

Задание 1.

Возьмите плату Arduino UNO, кабель USB-A-USB-B, семисегментный индикатор, макетную плату, 8 резисторов на 510 Ом. Перед сборкой схемы обратите внимание на маркировку семисегментного индикатора. По маркировке определите, индикатор имеет схему включения с общим анодом или с общим катодом. Различия между данными типами семисегментных индикаторов описаны на рисунке 1.

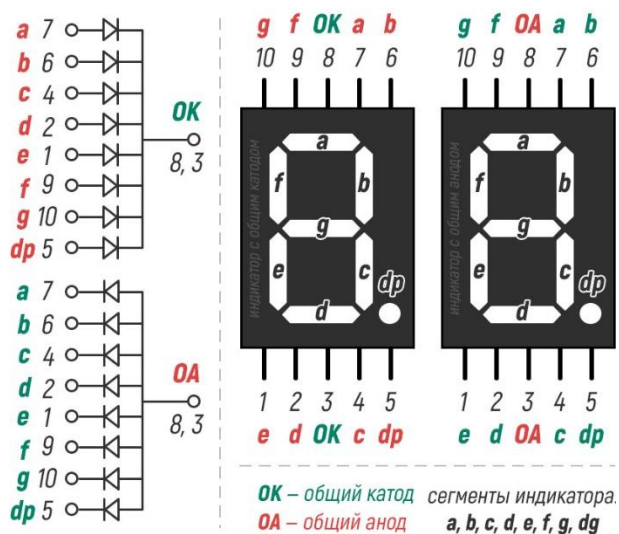


Рисунок 1 – Схемы индикаторов с общим катодом и с общим анодом

Как можно видеть, отличия заключаются в том, какие контакты светодиодов в индикаторе объединяются в один узел: аноды или катоды. В схеме с общим катодом аноды светодиодов выведены на контакты, а в общей точке соединяются катоды и светодиод включается с помощью подачи логической единицы. В схеме с общим анодом в общей точке соединяются аноды, поэтому включается светодиод при подаче на катод логического нуля.

Соберите схему, представленную на рисунках 2,3. Будьте внимательны, что при схеме с общим анодом к контактам 3,8 необходимо подключить напряжение питания +5 В, а при схеме с общим катодом – землю GND.

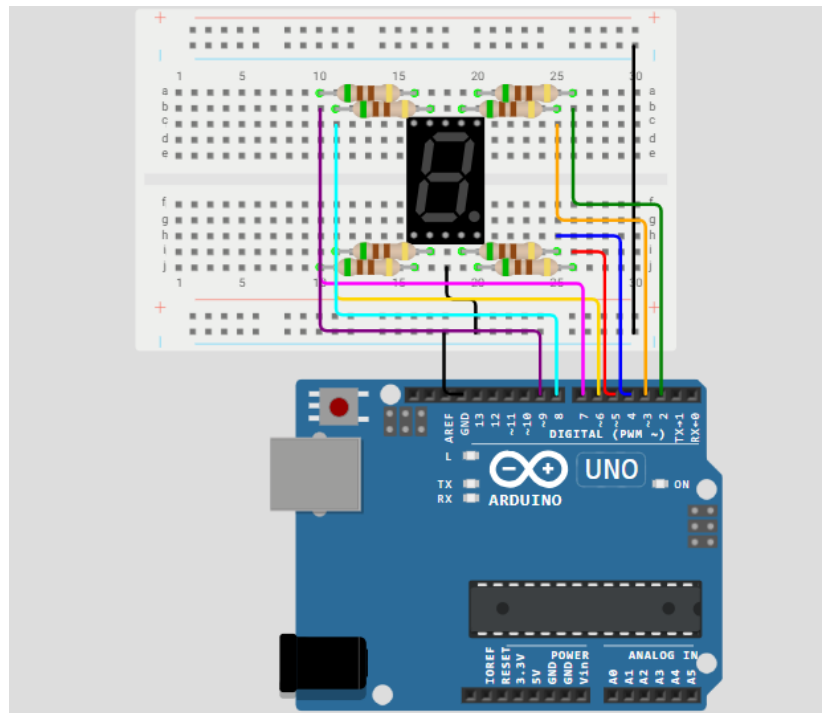


Рисунок 2 – Схема включения семисегментного индикатора с общим катодом

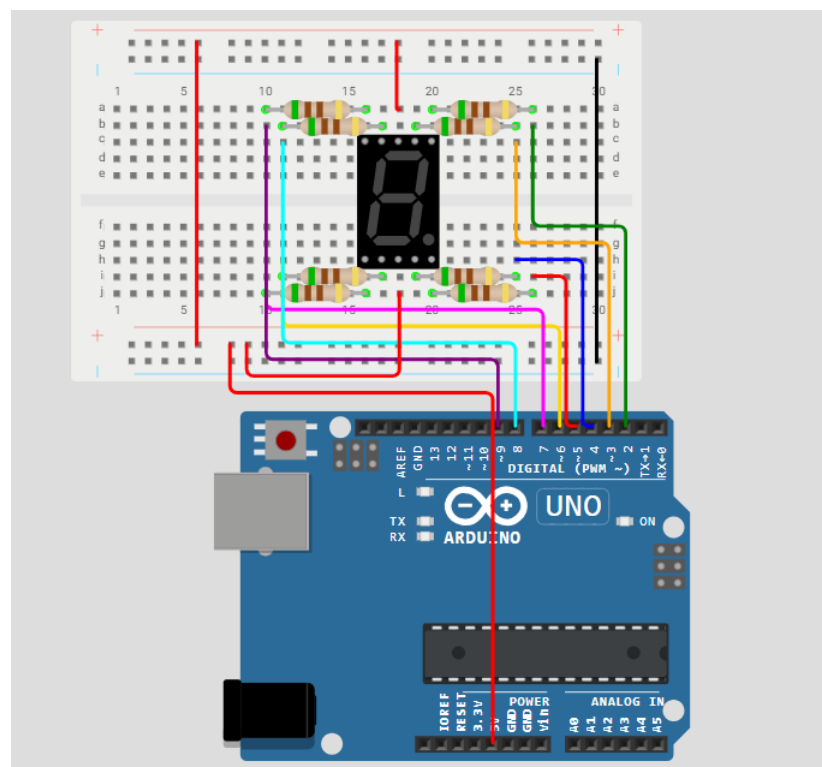


Рисунок 3 – Схема включения семисегментного индикатора с общим анодом

Программа для управления семисегментным индикатором схожа с программой для управления светодиодом, однако теперь таких светодиодов будет 8.

Для удобства написания программы и её лучшей читаемости введём макроопределения для наших выводов. Например, если сегмент А индикатора подключен ко 2 выводу, то можно оформить это следующим образом:

```
#define A_PIN 2
```

В таком случае, можно использовать в функциях `pinMode()` и `digitalWrite()` вместо цифры 2 макрос `A_PIN`, что позволяет не запутаться, какой из выводов микроконтроллера подключен к указанному выводу индикатора.

Пример программы инициализации вывода 2 и формирования на его выходе логической единицы представлен ниже:

```
#define A_PIN 2

void setup() {
  // put your setup code here, to run once:
  pinMode(A_PIN, OUTPUT);
  digitalWrite(A_PIN, HIGH)
}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Задание:

Сформируйте макросы для всех выводов, подключенных к индикатору, согласно примеру и инициализируйте их. Согласно рисунку 1 и схеме включения индикатора, реализуйте программу, которая выводит на индикатор цифру «0».

Покажите итог преподавателю.

Задание:

Разработайте программу, которая реализует поочередный вывод цифр от 0 до 3 в бесконечном цикле с задержкой в 1 секунду.

Покажите итог преподавателю.

Как можно наблюдать, код выглядит достаточно громоздким и если продолжать формировать цифры таким же образом в бесконечном цикле, то получится достаточно длинная программа. Для удобства, можно сформировать собственную функцию, которая будет выводить на индикатор цифру, в зависимости от переданного в неё аргумента (числа).

Например, для этого можно будет воспользоваться условным оператором. Прототип функции может выглядеть следующим образом:

```
void Indicate_Digit(char Digit)
{
    if(Digit == 0)
    {
        /*Программа для вывода цифры "0"*/
    }
    else if(Digit == 1)
    {
        /*Программа для вывода цифры "1"*/
    }
    /*И так далее*/
}
```

Будьте внимательны! Функция должна быть определена вне функций `void setup()` `void loop()`.

Тогда для вывода цифры «0» на индикатор в бесконечном цикле будет достаточно написать `Indicate_Digit(0)`.

Задание:

Завершите определение функции, чтобы ваша она была способна выводить цифры от 0 до 5. В бесконечном цикле выводите поочередно цифры от 0 до 5 с использованием вашей функции.

Покажите итог преподавателю.

Всё равно выглядит достаточно громоздко! Теперь это можно реализовать с помощью цикла, поместив функцию в цикл `for()`