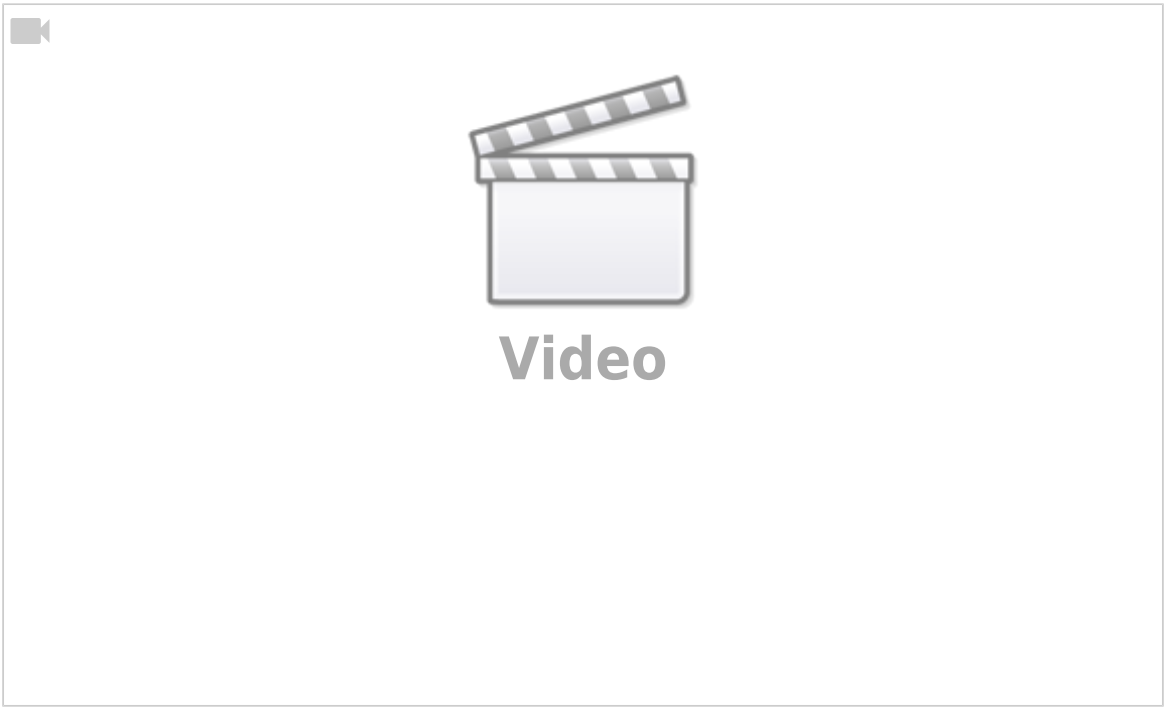
Урок 9. Собственный модуль

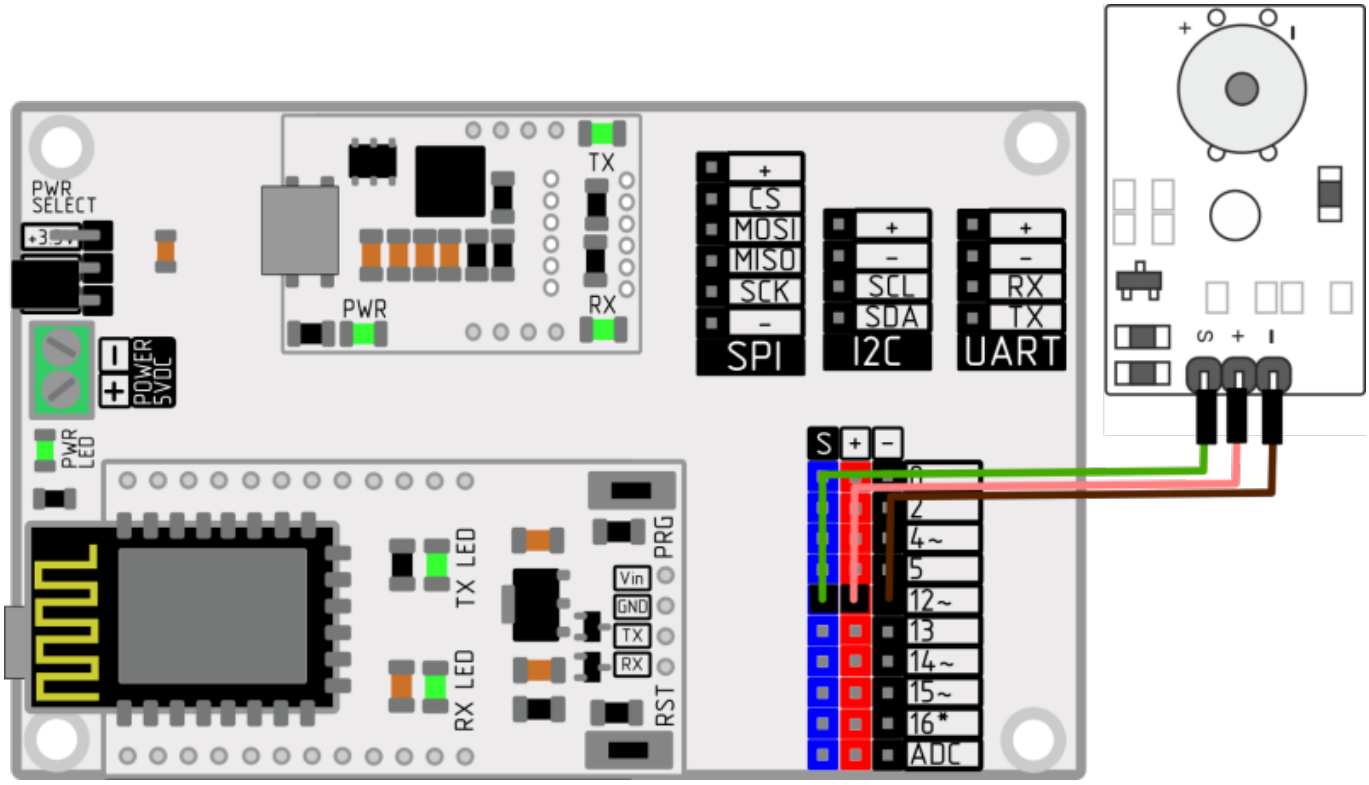


А теперь попробуем создать собственный музыкальный сервер. Принцип управления будет похожим. Мы хотели бы заходить на интернет-страницу нашего сервера, где выбирали бы мелодию для воспроизведения, после чего с помощью зуммера воспроизводится требуемая нам мелодия.

Для того чтобы наш код не выглядел слишком громоздко, мы уберём функции связанные с воспроизведением музыки в отдельный модуль. И, уже потом, импортируя его сможем пользоваться нужными нам функциями.

Напишем музыкальный модуль.

Соберите схему, подключив модуль зуммера к плате Гиккон Коннект, к выводу 12:



Возьмём за основу код из урока 14 первого раздела данного руководства. Для начала модифицируем функцию `sing_song`. В прошлом она воспроизводила единственную мелодию, мы изменим её, чтобы она могла воспроизводить любую переданную ей песню:

```
# функция для воспроизведения всех звуков
def sing_song(song, dur, pauses):
    for i in range(len(song)):
        note = song[i]
        sound(notes[note], dur[i], pauses[i])
```

Теперь, чтобы функция работала, нам нужно передать в неё массив с нотами, длительностями и паузами. Песню ёлочка, оформим в отдельную функцию:

```
def elochka():
    # список нот
    song = ["sol","mi","mi","sol","mi","mi","sol","fa","mi","re","do",
    "la","si","la","sol","mi","mi","sol","fa","mi","re","do"]

    # список длительности в миллисекундах
    dur = [400,200,200,400,200,200,200,200,200,200,400,400,200,200,400,200,
    200,200,200,200,200,400]

    # список пауз после ноты
    pauses =
[500,250,250,500,250,250,250,250,250,250,800,500,250,250,500,250,
    250,250,250,250,250,500]

    sing_song(song, dur, pauses)
```

Добавим похожую функцию, которая воспроизведёт песню Антошка:

```
def antoshka():
    # список нот
    song = ["la","sol","sol","la","fa","fa","la","sol","sol","mi","do",
"la","fa","la","sol","sol","la","fa","fa","la","sol","sol","mi","do","la","f
a"]

    # список длительности в миллисекундах
    dur = [150,150,200,150,150,200,150,150,150,150,150,400,150,
    150,200,150,150,200,150,150,150,150,150,200,400,200]

    # список пауз после ноты
    pauses =
[200,200,300,200,200,300,250,250,250,250,250,500,200,200,300,200,200,300,250
,250,250,250,250,250,500,800]

    sing_song(song, dur, pauses)
```

Функцию sound нам менять не нужно, она подойдёт в том виде что есть сейчас. Теперь у нас есть две мелодии, которые мы можем воспроизвести. Понять что всё работает правильно можно следующим образом.

При запуске модуля проверим он сейчас запущен как основной модуль или нет. Если как основной (а не вспомогательный), то воспроизведём наши мелодии по очереди. У модуля есть служебное свойство name, которое хранит имя запущенного модуля. Если модуль запущен как главный, то его имя будет “main”. Именно это и проверим, а затем воспроизведём две мелодии подряд.

```
if __name__ == "__main__":
    elochka()
    antoshka()
```

Теперь у нас готов собственный модуль для воспроизведения музыки.

```
# импорт модулей
from machine import Pin, PWM
from time import sleep_ms

# создадим ШИМ вывод  buz  на 12 выводе
buz = PWM(Pin(12, Pin.OUT))

# словарь нот
notes = {}
notes["do"] = 262
notes["re"] = 294
notes["mi"] = 330
notes["fa"] = 349
notes["sol"] = 392
notes["la"] = 440
notes["si"] = 494
```

```
def elochka():
    # список нот
    song = ["sol","mi","mi","sol","mi","mi","sol","fa","mi","re","do",
    "la","si","la","sol","mi","mi","sol","fa","mi","re","do"]

    # список длительности в миллисекундах
    dur = [400,200,200,400,200,200,200,200,200,200,400,400,200,200,400,200,
    200,200,200,200,200,400]

    # список пауз после ноты
    pauses =
[500,250,250,500,250,250,250,250,250,250,800,500,250,250,500,250,
    250,250,250,250,250,500]

    sing_song(song, dur, pauses)

def antoshka():
    # список нот
    song = ["la","sol","sol","la","fa","fa","la","sol","sol","mi","do",
"la","fa","la","sol","sol","la","fa","fa","la","sol","sol","mi","do","la","f
a"]

    # список длительности в миллисекундах
    dur = [150,150,200,150,150,200,150,150,150,150,150,400,150,
    150,200,150,150,200,150,150,150,150,150,200,400,200]

    # список пауз после ноты
    pauses =
[200,200,300,200,200,300,250,250,250,250,250,500,200,200,300,200,200,300,250
,250,250,250,250,250,500,800]

    sing_song(song, dur, pauses)

# функция для воспроизведения звука определённой частоты
def sound(freq, dur, pause):
    buz.freq(freq)  # установка частоты из параметра
    buz.duty(512)  # установить заполнение в 512
    sleep_ms(dur)  # задержка
    buz.duty(0)  # установить заполнение в 0
    sleep_ms(pause)  # пауза

# функция для воспроизведения всех звуков
def sing_song(song, dur, pauses):
    for i in range(len(song)):
        note = song[i]
        sound(notes[note], dur[i], pauses[i])


if __name__ == "__main__":

    elochka()
```

```
    antoshka()
```

Предыдущий урок

Следующий урок