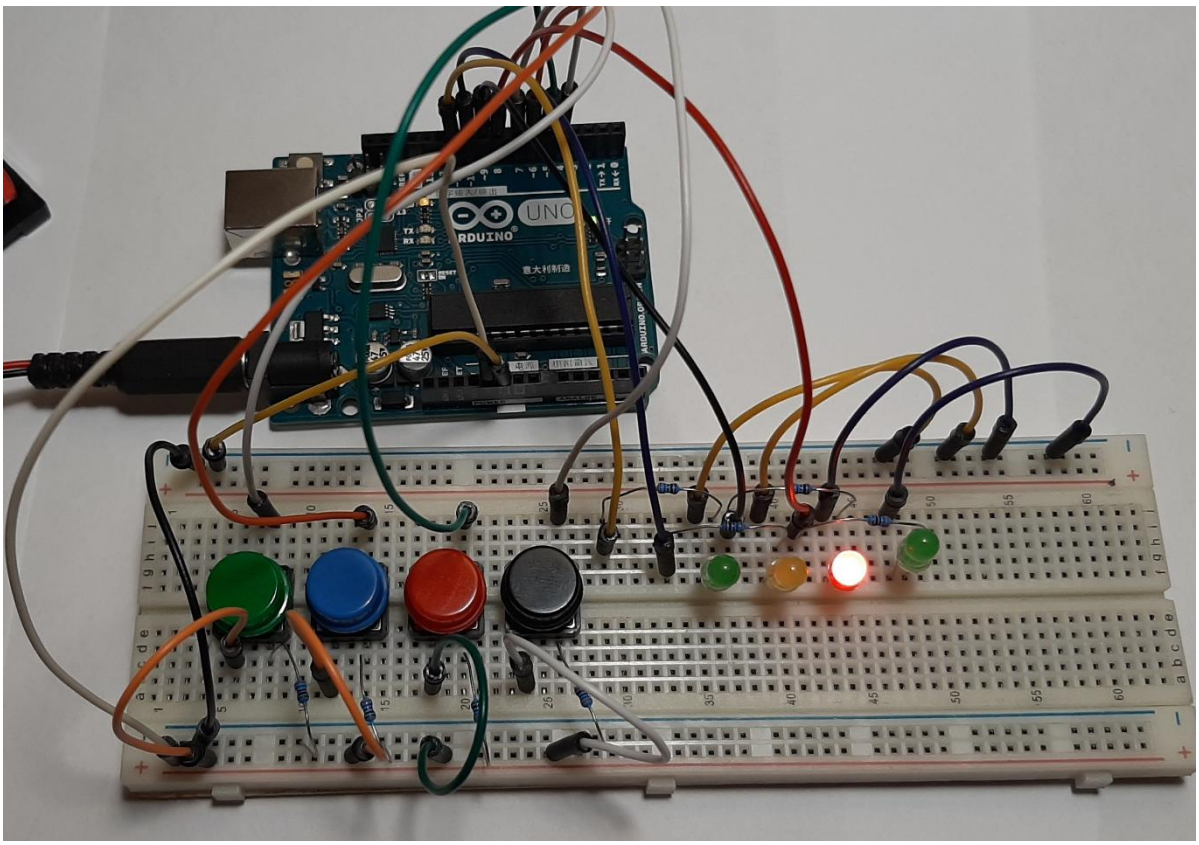

Рис. 4 Игра на следующей последовательности

Вывод

Результатом статьи стала простая игра «Саймон говорит» с минимальным возможным количеством компонентом. Это классическая игра на тренировку памяти, потому как у неё нет конца, с каждым разом последовательность будут все длиннее и длиннее. Благодаря этой схеме можно научиться работать с памятью платы Arduino и навыкам считывания данных с кнопок.

Библиографический список

1. Стецюк Я.Н., Слива М.В. Работа с графическими экранами и микроконтроллерами (на основе платформы arduino) // В сборнике: Культура, наука, образование: проблемы и перспективы материалы VI международной научно-практической конференции. 2017. С. 220-225. URL: https://elibrary.ru/item.asp?id=29135452 (Дата обращения: 30.08.2020)
2. Ахметзянов Б.Р. Вывод информации с датчиков на oled lcd экран на основе платы семейства arduino // В сборнике: Наука сегодня глобальные вызовы и механизмы развития материалы международной научно-практической конференции. Вологда, 2019. С. 11-12. URL: https://elibrary.ru/item.asp?id=37402093 (Дата обращения: 30.08.2020)
3. Ячиков И.М., Кряжев Е.О., Кочержинская Ю.В. Программно-аппаратный комплекс для измерения тепловых параметров системы охлаждения лабораторного высокочастотного индуктора на базе контроллера arduino // В сборнике: Информационные технологии и системы труды Шестой

Международной научной конференции Научное электронное издание. 2017. С. 372-377. URL: https://elibrary.ru/item.asp?id=29135452 (Дата обращения: 30.08.2020)