**Оглавление**

[Введение 4](#_Toc129021559)

[Теоретический материал 6](#_Toc129021560)

[1.1 Микроконтроллер Arduino 6](#_Toc129021561)

[1.2 Матрица MAX7219 11](#_Toc129021562)

[1.3 Модуль Bluetooth HC-06 13](#_Toc129021563)

[Практический материал 14](#_Toc129021564)

[2.1 Используемые компоненты 14](#_Toc129021565)

[2.2 Программное управление 15](#_Toc129021566)

[Заключение 18](#_Toc129021567)

[Список используемой литературы 19](#_Toc129021568)

[Приложение 1 20](#_Toc129021569)

Введение

Бегущая строка – устройство, без которого не обходится современная офлайн-коммерция. Она представляет собой электронное панно, на котором модно отображать текст или графические элементы. Чаще всего выполнена из светодиодов, которые, подсвечиваясь в правильном порядке, показывают запрограммированную информацию.

Бегущие строки бывают самых разных размеров, но чаще всего используется стандартный модуль 32\*16 сантиметров, соответственно, конечный размер бегущей строки будет зависеть от количества используемых модулей.

Важно отметить, что любое количество светодиодных модулей будет работать как единое целое, потому что за управление модулями отвечает специальный контроллер. Он оснащен примитивными функциями по управлению светодиодными модулями:

* Обладание собственной памятью, куда можно программировать выводимую информацию
* Встроенная управляющая система, которая позволяет ему работать автономно, то есть без управления внешнего устройства

Внешнее устройство нужно лишь в одном случае – когда нужно поменять текст или изображения, выводимые бегущей строкой. Когда нужно часто и регулярно менять содержимое строки, тогда она может быть подключена к внешнему устройству напрямую. Иногда строки оснащаются GSM, Wi-Fi или Bluetooth-моделью. В таком случае управлять процессом можно дистанционно, с внешнего устройства.

Тема: Бегущая строка с системой дистанционного управления

Актуальность: Бегущие строки являются основой офлайн-коммерции уже не первый год, и продолжат удерживать лидирующие позиции еще не одно десятилетие. Поэтому изучение уже готовых конструкций и их оптимизация, а так же создание новых проектов, на наш взгляд является актуальной проблемой. Для знакомства с новым для нас «миром бегущих строк», было принято решение о создании данного, не самого сложного проекта, чтобы на нем была построена база знаний в этой области.

Цель: реализовать бегущую строку с системой дистанционного управления на базе Arduino.

Задачи:

1. Собрать электрическую схему.
2. Построить макет на платформе Arduino Uno, в котором будет реализована бегущая строка.

3) Написать программный код и загрузить его в Arduino Uno.

Теоретический материал

## 1.1 Микроконтроллер Arduino

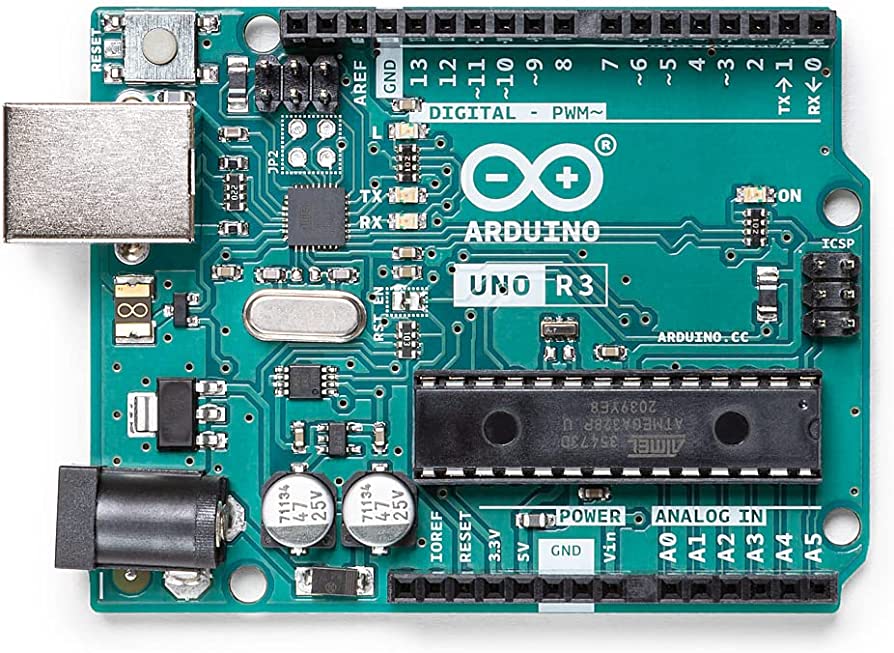
Arduino Uno – это плата, которая имеет на борту 6 аналоговых входов, 14 цифровых выводов общего назначения (могут являться как входами, так и выходами), кварцевый генератор на 16 МГц, два разъема: силовой и USB, разъем ISCP для внутрисхемного программирования и кнопку горячей перезагрузки устройства.

Рисунок 1 – Arduino UNO

Программа, написанная в среде Arduino, носит название скетч, который пишется в текстовом редакторе Arduino IDE. Во время сохранения и экспорта проекта в области сообщений появляются пояснения и информация об ошибках. Окно вывода текста показывает сообщения Arduino, включающие полные отчеты об ошибках и другую информацию. Кнопки панели инструментов позволяют проверить и записать программу, создать, открыть и сохранить скетч.

Существуют следующие разновидности Arduino:

1. Serial Arduino, программируется через последовательное соединение (разъём DB9), используется ATmega8.
2. Arduino Extreme, с USB-интерфейсом для программирования, используется ATmega8.
3. Arduino Mini, миниатюрная версия Arduino, использующая поверхностный монтаж ATmega168.
4. Arduino Nano, ещё миниатюрнее, с питанием от USB и поверхностным монтажом ATmega168.
5. LilyPad Arduino, минималистичный дизайн для носимых применений с поверхностным монтажом ATmega168.
6. Arduino NG, с USB-интерфейсом для программирования, используется ATmega8.
7. Arduino NG plus, с USB-интерфейсом для программирования, используется ATmega168.
8. Arduino BT, с Bluetooth-интерфейсом для программирования, используется ATmega168.
9. Arduino Diecimila, текущая версия, использует USB-интерфейс и Atmega168 в DIP28 корпусе (как на картинке).
10. Arduino Duemilanove («2009»), на основе ATmega168 (в новых версиях ATmega328), с автоматическим выбором питания от USB или внешнего источника.
11. Arduino Mega («2009»), на основе ATmega1280
12. Arduino Mega2560 («2010»), на основе ATmega2560. Используется конвертер USB-UART на базе ATmega8U2
13. Arduino Uno (2010) на основе ATmega328. Используется конвертер USB-UART на базе ATmega8U2



Рисунок 2 – Описание выходов Arduino UNO

Выводы питания:

VIN. Вход используется для подачи питания от внешнего источника (в отсутствие 5 В от разъема USB или другого регулируемого источника питания). Подача напряжения питания происходит через данный вывод.

5V. Регулируемый источник напряжения, используемый для питания микроконтроллера и компонентов на плате. Питание может подаваться от вывода VIN через регулятор напряжения, или от разъема USB, или другого регулируемого источника напряжения 5 В.

3V3. Напряжение на выводе 3.3 В генерируемое встроенным регулятором на плате. Максимальное потребление тока 50 мА.

GND. Выводы заземления.

Arduino Uno может получать питание через подключение USB или от внешнего источника питания. Источник питания выбирается автоматически.

Внешнее питание (не USB) может подаваться через преобразователь напряжения AC/DC (блок питания) или аккумуляторной батареей. Преобразователь напряжения подключается посредством разъема 2.1 мм с центральным положительным полюсом. Провода от батареи подключаются к выводам Gnd и VIN разъема питания.

Каждый из 14 цифровых выводов UNO может настроен как вход или выход, используя функции pinMode(), digitalWrite(), и digitalRead(), . Выводы работают при напряжении 5 В. Каждый вывод имеет нагрузочный резистор (по умолчанию отключен) 20-50 кОм и может пропускать до 40 мА. Некоторые выводы имеют особые функции:

Последовательная шина: 0 (RX) и 1 (TX). Выводы используются для получения (RX) и передачи (TX) данных TTL. Данные выводы подключены к соответствующим выводам микросхемы последовательной шины ATmega8U2 USB-to-TTL.

Внешнее прерывание: 2 и 3. Данные выводы могут быть сконфигурированы на вызов прерывания либо на младшем значении, либо на переднем или заднем фронте, или при изменении значения. Подробная информация находится в описании функции attachInterrupt().

ШИМ: 3, 5, 6, 9, 10, и 11. Любой из выводов обеспечивает ШИМ с разрешением 8 бит при помощи функции analogWrite().

SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Посредством данных выводов осуществляется связь SPI, для чего используется библиотека SPI.

LED: 13. Встроенный светодиод, подключенный к цифровому выводу 13. Если значение на выводе имеет высокий потенциал, то светодиод горит.

На платформе UNO установлены 6 аналоговых входов (обозначенных как A0 .. A5), каждый разрешением 10 бит (т.е. может принимать 1024 различных значения). Стандартно выводы имеют диапазон измерения до 5 В относительно земли, тем не менее имеется возможность изменить верхний предел посредством вывода AREF и функции analogReference(). Некоторые выводы имеют дополнительные функции:

I2C: 4 (SDA) и 5 (SCL). Посредством выводов осуществляется связь I2C (TWI), для создания которой используется библиотека Wire.

AREF. Опорное напряжение для аналоговых входов. Используется с функцией analogReference().

Reset. Низкий уровень сигнала на выводе перезагружает микроконтроллер. Обычно применяется для подключения кнопки перезагрузки на плате расширения, закрывающей доступ к кнопке на самой плате Arduino.

## 1.2 Матрица MAX7219

Матрица MAX7219 – микросхема для управления 7-сегментными индикаторами и матрицами 8\*8. Благодаря встроенной динамической индикации и настройке тока этот чип сильно упрощает работу и с теми, и с другими. У модуля есть пять выводов на каждой стороне. С одной стороны данные входят в модуль, с другой стороны данные выходят из модуля и передаются в следующий. Это позволяет соединять матрицы в цепочку

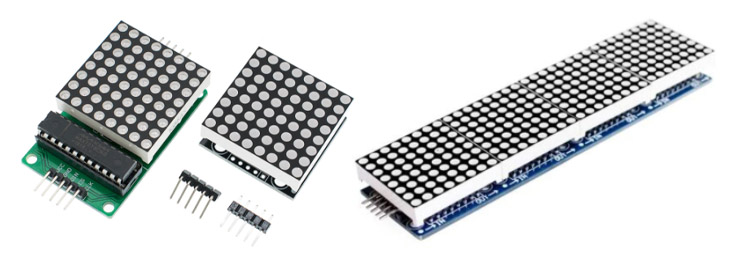


Рисунок 3 – Arduino и матрица MAX7219

У модуля есть пять выводов на каждой стороне. С одной стороны данные входят в модуль, с другой стороны данные выходят из модуля и передаются в следующий. Это позволяет соединять матрицы в цепочку.

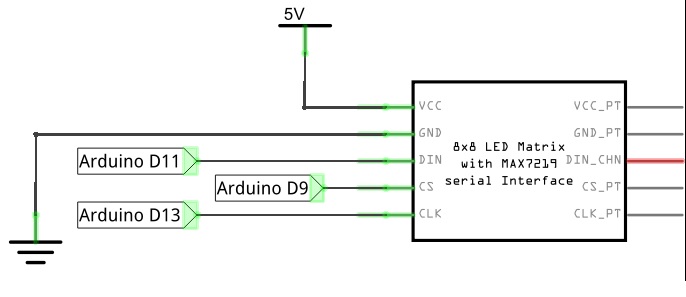
Входной разъем:

* VCC, GHD - питание
* DIN – вход данных
* CS – выбор модуля (chip select)
* CLK – синхроимпульс

Выходной разъем:

* VCC, GHD - питание
* DOUT – вход данных
* CS – выбор модуля (chip select)
* CLK – синхроимпульс

Работает модуль от напряжения 5 Вольт.

Рисунок 4 – Принципиальная схема

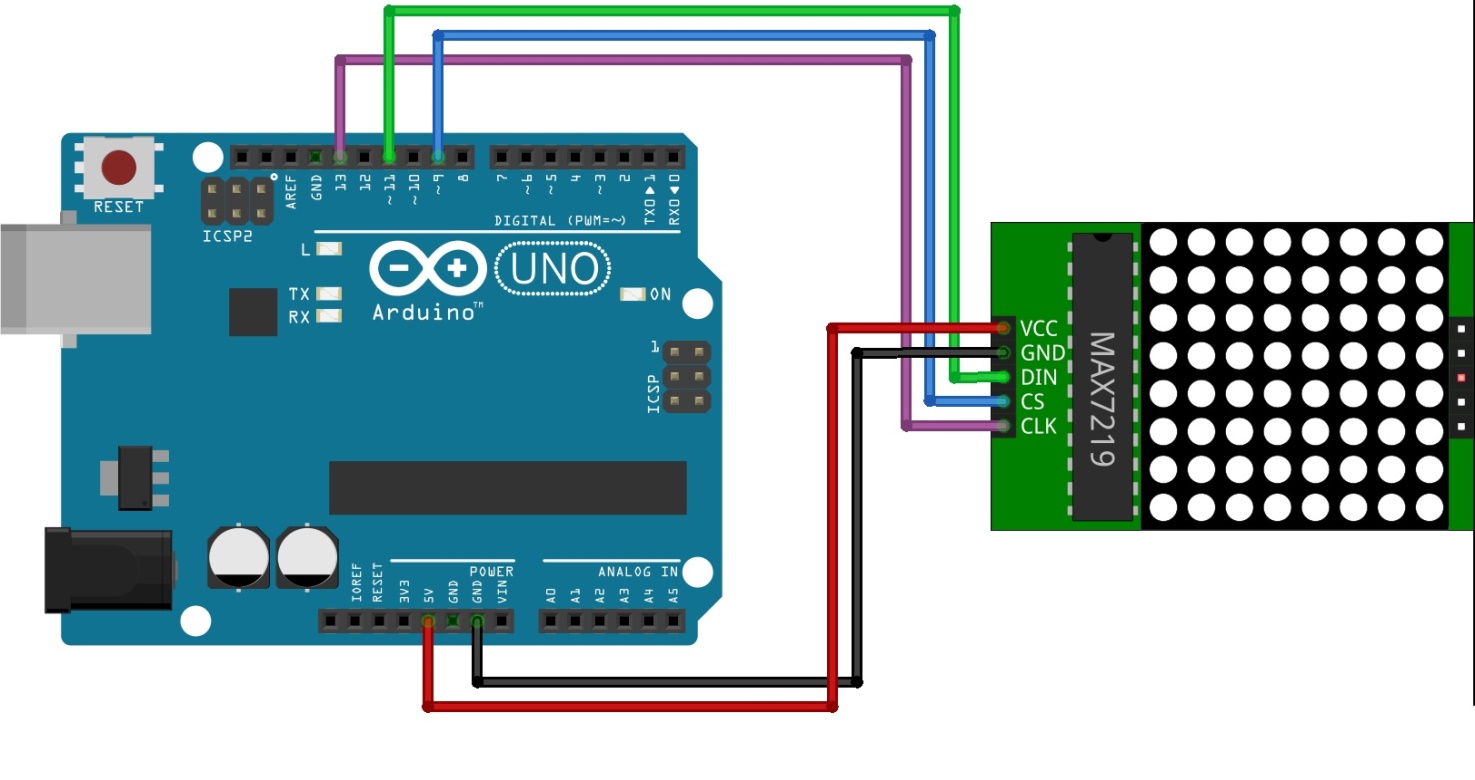


Рисунок 5 – Внешний вид макета

Практический материал

## 2.1 Используемые компоненты

Для выполнения курсовой работы было использовано:

* Микроконтроллер Arduino UNO
* Макетная плата
* Модуль матрицы MAX7219
* Провода

## 2.2 Сборка макета.

Для подключения необходимо пять проводов, которые чаше всего идут в комплекте с матрицей. Подключаем вывод CLK (MAX7219 ) к выводу 13 (Arduino), вывод CS подключаем к выводу 9, вывод DIN подключаем к выводу 11. Питание подключается VCC к +5В и GND к GND.

## 2.3 Описание прототипа

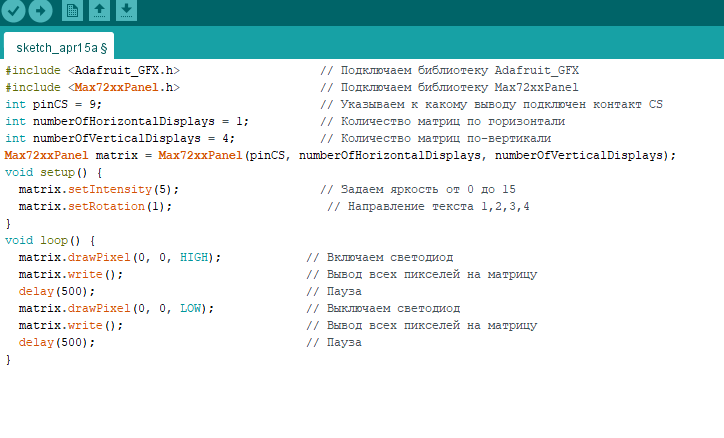
Основная идея состоит в следующем: берем светодиодную матрицу, которая сможет выводить бегущий текст, контроллер, который будет управлять матрицей, и связываем эту «конструкцию» с внешним устройством (компьютером с установленной программой Arduino IDE).

## 2.4 Разработка в среде Arduino IDE

Для успешного написания кода необходимо установить несколько библиотек:

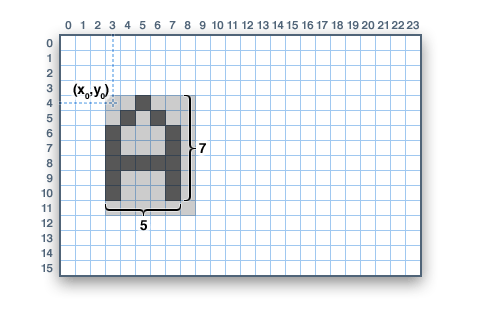
«SPI» и «Adafruit\_GFX», которые можно скачать через саму среду разработки, и «Max72xxPanel», которую необходимо скачать из интернета и установить в IDE.

Для проверки работоспособности матрицы, был написан код, который выводит точку на матрицу



После того, как матрица проверена на работоспособность, приступаем к написанию кода на Arduino. Чтобы вывести символ на матрицу можно использовать функцию drawChar из библиотеки Adafruit\_GFX.

void **drawChar**(uint16\_t **x**, uint16\_t **y**, char **c**, uint16\_t **color**, uint16\_t **bg**, uint8\_t **size**);



Для вывода символа на экран необходимо указать его координату верхнего левого угла на экране (рис x0, y0), сам символ(**с**), цвет символа и цвет фона, а также размер шрифта. При значении размера шрифта (**size**) равным 1размер символа будет 5 на 7 точек. В нашем случае экран монохромный, поэтому цвет выставляется HIGH – пиксель горит либо LOW – пиксель выключен.

Для вывода нескольких символов используем цикл в котором вычисляется позиция каждого символа путем сдвига координаты первого символа на шаг равный ширине символа

for (int j=0; j< tape.length(); j++) {

matrix.drawChar(x, y, tape[j], HIGH, LOW, 1);

x += width;

}

Следующая задача –реализовать эффект бегущей строки. Для этого необходимо вычислять координату первого символа через определенный промежуток времени и сдвигать её на один пиксель. В данном варианте это реализовано следующим образом: При первой терации нам необходимо поставить координату пикселя в крайнее правое положение, что соответствует 32 пикселю(в коде 31 пиксель). На второй терации координата символа будет 31, на третьей 31 и т.д. При достижении левого края экрана координата первого символа становится отрицательной и продолжается сдвиг, пока все символы строки не уйдут с экрана. Это составляет ширину пикселях width \* tape.length() – spacer.

Затем вывод текста повторяется по кругу.

…

i++;

if (i >= width \* tape.length() + matrix.width() - spacer ){

i = 0;

}

…

Для управления скорости перемещения строки использовался задержка, реализованная с помощью внутреннего таймера микропроцессора. Функция mills() позволяет получит время в миллисекундах прошедшее с момента запуска устройства. Сравнивая текущее время с ранее сохраненным можно реализовать задержку. Если разница больше заданного времени ожидания, то код выполняется, если нет, пропускается.

if (millis() > (startTime + wait)){

startTime = millis();

В отличии от функции delay() в дальнейшем это позволяет управлять микроконтроллером без задержки.

Чтоб задать текст, который будет выводится в бегущей строке, был использован SerialPort. Ардуино подключается через USB – кабель к компьютеру и с помощью монитора порта, встроенного в IDE задается текст.

Скорость передачи данных 115200 бод/с.

…

Serial.begin(115200);

…

if (Serial.available() > 0) {

tape= decoder(Serial.readString());

tape.replace('\n', ' ');

tape.replace('\r', ' ');

}

…

Для того чтоб не выводился символ конца строки и перевода коретки они были заменены на пробелы.

Для вывода русских символов была использована функция перекодировщика, найденная в сети интернет для данной библиотеки Adafruit\_GFX. В реализации эта функция называется decoder. Она принимает строку и возвращает её преобразованную к набору символов, которые может отображать матрица

Заключение

При выполнении курсовой работы была разработана и реализована бегущая строка на базе микроконтроллера Arduino. Установка текста бегущей строки осуществляется с пмощью пк, подключенного к ардуино по USB – порту. На примере конструкции были узнаны основные моменты, связанные с изготовлением примитивных бегущих строк.

Модель, которая была собрана, способна выводить текст, вводимый с внешнего устройства.

Список используемой литературы

1. Интернет-ресурс «Программирование на Ardiuno»: <http://arduino.ru/>. Дата обращения 16.12.2021.

2.Интернет ресурс «Обучение по Tinkercad»: <https://www.tinkercad.com/learn/designs>. Дата обращения: 17.11.2021.

3. Муханин Л.Г. Схемотехника измерительных устройств. Учебное пособие: Лань, 2019.-284с

4. Интернет ресурс Роботехника: <https://роботехника18.рф/ардуино-уно/>. Дата обращения 11.11.2021.

5. Интернет ресурс «Обучение по Tinkercad»: <https://www.tinkercad.com/learn/designs>. Дата обращения: 17.11.2021.

6. Интернет-ресурс «Амперка»: <https://amperka.ru/product/particle-photon-headers>. Дата обращения 06.11.2021.

Приложение 1

#include <SPI.h> // Подключаем библиотеку SPI

#include <Adafruit\_GFX.h> // Подключаем библиотеку Adafruit\_GFX

#include <Max72xxPanel.h> // Подключаем библиотеку Max72xxPanel

String a;

int pinCS = 9; // Указываем к какому выводу подключен контакт CS

int numberOfHorizontalDisplays = 1; // Количество матриц по горизонтали

int numberOfVerticalDisplays = 4; // Количество матриц по-вертикали

/\* Функция перекодировки русских букв в формат матрицы \*/

String decoder(String source)

{

int i, k;

String target;

unsigned char n;

char m[2] = { '0', '\0' };

k = source.length(); i = 0;

while (i < k) {

n = source[i]; i++;

if (n >= 0xC0) {

switch (n) {

case 0xD0: {

n = source[i]; i++;

if (n == 0x81) {

n = 0xA8;

break;

}

if (n >= 0x90 && n <= 0xBF) n = n + 0x2F;

break;

}

case 0xD1: {

n = source[i]; i++;

if (n == 0x91) {

n = 0xB7;

break;

}

if (n >= 0x80 && n <= 0x8F) n = n + 0x6F;

break;

}

}

}

m[0] = n; target = target + String(m);

}

return target;

}

Max72xxPanel matrix = Max72xxPanel(pinCS, numberOfHorizontalDisplays, numberOfVerticalDisplays);

String tape = decoder("123мир");

int wait = 250; // интервал, чем меньше тем бытрее бежит строка

int spacer = 1; // Промежуток между символами (кол-во точек)

int width = 5 + spacer; // Ширина шрифта составляет 5 пикселей

void setup() {

Serial.begin(115200);

matrix.setIntensity(1); // Задаем яркость от 0 до 15

matrix.setRotation(1); // Направление текста 1,2,3,4

}

int i = 0;

unsigned long startTime = millis();

void loop() {

if (Serial.available() > 0) {

tape= decoder(Serial.readString());

tape.replace('\n', ' ');

tape.replace('\r', ' ');

}

if (millis() > (startTime + wait)){

startTime = millis();

// Обновляем экран

matrix.fillScreen(LOW);

// Вычисляем координаты первого символа

int x = (matrix.width() - 1) - i;

int y = 0;

for (int j=0; j< tape.length(); j++) {

matrix.drawChar(x, y, tape[j], HIGH, LOW, 1);

x += width;

}

matrix.write(); // выведим значения на матрицу

i++;

if (i >= width \* tape.length() + matrix.width() - spacer ){

i = 0;

}

}

}