**Федеральное государственное бюджетное образовательное учреждение высшего образования "Белгородский государственный технологический университет им. В.Г. Шухова"**

Кафедра программного обеспечения вычислительной техники и

автоматизированных систем.

**Лабораторная работа работа № 3**

Изучение среды разработки arduino IDE и стандартных библиотек для работы с портами и интерфейсами микроконтроллера ATMEGA 328p.

Вариант 13

Выполнил:

Студент группы КБ-211

Коренев Д.Н.

Принял:

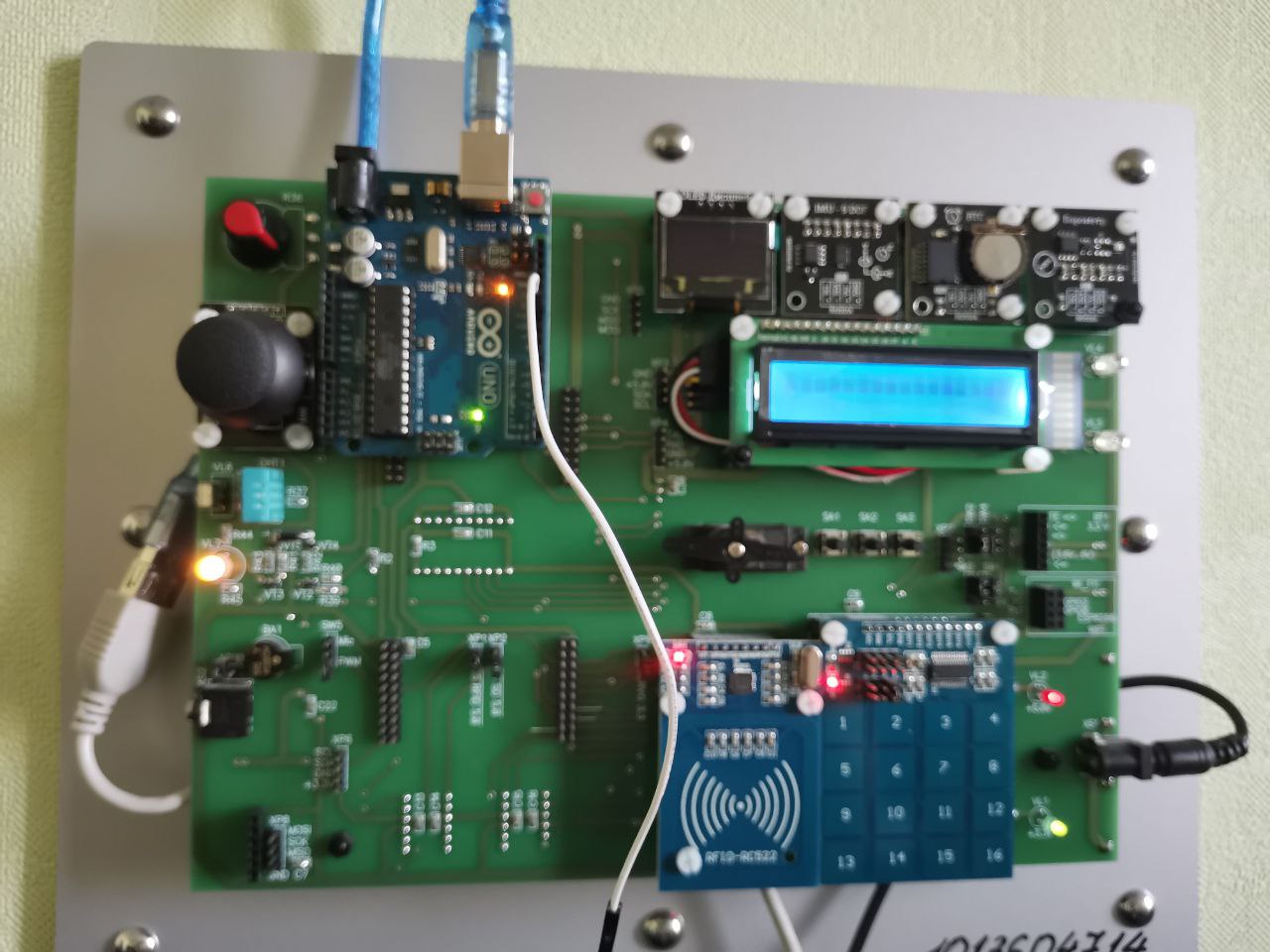
Шамраев А.А.

*Цель работы:* изучить функциональные возможности среды разработки Arduino IDE, научиться компилировать программу и загружать ее в микроконтроллер. Изучить стандартные библиотеки среды Arduino IDE для работы с портами ввода вывода, интерфейсами микроконтроллера Atmega 328p и научиться применять их на практике.

**Задание**

1. Получить навыки работы с GPIO портами. Зажечь светодиод, подключенный к порту D13, помигать им.

1. void setup() {
2. pinMode(13, OUTPUT);
3. }
4. void loop() {
5. digitalWrite(13, HIGH);
6. delay(1000);
7. digitalWrite(13, HIGH);
8. delay(1000);
9. }



2. Воспользоваться стандартными функциями для генерации ШИМ:

В соответствии со своим номером в списке, сделать задания:

Номер меньше 8:

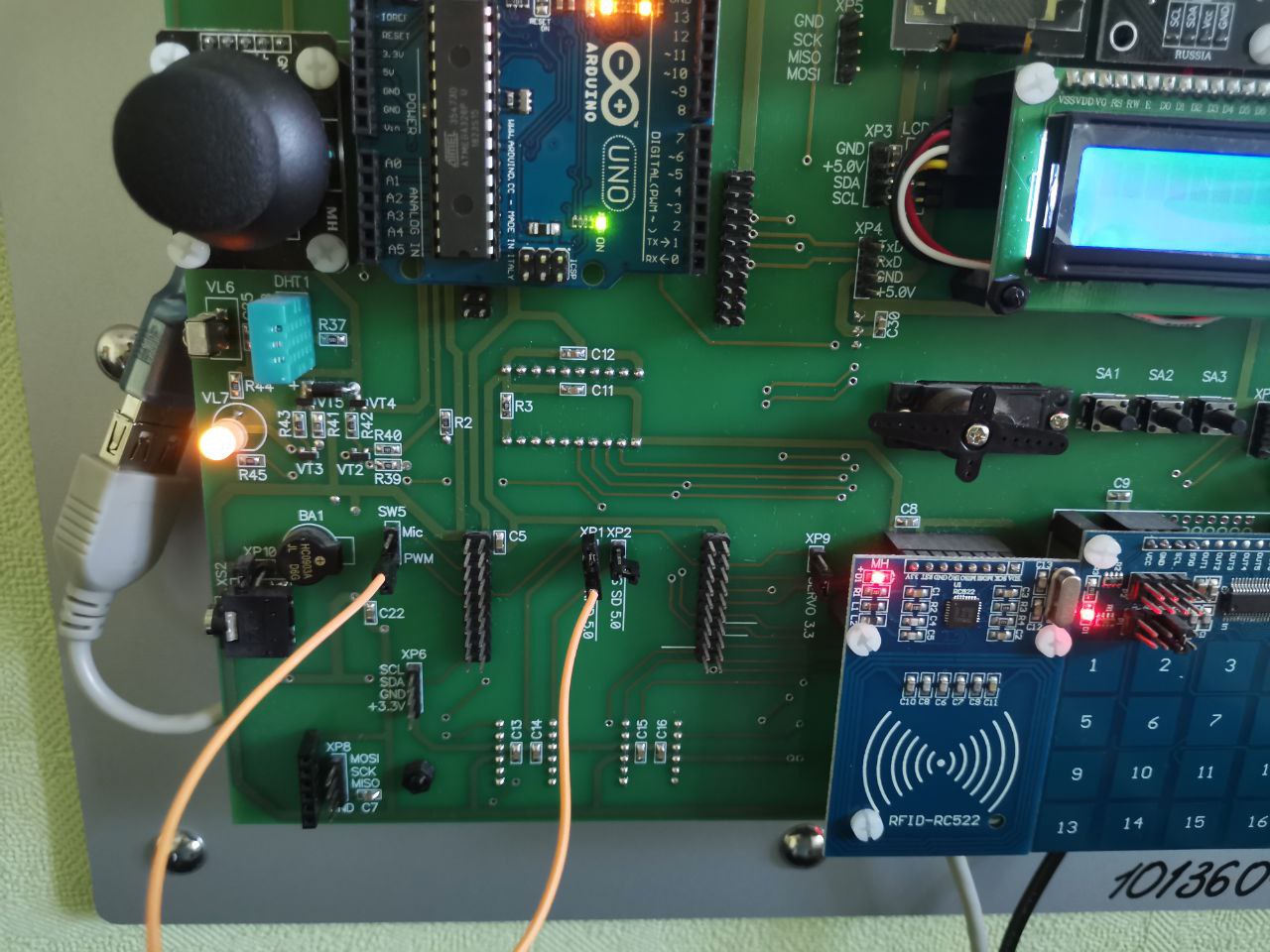
Сгенерировать ШИМ на 13 выводе Arduino. Плавно зажигать и гасить светодиод с некоторым периодом, яркостью в пределах от 0 до номера в журнале, умноженного на 30 с шагом, равным вашему номеру в журнале.

Управление по ШИМ недоступно на 13 выводе Arduino.

Номер больше или равен 8:

Установить перемычку для работы со спикером. Сделать сирену, изменяя скважность ШИМ от 30 до 150.

Период выбрать (номер в журнале)\*50 мс.

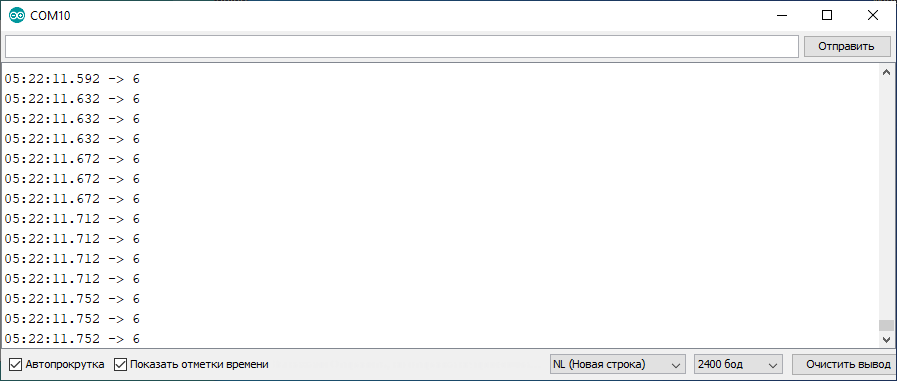


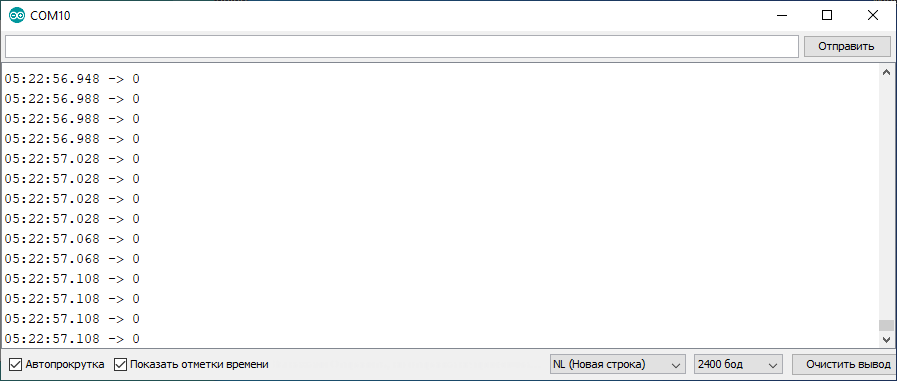
1. void setup() {
2. pinMode(10, OUTPUT);
3. }
4. void loop() {
5. sise\_t i;
6. for (i = 0; i <= 150; i += 10)(
7. analogWrite(10, i);
8. delay(650);
9. )
10. while (i > 30) {
11. i -= 10;
12. analogWrite(10, i);
13. delay(650);
14. }
15. }

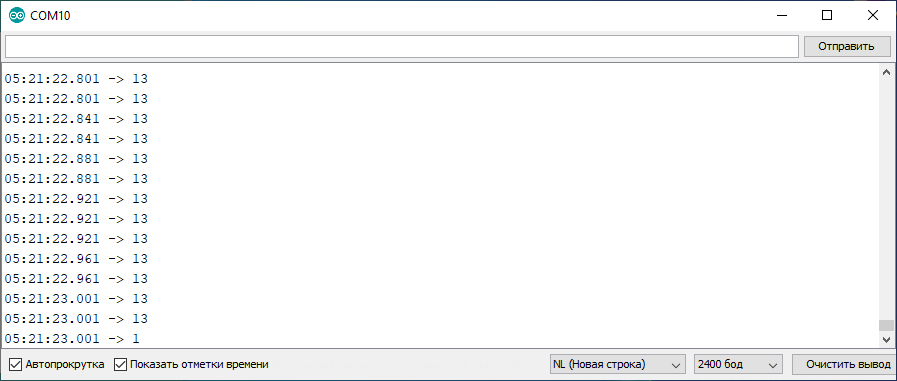
3. Получить напряжение на центральном выводе потенциометра при помощи АЦП и вывести его в COM порт. Вращая ручку, убедиться, что напряжение изменяется. С помощью функции map изменить диапазон получаемого значения в диапазон от 0 до номера в журнале. Если диапазон получился слишком маленьким, умножить максимальное значение на 10.

1. void setup() {
2. pinMode(A0, INPUT);
3. Serial.begin(2400);
4. }
5. void loop() {
6. int input = analogRead(A0);
7. int value = map(input, 0, 1023, 0, 14);
8. Serial.println(value);
9. }

Вращаем ручку:







4. С помощью интерфейса I2C изменить состояние расширителя портов. Зажечь несколько светодиодов на световой шкале, зажечь RGB светодиоды. Полоски на шкале загораются при низком уровне на выводе (схема с общим анодом), а RGB светодиоды при высоком.

Сделать задание соответствующее своему варианту (номер в журнале % 6 + 1)

1. Зажечь полоски световой шкалы через одну, начиная с первой

2. Зажечь полоски световой шкалы через одну, начиная со второй

3. Зажечь оба RGB светодиода синим цветом

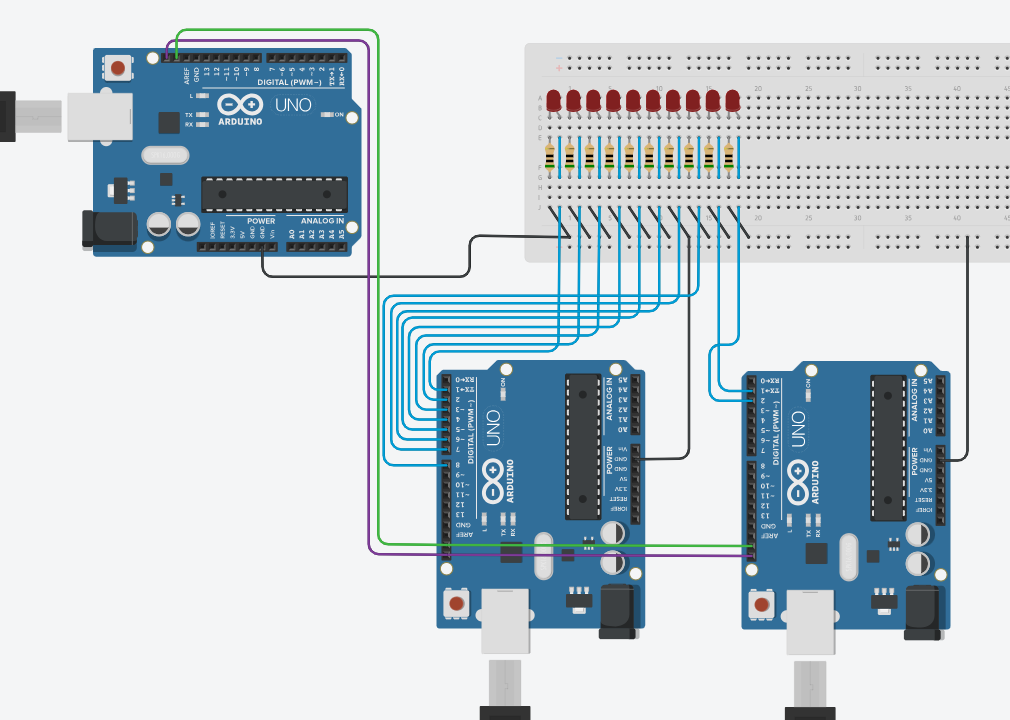
4. Зажечь половину световой шкалы и один из rgb светодиодов желтым цветом

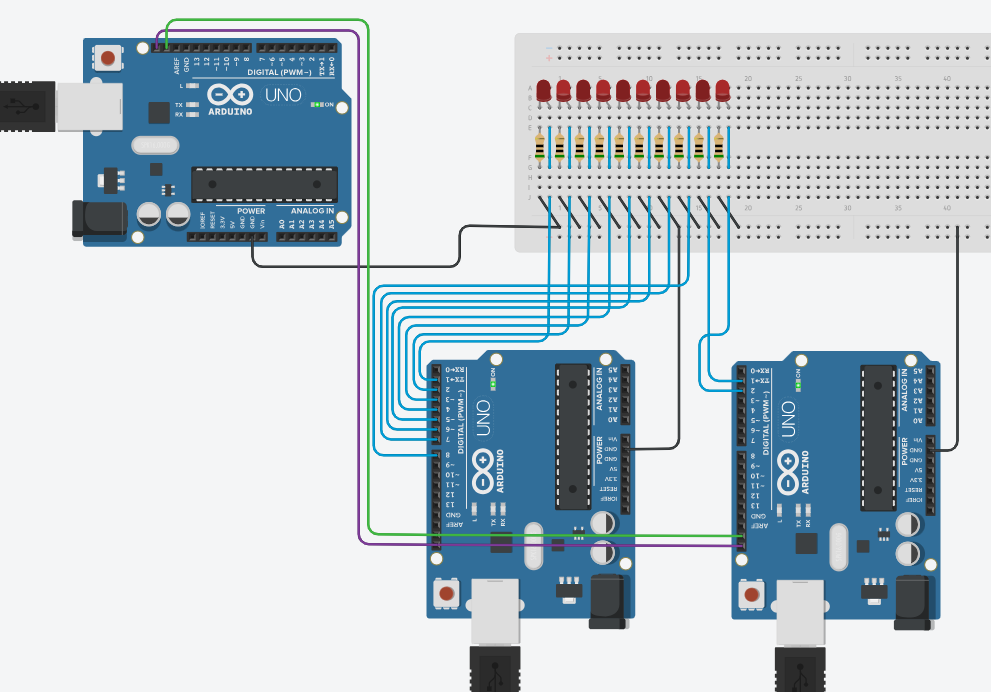
5. Зажечь нижний RGB светодиод зеленым, а верхний красным

6. Зажечь оба RGB светодиода всеми цветами (смесь R, G, B), полоски на световой шкале сериями по 2 через 1.

1. #include <Wire.h>
2. void setup()
3. {
4. Wire.begin(0);
5. }
6. byte lampCode0 = 0b10101010; // 0x73
7. byte lampCode1 = 0b00000010; // 0x70
8. void loop()
9. {
10. Wire.beginTransmission(0x73);
11. Wire.write(lampCode0);
12. Wire.endTransmission();
13. Wire.beginTransmission(0x70);
14. Wire.write(lampCode1);
15. Wire.endTransmission();
16. }

Для задания в среде Tinkercad собрал схему (нижние платы Arduino замещают собой две PCA9538, верхняя работает в режиме “master”):





Код для плат Arduino в режиме “slave”:

1. #include <Wire.h>
2. #define WIRE\_PORT 0x73
3. #define ZERO\_PIN 1
4. void setup()
5. {
6. Wire.begin(WIRE\_PORT);
7. Wire.onReceive(receiveEvent);
8. }
9. void loop()
10. {
11. delay(100);
12. }
13. void receiveEvent(int howMany)
14. {
15. byte c;
16. byte mask = 0b00000001;
17. c = Wire.read();
18. for (char i=0; i<8; i++)
19. {
20. if (c & mask){
21. digitalWrite(i+ZERO\_PIN, HIGH);
22. } else {
23. digitalWrite(i+ZERO\_PIN, LOW);
24. }
25. c >>= 1;
26. }
27. }

5. Написать микро-библиотеку (несколько функций), реализующую взаимодействие с расширителем портов PCA9538. Должна быть возможность устанавливать режим любого порта на расширителе, считывать уровень сигнала на нем, устанавливать выходной уровень, включать инверсию по входу.

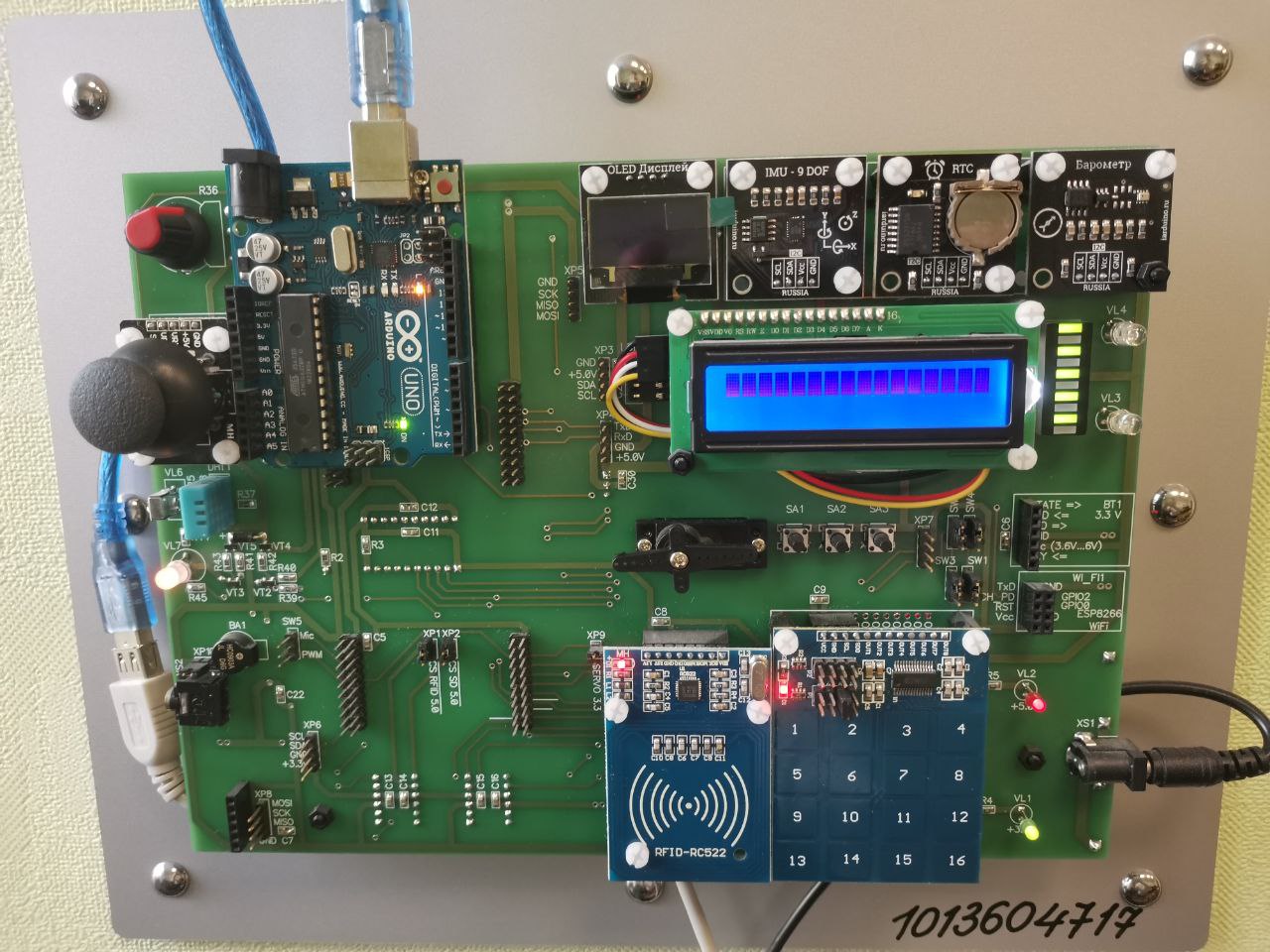
Формировать нужные значения регистров можно при помощи маски/битового поля.

Номер в журнале меньше 8:

Используя полученную библиотеку, реализовать отображение числа при помощи световой шкалы. Задействована должна быть вся шкала, включая 2 нижних полоски. Сначала должно произойти тестирование шкалы – отображение чисел от 1 до 10, с шагом 1 за 1 секунду. Затем отображать остаток от деления значения, полученного с помощью analogRead на выводе A0 (U\_R, центральный вывод потенциометра), на номер в журнале. При вращении ручки потенциометра, отображаемое число должно меняться.

Номер в журнале больше или равен 8:

Используя сирену из задания 2, сделать световой сигнал для сирены с помощью RGB светодиодов. Состояние расширителей портов менять при помощи реализованной функции. С помощью потенциометра должен регулироваться период сирены. Начальное положение (при 0), должно соответствовать периоду в задании 2.



Файл “pca95xx.h”:

1. #ifndef PCA95XX\_H\_
2. #define PCA95XX\_H\_
3. #include <Arduino.h>
4. #include <Wire.h>
5. #define PCA95XX\_IN\_REG 0x00
6. #define PCA95XX\_OUT\_REG 0x01
7. #define PCA95XX\_POL\_REG 0x02
8. #define PCA95XX\_CONF\_REG 0x03
9. #define PCA95XX\_IN\_ALL 0xFF
10. #define PCA95XX\_OUT\_ALL 0x00
11. #define PCA95XX\_POL\_DEFAULT\_ALL 0x00
12. #define PCA95XX\_POL\_INVERTED\_ALL 0xFF
13. #define PCA95XX\_PIN\_RESET\_MODE OUTPUT
14. // Отправляет сигнал RESET для PCA9538.
15. // Parameters
16. // ----------
17. // resetPin(uint8\_t): Вывод сброса
18. // Returns
19. // -------
20. // NONE
21. void pca95xx\_reset(int8\_t resetPin);
22. // Настраивает все выводы в режим OUT по умполчанию.
23. // Parameters
24. // ----------
25. // addr(uint8\_t): Адрес микросхемы PCA95XX на I2C
26. // polInv(uint8\_t): Инверсия выводов (байт настройки)
27. // io(uint8\_t): Режим ввод/вывод (байт настройки)
28. // Returns
29. // -------
30. // NONE
31. void pca95xx\_configure(uint8\_t addr, uint8\_t polInv = PCA95XX\_POL\_DEFAULT\_ALL, uint8\_t io = PCA95XX\_OUT\_ALL);
32. // Получает данные из PCA95XX.
33. // Parameters
34. // ----------
35. // addr(uint8\_t): Адрес микросхемы PCA95XX на I2C
36. // Returns
37. // -------
38. // uint8\_t: Данные, полученные от PCA95XX
39. uint8\_t pca95xx\_in(uint8\_t addr);
40. // Отправляет данные в PCA95XX.
41. // Parameters
42. // ----------
43. // addr(uint8\_t): Адрес микросхемы PCA95XX на I2C
44. // data(uint8\_t): Байт данных
45. // Returns
46. // -------
47. // NONE
48. void pca95xx\_out(uint8\_t addr, uint8\_t data);
49. #endif // PCA95XX\_H\_

Файл “pca95xx.cpp”:

1. #include "pca95xx.h"
2. // Внешний буфер для результатов работы функций
3. uint8\_t pca95xx\_i2c\_buf = 0u;
4. // Отправляет байт data с командой cmd в PCA95XX
5. // с адресом addr.
6. // Parameters
7. // ----------
8. // addr(uint8\_t): Адрес микросхемы PCA95XX на I2C
9. // cmd(uint8\_t): Код операции
10. // data(uint8\_t): Байт данных
11. // Returns
12. // -------
13. // bool: Возвращает 1 в случае ошибки,
14. // 0 при удачном выполнении
15. bool pca95xx\_i2c\_sendData(uint8\_t addr, uint8\_t cmd,
16. uint8\_t data = 0u)
17. {
18. Wire.beginTransmission(addr);
19. Wire.write(cmd);
20. if (cmd != PCA95XX\_IN\_REG)
21. Wire.write(data);
22. if (Wire.endTransmission())
23. return 1;
24. return 0;
25. }
26. // Получает байт данных из PCA95XX
27. // с адресом addr.
28. // Записывает результат в буфер.
29. // Parameters
30. // ----------
31. // addr(uint8\_t): Адрес микросхемы PCA95XX на I2C
32. // Returns
33. // -------
34. // bool: Возвращает 1 в случае ошибки,
35. // 0 при удачном выполнении
36. bool pca95xx\_i2c\_receiveData(uint8\_t addr)
37. {
38. if (pca95xx\_i2c\_sendData(addr, PCA95XX\_IN\_REG)){
39. return 1;
40. }
41. delay(15);
42. Wire.requestFrom(addr, 1u, 1u);
43. while (Wire.available() < 1);
44. pca95xx\_i2c\_buf = Wire.read();
45. return 0;
46. }
47. // Отправляет сигнал RESET для PCA9538.
48. // Parameters
49. // ----------
50. // resetPin(uint8\_t): Вывод сброса
51. // Returns
52. // -------
53. // NONE
54. void pca95xx\_reset(int8\_t resetPin)
55. {
56. if (resetPin < 0)
57. return;
58. pinMode(resetPin, PCA95XX\_PIN\_RESET\_MODE);
59. digitalWrite(resetPin, LOW);
60. delay(1);
61. digitalWrite(resetPin, HIGH);
62. delay(1);
63. }
64. // Настраивает все выводы в режим OUT по умполчанию.
65. // Parameters
66. // ----------
67. // addr(uint8\_t): Адрес микросхемы PCA95XX на I2C
68. // polInv(uint8\_t): Инверсия выводов (байт настройки)
69. // io(uint8\_t): Режим ввод/вывод (байт настройки)
70. // Returns
71. // -------
72. // NONE
73. void pca95xx\_configure(uint8\_t addr, uint8\_t polInv, uint8\_t io)
74. {
75. pca95xx\_i2c\_sendData(addr, PCA95XX\_POL\_REG, polInv);
76. pca95xx\_i2c\_sendData(addr, PCA95XX\_CONF\_REG, io);
77. delay(15);
78. }
79. // Получает данные из PCA95XX.
80. // Parameters
81. // ----------
82. // addr(uint8\_t): Адрес микросхемы PCA95XX на I2C
83. // Returns
84. // -------
85. // uint8\_t: Данные, полученные от PCA95XX
86. uint8\_t pca95xx\_in(uint8\_t addr)
87. {
88. if (!pca95xx\_i2c\_receiveData(addr)) return 0u;
89. return pca95xx\_i2c\_buf;
90. }
91. // Отправляет данные в PCA95XX.
92. // Parameters
93. // ----------
94. // addr(uint8\_t): Адрес микросхемы PCA95XX на I2C
95. // data(uint8\_t): Байт данных
96. // Returns
97. // -------
98. // NONE
99. void pca95xx\_out(uint8\_t addr, uint8\_t data)
100. {
101. pca95xx\_i2c\_sendData(addr, PCA95XX\_OUT\_REG, data);
102. }

Файл “main.ino”:

1. #include <Wire.h>
2. #include "pca95xx.h"
3. void setup()
4. {
5. Wire.begin(0);
6. }
7. byte lampCode0 = 0b10101010; // 0x73 // BRG
8. byte lampCode1 = 0b00000010; // 0x70
9. #define COM\_BYTE 0x03
10. #define COM\_BYTE\_OUT 0x01
11. void loop()
12. {
13. pca95xx\_configure(0x73);
14. pca95xx\_out(0x73, lampCode0);
15. pca95xx\_configure(0x70);
16. pca95xx\_out(0x70, lampCode1);
17. delay(1000);
18. pca95xx\_configure(0x73);
19. pca95xx\_out(0x73, lampCode0 ^ 0xff);
20. pca95xx\_configure(0x70);
21. pca95xx\_out(0x70, lampCode1 ^ 0b00000011);
22. delay(1000);
23. }

**Вывод**: в ходе лабораторной работы мы изучили функциональные возможности среды разработки Arduino IDE, научились компилировать программу и загружать ее в микроконтроллер. Изучили стандартные библиотеки среды Arduino IDE для работы с портами ввода вывода, интерфейсами микроконтроллера Atmega 328p и научились применять их на практике.