Всем привет, в этом выпуске цикла уроков ардуино я расскажу вам про расширенную обработку нажатия кнопки.

В прошлом выпуске мы остановились на том, что включали кнопкой светодиод на плате, на 13 цифровом пине. Пока кнопка была нажата, светодиод горел, как только отпускали кнопку, он гас.

Давайте сделаем так, чтобы кнопка переключала состояние светодиода. То есть нажал отпустил, зажёгся, нажал отпустил - погас. Вспомните, цикл луп бесконечный и крутится тысячи раз в секунду. А нам нужно каким то образом выполнить одно действие и чтобы оно больше не выполнялось. Для этого пригодится такая штука как флажок, то есть логическая переменная, в которой хранится состояние объекта.

Нам нужно написать обработчик нажатий кнопки. Это очень полезный алгоритм, я использую его всегда когда работаю с тактовыми кнопками. Итак, есть логическая переменная батт флаг. Она будет запоминать последнее состояние кнопки. Где то здесь должна быть шутка про парня с прозвищем флагшток кун, но не все его знают, а кто знает сейчас уже во всю смеется. Изначально флажок опущен, то есть при объявлении переменная равна 0, но многие меня упрекают что это ненадежно и лучше руками приравнять к 0. Окей.

Кнопка подключена на 3 цифровой пин, используя пинмод настраиваем 3 пин как инпут пуллап.

Переменная butt будет принимать текущее значение кнопки. Мы используем подтяжку инпут пуллап, поэтому Я поставлю восклицательный знак, чтобы инвертировать сигнал, так будет более логично. Смотрите, кнопка нажата, батт равен 1. Кнопка не нажата, батт = 0.

Создаём условия. Если сигнал на пине кнопки равен единице, то есть если кнопка нажата. Второе условие, и если кнопка не была нажата, то есть батт флаг равен нулю. Если оба условия выполнены, то поднимаем флажок, то есть пишем на него единичку. Теперь внимательно прокрутите в голове это условие: изначально баттфлаг равен 0. Мы нажали кнопку, батт стал равен 1, условие выполнилось, флажок изменился на 1. И при следующей итерации цикла луп условие уже не выполняется, потому что флажок теперь равен 1, то есть код в фигурных скобках выполнился только один раз при нажатии кнопки. Продолжим.

Если кнопка отпущена и флаг равен единице. Это условие срабатывает только тогда, когда кнопка была только что нажата, а затем ее отпустили. И вот здесь вот мы опускаем флаг. Смотрите что получается, при помощи двух условий и одного флажка мы создали в бесконечном цикле конструкцию, которая срабатывает только один раз, при нажатии или отпускании кнопки. Причем первая часть выполняется при нажатии, а вторая при отпускании кнопки. Чтобы в этом убедиться пошлем в порт пару слов при нажатии кнопки. То есть откроем порт, и пошлем одну строчку из первого условия, и другую из второго. Кстати такое использование общения через порт называется отладкой, мы можем видеть при каких действиях какие куски кода выполняются или не выполняются.

Добавим управление светодиодом, причём сделаем это при помощи ещё одного флажка. Флажок лед флаг, будет хранить в себе состояние светодиода. 0 выключен, 1 включен. Светодиод у нас на 13 пине, настраиваем его как выход. Теперь внимательно, пусть состояние светодиода меняется, когда я нажимаю на кнопку. Переменная логическая, инвертируем её вот таким образом. То есть присваиваем флажку значение, обратное самому себе. Есть более красивая запись этого действия, выглядит вот так. И теперь значение флажка даём на функцию дижтл райт. Проверяем. Что происходит: как только я нажимаю кнопку, флажок состояния светодиода меняется с 1 на 0 или с 0 на 1, а затем это значение подставляется в функцию жижтл райт. Можно сделать так, чтобы состояние менялось, когда я отпускаю кнопку. Просто переносим вот эти две команды вот сюда. Вот такие пироги.

У механических кнопок есть проблема, называется она дребезг. Это когда мы вроде бы нажали кнопку, но механический контакт плохой и кнопка нажимается отжимается несколько раз. В качестве защиты от дребезга можно поставить задержку дилей, таким образом как только вы нажали кнопку, у вас будет несколько

миллисекунд чтобы убрать палец, и на протяжении этого времени ардуина будет игнорировать дребезг. Но я говорил вам что дилей очень бичовская команда и лучше вообще ей не пользоваться. Лучше использовать конструкцию с миллис из урока номер 4. То есть сделать так, чтобы сразу после нажатия кнопка перестала опрашиваться, чтобы исключить дребезг контактов. То есть добавить переменную, хранящую время прошлого нажатия, и момент отпускания кнопки проверять по еще одному условию: если кнопка была нажата и с этого момента прошло больше чем указанное число миллисекунд. Остановите видео и прокрутите в голове эту конструкцию. А я пока напомню что вся показанная в виде конспекта часть доступна для скачивания в формате пдф у меня на сайте, ссылка в описании под каждым видеоуроком здесь на ютубе.

Вы наверное спросите, а как поставить отработку не просто нажатия, а например удержания кнопки, или двойного нажатия. Есть замечательная библиотека onebutton, которая позволяет отрабатывать все перечисленные вариант нажатия кнопки. Я не знал об этой библиотеке и писал отработку сам, в итоге у меня есть черновик с кусочком кода, который как раз занимается отработкой разных нажатий. Поидее он не такой удобный как библиотека, но зато вы можете посмотреть как оно вообще работает, код максимально простой, закомментированный, и состоит чисто из флажков и задержек через миллис. Ссылку на этот скетч я оставлю в описании под видео и на сайте на странице с уроками. Читайте, разбирайтесь, и пробуйте внедрять его в свои проекты. В качестве демонатрации я покажу как оно работает, результаты отработки выводятся в порт в виде текста. Одиночное нажатие с защитой от дребезга. Двойное нажатие. Нажатие и удержание, причё1м код выполняется всё время пока кнопка удерживается. Также есть вариант отработки случая, когда кнопка была нажата, чуть чуть удержана и отпущена. Таким образом одна кнопка может заменить сразу несколько кнопок, главное не забыть на какой тип нажатия какая функция была повешена.

На этом всё, в следующем уроке научимся подключать внешние светодиоды и другую нагрузку к ардуино. Спасибо за внимание, всем пока.