Эксперимент 5.1. Определение длинного нажатия кнопки

Данный эксперимент является продолжением Эксперимента 5. «Эмуляция кнопки с фиксацией». Попробуем сделать кнопку при коротком нажатии, на которую включается/выключается первый светодиод, при длинном – второй.

Схема эксперимента

В качестве базовой используется схема из эксперимента 5. Дополнительно нужно подключить светодиод LED3 к GIPO12.

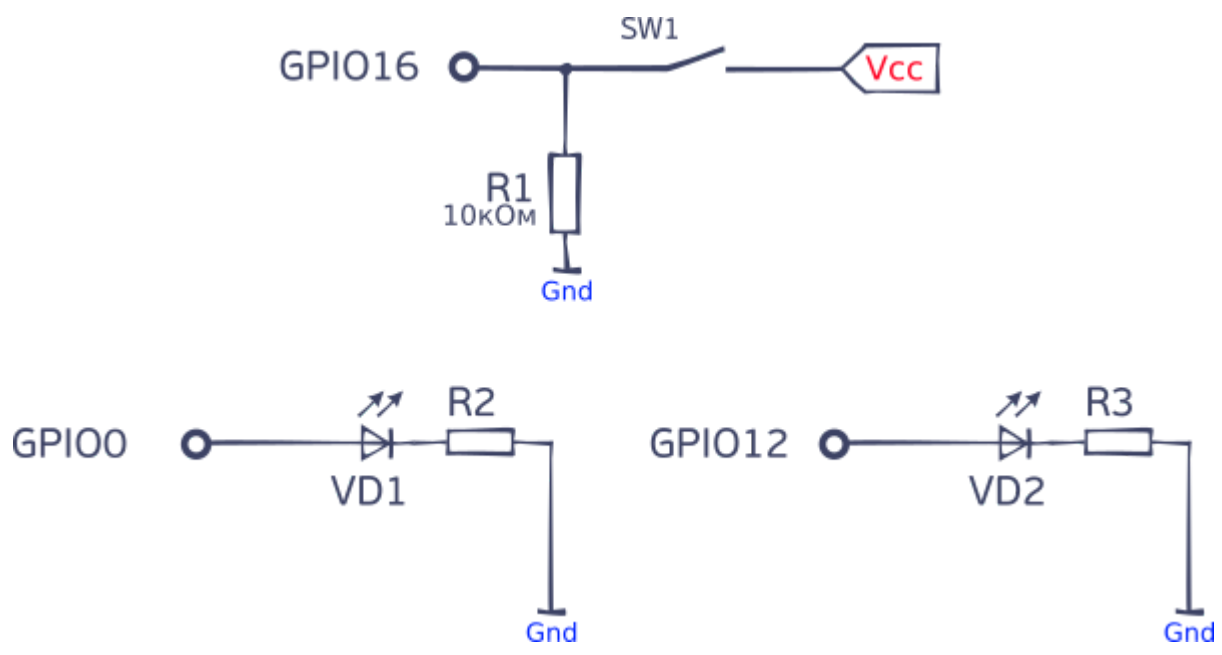

Рисунок 1. Электрическая принципиальная схема эксперимента

На рисунке изображены токоограничительные резисторы последовательно со светодиодами. При сборке схемы мы не будем устанавливать их сами так как они уже установлены на плате конструктора рядом с каждым светодиодом.

Соберем эту схему:

Рисунок 2. Монтажная схема эксперимента

Программный код эксперимента

Для начала измените в коде эксперимента 5 в строке 14 «1» на 0. Запустите скрипт и понажимайте кнопку. Заметили разницу? Теперь светодиод загорается не при нажатии кнопки, а при ее отпускании.

`Pin.value()` можно использовать для определения текущего состояния вывода, сконфигурированного не только как вход (Pin.IN), но и как выход (Pin.OUT). Кроме того `Pin.value(1)` эквивалентно Pin.on(), Pin.value(0) – Pin.off().

В нашем случае это позволит избавиться от переменной LedState и заменить конструкцию if-else в строках 15–20 одной строчкой: `Led.value(not Led.value())`

```
1.  from machine import Pin
2.  _init()
3.
4.  ButtonPin = 16
5.  LedPin = 0
6.  old_button_value = 0
7.
8.  Button = Pin(ButtonPin, Pin.IN)
9.  Led = Pin(LedPin, Pin.OUT)
10.
11. while True:
12.     button_value = Button.value()
13.     if old_button_value != button_value and button_value == 0:
14.         Led.value(not Led.value())
15.
16.     old_button_value = button_value
```

Теперь перейдем к коду нового эксперимента:

Exp5.1.py

```
1.  from machine import Pin
2.  import time
3.  _init()
4.
5.  ButtonPin = 16
6.  LedPin = 0
7.  LedPin2 = 12
8.  old_button_value = 0
9.  number_checks = 20
10. counter_checks = 0
11.
12. Button = Pin(ButtonPin, Pin.IN)
13. Led = Pin(LedPin, Pin.OUT)
14. Led2 = Pin(LedPin2, Pin.OUT)
15.
16. while True:
17.     button_value = Button.value()
18.     if button_value == 0:
19.         if button_value !=old_button_value:
20.             if counter_checks < number_checks:
21.                 Led.value(not Led.value())
22.                 counter_checks = 0
23.             else:
24.                 Led2.value(not Led2.value())
25.                 counter_checks = 0
26.     else:
27.         counter_checks += 1
28.     old_button_value = button_value
29.     time.sleep_ms(50)
```

Программа в бесконечном цикле while проверяет состояние кнопки. Если кнопка нажата, инкрементируем (увеличиваем на один) счетчик времени нажатия. Если кнопка отпущена, то определяем была ли она отпущена только что. Если это так, то определяем время сколько она была нажата. Если время нажатия меньше порога длинного нажатия, то считаем, что это было короткое нажатие. Если время нажатия было длиннее − длинное.

`number_checks = 20` это количество проверок состояния кнопки после которого будет считаться, что кнопка была нажата длительно.

`counter_checks = 0` это счетчик проверок состояния кнопки который увеличивается каждый раз при выполнении цикла while, если в момент проверки кнопка была нажата.

`time.sleep_ms(50)` – работает аналогично time.sleep(), но значение задается не в секундах, а в миллисекундах. 1 секунда = 1000 миллисекунд. Так мы задаем время между проверками кнопок. В данном случае мы проверяем состояние кнопка каждые 50 миллисекунд.

Например, мы считаем что длительное нажатие кнопки это более 1 секунды. Тогда нужное количество проверок равно 20, т.к. 20 проверок умножить на 50 мс = 1000 мс = 1 с.

В цикле сначала проверяется текущее состояние кнопки button_value == 0. Если кнопка не нажата, сравнивается текущее состояние кнопки с предыдущим `button_value !=old_button_value` . В случае различия состояний сравнивается значения счетчика с количеством проверок состояния кнопки `counter_checks < number_checks` (так мы определяем время нажатия кнопки): если значение счетчика меньше, то переключается первый светодиод, в противном случае второй светодиод. После отпускания кнопки сбрасывается значение счетчика нажатия кнопки `counter_checks = 0` .