**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 № 5**

Изучение принципа работы аналого-цифрового преобразователя.

Вариант 13

Выполнил:

Студент группы КБ-211

Коренев Д.Н.

Принял:

Шамраев А.А.

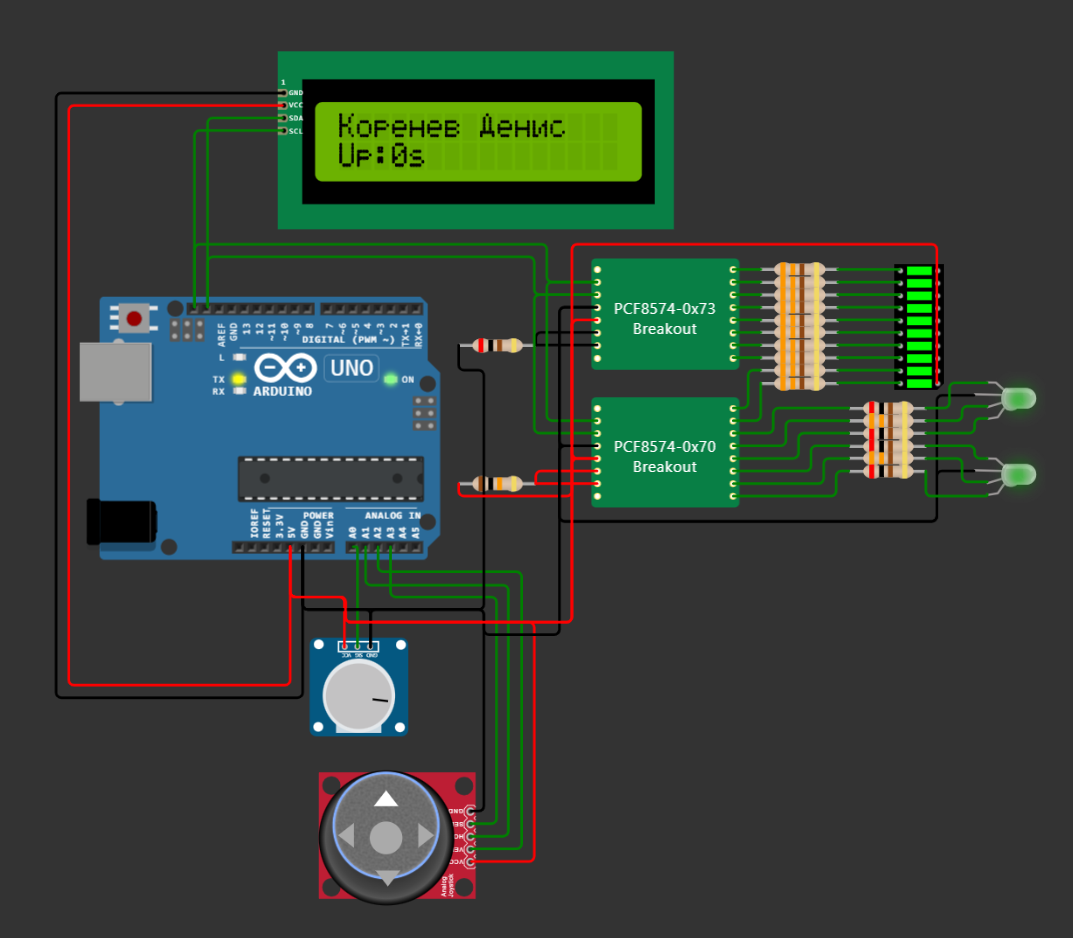
*Цель работы:* изучить принцип работы аналого-цифрового преобразователя (АЦП). Разработать алгоритм и программу соответствующие выбранному варианту.

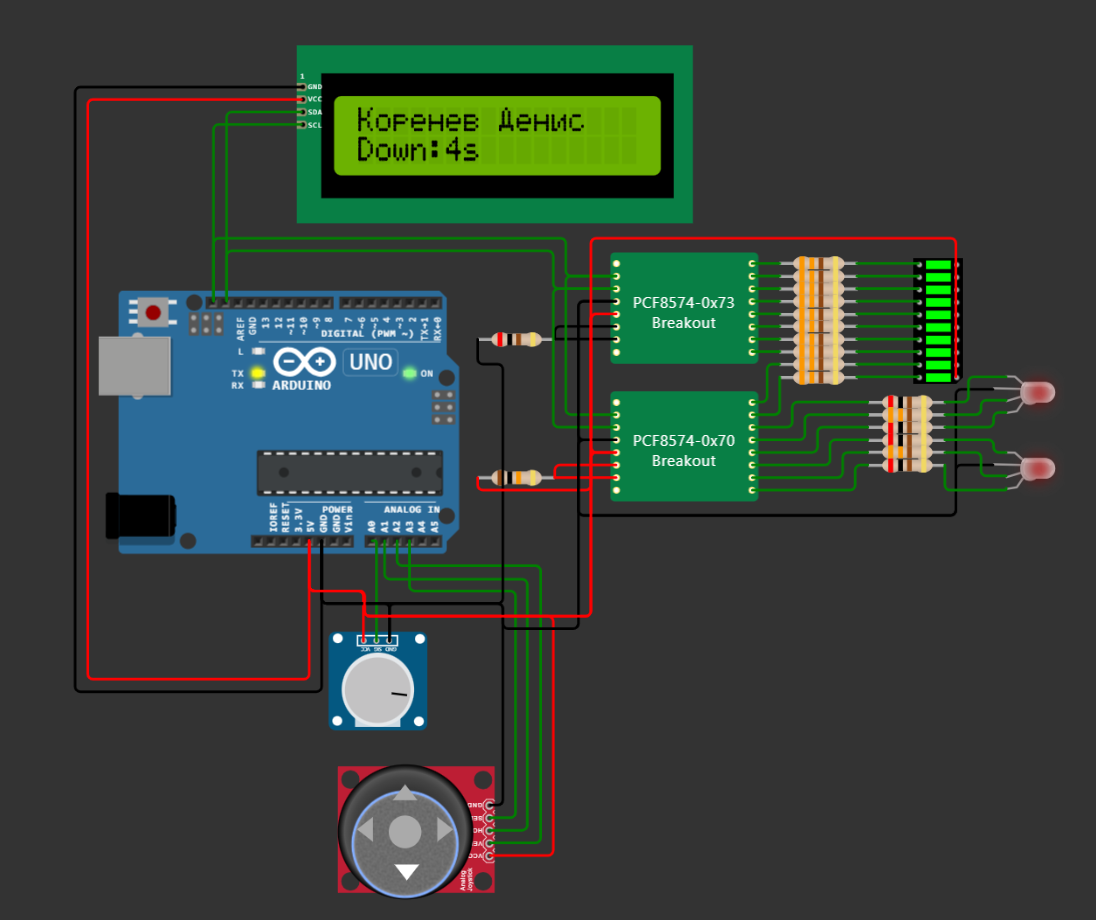
**Задание**

1. Используя стандартные функции для работы с АЦП, реализовать программу в соответствии с вариантом задания (номер в журнале % 2 + 1):

1) На первой строке ЖК-экрана выводить свое имя, фамилию, на второй строке положение ручки потенциометра в процентах.

2) При отклонении джойстика вниз зажигать красный светодиод, при отклонении вверх – зеленый, а при отклонении влево/вправо – мигать верхним/нижним желтым. На экране вывести свое имя, фамилию и считать время нахождения джойстика в каждом положении в секундах.





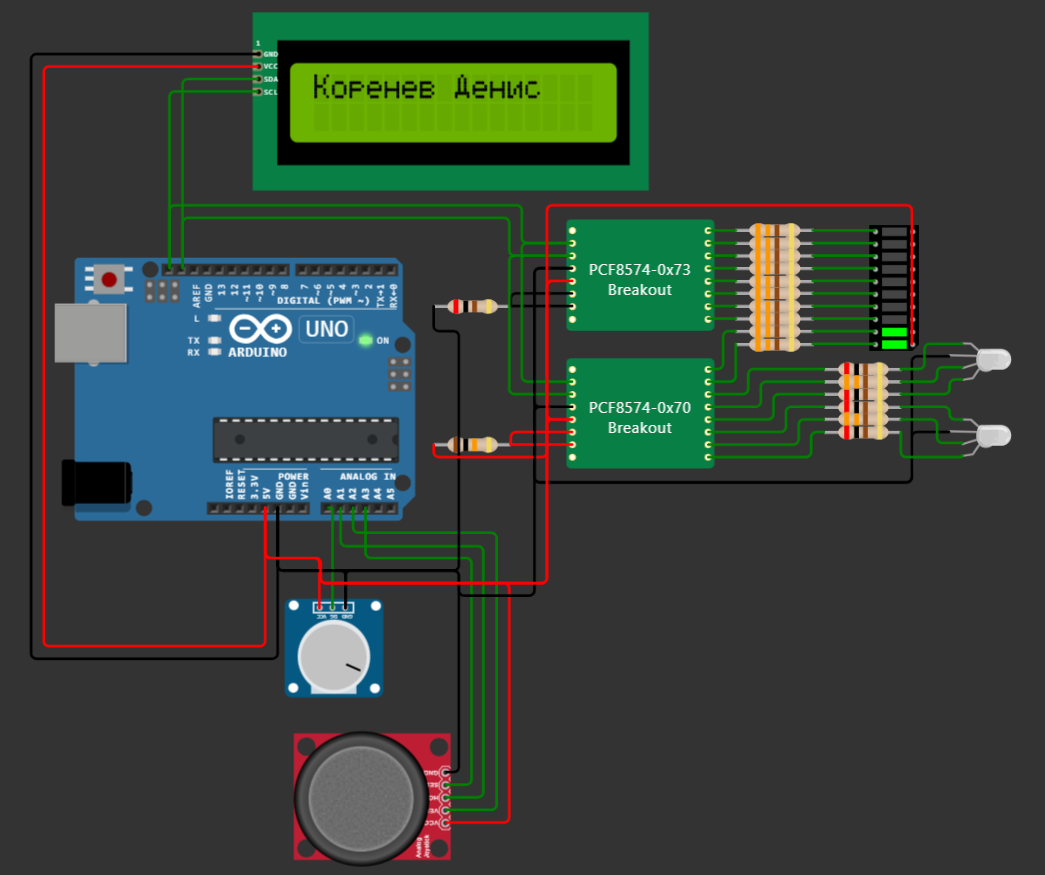
1. int xValue = analogRead(joyX);
2. int yValue = analogRead(joyY);
3. buttonState = digitalRead(button);
4. lcd.clear();
5. lcd.setCursor(0, 0);
6. lcd.write((uint8\_t)0);
7. lcd.print("ope");
8. lcd.write((uint8\_t)2);
9. lcd.print("e");
10. lcd.write((uint8\_t)1);
11. lcd.print(" ");
12. lcd.write((uint8\_t)4);
13. lcd.print("e");
14. lcd.write((uint8\_t)2);
15. lcd.write((uint8\_t)5);
16. lcd.print("c");
17. lcd.setCursor(0, 1);
18. unsigned long timeStart = millis();
19. delay(250);
20. if (yValue < 470)
21. {
22. // right
23. lcd.print("Right:");
24. lcd.print(String(timeRIGHT / 1000));
25. lcd.print("s");
26. timeRIGHT += 2.1 \* (millis() - timeStart);
27. }
28. else if (yValue > 550)
29. {
30. // left
31. // int timeStart = millis();
32. lcd.print("Left:");
33. lcd.print(String(timeLEFT / 1000));
34. lcd.print("s");
35. timeLEFT += 2.1 \* (millis() - timeStart);
36. }
37. else if (xValue < 470)
38. {
39. // up
40. lcd.print("Up:");
41. lcd.print(String(timeUP / 1000));
42. lcd.print("s");
43. timeUP += 2.1 \* (millis() - timeStart);
44. }
45. else if (xValue > 550)
46. {
47. // down
48. lcd.print("Down:");
49. lcd.print(String(timeDOWN / 1000));
50. lcd.print("s");
51. timeDOWN += 2.1 \* (millis() - timeStart);
52. }
53. else
54. {
55. // unpressed
56. lcd.print("                ");
57. timeUP = 0;
58. timeDOWN = 0;
59. timeLEFT = 0;
60. timeRIGHT = 0;
61. }
62. // При отклонении джойстика вниз зажигать красный светодиод, при отклонении вверх – зеленый, а
63. // при отклонении влево/вправо – мигать верхним/нижним желтым. На экране вывести свое имя,
64. // фамилию и считать время нахождения джойстика в каждом положении в секундах.
65. if (yValue < 470 || yValue > 550)
66. {
67. pca95xx\_out(0x70, pca95xx\_in(0x70) & RGB\_ALL\_REMOVE);
68. pca95xx\_out(0x70, pca95xx\_in(0x70) | RGB\_RED\_UP\_ADD | RGB\_GREEN\_UP\_ADD);
69. delay(250);
70. pca95xx\_out(0x70, pca95xx\_in(0x70) & RGB\_ALL\_REMOVE);
71. pca95xx\_out(0x70, pca95xx\_in(0x70) | RGB\_RED\_BOTTOM\_ADD | RGB\_GREEN\_BOTTOM\_ADD);
72. }
73. else if (xValue < 470)
74. {
75. pca95xx\_out(0x70, pca95xx\_in(0x70) & RGB\_ALL\_REMOVE);
76. pca95xx\_out(0x70, pca95xx\_in(0x70) | RGB\_GREEN\_UP\_ADD | RGB\_GREEN\_BOTTOM\_ADD);
77. }
78. else if (xValue > 550)
79. {
80. pca95xx\_out(0x70, pca95xx\_in(0x70) | RGB\_RED\_UP\_ADD | RGB\_RED\_BOTTOM\_ADD);
81. }
82. else
83. {
84. pca95xx\_out(0x70, pca95xx\_in(0x70) & RGB\_ALL\_REMOVE);
85. }
86. delay(100);

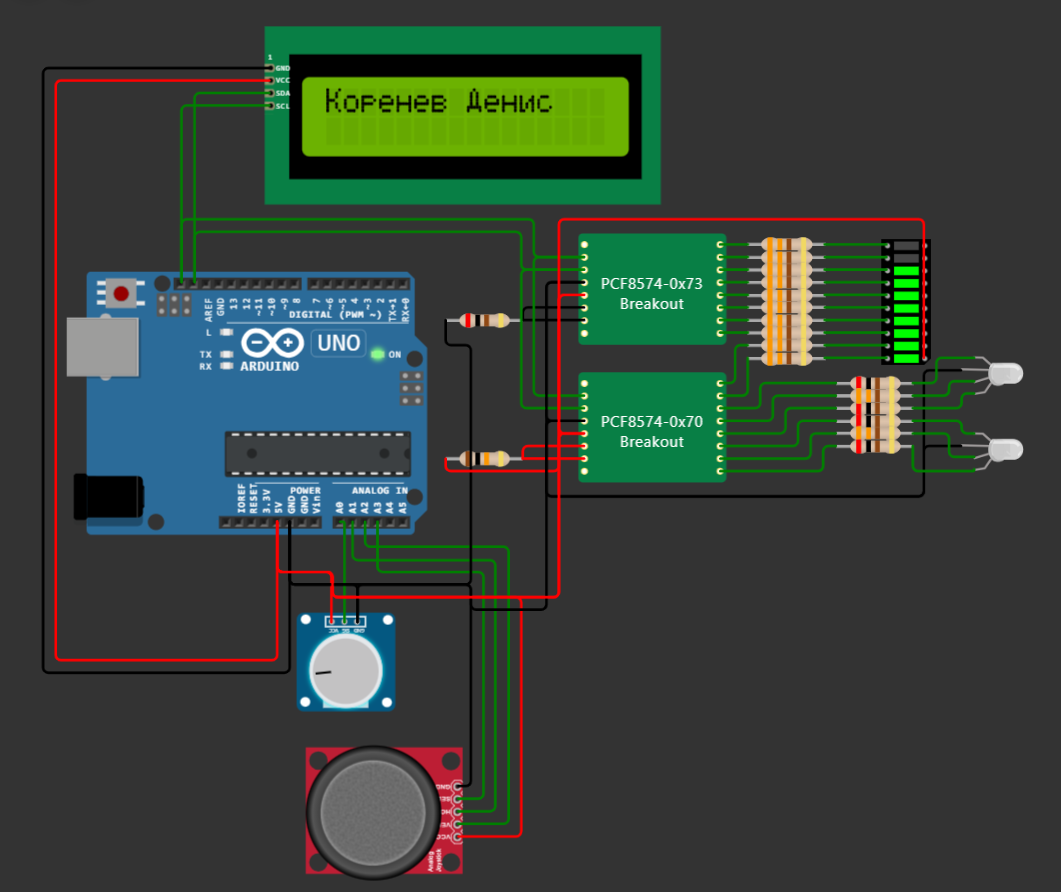
2. Используя стандартные функции для работы с АЦП, реализовать программу в соответствии с вариантом задания (номер в журнале % 6 + 1):

1) Вывести строку на ЖК-экран, перемещать её влево/вправо вращением ручки потенциометра

2-5) Отображать положение потенциометра на световой шкале, масштабируя в диапазоне от 0 до (номер варианта + 5)

6) Перемещать курсор на ЖК-экране с помощью джойстика влево/вправо. По нажатию на кнопку в джойстике выводить на текущем положении курсора первую букву своего имени.





1. int potentia = adc.read\_its();
2. int potentia\_mapped = map(potentia, 0, 1023, 0, 10);
3. displayLEDcolumn(potentia\_mapped);
4. delay(150);
5. }
6. void displayLEDcolumn(int value)
7. {
8. uint16\_t mask = 0b1111110000000000;
9. pca95xx\_out(0x73, pca95xx\_in(0x73) & (uint8\_t) mask);
10. pca95xx\_out(0x70, pca95xx\_in(0x70) & ((uint8\_t) (mask >> 8)));
11. uint16\_t toOutput = 0b0;
12. for (int i = value; i > 0; i--)
13. {
14. toOutput |= (1 << (10 - i));
15. }
16. toOutput ^= 0b1111111111;
17. pca95xx\_out(0x73, pca95xx\_in(0x73) | (uint8\_t) toOutput);
18. pca95xx\_out(0x70, pca95xx\_in(0x70) | ((uint8\_t) (toOutput >> 8)));
19. }

3. Взаимодействовать с АЦП напрямую с помощью регистров. Выполнить вариант задания из 2).

Проект Wokwi: <https://wokwi.com/projects/379824026508643329>

1. #ifndef ACD\_REG\_H\_\_
2. #define ACD\_REG\_H\_\_
3. class adc\_atmega328p {
4. public:
5. void begin(){
6. DDRC = B000000; // назначает выводы с 14 по 19 входными
7. PORTC = B000000; // устанавливает HIGH на выводах с 14 по 19 (на 19 LOW)
8. ADCSRA |= (1 << ADEN) // Включаем АЦП
9. | (1 << ADPS2)|(1 << ADPS1)|(1 << ADPS0); // устанавливаем предделитель преобразователя на 128
10. ADMUX |= (0 << REFS1)|(1 << REFS0) // выставляем опорное напряжение Vcc
11. | (1 << MUX0)|(1 << MUX1)|(0 << MUX2)|(0 << MUX3); // снимать сигнал будем с входа AC3
12. analogReference(DEFAULT);
13. }
14. int read\_its(){
15. auto pin = 0;
16. ADMUX = (pin & 0x07);
17. ADCSRA |= (0 << ADIF);
18. ADCSRA |= (1 << ADSC);
19. while ((ADCSRA & (1 << ADIF)) == 0); //ожидаем окончания преобразования
20. return (ADCL|ADCH << 8);
21. }
22. float read\_voltage(){
23. auto pin = 0;
24. ADMUX = (pin & 0x07);
25. ADCSRA |= (0 << ADIF);
26. ADCSRA |= (1 << ADSC);
27. while ((ADCSRA & (1 << ADIF)) == 0); //ожидаем окончания преобразования
28. int data = (ADCL|ADCH << 8); // Считываем полученное значение
29. float V = (float) data \* 0.0047031; // Переводим в вольты
30. /\* 0.0047031 = Vcc / 1024 \*/
31. return V;
32. }
33. };
34. #endif // ACD\_REG\_H\_\_
35. int potentia = adc.read\_its();
36. int potentia\_mapped = map(potentia, 0, 1023, 0, 10);
37. displayLEDcolumn(potentia\_mapped);
38. delay(150);

**Вывод**: в ходе лабораторной работы мы изучили принцип работы аналого-цифрового преобразователя (АЦП), разработали алгоритм и программу соответствующие выбранному варианту.