

## СООБЩЕНИЕ

УДК 519.24

### РАЗРАБОТКА СХЕМЫ И ПРОГРАММИРОВАНИЕ МИКРОКОНТРОЛЛЕРА ATMELE ATTINY45 ДЛЯ УПРАВЛЕНИЯ RGB-МАТРИЦЕЙ\*

А.И. НЕЗВАНОВ<sup>1</sup>, И.И. МОСКВИЧЁВ<sup>2</sup>, Е.Д. ЖАРОВ<sup>3</sup>

<sup>1</sup> 630073, РФ, г. Новосибирск, пр. Карла Маркса, 20, Новосибирский государственный технический университет, студент факультета автоматики вычислительной техники. E-mail: nezvanovmi@mail.ru

<sup>2</sup> 630073, РФ, г. Новосибирск, пр. Карла Маркса, 20, Новосибирский государственный технический университет, студент факультета автоматики вычислительной техники. E-mail: ivanmoskvi-chev96@mail.ru

<sup>3</sup> 630073, РФ, г. Новосибирск, пр. Карла Маркса, 20, Новосибирский государственный технический университет, студент факультета автоматики вычислительной техники. E-mail: edar.vih.esrr@yandex.ru

Использование микроконтроллеров находит все большее распространение в различных областях обработки информации. Актуальные вопросы по реализации разнообразных алгоритмов, а именно разработка программного обеспечения, рассматриваются во все большем количестве работ. Некоторые алгоритмы довольно простые, но требуют тщательной проработки программного обеспечения. Но есть алгоритмы и более сложные, которые должны моделировать функционирование, например, нейронных сетей и сетей Петри. В данной работе показана последовательность проектирования не сложного, на первый взгляд, но требующего тщательной проработки, устройства. В исследовании излагается ход работ над проектом по созданию светодиодной матрицы для отображения какой-либо информации.

**Ключевые слова:** микроконтроллер, ATTINY45, сдвиговый регистр, RGB-светодиод, светодиодная матрица, программирование, C++, пайка

DOI: 10.17212/2307-6879-2016-4-144-159

## ВВЕДЕНИЕ

Реализация многочисленных алгоритмов в настоящее время может выполняться на разнообразных микроконтроллерах [1–4, 6, 9, 10, 12, 13, 15].

---

\* Статья получена 10 ноября 2016.

К таким алгоритмам можно отнести алгоритмы, реализующие автоматизацию технологических процессов, управление роботами, обработку информации и т. д. Особый интерес представляют алгоритмы, математическая реализация которых осуществлена на сетях Петри и нейронных сетях [5, 7, 8, 11, 14]. При использовании микроконтроллеров даже простая обработка информации требует особого внимания и тщательной проработки.

Ниже описана последовательность действий по разработке цифрового устройства отображения информации, реализованного на светодиодной матрице. В качестве экрана была выбрана RGB-матрица размером  $8 \times 8$  пикселей, а для управления отображением – микроконтроллер ATMEL ATTINY45.

## 1. МАТЕРИАЛЫ И СХЕМА РАБОТЫ

ATtiny 45 – экономичный 8-разрядный КМОП микроконтроллер (рис. 1), выполненный по усовершенствованной AVR RISC-архитектуре. За счет выполнения большинства инструкций за один машинный цикл микроконтроллеры семейства ATtiny 25/45/85 достигают производительности 1млн операций в секунду при тактировании частотой 1МГц. Используемый в данной работе МК имеет встроенную память 4КБ, этого достаточно для размещения программы и около 80 символов для отображения в режиме бегущей строки.

Программирование данного микроконтроллера возможно с использованием ARDUINO UNO при условии, что он запрограммирован как ISP-программатор. В качестве среды программирования была выбрана Arduino IDE, которая позволяет на языке C++ написать программу для МК. Также для этой цели можно использовать среду разработки от производителя МК и язык AVR-СИ, использование которого для имеющейся цели не имеет практического смысла, так как программирование на нем более эффективно, но и более трудоемко.

Для работы будут задействованы только три вывода МК (рис. 2), так как для управления светодиодной матрицей (рис. 3) отлично подходят сдвиговые регистры, для задания параметров которым достаточно трех управляющих линий:

- тактовый выход Serial Clock (PB2);
- выход битовых импульсов Serial In (PB3);
- выход-защелка RCK (PB1), переводящий сдвиговый регистр (74HC595TN) в состояние вывода или состояние записи.

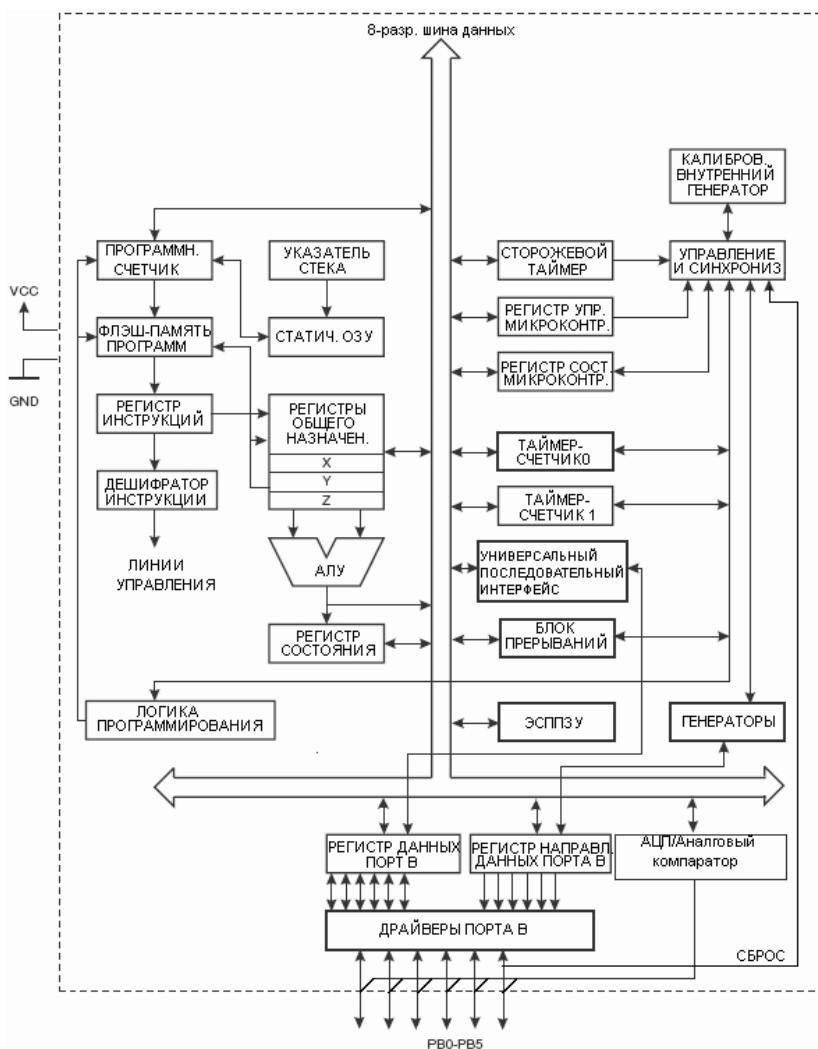


Рис. 1. Структура МК ATtiny45

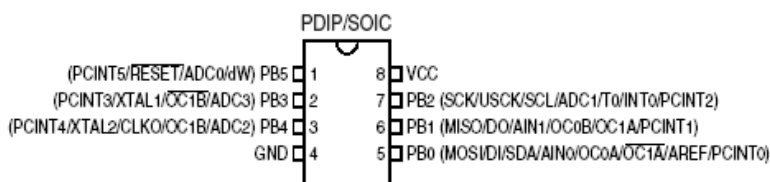


Рис. 2. Выводы МК ATTINY45

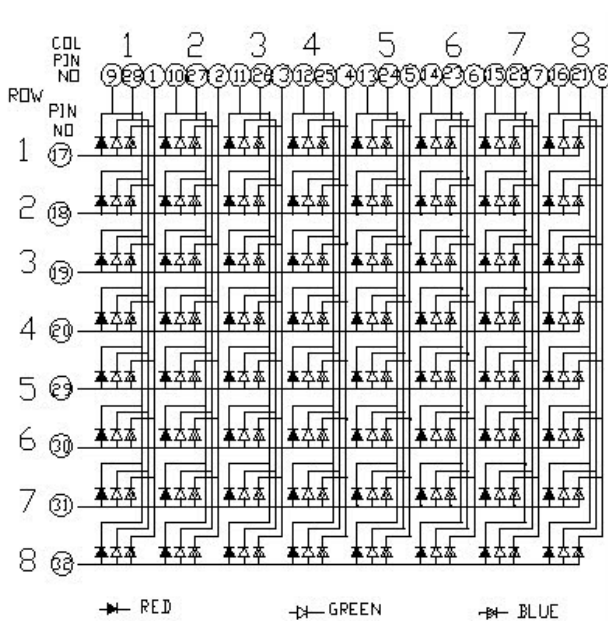


Рис. 3. Схема матрицы 2088RGB-5

Используемая матрица является сборкой с общим анодом, т. е. для подключения имеется три группы катодов (по 8 на каждый цвет в столбце) и одна группа анодов (по одному на каждую строку). Из этой структуры следует, что развертку придется производить построчно, т. е. за 8 шагов. Но в ходе прототипирования на макетной плате было выяснено, что если в каждой строке за один шаг отрисовывать все три цвета (red, green, blue), то в зависимости от заполнения строки яркость свечения отличается от соседних строк. Для решения этой проблемы было решено увеличить количество шагов отрисовки с 8 до 24, т. е. в каждой строке отдельно отрисовывается каждый цвет. Также в

ходе экспериментов было выяснено, что частоты 1 МГц недостаточно для статичного вывода, а частоты в 8 МГц, т. е. максимальной при использовании встроенного тактирующего элемента, достаточно для того, чтобы человеческий глаз не мог заметить мерцания. Также возможно использование внешнего кварцевого осциллятора на 16 МГц, но для поставленной задачи этого не требуется.

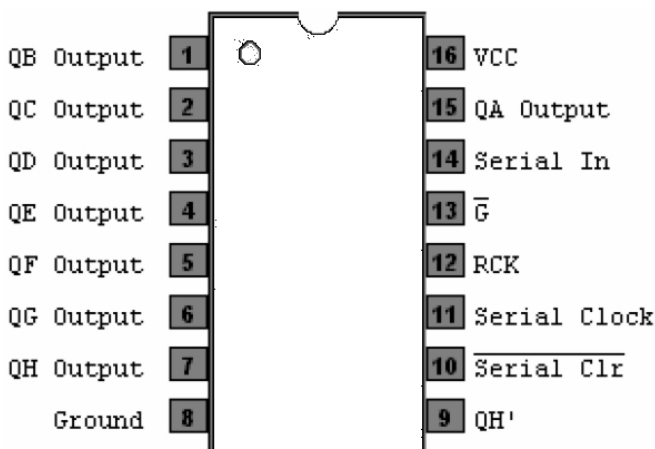


Рис. 4. Схема сдвигового регистра 74HC595N

Работа со сдвиговым регистром (рис. 4) позволяет получить большое число выходов (кратно 8) путем последовательного соединения. Управление сдвиговыми регистрами происходит по следующему алгоритму.

- Установка защелки (LATCH) в состояние LOW переводит регистр в режим задания выходов.
- Передача импульса состояния первого выхода, затем перевод CLOCK в состояние HIGH и обратно в состояние LOW для фиксирования переданного импульса.
- Повторение предыдущего пункта для всех выходов сдвигового регистра/сборки регистров.
- Установка защелки (LATCH) в состояние HIGH переводит регистр в режим работы.

В работе используется четыре сдвиговых регистра, так как матрица имеет 32 ножки для подключения

## 2. ПРОЕКТИРОВАНИЕ СХЕМЫ И ПОДГОТОВКА ПЛАТЫ

Для разработки схемы была использована программа, позволяющая размещать элементы и соединяющие дорожки. В ходе разработки были добавлены светодиод для контроля питания платы, стабилизатор LM7805 для получения напряжения в 5В и выключатель для разрыва цепи питания. Получившаяся схема показана на рис. 5.

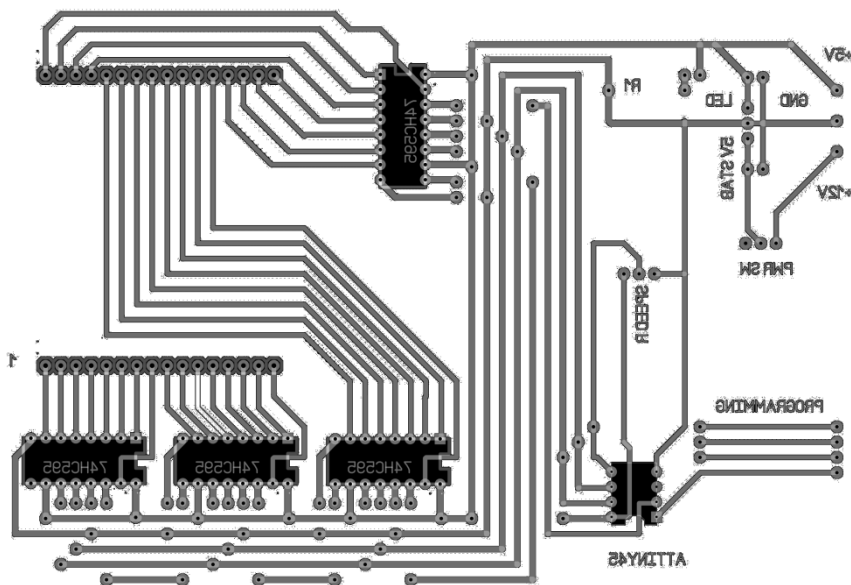


Рис. 5. Схема печатной платы, разработанная в программе

Для перевода на текстолит была применена так называемая лазерно-утюжная технология, суть которой заключается в том, что схема платы распечатывается на глянцевой бумаге лазерным принтером, после чего лист прикладывается к текстолиту и с помощью утюга (т. е. нагрева тонера на листе) всё с листа переводится на лист текстолита. Затем происходит удаление листа и травление текстолита в растворе хлорного железа. Затем необходимо с помощью тонкого сверла просверлить все посадочные отверстия для установки элементов, и плата готова для сборки.

Результат сборки представлен на рис. 6.

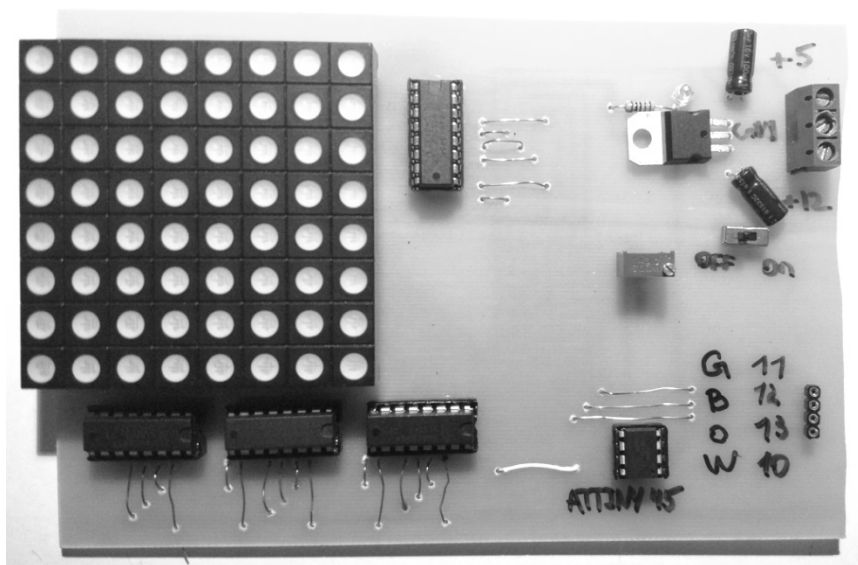


Рис. 6. Результат сборки

### 3. РАЗРАБОТКА ПРОГРАММЫ ДЛЯ ФОРМИРОВАНИЯ ТЕКСТА

Для задания отображаемого текста бегущей строки используется массив байтов (с целью уменьшения занимаемой памяти), количество элементов должно быть кратным 8.

Возможные значения в массиве:

- 0 – белый цвет,
- 1 – розовый цвет,
- 2 – голубой цвет,
- 3 – синий цвет,
- 4 – желтый цвет,
- 5 – красный цвет,
- 6 – зеленый цвет,
- 7 – черный (пустая ячейка) цвет.

Пример – задание буквы «А» белым цветом:

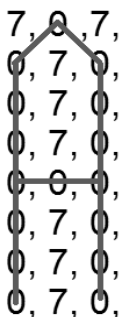


Рис. 7. Наглядный пример задания текста

Программа для генерации текста представляет собой набор запрограммированных массивов для русского алфавита, чисел 0–9 и знаков –, ?, !, которые форматируются согласно введенной строке и выводятся в файл, содержимое которого необходимо загрузить в МК.

Исходный текст программы:

```
#include <iostream>
#include <string>
#include <fstream>
#include <cstdio>

using namespace std;

string alph = "0123456789f,dult;pbqrkvyjghcnea[wxi]sm'.z ?!-";

int ret_num(char ch){
    for(int a = 0; a < alph.size(); a++) if(alph[a] == ch) return a;
    return 0;//вернуть знак вопроса
}

int main(int argc, char* argv[]){
    int sym[][50] = {
        //0
        {3,1,1,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,1},
        //1
        {3,0,1,0,1,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,1,1,1},
        //2
        {3,0,1,1,1,0,1,1,0,1,0,0,1,0,0,1,0,1,0,1,0,1,0,0,1,1,1},
        //3
```



```

{3,1,1,1,1,0,1,0,0,1,0,0,1,1,1,1,0,0,1,0,0,1,1,1,1},
//4
{3,1,0,1,1,0,1,1,0,1,1,0,1,1,1,1,0,0,1,0,0,1,0,0,1},
//5
{3,1,1,1,1,0,0,1,0,0,1,0,0,1,1,1,0,0,1,0,0,1,1,1,1},
//6
{3,1,1,1,1,0,1,1,0,0,1,0,0,1,1,1,1,0,1,1,0,1,1,1,1},
//7
{3,1,1,1,1,0,1,0,0,1,0,1,0,0,1,0,1,0,0,1,0,0,1,0,0},
//8
{3,1,1,1,1,0,1,1,0,1,1,0,1,1,1,1,1,0,1,1,0,1,1,1,1},
//9
{3,1,1,1,1,0,1,1,0,1,1,0,1,1,1,1,1,0,0,1,0,0,1,1,1,1},
//a
{3,0,1,0,1,0,1,1,0,1,1,0,1,1,1,1,1,0,1,1,0,1,1,0,1},
//6
{3,1,1,1,1,0,1,1,0,0,1,0,0,1,1,0,1,0,1,1,0,1,1,1,1},
//b
{3,1,1,1,1,0,1,1,0,1,1,0,1,1,1,0,1,0,1,1,0,1,1,1,1},
//г
{3,1,1,1,1,0,1,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0},
//д
{5,0,0,1,0,0,0,1,0,1,0,0,1,0,1,0,0,1,0,1,0,0,1,0,1,1,1,1,1,1,0,0,0,1
,1,0,0,0,1},
//e
{3,1,1,1,1,0,0,1,0,0,1,0,0,1,1,1,1,0,0,1,0,0,1,1,1},
//ж
{5,1,0,1,0,1,1,0,1,0,1,1,0,1,0,1,0,1,1,1,0,0,1,1,1,0,1,0,1,0,1,0,1
,1,0,1,0,1},
//з
{3,1,1,1,1,0,1,0,0,1,0,0,1,1,1,1,0,0,1,1,0,1,1,1,1},
//и
{4,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,1,1,0,1,1,1,0,1,0,0,1},
//й
{4, 0,1,1,0, 0,0,0,0, 1,0,0,1,1,0,0,1,1,0,0,1,1,0,1,1,1,1,0,1,1,0,0,1},
//к
{3,1,0,1,1,0,1,1,0,1,1,1,0,1,1,0,1,0,1,1,0,1,1,0,1},
//л
{3,0,0,1,0,1,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1},
//м
{5,1,0,0,0,1,1,1,0,1,1,1,0,1,0,1,1,0,1,0,1,1,0,1,0,1,1,0,1,0,1,0,1
,1,0,1,0,1},
//н
{3,1,0,1,1,0,1,1,0,1,1,0,1,1,1,1,1,0,1,1,0,1,1,0,1},
//o
{3,0,1,0,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,0,1,0},
//п
{3,1,1,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1},
//p

```

[illegible]

;

```

    cout << "Text generator for attiny45 RGB controller.\nVer 1.0\n\n";
    cout << "Print text (russian alph. and digits 0-9) in english keyboard
layer than press enter." << endl;
    string str;
    getline(std::cin, str);
    for(int b = 0; b < str.size(); b++) cout << ret_num(str[b]) << " ";
    cout << "Start converting..." << endl;
    string out = "";
    int len, num, count = 0;

    for(int s = 0; s < 8; s++){
        out += "7,7,7,7,7,7,7,7,";
        for(int a = 0; a < str.size(); a++){
            num = ret_num(str[a]);
            len = sym[num][0];
            if(s == 0) count += len;
            for(int b = 0; b < len; b++){
                if(sym[num][1+s*len+b] == 0) out += "7";
                else out += a%7+48;
                out+=",";
            }
            out += "7,"; if(s == 0) count += 1;
        }
        out += "7,7,7,7,7,7,7,7,\n";
        if(s == 0) count += 16;
    }

    ofstream fout("out.txt");
        fout << count << "\n" << out;
        fout.close();
    return 0;
}

```

#### 4. ИСХОДНЫЙ КОД ПРОГРАММЫ МК

```

#include <avr/pgmspace.h>
int dataPin= 3;
int clockPin = 2;
int latchPin = 1;
int len = 73;//длина строки
int print = 5;//скорость отображения

const PROGMEM byte ch[] = {
//массив текста
};

void setup() {

```

```

    len -= 8;
    pinMode(latchPin, OUTPUT);
    pinMode(clockPin, OUTPUT);
    pinMode(dataPin, OUTPUT);
}

int thisRow = 0, thisCol = 0;
int count = 0, count2 = 0;
void loop() {
    int count = 0;
    if(count2 == len) count2 = 0;
    while(count < print){
        for(int a = 0; a < 24; a++){
            digitalWrite(latchPin, LOW);
            for(int c = 0; c < 24; c++){
                if(c < 8){//BLUE
                    if(thisCol == 0){
                        if(pgm_read_byte(ch + thisRow*(8+len)+c+count2) >
3) digitalWrite( dataPin, HIGH );
                        else digitalWrite( dataPin, LOW);
                    }
                    else digitalWrite( dataPin, HIGH);
                } else if(c > 7 && c < 16){//GREEN
                    if(thisCol == 1){
                        if(pgm_read_byte(ch + thisRow*(8+len)+c-8+count2)
== 2 || pgm_read_byte(ch + thisRow*(8+len)+c-8+count2) == 3 ||
pgm_read_byte(ch + thisRow*(8+len)+c-8+count2) >5) digitalWrite( dataPin,
HIGH );
                        else digitalWrite( dataPin, LOW);
                    }
                    else digitalWrite( dataPin, HIGH);
                } else { //RED
                    if(thisCol == 2){
                        if(pgm_read_byte(ch + thisRow*(8+len)+c-16+count2)
== 1 || pgm_read_byte(ch + thisRow*(8+len)+c-16+count2) == 3 ||
pgm_read_byte(ch + thisRow*(8+len)+c-16+count2) == 5 || pgm_read_byte(ch +
thisRow*(8+len)+c-16+count2) == 7) digitalWrite( dataPin, HIGH);
                        else digitalWrite( dataPin, LOW);
                    }
                    else digitalWrite( dataPin, HIGH);
                }
            }
            digitalWrite( clockPin, HIGH );
            digitalWrite( clockPin, LOW );
        }
        for(int b = 0; b < 8; b++){
            if(thisRow == 0 && b == 3) digitalWrite( dataPin, HIGH
);
            else if(thisRow == 1 && b == 2) digitalWrite( dataPin,
HIGH );

```

```
else if(thisRow == 2 && b == 1) digitalWrite( dataPin,
HIGH );
else if(thisRow == 3 && b == 0) digitalWrite( dataPin,
HIGH );
else if(thisRow == 4 && b == 4) digitalWrite( dataPin,
HIGH );
else if(thisRow == 5 && b == 5) digitalWrite( dataPin,
HIGH );
else if(thisRow == 6 && b == 6) digitalWrite( dataPin,
HIGH );
else if(thisRow == 7 && b == 7) digitalWrite( dataPin,
HIGH );
else digitalWrite( dataPin, LOW );
digitalWrite( clockPin, HIGH );
digitalWrite( clockPin, LOW );
}
digitalWrite(latchPin, HIGH);
thisCol++;
if(thisCol == 3){
    thisCol = 0;
    thisRow++;
}
}
thisCol = 0;
thisRow = 0;
count++;
}
count2++;
}
```

## ЗАКЛЮЧЕНИЕ

В данной работе были достигнуты поставленные цели, собранная схема полностью готова для использования в других проектах. Также был получен навык в программировании МК ATTINY45, проектировании и сборке печатных плат.

## БЛАГОДАРНОСТИ

Авторы выражают свою искреннюю благодарность профессору кафедры автоматики А.А. Воеводе за помощь при выполнении работ, а также полезное обсуждение полученных результатов.

## СПИСОК ЛИТЕРАТУРЫ

1. Воевода А.А., Романников Д.О., Зимаев И.В. Применение UML диаграмм и сетей Петри при разработке встраиваемого программного обеспечения // Научный вестник НГТУ. – 2009. – № 4. – С. 169–174.
2. Воевода А.А., Романников Д.О. Редуцирование пространства состояний сетей Петри для объектов одного класса // Научный вестник НГТУ. – 2011. – № 4. – С. 136–139.
3. Коротиков С.В., Воевода А.А. Применение сетей Петри в разработке программного обеспечения центров дистанционного управления и контроля // Научный вестник НГТУ. – 2007. – № 4. – С. 15–32.
4. Воевода А.А., Романников Д.О. О компактном представлении языков раскрашенных сетей Петри // Сборник научных трудов НГТУ. – 2008. – № 3(53). – С. 105–108.
5. Воевода А.А., Марков А.В. Рекурсия в сетях Петри // Сборник научных трудов НГТУ. – 2012. – № 3 (69). – С. 115–122.
6. Евстифеев А.В. Микроконтроллеры AVR семейства Tiny. – М.: Додэка-XXI, 2007. – 290 с. – (Программируемые системы).
7. Евстифеев А.В. Микроконтроллеры AVR семейств Tiny и Mega фирмы Atmel. – М.: Додэка-XXI, 2005. – 546 с.
8. Мортон Дж. Микроконтроллеры AVR: вводный курс. – М.: Додэка-XXI, 2005. – 345 с.
9. Белов А.В. Микроконтроллеры AVR в радиолюбительской практике. – СПб.: Наука и техника, 2007. – 337 с.
10. Редькин П.П. Микроконтроллеры Atmel архитектуры AVR32 семейства AT32UC3. – М.: Техносфера, 2010. – 783 с.
11. Голубцов М.С. Микроконтроллеры AVR: от простого к сложному. – М.: Солон-Пресс, 2003. – 287 с.
12. Гадре Д., Мэлхотра Н. Занимательные проекты на базе микроконтроллеров tinyAVR. – СПб.: БВХ-Петербург, 2012. – 340 с.
13. Шпак Ю.А. Программирование на языке С для AVR и PIC микроконтроллеров. – Изд. 2-е, перераб. и доп. – СПб.: Корона-Век; Киев: Мк-Пресс, 2011. – 544 с.
14. Кривченко И.В. AVR-микроконтроллеры: очередной этап развития // Компоненты и технологии. – 2002. – № 3. – С. 98–103.
15. Баранов В.Н. Применение микроконтроллеров AVR: схемы, алгоритмы, программы. – 2-е изд. – М.: Додэка-XXI, 2006. – 288 с.

**Незванов Александр Игоревич**, студент факультета автоматики и вычислительной техники Новосибирского государственного технического университета. E-mail: nezvanovml@mail.ru

**Москвичёв Иван Игоревич**, студент факультета автоматики и вычислительной техники Новосибирского государственного технического университета. E-mail: ivanmoskvichev96@mail.ru

**Жаров Евгений Дмитриевич**, студент факультета автоматики и вычислительной техники Новосибирского государственного технического университета. E-mail: edar.vih.esrr@yandex.ru

## Development of the scheme and programming of the atmel attiny 45 microcontroller for RGB-MATRIX CONTROL\*

A.I. Nezvanov<sup>1</sup>, I.I. Moskvichev<sup>2</sup>, E.D. Zharov<sup>3</sup>

<sup>1</sup> Student of Novosibirsk State Technical University. E-mail: nezvanovml@mail.ru

<sup>2</sup> Student of Novosibirsk State Technical University. E-mail: ivanmoskvichev96@mail.ru

<sup>3</sup> Student of Novosibirsk State Technical University. E-mail: edar.vih.esrr@yandex.ru

The application of microcontrollers is becoming increasingly widespread in various areas of information processing. Actual questions on the implementation of various algorithms, the development of software, is encountered in an increasing number of papers. Some algorithms are fairly simple, but require careful software development. But there are more complex algorithms, that should simulate the functioning, for example, of neural networks and Petri nets. In this paper, we show the sequence of designing a device that, at the first sight, is not complex but which requires careful study. The study outlines the progress of the project to create an LED matrix for displaying any information.

**Keywords:** microcontroller, ATTINY45, shift register, RGB LED, LED matrix, programming, C++, soldering

DOI: 10.17212/2307-6879-2016-4-144-159

## REFERENCES

1. Voevoda A.A., Romannikov D.O., Zimaev I.V. Primenenie UML diagramm i setei Petri pri razrabotke vstraivaemogo programmnoo obespecheniya [An approach to the using UML and Petri nets for embedded software designing]. *Nauchnyi vestnik Novosibirskogo gosudarstvennogo tekhnicheskogo universiteta – Science bulletin of the Novosibirsk state technical university*, 2009, no. 4, pp. 169–174.
2. Voevoda A.A., Romannikov D.O. Redutsirovanie prostranstva sostoyanii setei Petri dlya ob"ektov odnogo klassa [Reducing the state space of Petri nets for objects of one class]. *Nauchnyi vestnik Novosibirskogo gosudarstvennogo tekhnicheskogo universiteta – Science bulletin of the Novosibirsk state technical university*, 2011, no. 4, pp. 136–139.
3. Korotikov S.V., Voevoda A.A. Primenenie setei Petri v razrabotke programmnoo obespecheniya tsentrov distantsionnogo upravleniya i kontrolya [Using

---

\* Received 10 November 2016.

Petri nets in software development of remote monitoring and control center]. *Nauchnyi vestnik Novosibirskogo gosudarstvennogo tekhnicheskogo universiteta – Science bulletin of the Novosibirsk state technical university*, 2007, no. 4, pp. 15–32.

4. Voevoda A.A., Romannikov D.O. O kompaktnom predstavlenii yazykov raskrashennykh setei Petri [On the compact representation of the languages of colored Petri nets]. *Sbornik nauchnykh trudov Novosibirskogo gosudarstvennogo tekhnicheskogo universiteta – Transaction of scientific papers of the Novosibirsk state technical university*, 2008, no. 3 (53), pp. 105–108.

5. Voevoda A.A., Markov A.V. Rekursiya v setyakh Petri [The concepts recursion in Petri nets]. *Sbornik nauchnykh trudov Novosibirskogo gosudarstvennogo tekhnicheskogo universiteta – Transaction of scientific papers of the Novosibirsk state technical university*, 2012, no. 3 (69), pp. 115–122.

6. Evstifeev A.V. *Mikrokontrollery AVR semeistva Tiny* [Microcontrollers Tiny of AVR type]. Moscow, Dodeka-XXI Publ., 2007. 290 p.

7. Evstifeev A.V. *Mikrokontrollery AVR semeistv Tiny i Mega firmy Atmel* [Microcontrollers Tiny, Mega of AVR type from Atmel]. Moscow, Dodeka-XXI Publ., 2005. 546 p.

8. Morton J. *AVR: an introductory course*. Oxford, Newnes, 2002 (Russ. ed.: Morton Dzh. *Mikrokontrollery AVR: vvodnyi kurs*. Moscow, Dodeka-XXI Publ., 2005. 345 p.).

9. Belov A.V. *Mikrokontrollery AVR v radiolyubitel'skoi praktike* [Microcontrollers AVR if radiotechnics practice]. St. Petersburg, Nauka i tekhnika Publ., 2007. 337 p.

10. Red'kin P.P. *Mikrokontrollery Atmel arkhitektury AVR32 semeistva AT32UC3* [Microcontrollers Atmelwith AVR32 architecture]. Moscow, Tekhnosfera Publ., 2010. 783 p.

11. Golubtsov M.S. *Mikrokontrollery AVR: ot prostogo k slozhnomu* [Microcontrollers AVR: from easy to hard]. Moscow, Solon-Press Publ., 2003. 287 p.

12. Gadre D.V., Malhotra N. *TinyAVR microcontroller projects for the evil genius*. New York, McGraw-Hill, 2011 (Russ. ed.: Gadre D., Melkhotra N. *Zanimatel'nye proekty na baze mikrokontrollerov tinyAVR*. St. Petersburg, BHV-Peterburg, 2012. 340 p.).

13. Shpak Ju.A. *Programmirovaniye na yazyke C dlya AVR i PIC mikrokontrollerov* [Programming on C for AVR & PIC microcontrollers]. 2<sup>nd</sup> ed. St. Petersburg, Korona-Vek Publ., Kiev, Mk-Press Publ., 2011. 544 p.

14. Krivchenko I.V. AVR-mikrokontrollery: ocherednoi etap razvitiya [AVR-microcontrollers: stage of evolution]. *Komponenty i tekhnologii – Components & Technologies*, 2002, no. 3, pp. 98–103.

15. Baranov V.N. *Primeneniye mikrokontrollerov AVR: skhemy, algoritmy, programmy* [Using microcontrollers AVR: schemes, algorithmes, programmes]. 2<sup>nd</sup> ed. Moscow, Dodeka-XXI Publ., 2006. 288 p.