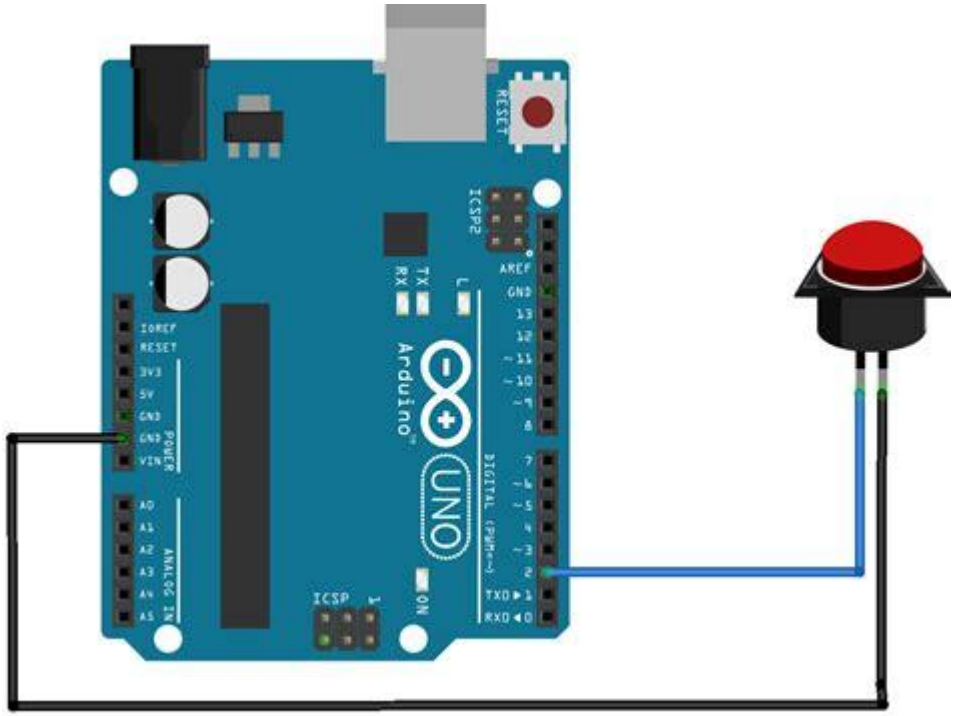
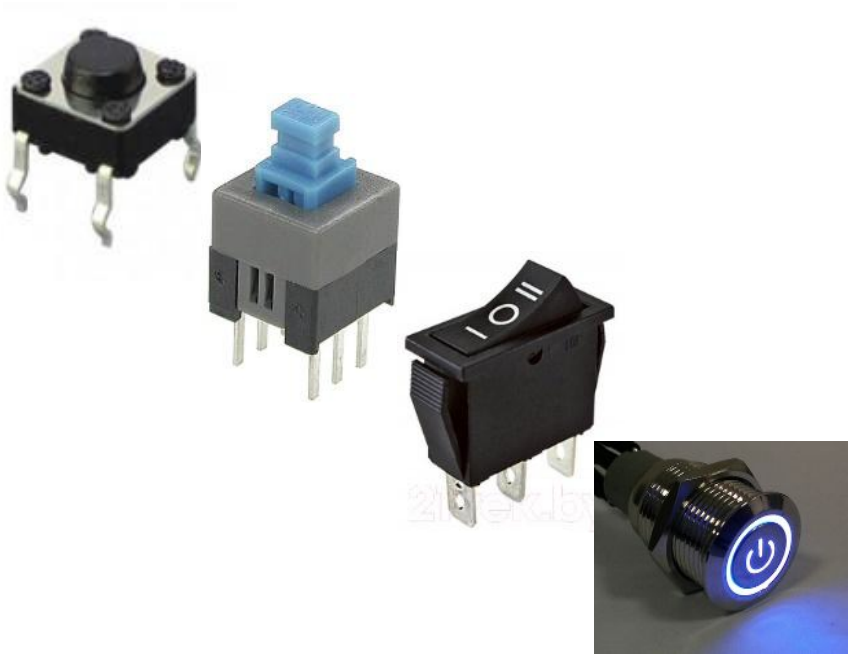
Подключение кнопки к Ардуино



Виды кнопок

- **Тактовые(без фиксации)**
- **С фиксацией**
- **Многопозиционные**
- **Индикаторные**

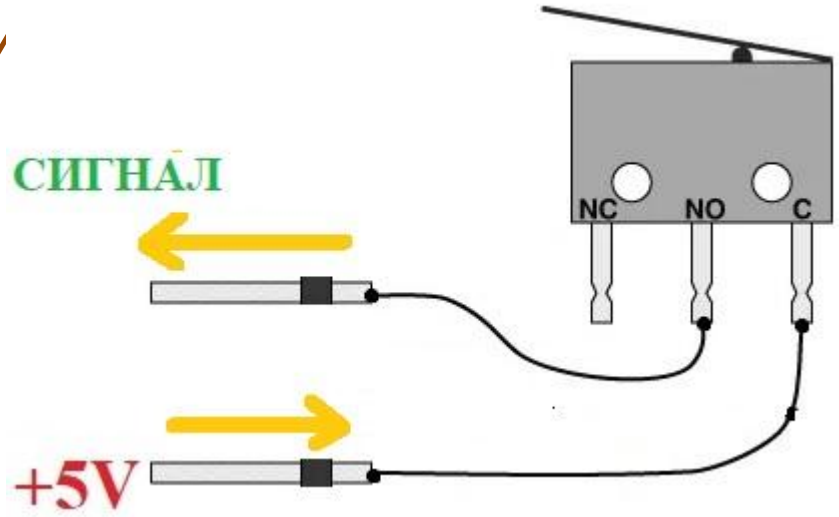


Кнопка выполняет простую функцию - замыкает и размыкает контакт

Кнопка является простейшим датчиком её можно использовать как:

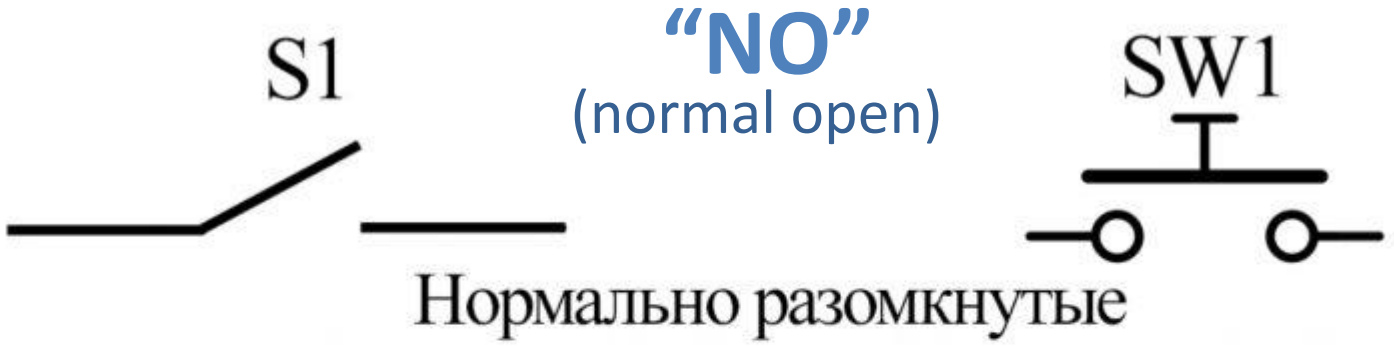
1. *Управляющий элемент (клавиатура);*
2. *Датчик препятствия(столкновения);*
3. *Датчик открытия/закрытия (двери, окна, корпуса);*
4. *Датчик предельного угла*

М



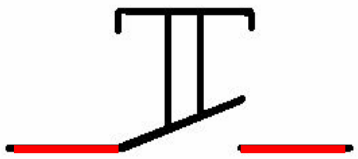
Различие по логике работы:

- ***Нормально разомкнутые(открытые)*** **«НО»**
- ***Нормально замкнутые*** **«НЗ»**

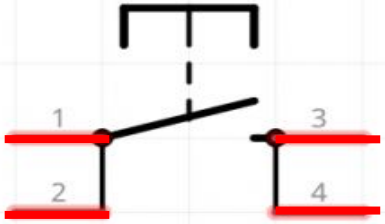


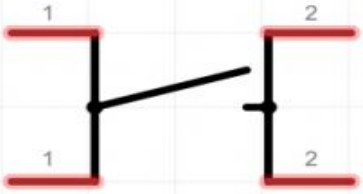
Обозначения на схемах:

Одна контактная группа:

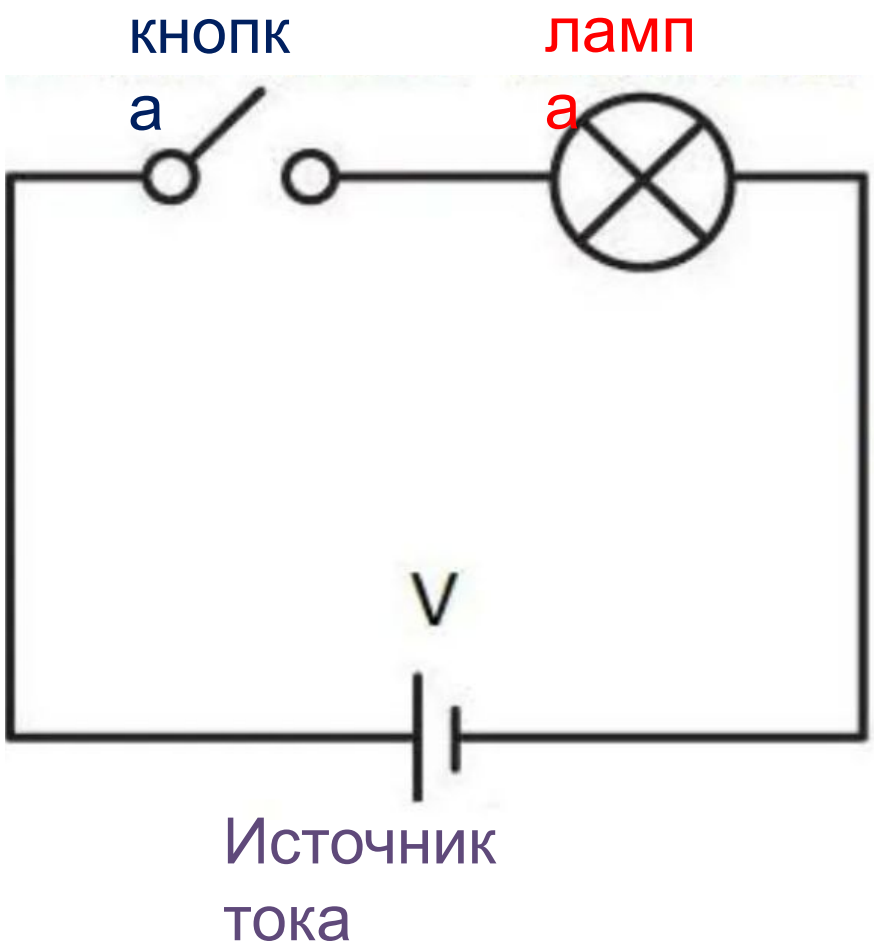


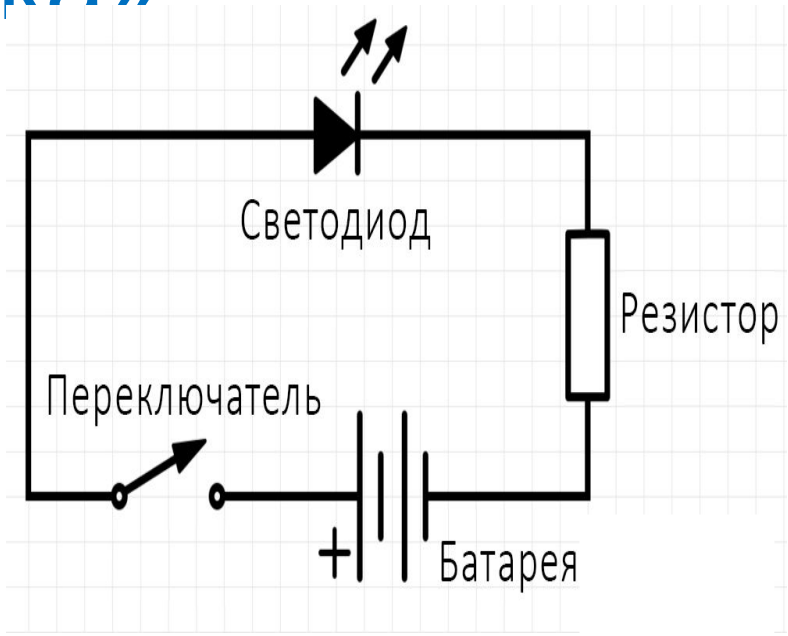
Две контактных группы:

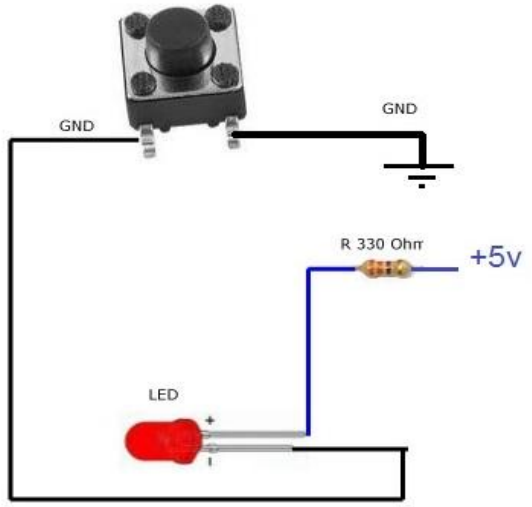




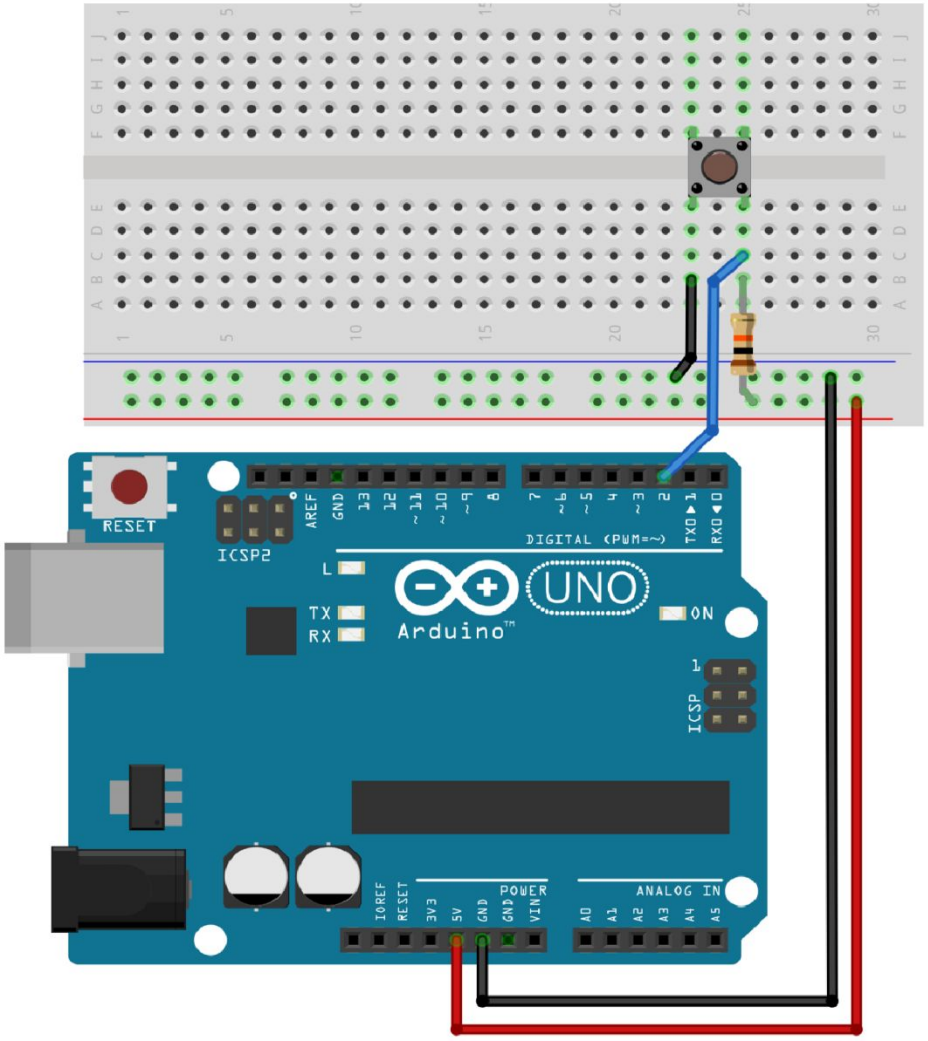
Цепь с кнопкой - «схема фонарика»



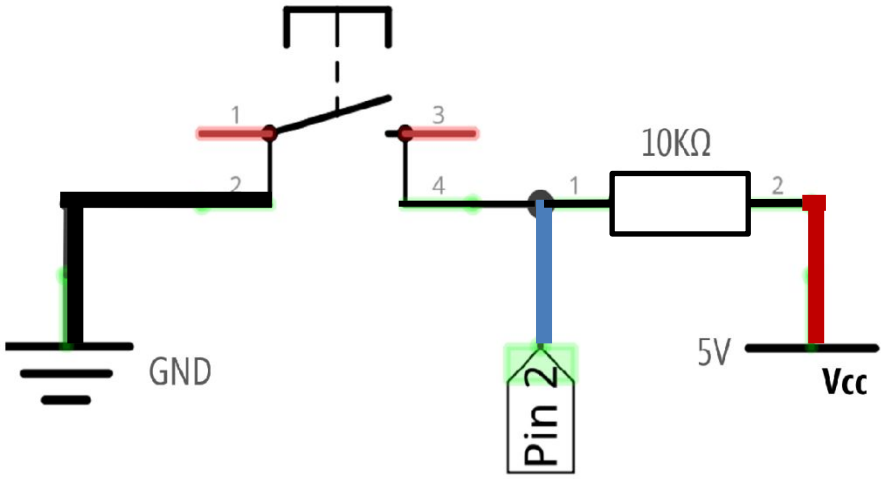




Пример подключения кнопки к Arduino через монтажную плату



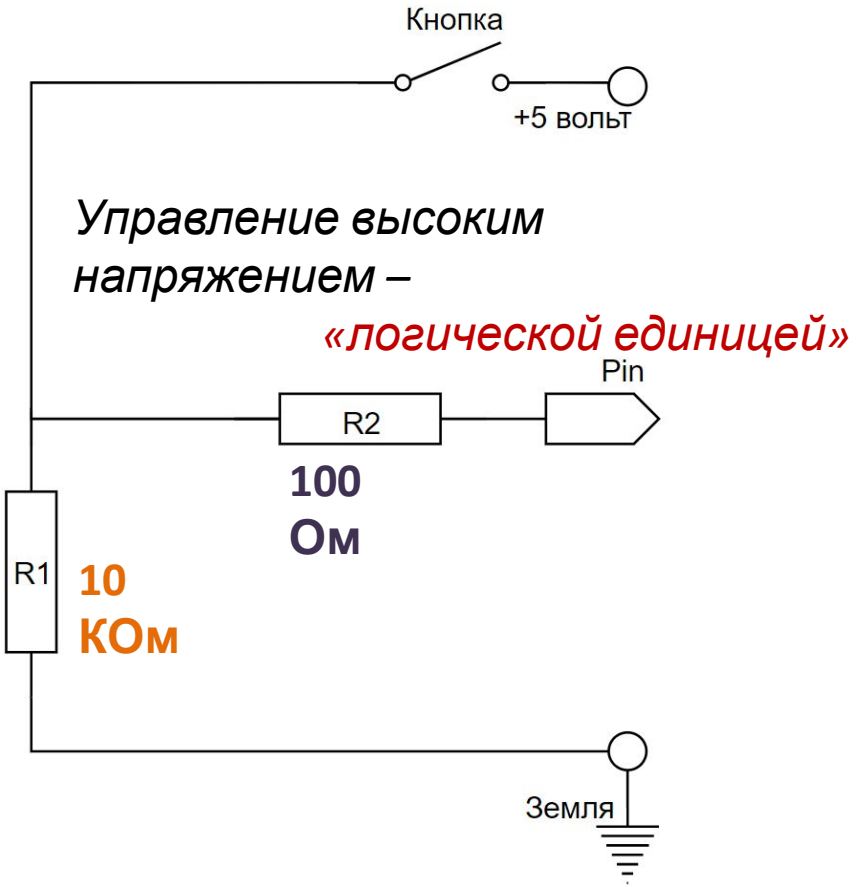
Принципиальная схема цепи (небезопасная)



Такое подключение ***ОПАСНО*** *возникновением* ***КОРОТКОГО ЗАМЫКАНИЯ*** *при нажатии кнопки, если пин окажется настроен на вывод*

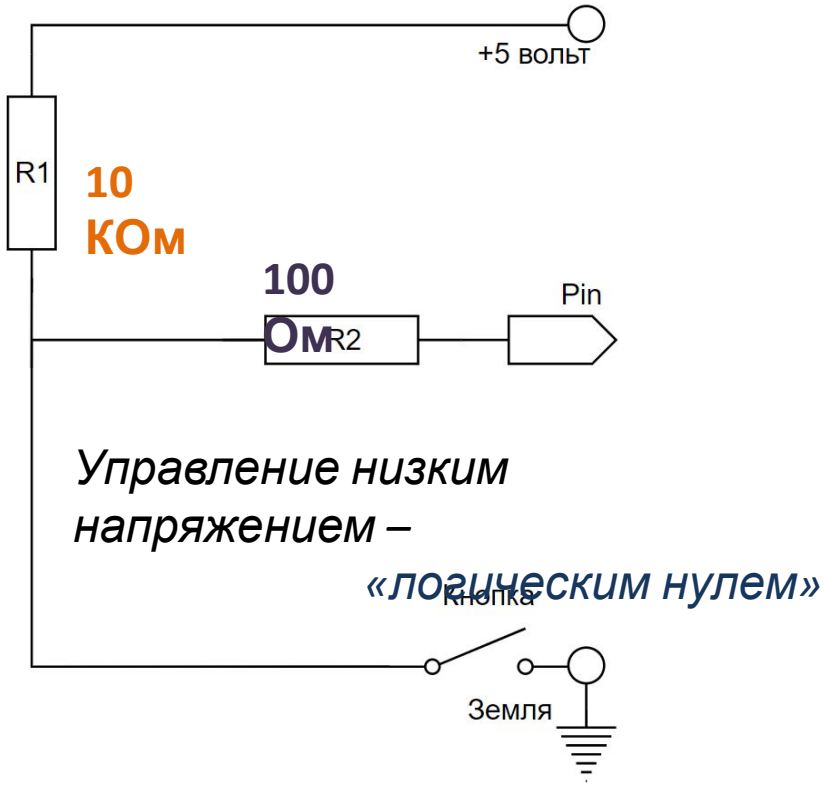
Защита от короткого замыкания ***токоограничивающим резистором*****.**

И от случайных шумов с помощью ***подтягивающего резистора*****.**



R1 - Подтягивающий резистор к земле

R2 - Токоограничивающий резистор 100 Ом - 10 кОм



R1 - Подтягивающий резистор к питанию или стягивающий резистор

R2 - Токоограничивающий резистор 100 Ом - 10 кОм

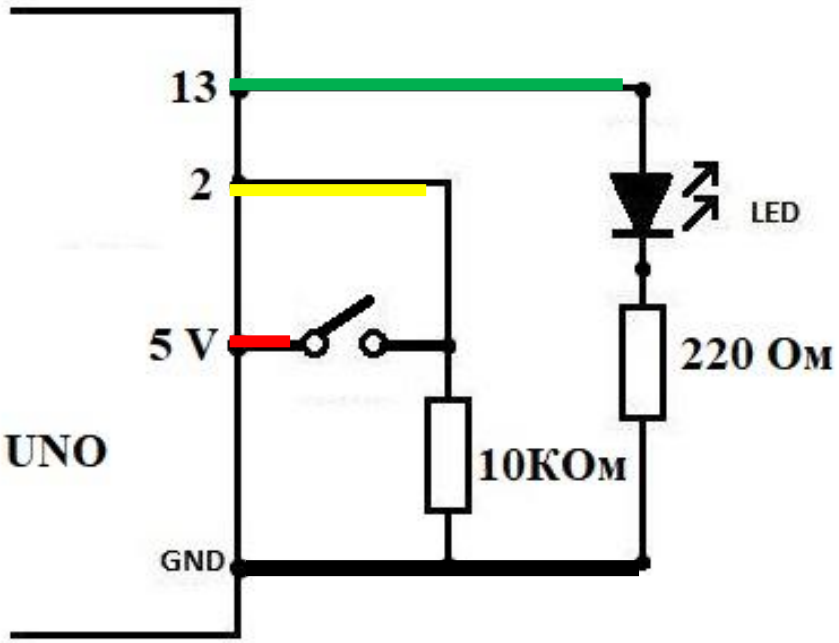
Настройка порта Ардуино на подключение кнопки

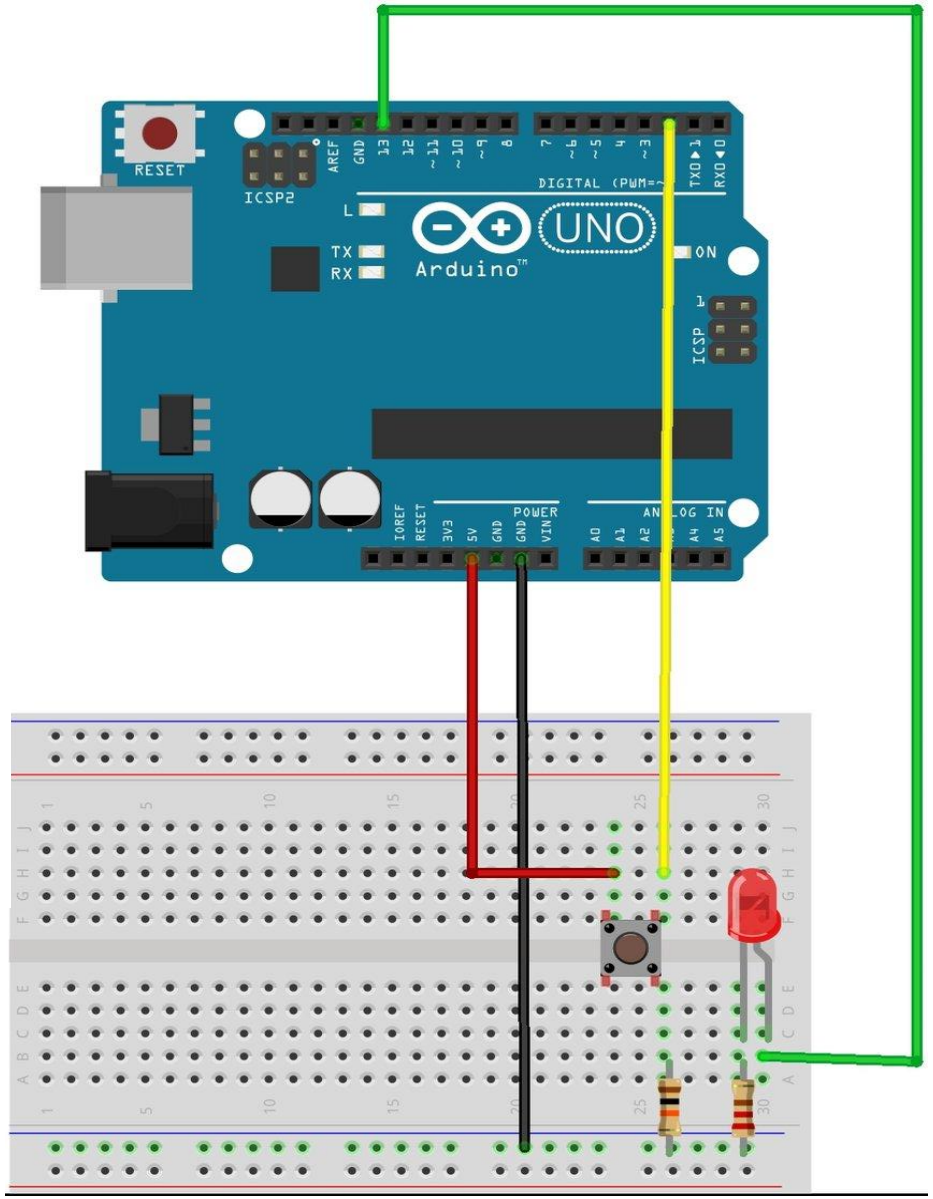
*Чтобы порт Ардуино работал **«на вход»***
*его необходимо настроить командой **«pinMode»**:*

```
pinMode(пин, INPUT);
```

*Значение второго параметра «**INPUT**»*
*задает **режим чтения** состояния пина,*
это позволяет роботу
***получать информацию** от датчиков.*

Управление кнопкой внешним светодиодом через Ардуино





```
const int buttonPin = 2;  // сохраняем номер пина с кнопкой в переменной
const int ledPin = 13;      // сохраняем номер пина светодиода в переменной

int buttonState = 0;        // переменная для хранения состояния кнопки

void setup() {  //  блок предстартовых настроек программы

  pinMode( ledPin, OUTPUT);    // настраиваем пин светодиода на выход

  pinMode( buttonPin, INPUT); // настраиваем пин кнопки на вход

}

void loop() {   //  основной цикл программы

  buttonState  = digitalRead( buttonPin ); // считываем состояние кнопки

  if ( buttonState == HIGH )  {     //  если кнопка нажата, то:
    digitalWrite( ledPin, HIGH ); // зажечь светодиод (подать на пин 5V)
  }
  else   { // иначе ( если кнопка не нажата и buttonState = LOW ):
    digitalWrite( ledPin, LOW );  // выключить светодиод ( 0V на пин)
  }
```

«**buttonPin**», «**ledPin**», «**buttonState**» - это переменные

Переменная *– это именованная область в памяти микроконтроллера, где может храниться информация: число, символ, строка, массив, структура и т.д.*

```
int buttonState = 0;
```

/ объявление числовой переменной целого типа, с присвоением ей значения «0» */*

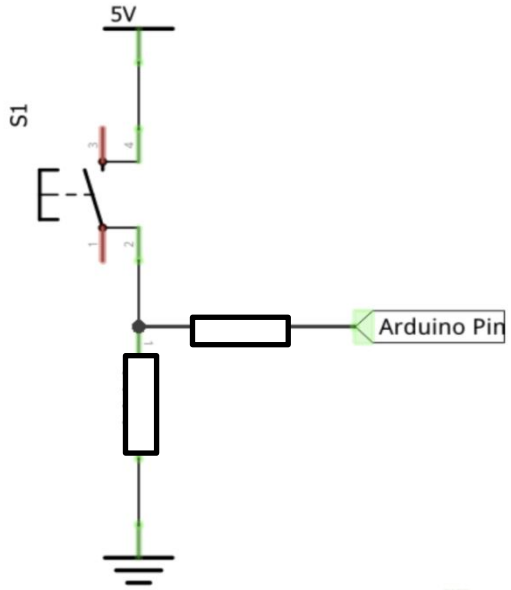
Переменные типа **int** *(от англ. “integer”) занимают в памяти* **16 бит** (**2 байта**) *информации и могут принимать только целые значения: от* **-32768** *до* **32767**

```
void setup(){…}    /*  Блок стартовых настроек.
                   Команды в фигурных скобках
                   выполнятся всего один раз.
                   До запуска основного цикла */

void loop(){…}     /* Основной цикл программы.
                   Команды в скобках
                   будут выполняться бесконечно */
```

digitalRead(номер пина);

// получить информацию о состоянии цифрового порта



*На порт Ардуино через нажатую кнопку подается, либо **высокое**, либо **низкое** напряжение.*

Это напряжение читается микроконтроллером как

«Логическая Единица» «HIGH» «TRUE»

при напряжении

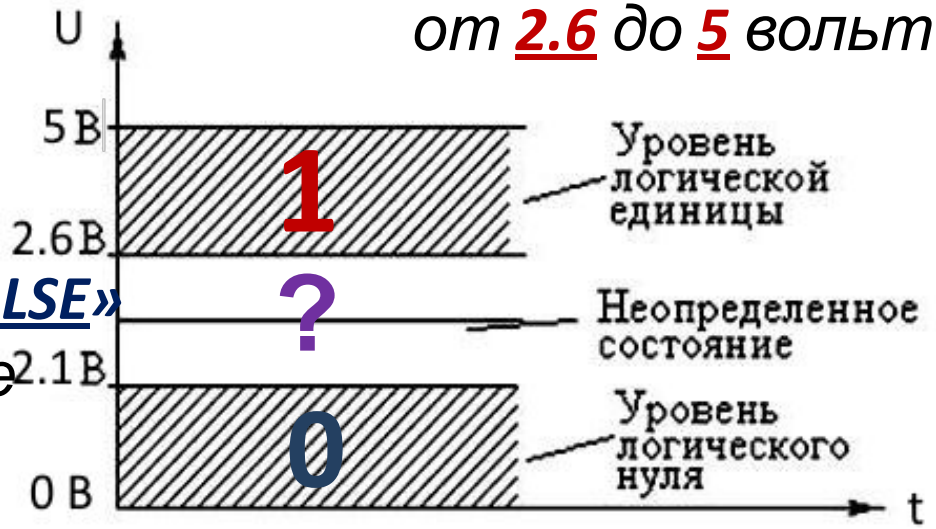
*от **2.6** до **5** вольт*

или как

«Логический Ноль» «LOW» «FALSE»

когда напряжение в диапазоне

*от **0** до **2.1** вольт*



Команда управления состоянием цифрового пина

```
digitalWrite( номер пина, HIGH );
// установить на цифровом пине напряжение 5V
```

```
digitalWrite( номер пина, LOW );
// установить на цифровом пине напряжение 0V
```

Выходы Ардуино имеют ограничение по силе тока в 40мА (40 миллиампер).
Более мощная нагрузка приведет к выходу из строя пина или микроконтроллера.
Поэтому нельзя подключать напрямую к портам моторы и прочие приборы с потреблением тока больше 40мА!

Условный оператор или «оператор ветвления»

```
if ( условие ) {
    // этот блок команд выполнится, если условие истинно
}
else {
    // этот блок команд выполнится, когда условие ложно
}
```

Двойное равно «==», оператор не присваивания, а **СРАВНЕНИЯ**. Если левая и правая части равны, то результатом его работы будет **TRUE**(истина). Если не равны, то **FALSE**(ложь).

Другие операторы сравнения: **> < >= <= !=**

Функция millis()

millis() – возвращает **время, прошедшее с момента запуска программы**

Тип возвращаемого значения:

unsigned long = { **1** … **4 294 967 295** } (мс)

После “переполнения” (~50 суток), отсчёт начинается с нуля.

Arduino Nano

BlinkWithoutDelay.ino

```
const int ledPin = LED_BUILTIN;  // номер пина со светодиодом

int ledState = LOW;  // переменная состояния светодиода

unsigned long previousMillis = 0;  // переменная для хранения времени последнего моргания

const long interval = 1000;  // интервал смены состояний свтодиода (milliseconds)

void setup() {
  // сконфигурировать пин со светодиодом на вывод
  pinMode(ledPin, OUTPUT);
}

void loop() {

  unsigned long currentMillis = millis(); // получить текущее время микроконтроллера

  if (currentMillis - previousMillis >= interval) {
    // сохранить время предыдущей смены состояния светодиода
    previousMillis = currentMillis;

    // если светодиод не горит, то зажечь и наоборот
    if (ledState == LOW) {
      ledState = HIGH;
    }
    else {
      ledState = LOW;
    }

    // set the LED with the ledState of the variable:
    digitalWrite(ledPin, ledState);
  }
}
```