

## Практична робота 2. Робота з RGB світлодіодом

**Мета:** дослідити роботу RGB світлодіода, закріпити навички роботи з цифровими портами, тактовими кнопками, формуванням ШІМ; навчитися програмувати режими роботи Arduino з використанням масивів.

**Завдання:** створити програму керування RGB світлодіодом з використанням тактових кнопок, зовнішніх переривань, АЦП та ШІМ лабораторного макету / віртуального стенду «Arduino Learner Kit» Arduino.

**Обладнання:** лабораторний макет/віртуальний стенд «Arduino Learner Kit»; USB – кабель; провідники-з'єднувачі.

### Загальні відомості

Масив — це набір змінних, доступ до яких здійснюється за індексом. Для роботи з масивом його потрібно створити (оголосити). Способи створення (оголошення) масивів:

```
int myInts[6];  
int myPins[] = {2, 4, 8, 3, 6};  
int mySensVals[6] = {2, 4, -8, 3, 2};  
char message[6] = "hello";
```

Можна оголосити масив без його ініціалізації, як *myInts*. У *myPins* оголошений масив без прямої вказівки його розміру. Компілятор сам порахує елементи та створить масив відповідного розміру. При створенні (ініціалізації) масиву можна вказати його розмір, як *mySensVals*. При оголошенні масиву типу *char*, в ньому необхідно місце для зберігання обов'язкового нульового символу, тому розмір масиву повинен бути на один символ більше.

Нумерація елементів масиву починається з нуля, тобто перший елемент масиву має індекс 0. Так *mySensVals[0] == 2*, *mySensVals[1] == 4*.

Це також означає, що в масиві з 10 елементів, останній елемент має індекс 9. Отже:

```
int myArray[10]={9,3,2,4,3,2,7,8,9,11};  
// myArray[9] дорівнює 11  
// myArray[10] буде помилка
```

Тому, необхідно бути уважним при зверненні до масивів. Звернення до елемента за межами масиву (коли зазначений індекс більше, ніж оголошений розмір масиву – 1) призведе до читання даних з комірки пам'яті, що використовується для інших цілей. Зчитування з цієї області призведе до збою в роботі програми. На відміну від BASIC або JAVA, компілятор C не перевіряє правильність індексів при зверненні до елементів масиву.

Запис значення до масиву — *mySensVals[0] = 10*. Зчитування запису з масиву — *x = mySensVals[4]*.

Робота з масивами часто здійснюється всередині циклів FOR, в яких лічильник циклу використовується як індексу кожного елемента масиву. Наприклад, програма налаштування режимів роботи портів Arduino може виглядати так:

```
const byte rgbPins[3] = {11,10,9};
void setup() {
    for( byte i=0; i < 3; i++ )
        pinMode( rgbPins[i], OUTPUT );
}
```

В одномірних масивах елементи визначаються просто порядковим номером. У двовимірних масивах (матриця або таблиця) кожен елемент має номер рядка та стовпця. Задається такий масив ось так:

```
// двовимірний масив, 5 рядків 10 стовпців
byte myTable [5][10] ;
// матриця 3x3
byte myMatrix [3][3] = {
    { 10, 11, 12 } ,
    { 13, 14, 15 } ,
    { 16, 17, 18 } ,
} ;
```

Після останнього члена масиву можна ставити кому, це не призведе до помилки (приклад коду вище). У розглянутому вище двовимірному масиві елемент з адресою 0, 2 (рядок 0 стовпець 2) *myMatrix [0] [2]* має значення 12.

Дребезг контактів — це явище, що відбувається в електромеханічних пристроях (кнопках, реле, герконах, перемикачах, контакторах), що триває

деякий час після замикання електричних контактів. Після замикання (натискання кнопки, включення реле і т.д.) відбуваються багаторазові неконтрольовані замикання та розмикання контактів за рахунок пружності матеріалів та деталей контактної системи. Перехідні процеси протікають дуже швидко (від 0,5 до декількох сотень мілісекунд). Тому їх не помічаємо, наприклад, коли включаємо світло в кімнаті. Лампа розжарювання не може змінювати свою яскравість з такою швидкістю. Але, обробляючи сигнал від кнопки на швидкому пристрої, як Arduino, повинні враховувати при програмуванні це явище.

Найпростішим способом боротьби з дребезгом кнопки є витримування паузи. Для цього робимо паузу 10-50 мілісекунд, як наведено у прикладі програми нижче.

```
int currentValue, prevValue;
void loop ( ) {
    currentValue = digitalRead (PIN_BUTTON) ;
    if ( currentValue != prevValue ) {
        delay ( 10 ) ;
        currentValue = digitalRead (PIN_BUTTON) ;
    }
    prevValue = currentValue;
    Serial.println (currentValue) ; }
```

Проблема з дребезгом настільки актуальна, що є спеціальні бібліотеки, в яких не потрібно організовувати очікування та паузи вручну — це все робиться всередині спеціального класу. Приклад популярної бібліотеки для боротьби з дребезгом кнопок — бібліотека Bounce.

Більш правильним способом боротьби з дребезгом є використання апаратного рішення, що згладжує імпульси з кнопки. Апаратний спосіб усунення дребезгу заснований на використанні RC фільтру або тригера Шмітта [1].

RGB світлодіод на схемі лабораторного макету позначений HL7 (рис. 1.5) та підключений за схемою з загальним катодом. Для його з'єднання з Arduino Nano використовується з'єднувач X6 (рис. 1.11).

### **Хід виконання роботи**

1. Підключити схему до комп'ютера через USB порт плати Arduino та/або запустити віртуальний стенд у середовищі Proteus 8.
2. Завантажити програму RGB1 (додаток Г) до лабораторного макета / віртуального стенду, попередньо виконати з'єднання RGB світлодіода у відповідність до програми. Дослідити роботу програми.
3. Завантажити програму RGB2 (додаток Г) до лабораторного макета / віртуального стенду. Дослідити роботу програми.
4. Завантажити програму RGB3 (додаток Г) до лабораторного макета / віртуального стенду. Дослідити роботу програми.
5. Завантажити програму RGB4 (додаток Г) до лабораторного макета / віртуального стенду. Дослідити роботу програми.
6. Завантажити програму RGB5 (додаток Г) до лабораторного макета / віртуального стенду, попередньо виконати з'єднання RGB світлодіода та тактової кнопки у відповідність до програми. Дослідити роботу програми.

### **Завдання**

1. Внести зміни до програми RGB5 (додаток Г). Для опрацювання моментів натиснення кнопки використати зовнішнє переривання. Для вибору режиму роботи RGB світлодіода використати оператор SWITCH.
2. Реалізувати програму з застосуванням переривань INT0, INT1. При натисканні кнопки, що підключена до INT0 змінюється колір світлодіода, а при натисканні кнопки, що підключена до INT1 змінюється частота мигання світлодіода даним кольором.
3. Реалізувати програму, у якій колір RGB світлодіода змінюється потенціометром RV1.

**Підготувати звіт** згідно ДСТУ 3008-95 (лістинг програми, висновки, перелік посилань).

### Контрольні питання

1. Що таке дребезг контактів? Як програмно усунути його?
2. Як апаратно усунути дребезг контактів? Наведіть практичні схеми.
3. Для чого використовується бібліотека Bounce? Наведіть приклад її застосування
4. Реалізуйте програму «вогонь, що біжить» з використанням одномірного масиву.
5. Реалізуйте програму «вогонь, що біжить» з використанням двомірного масиву.