**Отчет**

**Лабораторная работа №6**

**Лобанов Владимир, 2 курс 3 группа**

Дребезг контактов – это явление, возникающее в механических и электронных переключателях, при котором они вместо некоторого стабильного переключения производят случайные многократные неконтролируемые замыкания и размыкания контактов (происходит в момент переключения, приблизительно в течение 40—100 мс).

Решить проблему дребезга контактом можно двумя способами:

* Аппаратным
* Программным

Суть программного способа в том, чтобы внутри программы отличить случайные срабатывания из-за дребезга от настоящего нажатия. Самый простой способ: добавить задержку сразу после первого срабатывания. Тогда при срабатывании кнопки мы останавливаем выполнение программы и дожидаемся, пока контакты кнопки перестанут дребезжать. Ложных срабатываний не будет.

byte led=0; //Состояние светодиода

byte oldled=1; //Последнее состояние светодиода, для исключения ложных переключений

void setup()

{

pinMode(13, OUTPUT); pinMode(2, INPUT);

}

void loop()

{

if(digitalRead(2)==HIGH) { //Когда нажата кнопка delay(100); //Защита от дребезга

if (led==oldled) { //Проверка, что состояние кнопки изменилось led=!led;

}

} else { oldled=led;

}

digitalWrite(13,led); //Переключаем светодиод

}

Следующий вариант избавления от дребезга, засекать время, прошедшее

от первого срабатывания кнопки с помощью функции millis:

byte led=0; //Состояние светодиода

byte oldled=1; //Последнее состояние светодиода, для исключения ложных переключений long previousMillis = 0; //Время последнего нажатия кнопки

void setup()

{

pinMode(13, OUTPUT); pinMode(2, INPUT);

}

void loop()

{

if(digitalRead(2)==HIGH) { //Когда нажата кнопка

//Если от предыдущего нажатия прошло больше 100 миллисекунд if (millis() - previousMillis &amp;amp;amp;amp;gt; 100) {

//Запоминаем время первого срабатывания previousMillis = millis();

if (led==oldled) { //Проверка, что состояние кнопки изменилось led=!led;

}

}

} else { oldled=led;

}

digitalWrite(13,led); //Переключаем светодиод

}

Еще одним способом избавлением от дребезга является использование

библиотеки Bounce. В ней уже все предусмотрено, и для чтения значения на входе используются специальные функции:

// подключение библиотеки Bounce #include "Bounce.h";

// Создание объекта Bounce.

// В качестве параметров конструктору передаются номер пина кнопки

// и время дребезга в миллисекундах. Bounce bouncer = Bounce(2,5);

byte led=0; //Состояние светодиода

byte oldled=1; //Последнее состояние светодиода, для исключения ложных переключений

void setup() { pinMode(13,OUTPUT); pinMode(2,INPUT);

}

void loop() {

// Опрос изменения состояния кнопки if ( bouncer.update() ) {

if ( bouncer.read() == HIGH) { //Когда нажата кнопка

if (led==oldled) { //Проверка, что состояние кнопки изменилось led=!led;

}

} else { //Когда не нажата oldled=led;

}

}

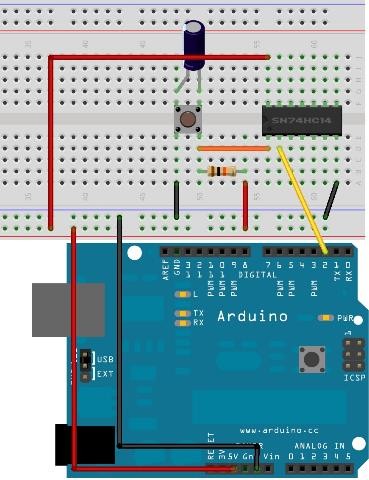
digitalWrite(13,led); //Переключаем светодиод

}

Аппаратные решения всегда ценились за их надёжность и возможность облегчить написание программы.

Добавив пару дополнительных элементов в схему, мы сможем полностью избавиться от дребезга, не усложняя при этом код. А добавив инвертирующий триггер Шмитта и конденсатор, мы построим RC-цепь с инвертирующим триггером.

Вот так это выглядит на схеме:



Резистор 10 кОм у нас теперь подтягивает вывод не к земле, а к +5В. Причём подтягивается не вход Ардуино, а вход инвертирующего триггера и заодно, конденсатор. Кнопка же, наоборот, подключена к земле. Сделано это в связи с тем, что триггер у нас инвертирующий. Суть такова: входящий аналоговый сигнал может быть восходящим, либо нисходящим. Внутри триггера определены пороговые значения: ~1,6 В — для нисходящего сигнала и ~0,9 В — для восходящего. Выход триггера становится логической единицей (5 В), при прохождении нисходящим сигналом через свой порог в 0,9 В, при этом нисхождение восходящим сигналом верхнего порога в 1,6 В будет проигнорировано. Аналогично, выход триггера становится логическим нулём (0 В), только когда восходящий сигнал проходит свой порог в 1,6 В, при этом про игнорируется прохождение нисходящим сигналом нижнего порога в 0,9 В. Зона неопределённости между порогами называется гистерезисом. То есть триггер переворачивает сигнал. Конденсатор с резистором образует RC-цепь. RC — от английских названий резистора и конденсатора (resistor и capacitor). RC-цепь замедляет затухание сигнала. То есть при снятии напряжения, оно пропадёт не сразу, а будет плавно снижаться. При нажатии кнопки мы прижмём RC-цепь к земле, конденсатор начнёт разряжаться и напряжение начнёт плавно падать.