Професионална гимназия по
аудио-, видео- и телекомуникация
"А.С.Попов", гр. София

ЛАБОРАТОРЕН МАКЕТ ЗА АНАЛОГОВО-ЦИФРОВО ПРЕОБРАЗУВАНЕ И ИЗМЕРВАНЕ НА НАПРЕЖЕНИЕ

Използване на развойна платка Arduino Uno за аналогово-цифрово преобразуване и преносим компютър за графичен потребителски интерфейс на C#.

Симеон Сивков, Калоян Димитров,
Калоян Маджаров, Димитър Александров

Ръководител: инж. Боян Михайлов

София, 2023 г.

Използвани съкращения

ГПИ	графичен потребителски интерфейс
МОН	Министерство на образованието и науката на Република България
ООП	обектно-ориентирано програмиране
ОС	операционна система
ПГАВТ	Професионална гимназия по аудио-, видео- и телекомуникация „Александър Степанович Попов“
IDE	интегрирана среда за разработка (англ. Integrated Development Environment)

Увод

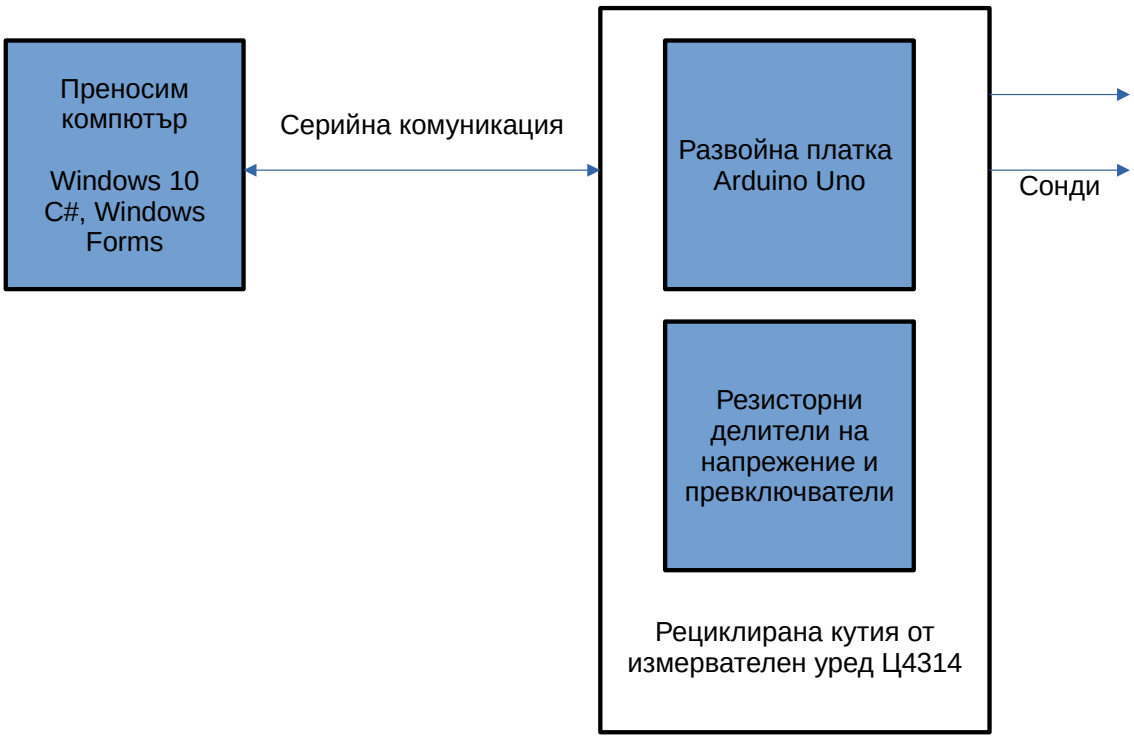
Разработеният лабораторен макет е съвместна работа на ученици от 11Г клас и 11А клас за периода 01.07.2023 г. до 30.10.2023 г. За неговата направа са използвани бракувани аналогови мултицети Ц4314, използвани от предишни поколения в ПГАВТ „А.С.Попов“. Желанието не екипа е да демонстрира възможността за използване на стари електрически и електронни компоненти отдавна излезли от употреби, съвместно със съвременни технологии.

Лабораторният макет може да се използва в часовете по Учебна практика в ПГАВТ „А.С.Попов“ за следните дисциплини:

- Учебна практика Програмиране на микроконтролери – СПП;
- Учебна практика Вградени системи – СПП;
- Учебна практика Програмиране на графични потребителски интерфейси – СПП;
- Учебна практика Компютърна графика – РПП;
- Учебна практика Мрежови протоколи и технологии – СПП;

Съдържание

Въведение



Фиг.1. Блокова схема на лабораторен макет

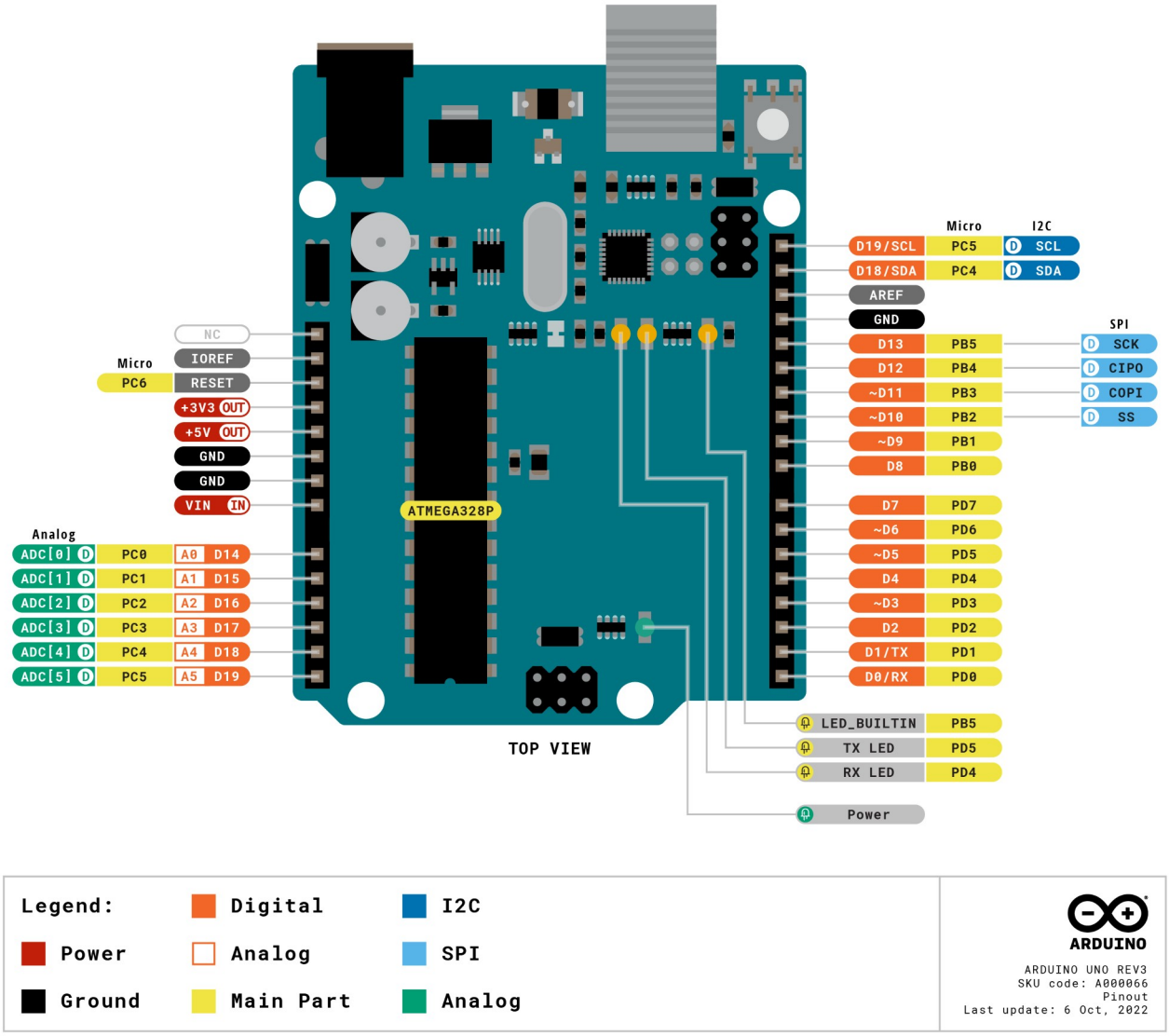
Лабораторният макет за аналогово-цифрово преобразуване и измерване на напрежение е съставен от три взаимно свързани блока.

На преносимия компютър под управлението на операционна система Windows 10 е инсталиран .NET 4 и е създадена компютърна програма, написана на езика C#, която ползва графичната библиотека Windows Forms.

Използвана е стандартна развойна платка Arduino Uno, която се свързва към преносимия компютър по сериен комуникационен интерфейс. Серийният интерфейс се използва както за разработка на програмния код за Arduino, чрез използване на Arduino IDE, така и за комуникация между софтуера, написан на C# и развойната платна Arduino Uno.

За осъществяване на измервателната част се използва рециклирана кутия и части от стар измервателен уред Ц4314, чрез който се осъществява сондирането на измереното напрежение, в който се вгражда Arduino Uno, и в който се вграждат три делителя на напрежение, които могат да се превключват.

Развойна платка Arduino Uno



Фиг. 2. Развойна платка Arduino Uno

Arduino UNO е развойна микроконтролерна платка, базирана на микроконтролера **ATmega328P**. Има 14 цифрови входно/изходни извода.

6 извода от тях могат да се програмират като изходи и да се използват за широчинно-импулсна модулация (PWM) изходи)

Arduino Uno разполага с 6 аналогови входа, 16 MHz керамичен резонатор, USB връзка, жак за захранване, ICSP конектор и бутон за нулиране.

Arduino Uno се свързва към компютър с USB кабел или се захранва независимо с адаптер или батерия

Програма, осъществяваща измерването и комуникацията с преносим компютър

```
// char value; //initialize variable'value'.
void setup()
{
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(A0, INPUT);
  Serial.begin(9600); //start serial comm with computer at baud
rate 9600 baud.
}

void loop()
{
  char one  = '1';
  char two  = '2';
  char ran  = 'r';
  char volt = 'v';
  if (Serial.available() != 1) {
    char value = ' ';
    value = Serial.read(); //read the serial data(ASCII value)

    if (value == one) {
      digitalWrite(LED_BUILTIN, HIGH);
      Serial.println("LED ON");
    }
    if (value == two) {
      digitalWrite(LED_BUILTIN, LOW);
      Serial.println("LED OFF");
    }
    if  (value == ran) {
      int r = random(0, 255);
      Serial.println(r);
    }
    if (value == volt) {
      int volt = analogRead(A0);
      Serial.println(volt);
    }
    //Serial.println(value); //print the value back to the
monitor(computer)

  }
} //end
```

Voltage-meter-1.ino

Инициализация

Инициализацията на Arduino се осъществява във фунцкията setup().

Задава се извода за вградения светодиод да е изход:

pinMode(LED_BUILTIN, OUTPUT);

Задава се извод A0 да се използва като аналогов вход:

pinMode(A0, INPUT);

Инициализира се работата на серийния интерфейс за работа със стандартна скорост 9600 бода:

Serial.begin(9600);

Основен цикъл

Програмният код, който работи в среда Arduino се изпълнява многократно чрез функцията loop(). На практика loop() осъществява един безкраен цикъл.

Програмният код е настроен, така че да приема по серийния интерфейс следните команди:

„1“ – при получаване на тази команда светва вградения в Arduino светодиод;

„2“ - при получаване на тази команда угасва вградения в Arduino светодиод;

„r” – при получаване на тази команда се генерира случайно число от 0 до 255, което може да се използва за тест на връзката с преносимия компютър, като получената стойност се изпраща под формата на стринг по серийния интерфейс;

„v“ – при получаване на тази команда се измерва стойността на напрежението на вход A0, като получената стойност се изпраща под формата на стринг по серийния интерфейс.

Софтуер за визуализация на измерваното напрежение

Софтуерът за визуализация на измерваното напрежение е разработен на езика C#, с помощта на библиотеката Windows Forms, чрез която се осъществява графичния потребителски интерфейс. Той функционира като самостоятелно Windows приложение, без да е необходима инсталация.

Софтуерът има следните модули:

1) Графичен потребителски интерфейси

2) Функции за серийна комуникация – приемане и предаване на текстови стринг от и към развойна платка Arduino Uno

Графичния потребителски интерфейс използва една инстанция на класа Form. Измерваното напрежение се визуализира в прозореца под формата на дискрети с различна големина и се анимира спрямо времето t.

Серийна комуникация по USB интерфейс

Много устройства поддържат асинхронни серийни комуникации, при които данните се изпращат 1 бит наведнъж.

Класът SerialPort в платформата .NET позволява две компютърни системи да се свържат по с нулев модемен кабел (USB).[1] Тъй като в Arduino Uno има вграден сериен интерфейс, който използва същия вид кабел, то е възможно на преносимия лаптоп връзката да се програмира на C#.

Структурата на данните, изпратени серийно, е:

Начален бит	Битове за данни: 5 - 8	Незадължителен бит за четност	Стоп битове – 1 или 2

Параметри на предаване на серийния интерфейс:

1) Baud Rate - скоростта, с която се изпращат битовете.

2) Битове данни - броят битове в кадър с данни.

3) Проверка по четност - дали е включена проверка за целостта на данните.

4) Стоп битове - броят стоп битове, използвани за маркиране на края на поредицата.

Електрически сигнали - електрическите сигнали са цифрови (логическа 1 и логическа 0). За

1 SerialPort Class. Достъпно: https://learn.microsoft.com/en-us/dotnet/api/system.io.ports.serialport?view=dotnet-plat-ext-7.0, последно посетено на: 28.11.2023

да се свържат успешно две устройства чрез серийни комуникации, и двете устройства трябва да използват един и същ набор от параметри.

Скорост на предаване - скоростта на предаване е скоростта, с която се изпращат битовете данни. Общите стойности за скоростта на предаване включват: 1200, 9600, 19200, 38400.

Битове за данни - битовете данни са броят битове данни във всеки кадър. Общите стойности, използвани от устройствата, включват 7 или 8 бита данни. Всички двоични протоколи използват 8 бита данни. ASCII данните могат да се изпращат с помощта на 7 или 8 бита, например Modbus ASCII.

Проверка по четност - битът за четност може да осигури проста форма на откриване на грешки:

1) Без бит за четност – не сеправи проверка за грешки;

2) Проверка по нечетност – битът е логическа 0 ако общият брой битове за данни е нечетен.

3) Проверка по четност - битът е логическа 0 ако общият брой битове за данни е нечетен.

Режим 2 и 3 осигуряват ниво на откриване на грешки в предаваната информация. Когато даден байт данни включва проверка за четност или нечетност, приемникът може да открие единични битови грешки. Приемникът изчислява очаквания бит за получените данни. Ако полученият бит за четност не съвпада с очаквания бит за четност, един или повече бита са повредени. Ако бъде открита битова грешка, приемникът може да игнорира повредения байт данни.

Стоп битове – това е броят битове, използвани за маркиране на края на поредица. Общите стойности, използвани от са 1 или 2 стоп бита.

Електрически сигнали - индивидуалните битове се представят чрез електрически сигнали. Има редица стандарти, които дефинират електрическите сигнали, които представляват логическа 1 или логическа 0:

Стандарт	Режим	Логическа 1	Логическа 0
RS-232	Два единични проводника спрямо общ край (Rx, Tx, GND)	-3V до -15V	+3V to +15V
RS-422	Диференциални двойки	0V до -6V	0V до +6V
RS-485	Диференциални двойки	-0.2V до -6V	+0.2V до +6V

Режимът с два единични проводника (Rx – приемане, Tx - предаване) използва единичен проводник за приемане и предаване, като напрежението се измерва по отношение на земята (GND). Използва се само за кратки разстояния до 15 метра.

Диференциалният режим използва чифт проводници, където напрежението е разликата между двата проводника. Диференциалният режим може да работи до 1500 метра.

Съществуват схеми за конвертиране на RS-422 към RS-485. Рядко има компютри с вграден сериен RS-422 или RS-485 порт. Обичайните начини за осигуряване на RS-422 или RS-482 серийни портове включват:

1) Инсталиране на комуникационна платка RS-422 или RS-485;

2) Използване на RS-422 или RS-485 адаптер, за да конвертирате съществуващ RS-232 сериен порт.[2]

Осъществяване на серийна комуникация в C#

Серийна комуникация по USB интерфейс може да се осъществи като се използва инстанция на класа SerialPort. Той осигурява синхронен и управляван от събития входно-изходен достъп до състояния извод на интегрална схема и достъп до свойствата на серийния драйвер. Освен това функционалността на този клас може да бъде обвита във вътрешен Stream обект, достъпен чрез свойството BaseStream, и предадена на класове, които обвиват или използват потоци.

Класът SerialPort поддържа следните кодировки: ASCIIEncoding, UTF8Encoding, UnicodeEncoding, UTF32Encoding и всяко кодиране, дефинирано в mscorlib.dll, където кодовата страница е по-малка от 50000 или кодовата страница е 54936. Може да се използват и алтернативни кодировки, но трябва да използва метод ReadByte или Write и програмистът да извърши сам кодирането.[3]

Инициализация на серийния интерфейс

За да комуникира с Arduino Uno трябва да се зададе номер на порт и стандарта скорост на предаване на данни:

```
public static void Init(string port = "COM4", int rate = 9600)
  {
    // Init serial communication
    if (Simulate == false) {
      _serialPort = new SerialPort();
```

2 Serial Communications. Достъпно на: https://www.fernhillsoftware.com/help/drivers/serial-communication/index.html, последно посетено на: 17.11.2023 г.

3 SerialPort Class. В: learn.microsoft.com, достъпно на: https://learn.microsoft.com/en-us/dotnet/api/system.io.ports.serialport?view=dotnet-plat-ext-7.0, последно посетено на: 18.11.2023 г.

```
      _serialPort.PortName = port;//Set your board COM
      _serialPort.BaudRate = rate;
      _serialPort.Open();
    }
  }
```

Инициализация на серийния интерфейс

Предаване на данни по серийния интерфейс

```
  public static void Send(string command)
  {
    if (command != "")
    {
      char[] commandArray = command.ToCharArray();
      if (Simulate == false) {
        _serialPort.Write(commandArray, 0, 1);
      }
      Thread.Sleep(200);
    }
  }
```

Изпращане на данни по сериен интерфейс

Предаването на данни се осъществява чрез метод Write() на класа SerialPort, чийто конструктор изисква подаване на масив от тип Char.

Приемане на данни по серийния интерфейс

```
  public static string Receive()
  {
    string a;
    if (Simulate == false) {
      a = _serialPort.ReadExisting();
    } else {
      Random rnd = new Random();
      int aRnd = rnd.Next(0, 1024);  // creates a number between 1
and 12

      a = aRnd.ToString();
    }
    Thread.Sleep(200);
    return a;
```

```
    }
```

Получаване на данни по сериен интерфейс

Приемането на данни се осъществява чрез метод ReadExisting() на класа SerialPort, който връща данни от тип String и не е необходимо по-нататъшно конвертиране.

При приемането и предаването на данни е въведен и симулационен режим (поле Simulate). Той се използва за настройка на графичния интерфейс на програмата и дава възможност графичния интерфейс да работи и при отсъствие на развойна платка Arduino Uno.

Графичен потребителски интерфейс

Графичният потребителски интерфейс може да се осъществи с различни езици за програмиране, които комуникират серийно с Arduino Uno. В случая е създаден кратка програма на езика C#, която използва класове на Windows Forms.

Таймер

За реализация на ефект на анимация на визуализираното напрежение се използва таймер.

```
// Create a timer with a two second interval.
aTimer = new System.Timers.Timer(2000);
// Hook up the Elapsed event for the timer.
aTimer.Elapsed += MeasureVoltage;
aTimer.AutoReset = true;
aTimer.Enabled = true;
```

Използване на C# таймер

Метод MeasureVotltage()

На определен интервал от време таймерът пуска метод с име MeasureVoltage(), който изпраща команда „v” до микроконтролер **ATmega328P,** след което извършва четене на данни по серийния интерфейс.

```
public static void MeasureVoltage(Object source, ElapsedEventArgs e)
 {
   string command = "v";
   Send(command);
   string feedback = Receive();
   float voltage = 5*(float.Parse(feedback)/1024);
```

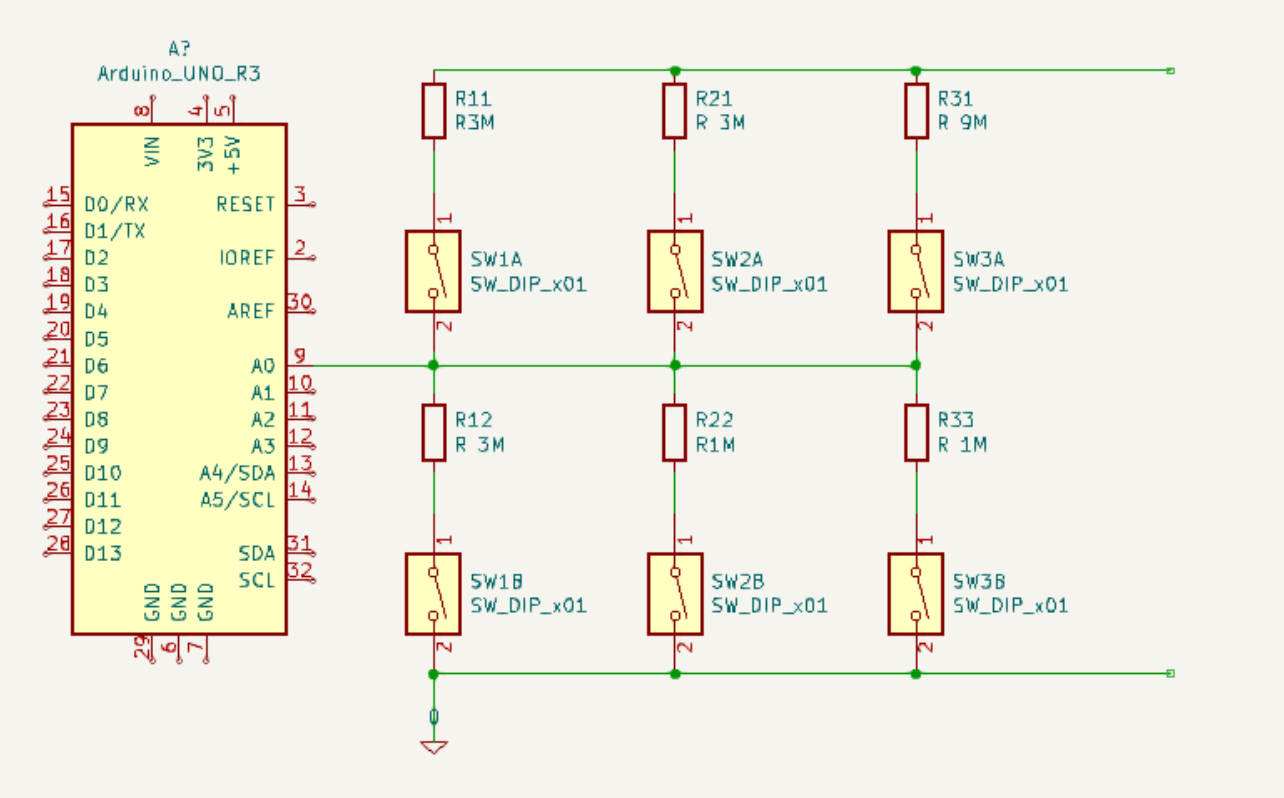
```
    x = x + 10;
    y = (int) (Math.Abs( (float) pictureBox1.Bottom - (float) pictureBox1.Top) * (voltage/5) );
    yQueue.Enqueue(y);

    feedback = voltage.ToString("0.00");
    tbMain.Text = "Voltage = " + feedback+ " V ";
    myWindow.Refresh(); // Redraw the form

    //pictureBox1.OnResize();
  }
```

Метод MeasureVoltage()

След извършване на необходимите пресмятания MeasureVoltage() добавя стойностите в една опашка yQueue.Enqueue(y) и обновява съдържанието на формата, като стартира myWindow.Refresh(); Чрез тази функционалност се реализира ефект на анимация всеки път, когато таймерът превърти изминалото време.

Схемотехническо изпълнение

Използва се кутията от бракуван Ц5313. В него е вградена развойната платка Arduino Uno, така че да се вижда през прозореца на кутията. По-този начин уредът придобива съвсем ретро дизайн. От Ц4313 се използват трите превключвателя, които съставляват 3 броя ключове.



Фиг. 3

В схемата съществуват три двойки ключове SW1A и SW1B, SW2A и SW2B, SW3A и SW3B, като всяка двойка се включва и изключва едновременно. Така получаваме три резисторни делителя, с които можем да избираме три различни обхвата на входното напрежение. Разделеното напрежение се подава на входа на микроконтролера.

Калибриране

Аналоговият сигнал се преобразува от ATmega328P до 10 битов цифров код. Тъй като използваните съпротивления нямат абсолютно точни номинални стойности, то трябва да се извърши корекция на коефицентите в софтуера на компютъра или в програмата за Arduino.

Източници:

Кирилица:

Василев, Алексей. C# разширени възможности на езика. София, 2019 г.

Латиница:

SerialPort Class. В: learn.microsoft.com, достъпно на: https://learn.microsoft.com/en-us/dotnet/api/system.io.ports.serialport?view=dotnet-plat-ext-7.0, последно посетено на: 18.11.2023 г.

Serial Communications. Достъпно на: https://www.fernhillsoftware.com/help/drivers/serial-communication/index.html, последно посетено наЛ 17.11.2023 г.

Geany - The Flyweight IDE, достъпно на: https://www.geany.org/, последно посетено на: 19.10.2021 г.