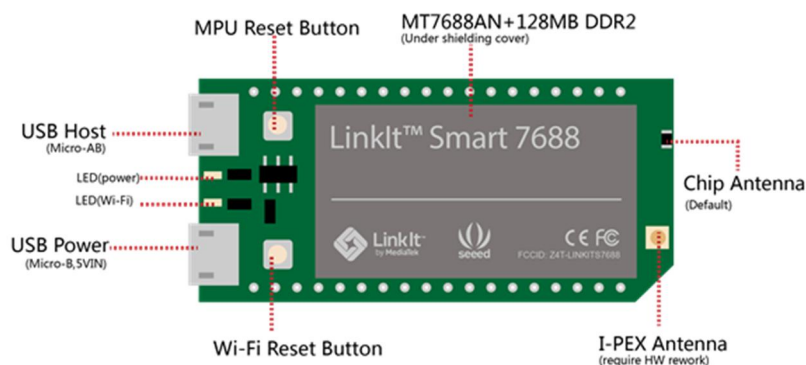


Указания к практическим работам с модулем Linkit Smart 7688



Содержание

Установка необходимых программ	2
Работа 1. Первое подключение к модулю Linkit Smart 7688 через сеть WiFi	3
Работа 2. Написание первых программ на Python: мигание светодиодом	12
Работа 3. Управляем яркостью светодиода с помощью PWM	20
Работа 4. Подключаем кнопку	23
Работа 5. Использование прерывания	26
Работа 6. Программная реализация бегущих огней	29
Работа 7. Подключаем электронный компас по интерфейсу I2C	34
Подключение к WiFi роутеру и сети Internet	40
Перевод модуля в режим точки доступа	43
Установка нового имени (dns-адреса) модуля	44
Бонусная работа. Создание веб-сервера на Python	47
Определение IP-адреса модуля	51
Обновление прошивки модуля	52
Восстановление стандартных настроек модуля	55
Индикация режимов работы модуля красным светодиодом	56
Описание линий ввода-вывода модуля	57
Ссылки	59

Установка необходимых программ

1. Для доступа к модулю Linkit Smart 7688 через сеть WiFi из операционной системы Windows установите **Bonjour print service** (для Linux и Mac OS X этого делать не нужно):

https://support.apple.com/kb/DL999?locale=en_US

2. Для подключения к модулю Linkit Smart 7688 из ОС Windows через зашифрованное SSH соединения установите программу **Putty** (для Linux и Mac OS X этого делать не нужно):

- для x32 Windows:

<https://the.earth.li/~sgtatham/putty/latest/w32/putty-0.68-installer.msi>

- для x64 Windows:

<https://the.earth.li/~sgtatham/putty/latest/w64/putty-64bit-0.68-installer.msi>

3. Для загрузки файлов на модуль по протоколу SCP через защищенное SSH соединение с компьютера под управлением ОС Windows установите программу **WinSCP** (для Linux и Mac OS X этого делать не нужно) :

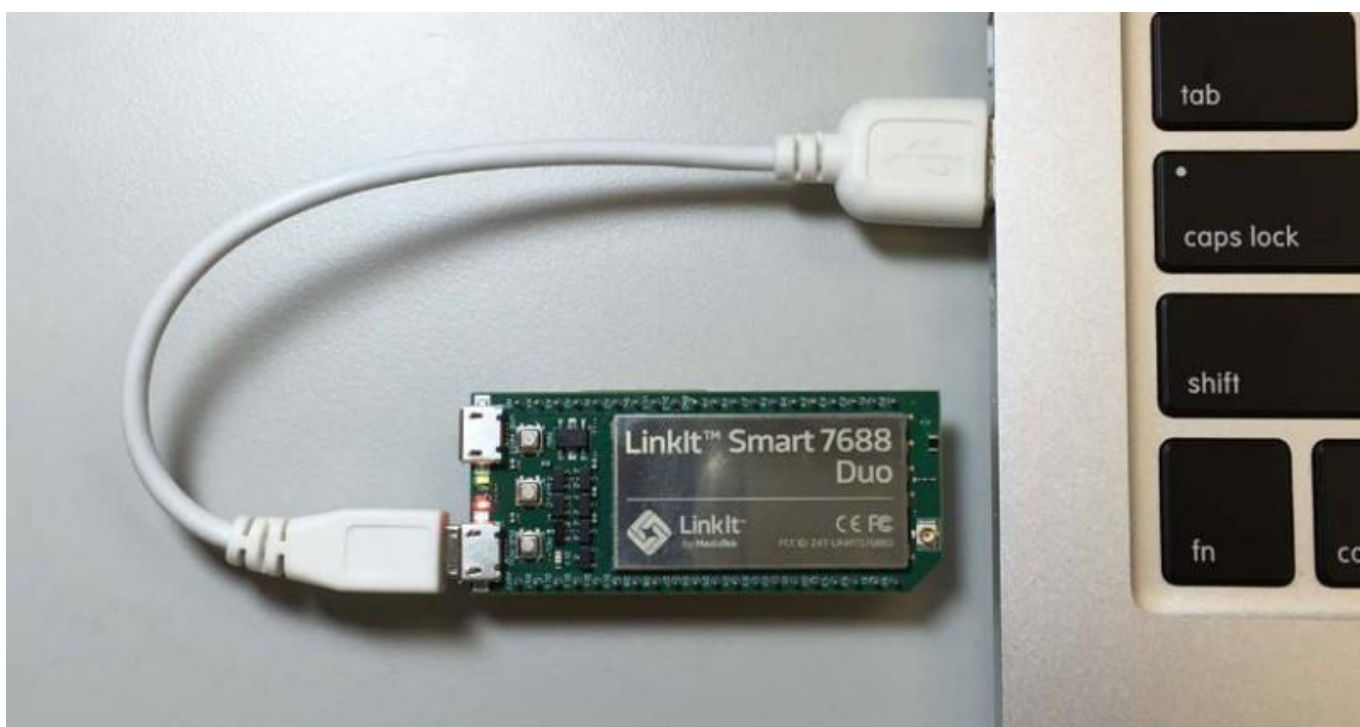
- детальное описание программы: <https://winscp.net/eng/docs/lang:uk>
- страница загрузки: <https://winscp.net/eng/download.php>
- прямая ссылка на загружаемый файл:

<https://winscp.net/download/WinSCP-5.9.4-Setup.exe>

Работа 1.

Первое подключение к модулю Linkit Smart 7688 через сеть WiFi

1. **Подайте напряжение питания** на модуль Linkit Smart 7688 подключив его к USB порту компьютера. Вставьте один конец кабеля в USB-порт компьютера, а другой конец подключите к обозначенному PWR microUSB-порту модуля Linkit Smart 7688. В результате зеленый светодиод должен гореть, а красный светодиод сначала будет гореть примерно 30 секунд (в течение загрузки модуля), а затем начнет мигать (после установки WiFi соединения).



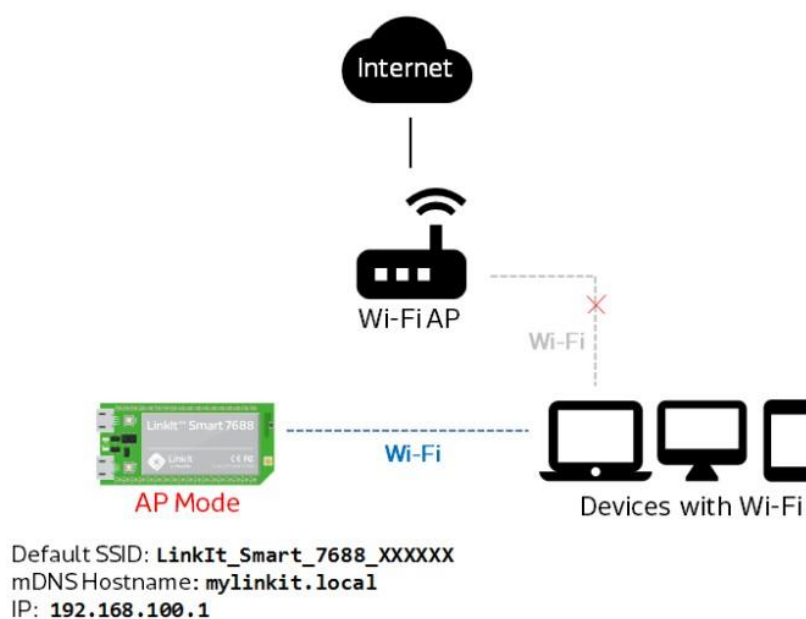
2. Подключитесь к модулю Linkit Smart 7688 через WiFi

После того, как модуль загрузился, он начинает работать как WiFi точка доступа с именем **LinkIt_Smart_7688_XXXXXX**. Подключитесь к этой точке доступа стандартными средствами своего ноутбука.



После того, как вы подключитесь к модулю Linkit Smart 7688 в таком режиме, его красный светодиод начнет мигает с частотой 3 Гц.

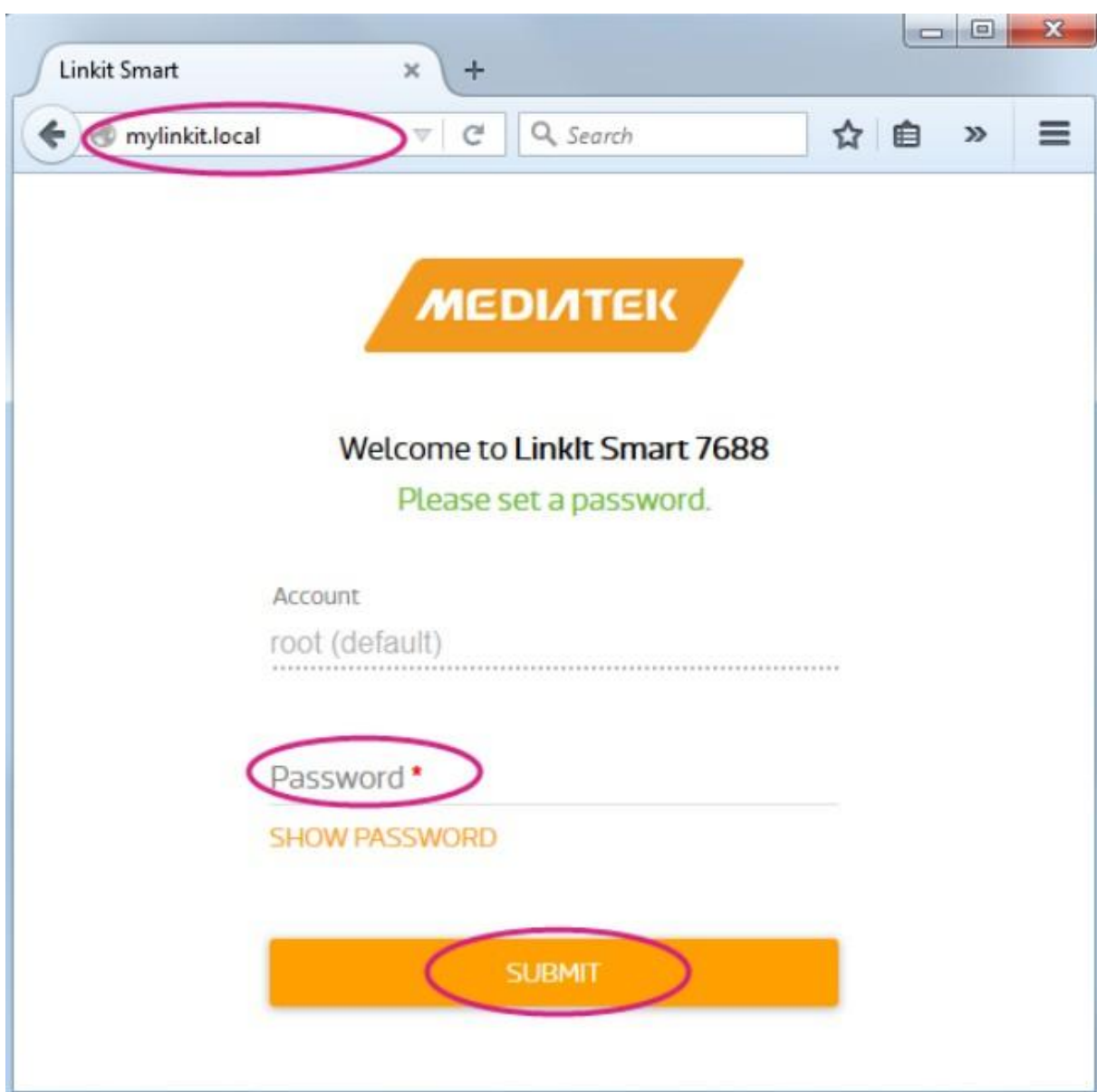
Обратите внимание на то, что после того, как вы подключитесь через WiFi к модулю Linkit Smart 7688 в таком режиме, доступ к сети Интернет возможен только через проводное соединение



Если вы были подключены к Internet через WiFi канал, то теперь это подключение будет потеряно, поскольку модуль WiFi вашего ноутбука подключен не к WiFi роутеру, а к модулю Linkit Smart 7688. Эту проблему можно решить, переведя модуль Linkit Smart 7688 в режим WiFi станции, которая будет подключена к тому же WiFi роутеру, что и ваш ноутбук. В одном из следующих разделов будет показано как это сделать.

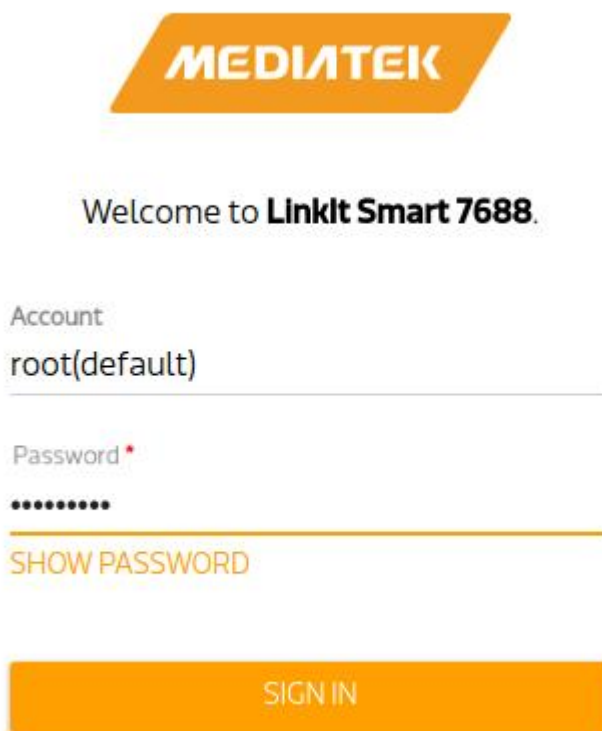
3. Зайдите в веб-интерфейс управления модулем Linkit Smart 7688

После подключения по WiFi к Linkit Smart 7688, запустите браузер и откройте в нем страницу веб-интерфейса управления модулем по адресу <http://mylinkit.local>



Если вы делаете это первый раз, то вам необходимо ввести пароль, который вы будете в дальнейшем использовать для подключения к модулю и нажать **SUBMIT**.

После этого появится стандартное окно авторизации доступа к веб-интерфейсу управления модулем, где вам будет предложено ввести только заданный пароль.



После ввода пароля необходимо нажать кнопку входа **SIGN IN**.

Если вы забудете пароль, вы не сможете подключиться к модулю. Единственная возможность после этого - сбросить пароль вместе со всеми настройками модуля. В одном из следующих разделов будет показано как это сделать. Однако следует иметь в виду, что во время процедуры сброса пароля будут удалены все файлы, созданные вами и загруженные в модуль, а также все установленные программы. Поэтому лучше хорошо запомнить пароль!

После ввода правильного пароля и нажатия на кнопку **SIGN IN**, вы попадаете на страницу веб-интерфейса упрощенных настроек модуля. Здесь вы можете просмотреть имя и IP-адрес модуля, задать новый пароль (нажав кнопку **CONFIGURE**), обновить прошивку (как это сделать будет описано далее), а нажав кнопку **RESET** выполнить сброс всех настроек прошивки к стандартным значениям.

System information	Network
<h3>Platform information</h3> <p>Device name mylinkit</p> <hr/> <p>Current IP address 192.168.100.1</p> <hr/> <h3>Account information</h3> <p>Account root(default)</p> <hr/> <p>Password *</p> <hr/> <p>CONFIGURE</p>	
<h3>Software information</h3> <p>Boot loader version v0.8.2</p> <hr/> <p>Firmware version v0.9.4</p> <hr/> <p>UPGRADE FIRMWARE</p>	
<h3>Factory reset</h3> <p>Reset the device to its default settings. Important: This action will remove all your data and settings from the device.</p> <hr/> <p>RESET</p>	

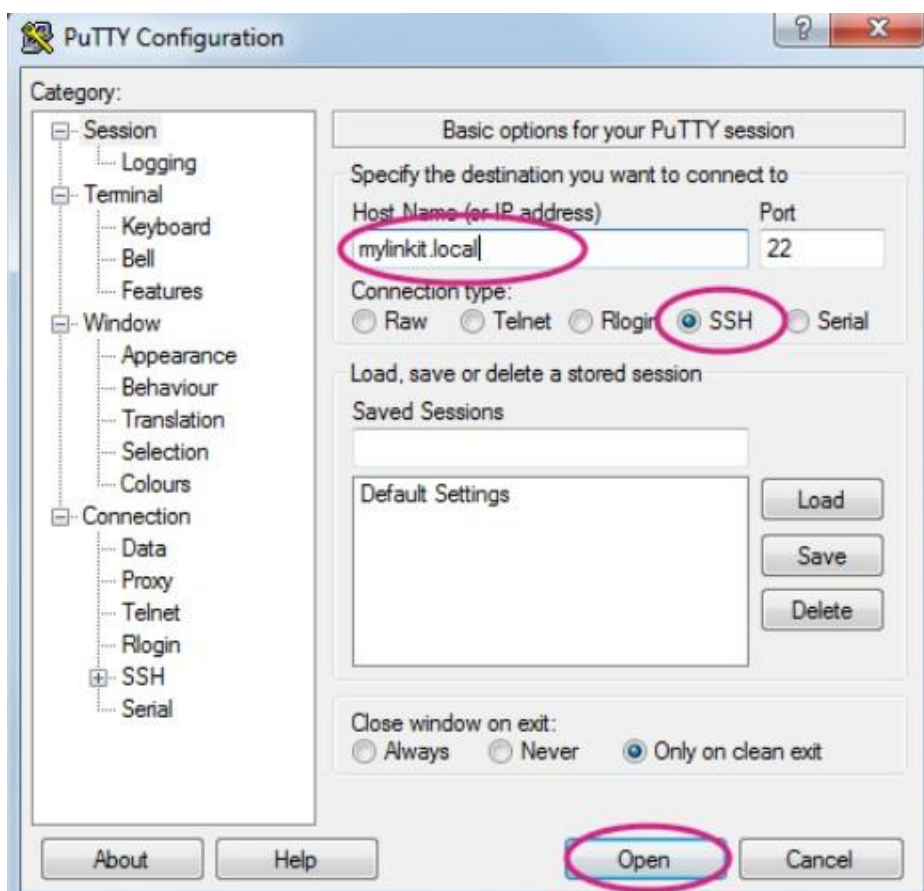
Для перехода в режим WiFi станции и подключения к WiFi роутеру необходимо нажать кнопку **Network** в правом верхнем углу. Далее будет подробно описано как подключить модуль Linkit Smart 7688 к WiFi роутеру.

Для перехода в режим расширенных настроек необходимо нажать ссылку **OpenWrt** в верхнем правом углу. Однако сейчас этого делать не надо.

4. Подключение к терминалу модуля Linkit Smart 7688 через SSH

Инструкция для Windows

Запустите программу PuTTY, которую вы установили ранее. В поле Host Name введите адрес модуля mylinkit.local (или IP адрес модуля. Как определить IP адрес Linkit Smart 7688 будет рассказано далее). Установите тип связи (connection type) SSH и нажмите на кнопку **Open** для подключения к модулю.



При первой попытке подключения появится информационное окно с предупреждением. Просто нажмите **YES**.



Поздравляем! Вы получили доступ к консоли (терминалу) Linux OpenWrt!

```
lampa@lampa ~ $ ssh root@mylinkit.local
The authenticity of host 'mylinkit.local (192.168.100.1)' can't be established.
RSA key fingerprint is SHA256:BhswVqUBVOZ0UNkR1+Xy0a8pu2/gbqZpmp55c7cewcl.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'mylinkit.local,192.168.100.1' (RSA) to the list of known hosts.
root@mylinkit.local's password:

BusyBox v1.23.2 (2016-09-27 07:54:34 CEST) built-in shell (ash)

  _   _   _   _   _   _   _   _   _   _   _   _   _   _   _   _
 | |   | |   | |   | |   | |   | |   | |   | |   | |   | |   | |   | |
 |_|   |_|   |_|   |_|   |_|   |_|   |_|   |_|   |_|   |_|   |_|   |_|   |_|

-----
CHAOS CALMER (15.05.1, r49203)
-----
* 1 1/2 oz Gin           Shake with a glassful
* 1/4 oz Triple Sec     of broken ice and pour
* 3/4 oz Lime Juice    unstrained into a goblet.
* 1 1/2 oz Orange Juice
* 1 tsp. Grenadine Syrup
-----
root@mylinkit:~#
```

5. Запуск первой программы на Python

Сейчас мы запустим первую программу на Python, она будет мигать красным светодиодом модуля Linkit Smart 7688.

Программа уже находится в модуле. Для просмотра кода программы выполните в терминале модуля Linkit Smart 7688 команду **cat/IoT/examples/blink-gpio44.py**

```
lampa@lampa ~ $ ssh root@mylinkit.local
root@mylinkit.local's password:

BusyBox v1.23.2 (2016-09-27 07:54:34 CEST) built-in shell (ash)

  _   _   _   _   _   _   _   _   _   _   _   _   _   _   _   _
 | |   | |   | |   | |   | |   | |   | |   | |   | |   | |   | |   | |
 |_|   |_|   |_|   |_|   |_|   |_|   |_|   |_|   |_|   |_|   |_|   |_|   |_|

-----
CHAOS CALMER (15.05.1, r49203)
-----
* 1 1/2 oz Gin           Shake with a glassful
* 1/4 oz Triple Sec     of broken ice and pour
* 3/4 oz Lime Juice    unstrained into a goblet.
* 1 1/2 oz Orange Juice
* 1 tsp. Grenadine Syrup
-----
root@mylinkit:~# cat /IoT/examples/blink-gpio44.py
```


Работа 2.

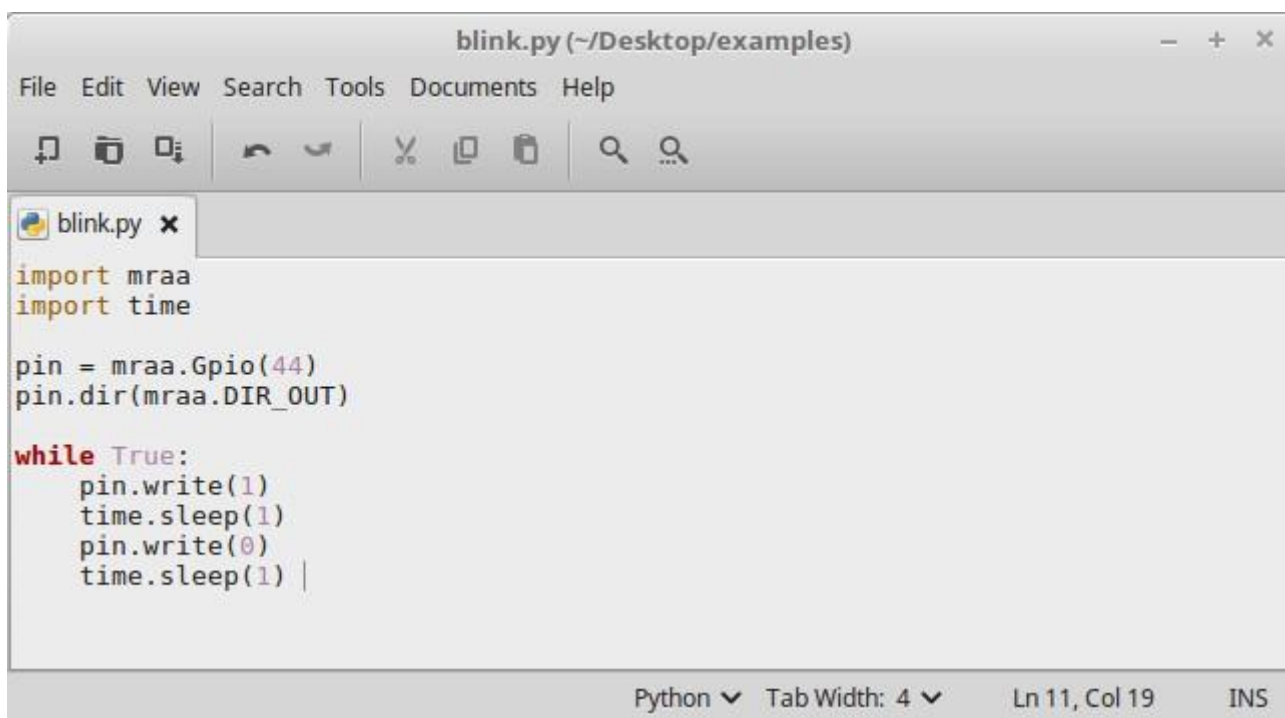
Написание первых программ на Python: мигание светодиодом.

Создадим программу для мигания красным светодиодом, который находится на плате модуля Linkit Smart 7688.

В конце предыдущей работы вы запустили программу, которая мигала красным светодиодом модуля Linkit Smart 7688. А теперь попробуйте написать такую программу самостоятельно.

На своем компьютере создайте текстовый файл с именем "blink.py". Добавьте в файл следующий код на языке Python:

С помощью инструкции **import mraa** подключается библиотеку mraa (она портирована с Intel Edison), которая содержит функции для управления аппаратной



```
blink.py (~/Desktop/examples)
File Edit View Search Tools Documents Help
blink.py x
import mraa
import time

pin = mraa.Gpio(44)
pin.dir(mraa.DIR_OUT)

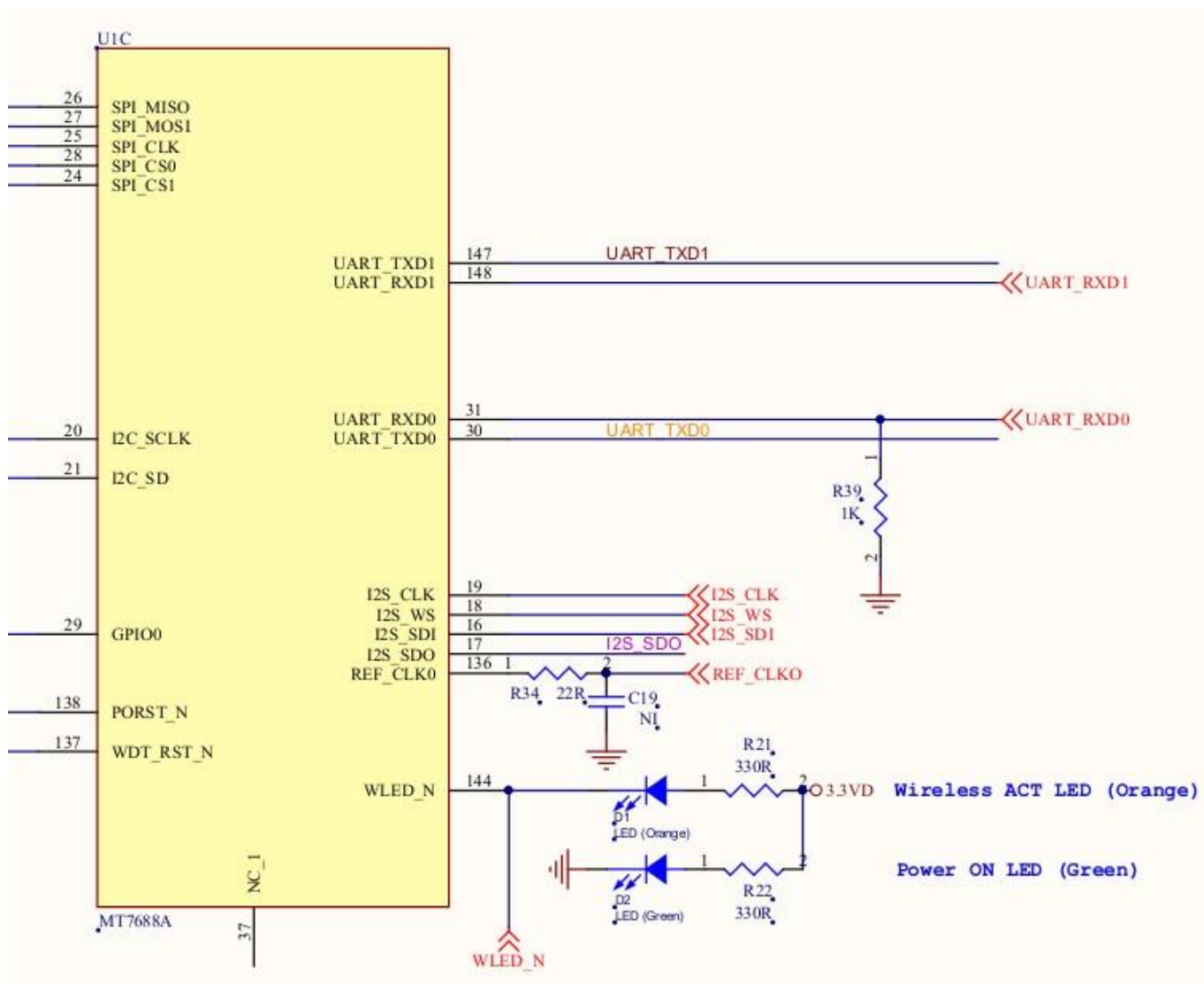
while True:
    pin.write(1)
    time.sleep(1)
    pin.write(0)
    time.sleep(1) |

Python Tab Width: 4 Ln 11, Col 19 INS
```

частью модуля из программ на языке Python. С помощью функций этой библиотеки мы будем управлять напряжением на выводе модуля.

С помощью инструкции **import time** подключается библиотеку time, которая содержит функцию задержки sleep(...), она останавливает выполнение программы на заданный в секундах интервал времени.

Создаем переменную `pin = mraa.Gpio(44)`, которая будет описывать контакт модуля с номером 44. Именно к этому контакту подключен красный светодиод по следующей схеме:



WLED_N - это и есть контакт №44. Как видно из схемы, чтобы включить красный светодиод необходимо 44-й контакт соединить с землей. Если же на 44-й контакт подать напряжение 3.3 В, то светодиод гаснет.

Вызов функции `pin.dir(mraa.DIR_OUT)` определяет пин №44 как выходной (то есть он будет формировать определенное напряжение - 0 В или 3.3 В).

Вызов функции `pin.write(1)` приводит к появлению на выводе №44 напряжения 3.3 В и светодиод гаснет.

Вызов функции `pin.write(0)` приводит к появлению на выводе №44 напряжения 0 В (вывод соединяется с землей) и светодиод начинает гореть.

Вызов функции **time.sleep(1)** останавливает выполнение программы на 1 секунду (создает задержку).

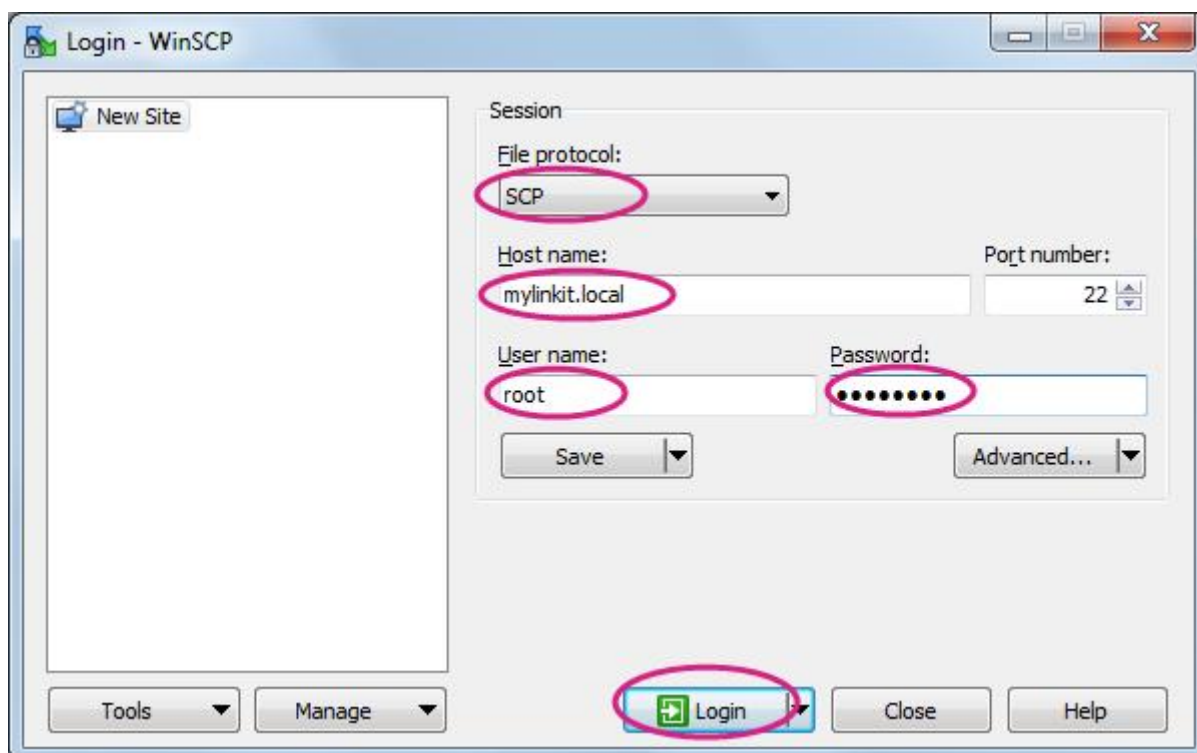
while True: создает бесконечный цикл. Обратите внимание на то, что отступы в виде табуляции позволяют определить, какая часть программы работает внутри цикла.

Таким образом, вы создали файл с первой программой на Python. Теперь необходимо загрузить этот файл в модуль Linkit Smart 7688, в папку /root. Сделайте это, используя приведенные ниже указания.

Загрузка файлов в модуль Linkit Smart 7688 через зашифрованное соединение

Инструкция для Windows

Запустите установленную ранее программу WinSCP.



В поле File protocol выберите **SCP**

В поле Host name укажите адрес модуля Linkit Smart 7688: mylinkit.local

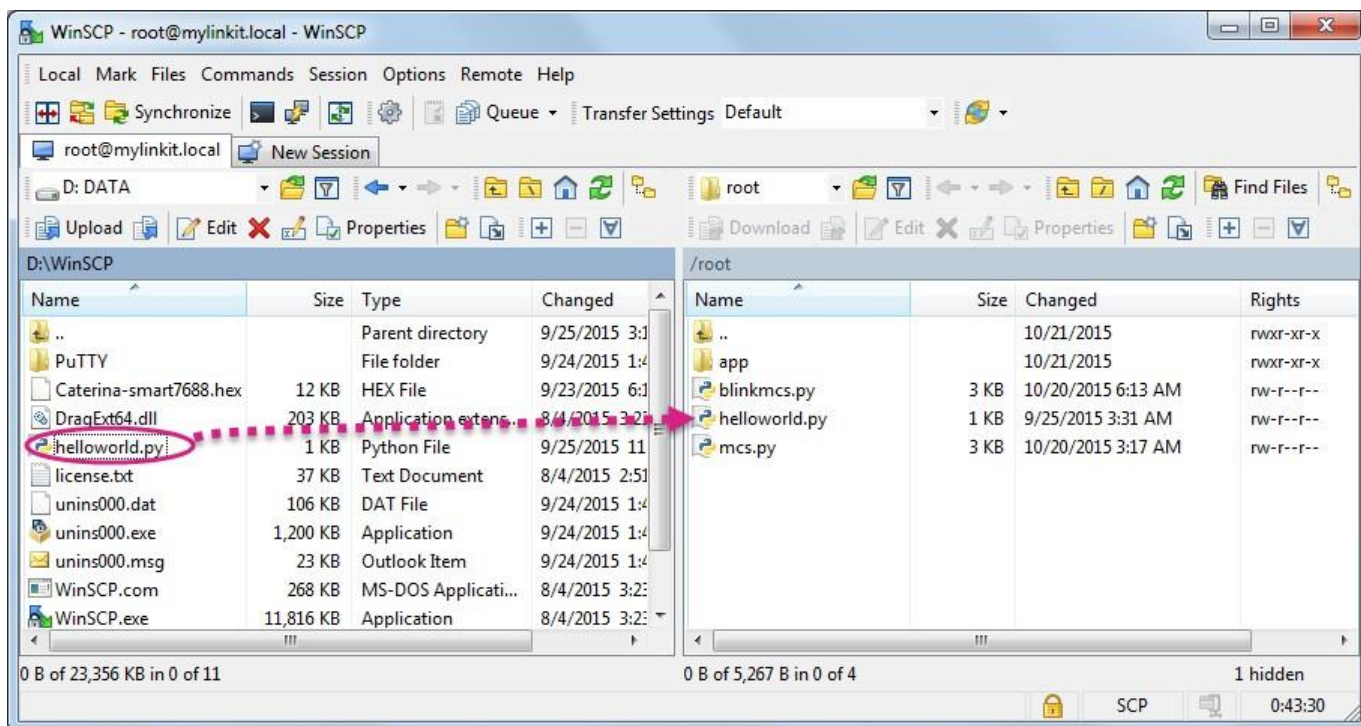
Номер порта: 22

В поле User name укажите: root

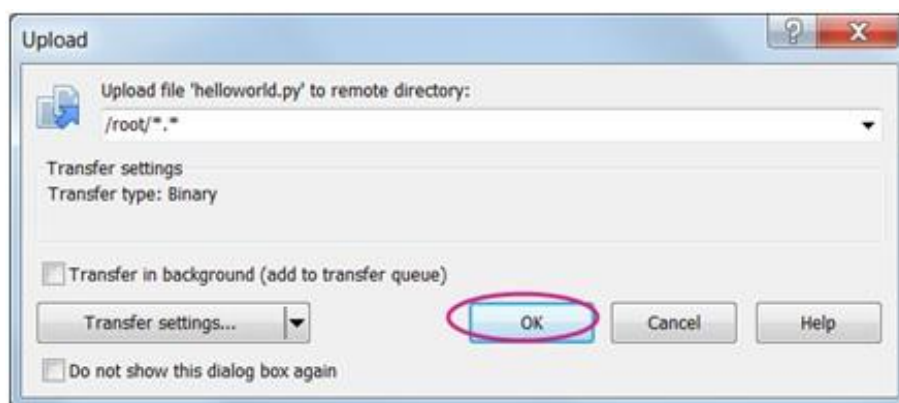
В поле Password введите пароль, который вы ранее задали в веб-интерфейсе управления модулем

Нажмите **Login**. При появлении предупреждения нажмите **OK**.

Откроется окно обмена файлами.



В левой панели выберите файл на компьютере, который вы хотите загрузить в модуль Linkit Smart 7688 и перетащите файл в правую панель. Появится окно **Upload** в котором необходимо будет подтвердить перемещение файла нажав **OK**.



Инструкция для Linux i Mac

Выполните из терминала команду

```
scp /home/lampa/Desktop/blink.py root@mylinkit.local:/root
```

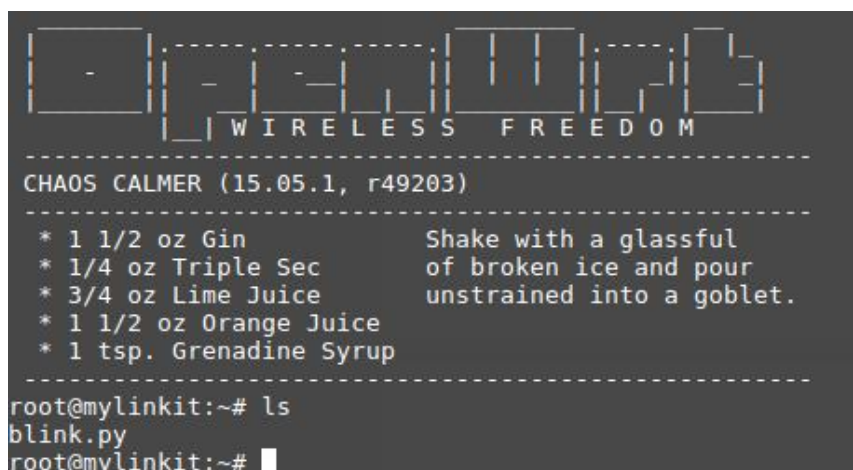
Данная команда копирует файл `blink.py`, который находится на вашем компьютере (в данном примере, файл находится на рабочем столе пользователя `lampa` в Linux), в папку `/root`, которая находится в модуле Linkit Smart 7688. Папка `/root` это домашняя папка пользователя `root` модуля Linkit Smart 7688, в эту папку вы попадаете, когда подключаетесь к модулю по `ssh`.

После запуска команды необходимо будет ввести пароль доступа к модулю Linkit Smart 7688 и нажать `Enter`. Файл будет скопирован.

Запуск программы `blink.py` на модуле Linkit Smart 7688

Для запуска программы `blink.py` на модуле Linkit Smart 7688, подключитесь к терминалу (консоли) модуля с помощью SSH (как это делать мы рассматривали в предыдущем разделе). После подключения вы находитесь в домашней папке пользователя `root`. Именно сюда вы скопировали файл `blink.py`.

Чтобы выяснить, какие файлы есть в данной папке, введите в консоли команду `ls` и нажмите `Enter`.



```
-----  
|_| W I R E L E S S F R E E D O M  
-----  
CHAOS CALMER (15.05.1, r49203)  
-----  
* 1 1/2 oz Gin           Shake with a glassful  
* 1/4 oz Triple Sec     of broken ice and pour  
* 3/4 oz Lime Juice     unstrained into a goblet.  
* 1 1/2 oz Orange Juice  
* 1 tsp. Grenadine Syrup  
-----  
root@mylinkit:~# ls  
blink.py  
root@mylinkit:~#
```


Как видите, файл **blink.py** присутствует.

Для запуска программы **blink.py** запустите с консоли команду **python blink.py**

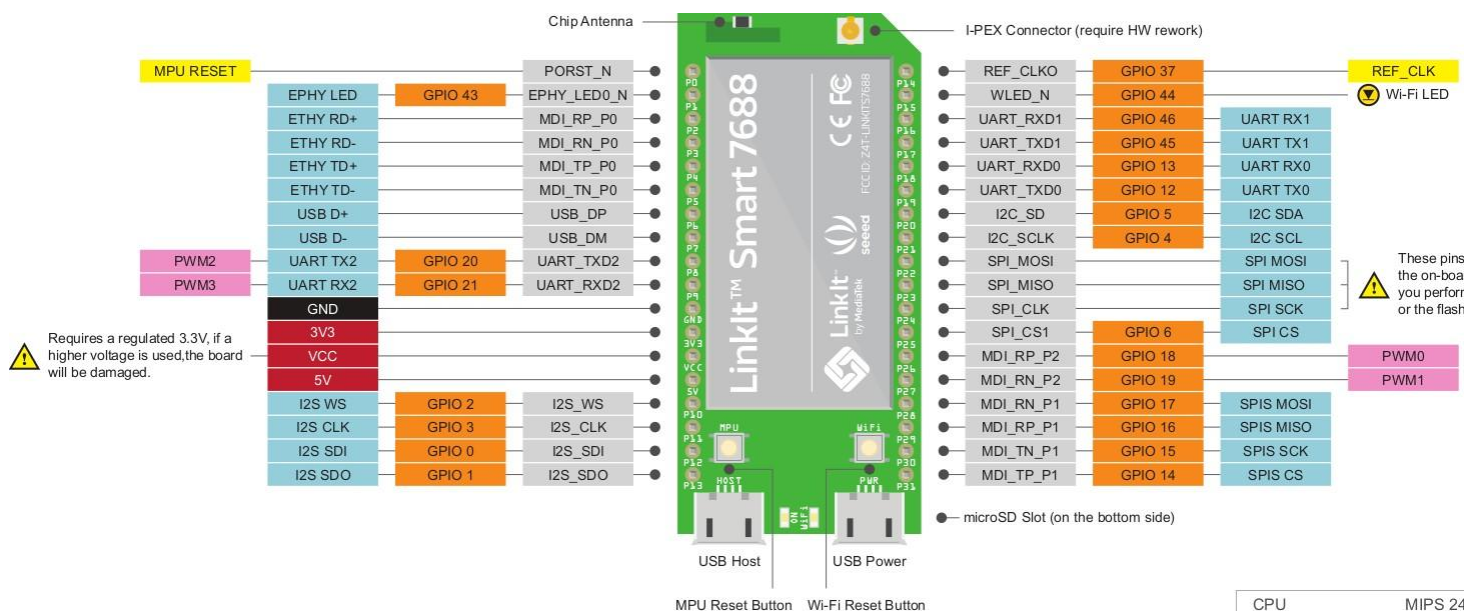
```
-----
|_ | W I R E L E S S   F R E E D O M
-----
CHAOS CALMER (15.05.1, r49203)
-----
* 1 1/2 oz Gin           Shake with a glassful
* 1/4 oz Triple Sec     of broken ice and pour
* 3/4 oz Lime Juice    unstrained into a goblet.
* 1 1/2 oz Orange Juice
* 1 tsp. Grenadine Syrup
-----
root@mylinkit:~# ls
blink.py
root@mylinkit:~# python blink.py
```

Убедитесь, что после запуска программы красный светодиод начал мигать, изменяя свое состояние (светится/не светится) каждую секунду.

Для остановки программы нажмите клавиш **Ctrl + C**

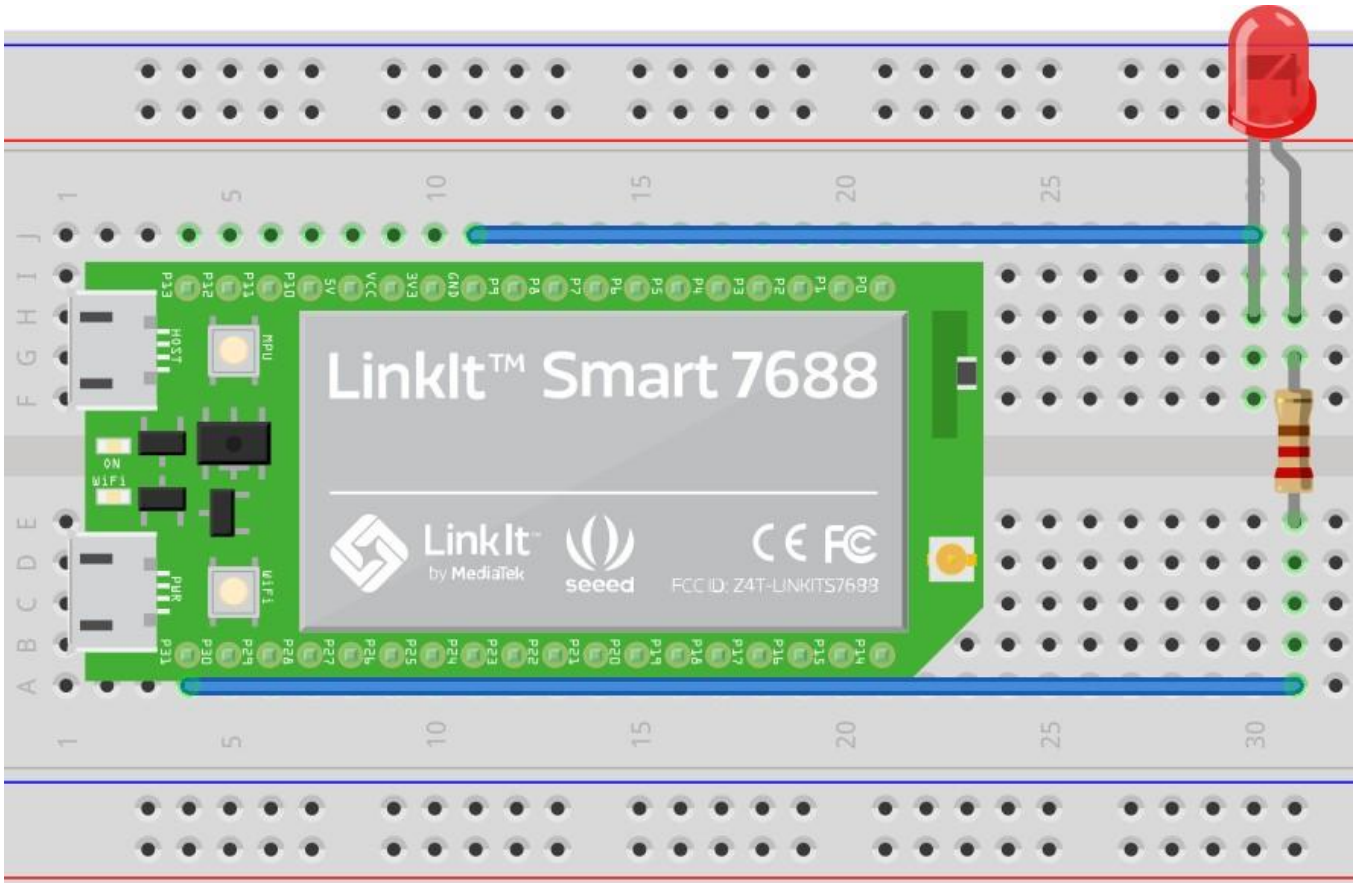
Подключим свой светодиод к модулю Linkit Smart 7688.

Теперь подключите красный светодиод к контакту №14. Этот контакт находится у USB разъема питания и обозначен как P31. Обратите внимание на то, что порядковые номера контактов, написанные на плате модуля, не соответствуют номерам, которые необходимо использовать в программах. Где находится необходимый вам контакт можно определить по рисунку:

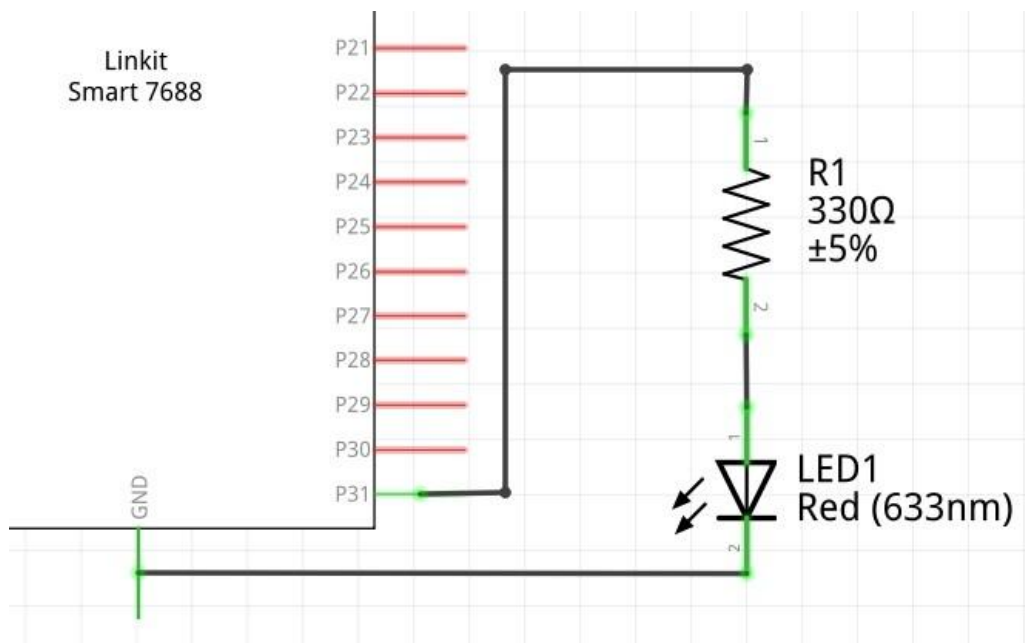


Этот рисунок в увеличенном виде приведен в конце данного документа.

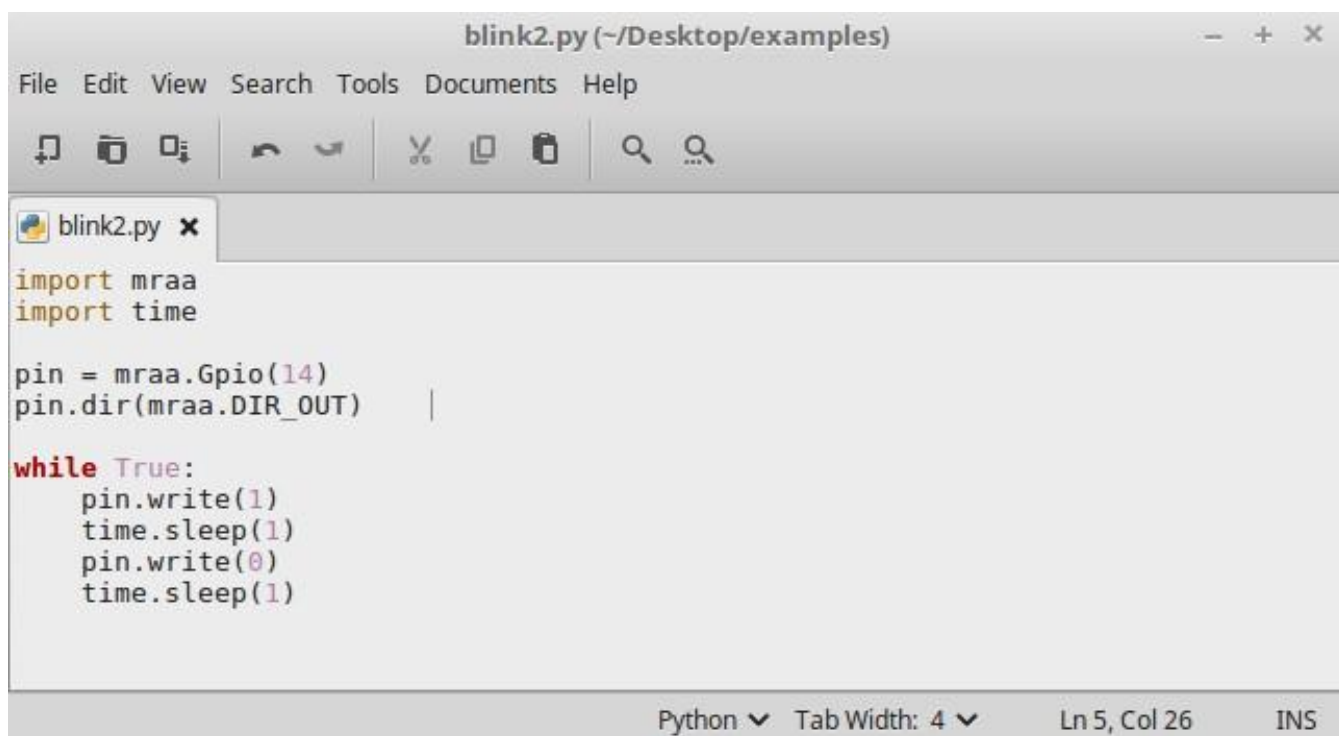
Схема подключения на макетной плате показана на рисунке ниже. Сопротивление резистора необходимо выбрать порядка 330 Ом. К резистору необходимо подключить длинный вывод светодиода (анод), а к контакту модуля с надписью GND подключите короткий вывод светодиода (катод).



Обычная (принципиальная) схема подключения светодиода выглядит так:



На своем компьютере создайте текстовый файл с именем "**blink2.py**". Добавьте в файл следующий код на языке Python:



```
blink2.py (-/Desktop/examples)
File Edit View Search Tools Documents Help
[Icons]
blink2.py x
import mraa
import time

pin = mraa.Gpio(14)
pin.dir(mraa.DIR_OUT)

while True:
    pin.write(1)
    time.sleep(1)
    pin.write(0)
    time.sleep(1)

Python Tab Width: 4 Ln 5, Col 26 INS
```

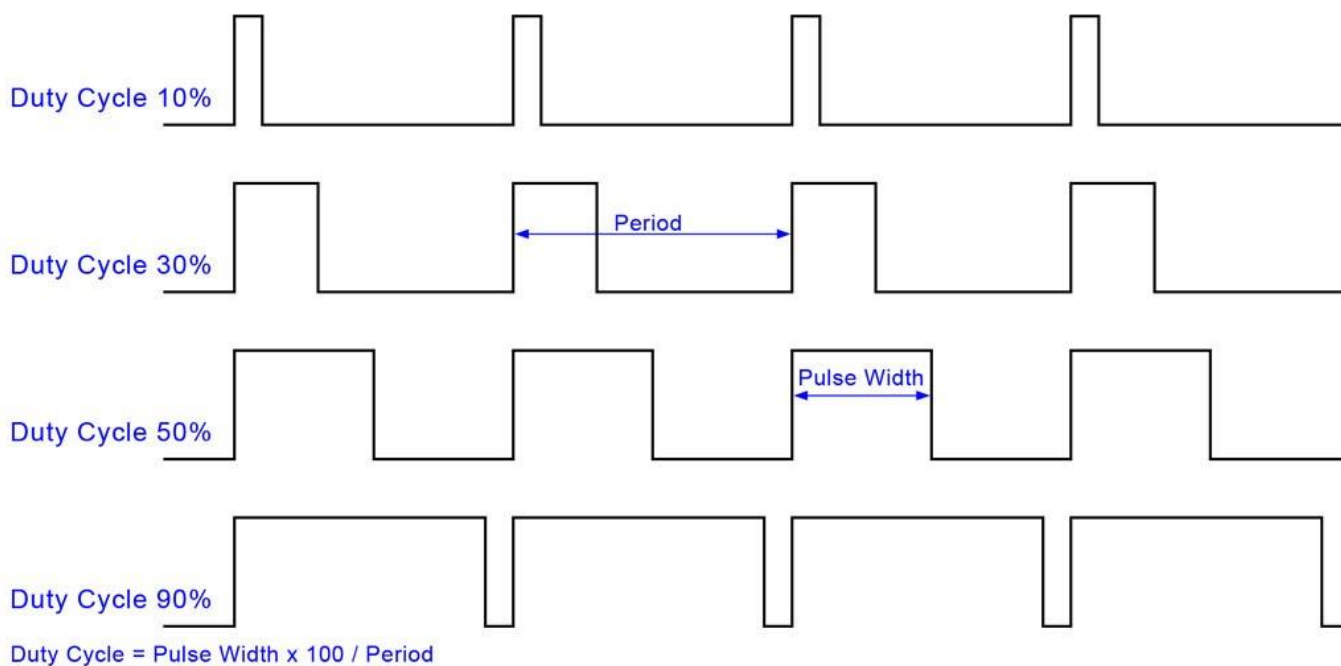
Обратите внимание на то, что данная программа отличается от предыдущей. Во-первых, здесь мы работаем с выводом модуля №14, к которому подключен светодиод. Во-вторых, теперь светодиод загорается после появления на выводе №14 напряжения 3.3В (**pin.write(1)**) и перестает гореть после появления на выводе №14 нулевого напряжения (**pin.write(0)**).

Загрузите данную программу в модуль, запустите ее и проверьте, что светодиод действительно мигает. Попробуйте изменять задержку функции **time.sleep(1)**: задать задержку не 1 секунда, а 0.3 секунды.

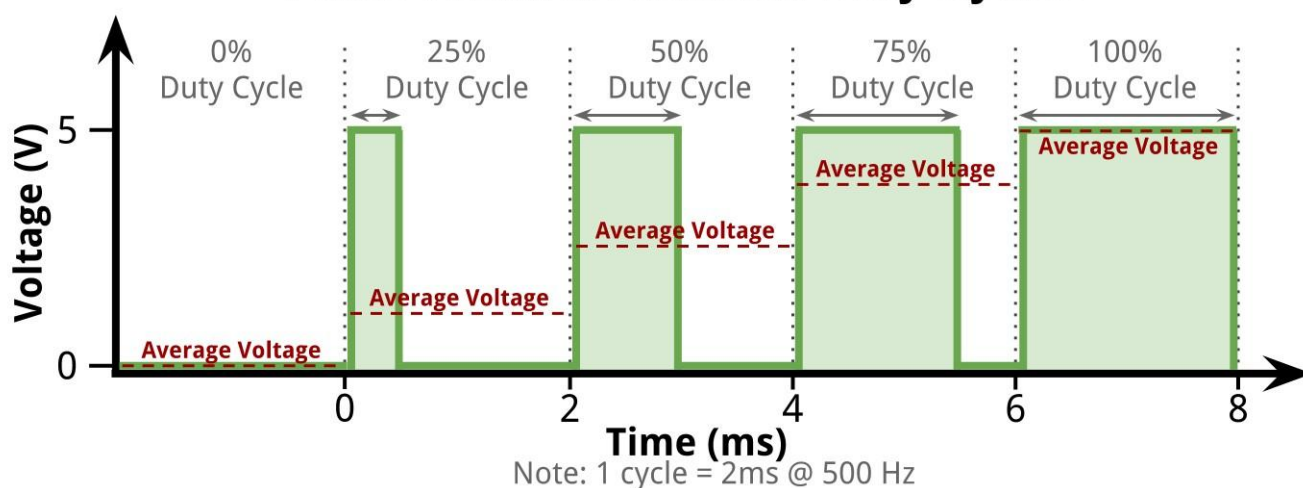
Работа 3.

Управляем яркостью светодиода с помощью широтно-импульсной модуляции (PWM)

Широтно-импульсная модуляция (PWM) позволяет изменять среднее напряжение на выходе микросхемы за счет изменения длительности импульса при неизменном периоде импульсов.

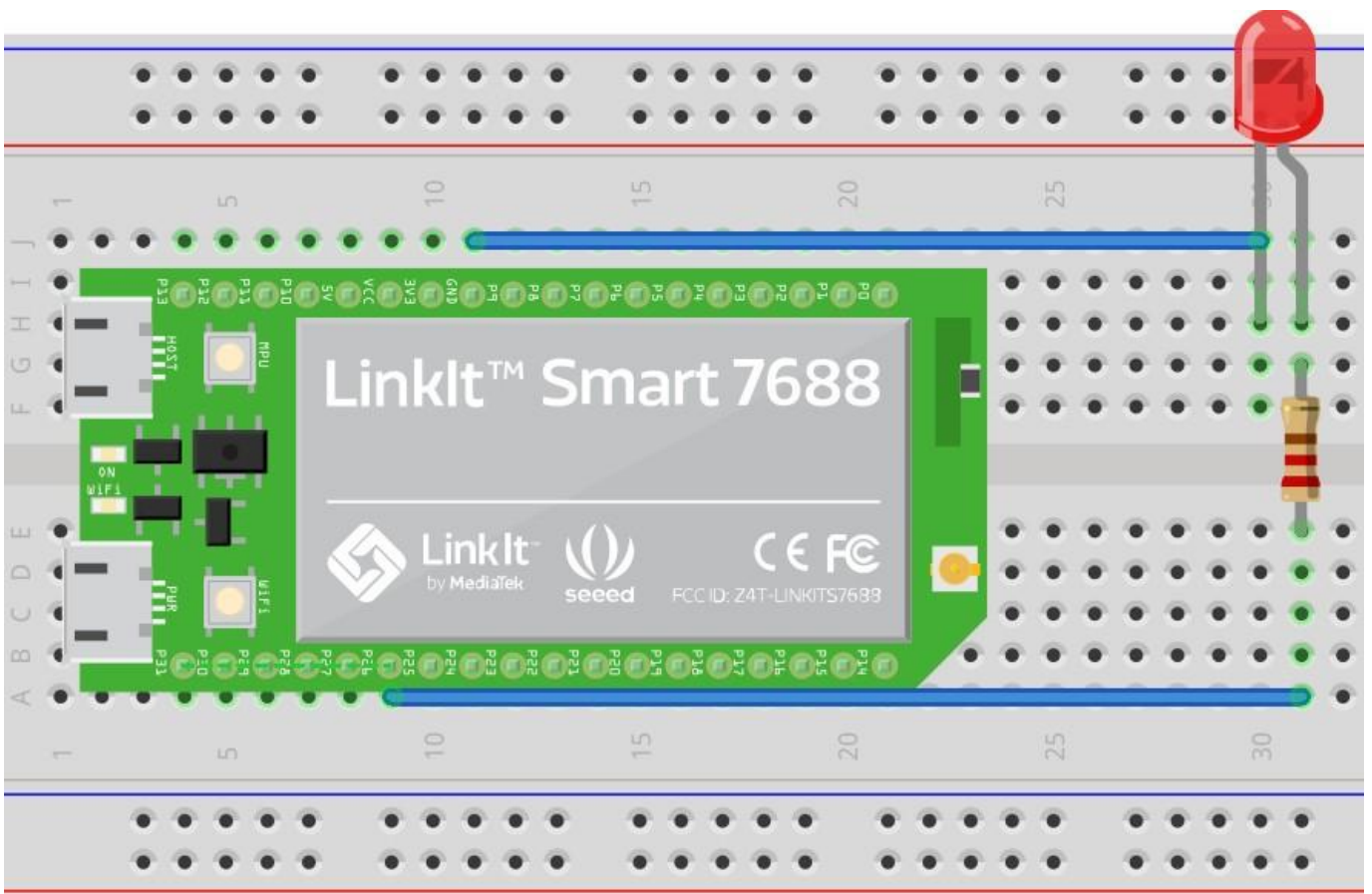


Pulse Width Modulation Duty Cycles



С помощью PWM мы будем менять среднее напряжение, которое подается на светодиод, что приведет к изменению яркости его свечения.

Соберите схему, в которой светодиод подключен к выходу PWM:



На своем компьютере создайте текстовый файл с именем "pwm.py". Добавьте в файл следующий код на языке Python:

```
pwm.py (~/Desktop/examples)
File Edit View Search Tools Documents Help
import mraa
import time

pin = mraa.Pwm(18)
pin.period_ms(2)
pin.enable(True)
pin.write(0.25)

while True:
    time.sleep(1)
```

С помощью **pin = mraa.Pwm(18)** создается PWM сигнал на выводе 18.

С помощью **pin.period_ms(2)** задается частота сигнала PWM равная 500 Гц (период 2 мс).

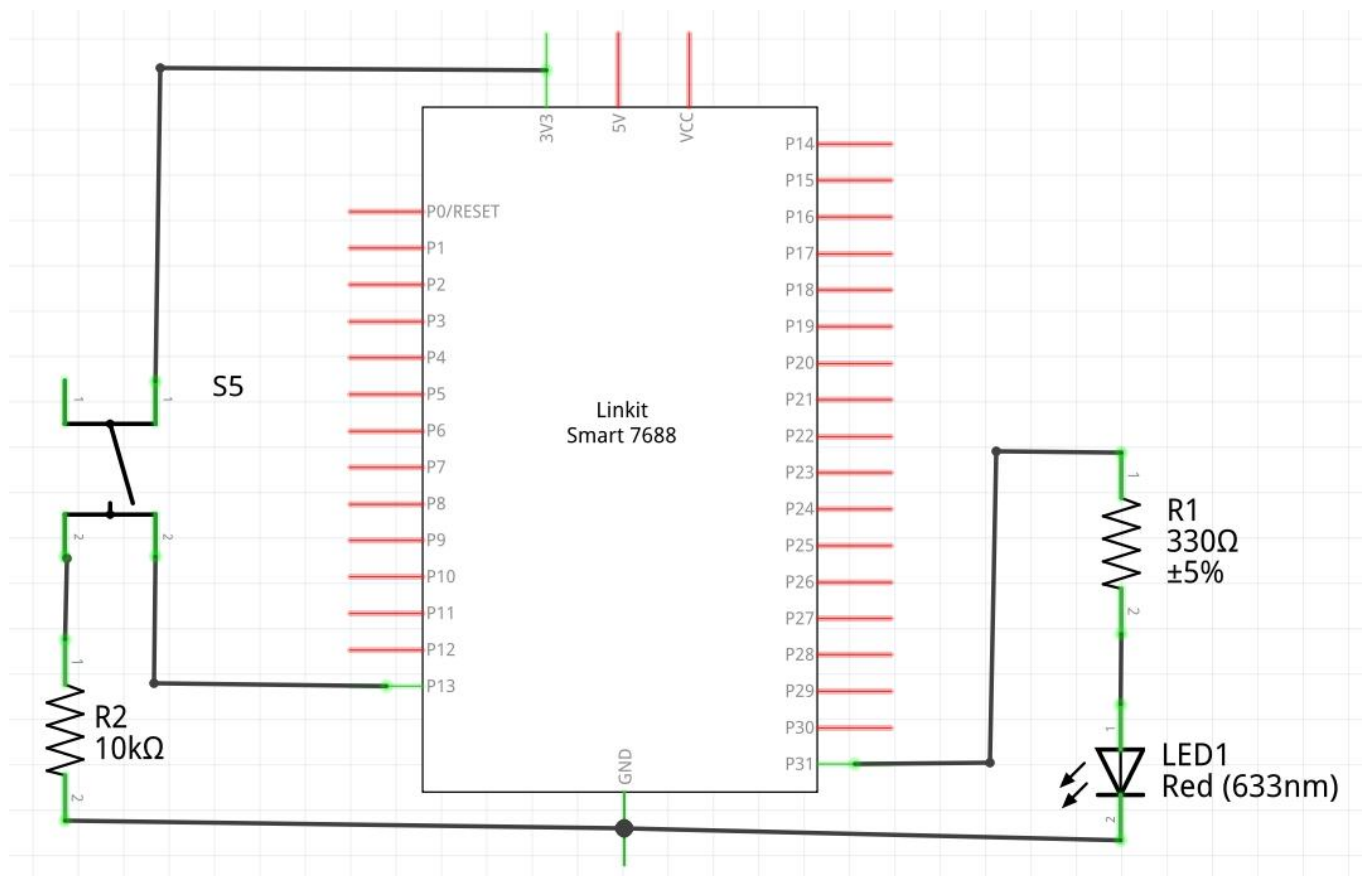
С помощью **pin.enable(True)** активируется PWM выход.

С помощью **pin.write(0.25)** задается коэффициент заполнения (duty cycle).

Загрузите программу **pwm.py** в модуль и запустите ее. Попробуйте изменять коэффициент заполнения, делая его равным 0.1, 0.3, 0.5, 0.75. Убедитесь в том, что это приводит к изменению яркости свечения светодиода.

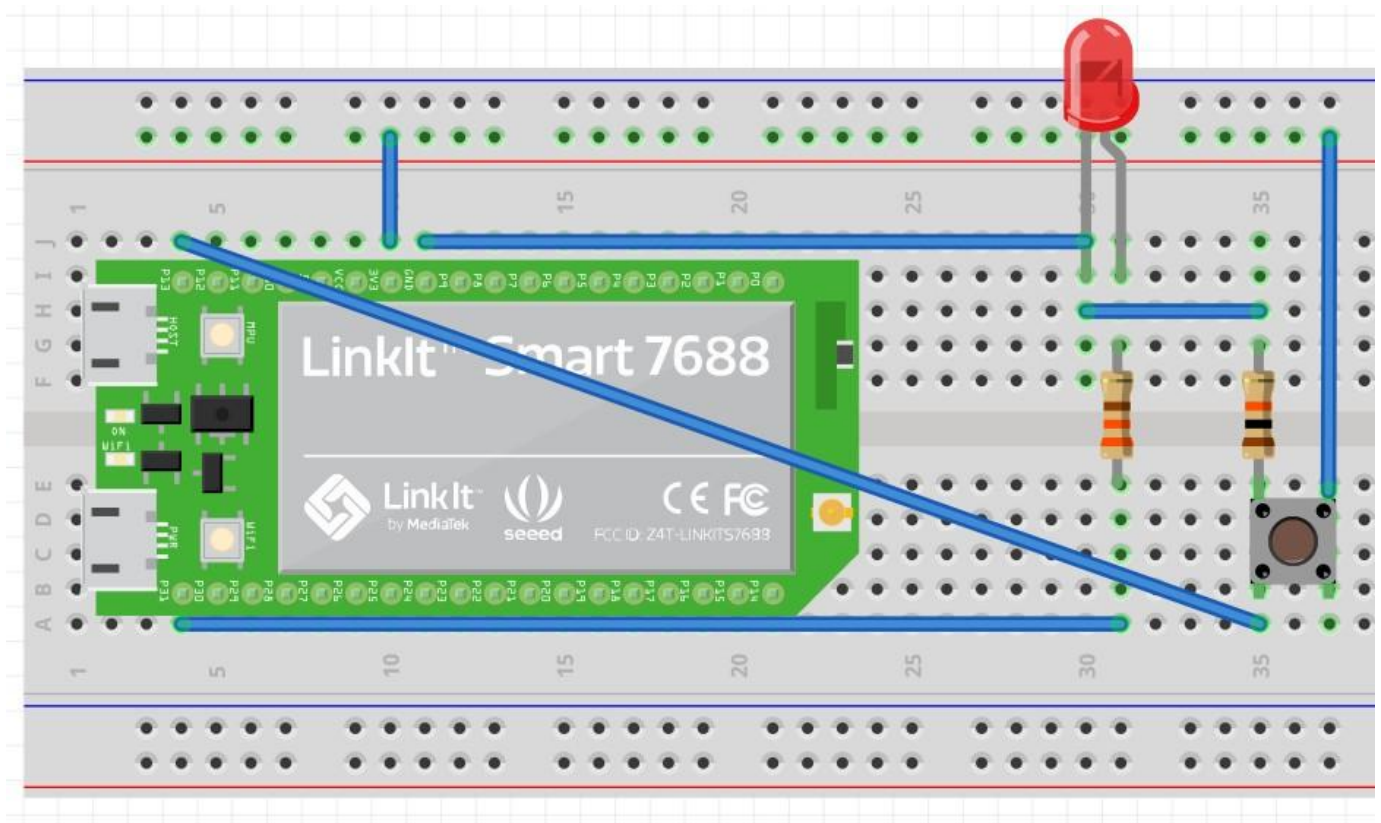
Работа 4. Подключаем кнопку

Сейчас мы подключим к модулю Linkit Smart 7688 кнопку. Схема подключения приведена ниже. Когда мы с помощью USB кабеля подаем питание на модуль, на выводе 3V3 появляется напряжение 3.3 В. Когда переключатель разомкнут (кнопка отпущенная), контакт №1 (обозначен на плате модуля как P13) будет соединен с землей (GND, потенциал 0 В) через резистор. Если мы нажмем кнопку, переключатель замкнется, через резистор начнет протекать ток и к резистору будет приложено напряжение 3.3 В, которое также будет присутствовать на контакте №1. Сопротивление резистора, подключенного к кнопке выбрано равным 10 кОм.



Обратите внимание на то, что к модулю Linkit Smart 7688 также подключен светодиод. Далее мы напишем программу на Python, которая будет включать светодиод, когда кнопка нажата. Если кнопки не нажата, светодиод не будет гореть.

Соберите приведенную выше схему на макетной плате:



На своем компьютере создайте текстовый файл с именем "button.py". Добавьте в файл следующий код на языке Python:

```
button.py (~/Desktop/examples)
File Edit View Search Tools Documents Help
button.py x blink2.py x
import mraa
import time

button = mraa.Gpio(1)
button.dir(mraa.DIR_IN)

led = mraa.Gpio(14)
led.dir(mraa.DIR_OUT)

button_state = 0

while True:
    button_state = button.read()
    print "GPIO1 state:", button_state
    led.write(button_state)
    time.sleep(0.1)
```

Часть программы, управляющая светодиодом, уже знакома вам по предыдущим примерам.

Работа 5. Использование прерывания.

В программе из предыдущего примера есть недостаток. Необходимо достаточно часто считывать значение кнопки, чтобы не пропустить момент, когда ее нажимают или отпускают. Если программа вместо считывания значения кнопки будет делать что-то другое (например, управлять светодиодом, отображать код веб-страницы и т.д.) и в это время кто-то нажмет кнопку, а затем быстро ее отпустит, программа этого не заметит, поскольку она не считывала значение кнопки в этот момент.

Для решения такой проблемы используют принцип прерываний. Сначала пишут основную программу, которая выполняет определенную функцию. Затем создают вспомогательную программу, которая называется "подпрограмма обработки прерывания". Определяют условие, после выполнения которого, запустится подпрограмма обработки прерывания. Идея заключается в том, что большую часть времени выполняется основная программа. Если выполняется определенное условие, основная программа останавливается (прерывается) и запускается подпрограмма обработки прерывания. После окончания подпрограммы обработки прерывания, основная программа продолжает свою работу с того места, где ее прервали. Если подпрограммы обработки прерываний будут короткими и их выполнение будет занимать мало времени, они почти не влияют на работу основной программы.

Рассмотрим принцип работы прерываний на примере. Будем в основной программе мигать светодиодом, а на нажатие кнопки реагировать в в программе обработки прерывания. Электронную схему оставим из предыдущего задания, но изменим программу на Python.

На своем компьютере создайте текстовый файл с именем "interrupt.py". Добавьте в файл следующий код на языке Python:

```
interrupt.py (~/Desktop/examples)
File Edit View Search Tools Documents Help
interrupt.py x
import mraa
import time

def callback(userdata):
    print button.read()

global button
button = mraa.Gpio(1)
button.dir(mraa.DIR_IN)
button.isr(mraa.EDGE_BOTH, callback, None)

led = mraa.Gpio(14)
led.dir(mraa.DIR_OUT)

while True:
    led.write(1)
    time.sleep(0.5)
    led.write(0)
    time.sleep(0.5)
```

Код для мигания светодиодом вам уже знаком с предыдущих примеров.

С помощью

```
def callback(userdata):  
    print button.read()
```

описываем подпрограмму обработки прерывания, которая будет вызываться после выполнения определенного условия. Имя подпрограммы обработки прерывания - **callback**. Однако вы можете задать другое имя. Единственное, что делает подпрограмма обработки прерывания в данном примере - с помощью функции **print** выводит на консоль модуля Linkit Smart 7688 значения на входе, к которому подключена кнопка, которое считано с помощью **button.read()**. Обратите внимание на то, что переменная **button**, описывающая вывод модуля к которому подключена кнопка, создается в основной программе, а считывается она в подпрограмме обработки прерывания. Для того, чтобы переменная **button**, созданная в основной программе, была доступна в подпрограмме обработки прерывания, необходимо сделать эту переменную глобальной, написав в основной программе **global button** перед записью в переменную **button**.

С помощью

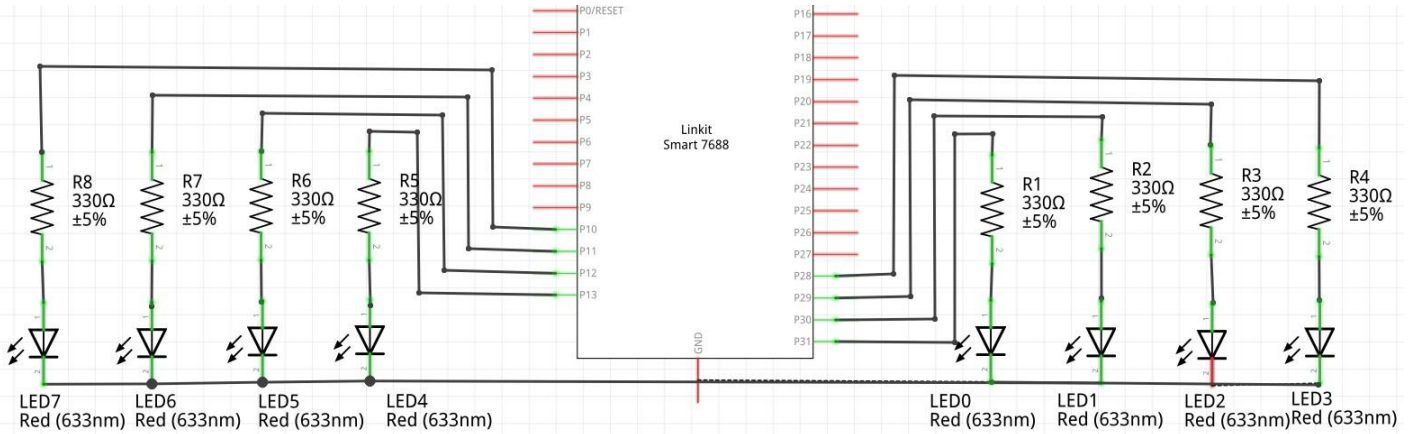
button.isr(mraa.EDGE_BOTH, callback, None)

мы определяем, что после изменения напряжения на входе, к которому подключена кнопка **button**, необходимо сделать прерывание и вызвать программу с именем **callback**. **EDGE_BOTH** означает, что программу обработки прерывания необходимо выполнять как при изменении напряжения на входе от 0 до 3.3 В, так и при изменении напряжения на входе от 3.3 до 0 В. Третий аргумент всегда должен быть **None**.

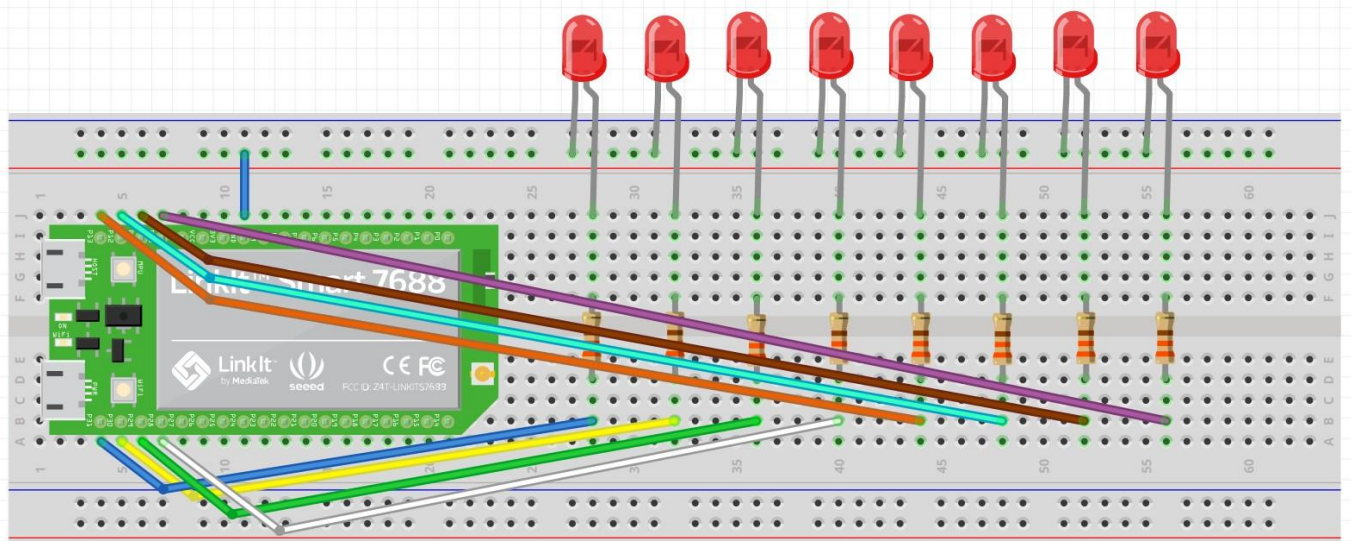
Загрузите программу **interrupt.py** в модуль и запустите ее. Убедитесь в том, что светодиод мигает и при этом сразу после нажатия на кнопку на консоль выводится число 1, а сразу после отпускания кнопки на консоль выводится число 0. Как видите, подпрограмма обработки прерывания вызывается только после изменения напряжения на входе, к которому подключена кнопка (при нажатии или отпускании кнопки).

Работа 6. Программная реализация бегущих огней

В соответствии со следующей схемой подключите к модулю Linkit Smart 7688 восемь светодиодов:



Соединение на макетной плате:



Сопротивление резисторов должно быть в пределах нескольких сотен Ом (например, 330 Ом)

На своем компьютере создайте текстовый файл с именем **"running_leds.py"**. Добавьте в файл следующий код на Python

```
running_leds.py x
import mraa
import time

def drive_leds(leds, state):
    for i in range(len(leds)):
        leds[i].write(state >> i & 0x01)

led0 = mraa.Gpio(14)
led1 = mraa.Gpio(15)
led2 = mraa.Gpio(16)
led3 = mraa.Gpio(17)
led4 = mraa.Gpio(1)
led5 = mraa.Gpio(0)
led6 = mraa.Gpio(3)
led7 = mraa.Gpio(2)

led0.dir(mraa.DIR_OUT)
led1.dir(mraa.DIR_OUT)
led2.dir(mraa.DIR_OUT)
led3.dir(mraa.DIR_OUT)
led4.dir(mraa.DIR_OUT)
led5.dir(mraa.DIR_OUT)
led6.dir(mraa.DIR_OUT)
led7.dir(mraa.DIR_OUT)

leds = [led0, led1, led2, led3, led4, led5, led6, led7]

state = 1

while True:
    drive_leds(leds, state)
    state = state << 1 | ~(state >> 7) & 0x01
    time.sleep(0.5)
```

Как видите, мы создаем восемь переменных (**led0-led7**), каждая из которых описывает контакт модуля, к которому подключен светодиод. Далее настраиваем эти восемь контактов для работы на выход (чтобы можно было программно устанавливать на каждом выходе напряжение 0 В или 3.3 В). После этого создаем массив **leds**, элементами которого являются переменные **led0-led7**:

```
leds = [led0, led1, led2, led3, led4, led5, led6, led7]
```

Далее создаем переменную **state**, каждый бит которой будет определять светится соответствующий светодиод или нет. Бит переменной **state** с номером 0 соответствует светодиода с номером 0, бит переменной **state** с номером 7 соответствует светодиода с номером 7 и т.д. Если соответствующий бит принимает значение 1, светодиод должен светиться. Если соответствующий бит принимает значение 0, светодиод не должен светиться. При создании переменной **state** присваиваем ей значение 1. Это означает, что в таком

случае должен светиться только светодиод с номером 0, так как только бит с номером 0 принимает значение 1.

Функция `drive_leds` зажигает светодиоды массива `leds` в соответствии со значениями битов переменной `state`:

```
def drive_leds(leds, state):  
    for i in range(len(leds)):  
        leds[i].write(state >> i & 0x01)
```

Как видите, функция принимает два аргумента - массив `leds` и переменную `state`. Внутри функции, в цикле `for`, проходим по каждому биту переменной `state` (с номерами от 0 до 7) и в зависимости от значения бита светодиод с соответствующим номером или зажигается или гаснет. Переменная `i` содержит номер итерации цикла. Функция `len` - это стандартная функция Python, которая возвращает размер (количество элементов) массива, который передается в функцию `len`. Функция `range` (`arg`) возвращает массив чисел в количестве `arg` со значениями от 0 до `arg-1`. Например:

```
>>> range(10)  
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Таким образом, цикл `for i in range (arg)` выполнит операции тела цикла `arg` раз, при этом на каждой итерации цикла переменная `i` будет принимать новое значение из диапазона от 0 до `arg-1`. Изменение значений переменной `i` будет происходить упорядочено: `0, 1, 2, 3, ..., arg-2, arg-1`. Для доступа к элементу массива `leds` с номером `i` используется `leds [i]`.

Конструкция `state >> i & 0x01` позволяет выделить значение `i`-го бита переменной `state`. Для этого сдвигаем переменную `state` вправо на `i` бит (`i`-й бит становится битом с номером 0, то есть самым младшим), а затем выполняем побитовую операцию AND результата с числом 1, то есть обнуляем все биты кроме младшего. Если `i`-й бит переменной `state` принимал значение 1, то в результате данной операции получим значение 1. Если `i`-й бит переменной `state` принимал значение 0, в результате данной операции получим значение 0.

В бесконечном цикле основной программы выполняется следующее:

while True:

drive_leds(leds, state)

state = state << 1 | ~(state >> 7) & 0x01)

time.sleep(0.5)

На каждой итерации цикла с помощью **drive_leds (leds, state)** выводим на светодиоды значение битов переменной **state**.

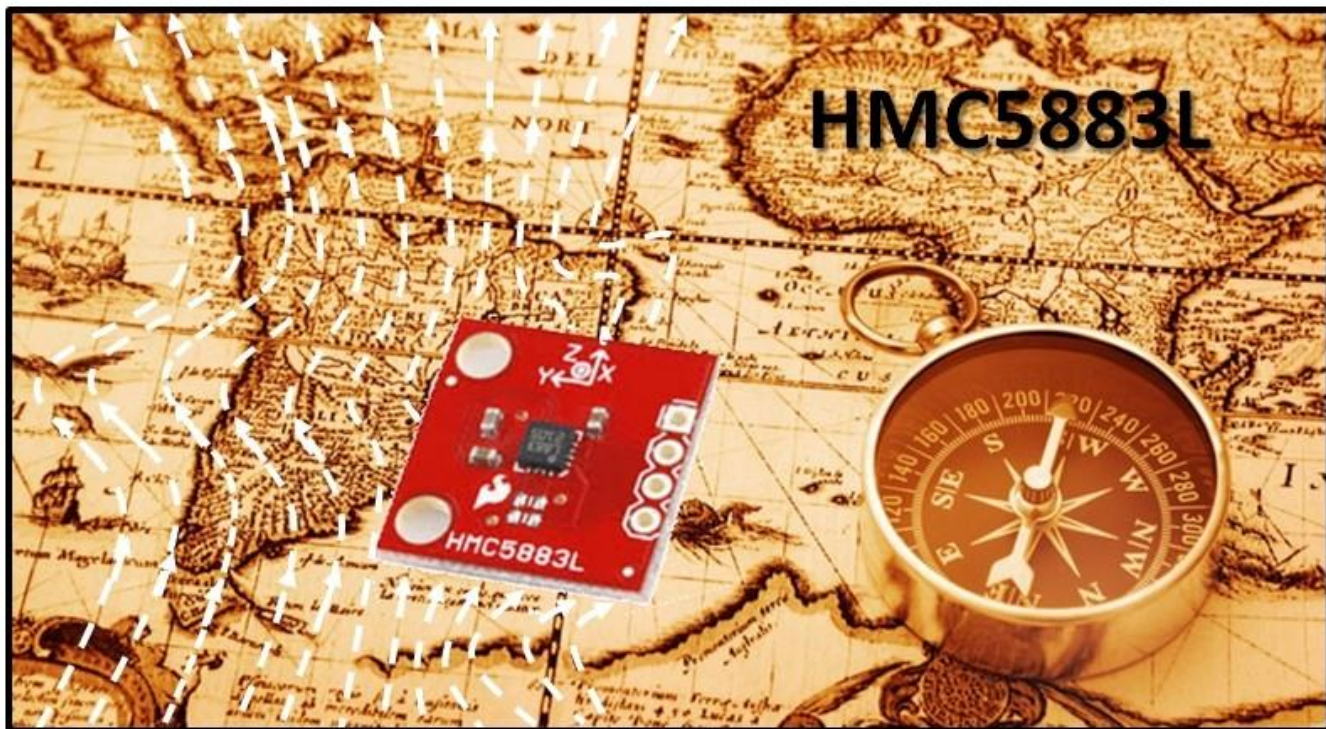
После этого с помощью **state << 1 | (~ (State >> 7) & 0x01)** сдвигаем содержимое переменной **state** на один разряд влево и при этом переписываем инвертированное значение 7-го бита старого значения **state** в нулевой бит результата выражения. Полученный результат записываем в переменную **state**. Попробуйте представить, как это происходит.

Далее делаем задержку на 0.5 секунд с помощью **time.sleep(0.5)**. После завершения задержки переходим к следующей итерации бесконечного цикла и описанные действия повторяются.

Загрузите программу **running_leds.py** в модуль и запустите ее. Убедитесь, что светодиоды мигают в соответствии с алгоритмом работы программы.

Работа 7. Подключаем электронный компас по интерфейсу I2C

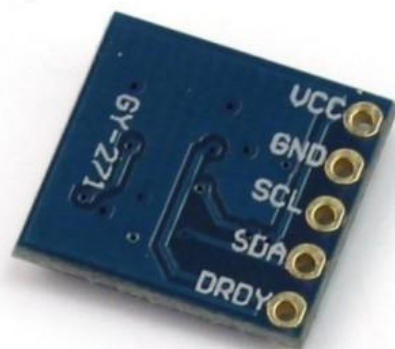
В данной работе мы подключим к модулю Linkit Smart 7688 электронный компас (магнитометр) HMC5883L через интерфейс I2C. Будем считывать значения из компаса и выводить их в терминал модуля.



Электронный компас (магнитометр) HMC5883L может выглядеть по-разному:



Gnd
Vcc
SDA
SCL



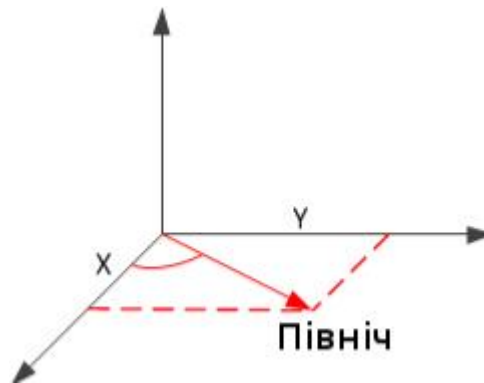
Однако у всех вариантов модуля есть общие контакты: VCC, GND, SDA, SCL.

К контакту VCC подключают «+» источника напряжения 3 ... 5 В.

К контакту GND подключают "-" этого же источника напряжения.

Контакты SDA и SCL используются для передачи данных по интерфейсу I2C

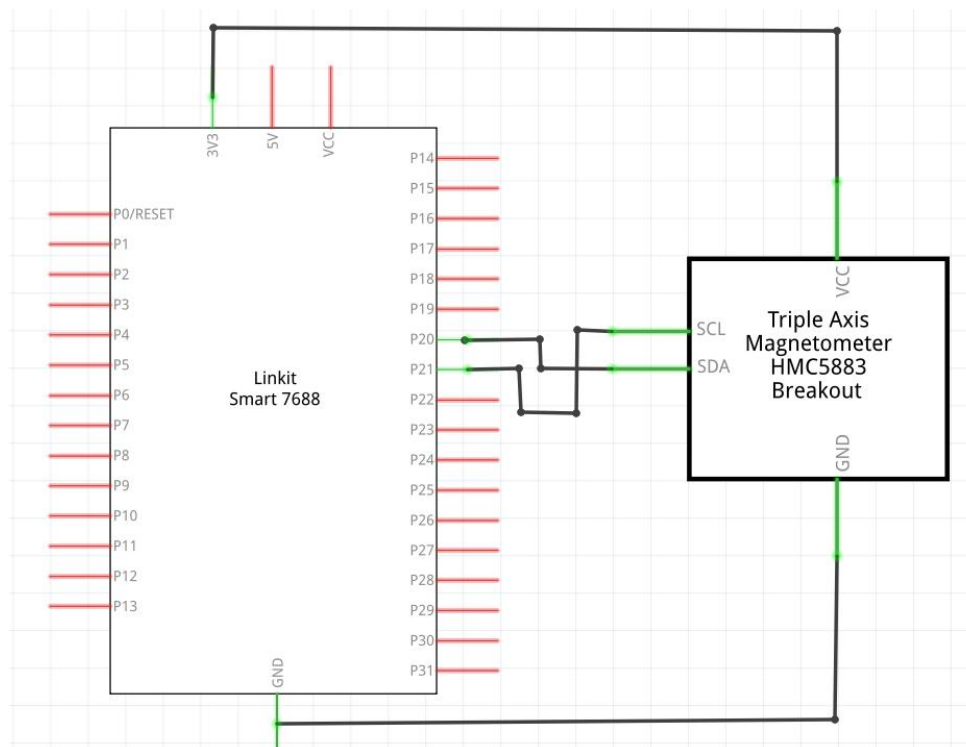
Электронный компас (магнитометр) HMC5883L позволяет измерять компоненты напряженности (силы) магнитного поля Земли по осям X, Y, Z. Из этих данных можно рассчитать угол между направлением на север и направлением на который указывает ось x компаса, как $\arctg(H_y/H_x)$, где H_y и H_x это напряженности магнитного поля Земли по осям Y и X компаса HMC5883L:



Для модуля Linkit Smart 7688 существует библиотека UPM (портированная с Intel Edison), которая позволяет очень просто взаимодействовать с множеством датчиков из программ на языках C, Python, NodeJS. Примеры работы с разными датчиками с использованием UPM и программ на Python: <https://github.com/intel-iot-devkit/upm/tree/master/examples/python>

Датчик HMC5883L также поддерживается библиотекой UPM.

Сначала подключите HMC5883L к модулю Linkit Smart 7688:



Контакт GND датчика HMC5883L необходимо подключить к контакту GND модуля Linkit Smart 7688.

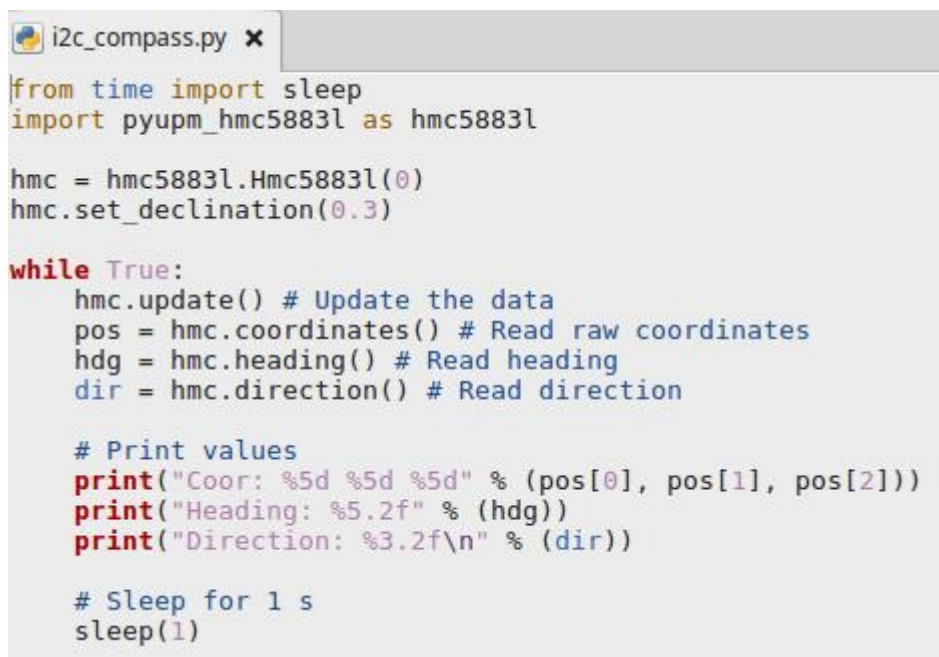
Контакт VCC датчика HMC5883L необходимо подключить к контакту 3.3V модуля Linkit Smart 7688.

Контакт SCL датчика HMC5883L необходимо подключить к контакту модуля Linkit Smart 7688 с обозначением P21 (выход SCL модуля Linkit Smart 7688).

Контакт SDA датчика HMC5883L необходимо подключить к контакту модуля Linkit Smart 7688 с обозначением P20 (это контакт SDA модуля Linkit Smart 7688).

На своем компьютере создайте текстовый файл с именем "i2c_compass.py".

Добавьте в этот файл следующий код на языке Python:



```
i2c_compass.py x
from time import sleep
import pyupm_hmc5883l as hmc5883l

hmc = hmc5883l.Hmc5883l(0)
hmc.set_declination(0.3)

while True:
    hmc.update() # Update the data
    pos = hmc.coordinates() # Read raw coordinates
    hdg = hmc.heading() # Read heading
    dir = hmc.direction() # Read direction

    # Print values
    print("Coord: %5d %5d %5d" % (pos[0], pos[1], pos[2]))
    print("Heading: %5.2f" % (hdg))
    print("Direction: %3.2f\n" % (dir))

    # Sleep for 1 s
    sleep(1)
```

Программа довольно проста. С помощью `hmc = hmc5883l.Hmc5883l(0)` создаем переменную `hmc`, которая будет описывать электронный компас. Магнитное поле в разных частях Земли несколько искривляется и для коррекции полученных значений необходимо выполнить `hmc.set_declination(0.3)`. Подробнее об определении этого параметра можно почитать по ссылке:

<http://www.meccanismocomplesso.org/en/arduino-magnetic-magnetic-magnetometer-hmc5883l>

На каждой итерации бесконечного цикла считываем показания датчика HMC5883L с использованием `hmc.update()`. С помощью `pos = hmc.coordinates()` считываем компоненты напряженности магнитного поля Земли по осям X, Y, Z в массив `pos`. С помощью `dg = hmc.heading()` считываем угол между направлением на север и направлением, на который указывает ось x компаса. Далее выводим полученные значения на консоль модуля Linkit Smart 7688, делаем задержку на 1 секунду и переходим к следующей итерации бесконечного цикла.

Загрузите программу `i2c_compass.py` в модуль и запустите ее. Убедитесь в том, что при повороте макетной платы изменяется величина угла между направлением на север и направлением, на который указывает ось x компаса (Heading).

Для студентов, имеющих опыт программирования и работы с интерфейсами электронных устройств, есть задачи повышенной сложности. С датчиком HMC5883L можно работать не только с помощью библиотеки UPM, а и напрямую через интерфейс I2C (библиотека MRAA позволяет работать напрямую с I2C). Вы можете считывать и записывать данные в регистры датчика, подключенного к модулю с помощью интерфейса I2C.

Программный код считывания показаний датчика HMC5883L напрямую через интерфейс I2C из программы на языке Python приведен ниже:

```
root@mylinkit:~# python i2c_compass.py
Coord: 359 -98 -396
Heading: 1.92
Direction: 0.03

Coord: 359 -96 -398
Heading: 2.22
Direction: 0.04

Coord: 357 -97 -399
Heading: 1.99
Direction: 0.03
```

```
i2c_compass2.py x
import mraa
from math import atan2
from time import sleep

def toSigned(val):
    if (val >= 32768):
        return val - 65535 + 1
    else:
        return val

i2c = mraa.I2c(0)

i2c.address(0x1e)

i2c.writeByte(0x02)
i2c.writeByte(0x00)

while True:

    i2c.writeByte(0x03)
    x = i2c.read(2)
    xval = toSigned(x[0] << 8 | x[1]);

    i2c.writeByte(0x05)
    z = i2c.read(2)
    zval = toSigned(z[0] << 8 | z[1])

    i2c.writeByte(0x07);
    y = i2c.read(2);
    yval = toSigned(y[0] << 8 | y[1])

    direction = atan2(yval, xval) + 0.31;

    heading = direction * 180/3.14;

    if (heading < 0):
        heading += 360.0;

    print "x=%d" % ( xval)
    print "y=%d" % ( yval )
    print "z=%d" % ( zval)
    print "direction = %d" % (direction)
    print "heading = %d" % (heading)

    sleep(1)
```

Вначале надо задать адрес датчика на шине I2C (это константа 0x1E) с помощью **i2c.address(0x1e)**. Чтобы записать данные в определенный регистр модуля, необходимо сначала записать в модуль номер регистра и, сразу после этого, значение регистра. Например, для того, чтобы перевести датчик HMC5883L в режим постоянного считывания необходимо записать 0 в регистр с адресом 0x02. Для этого необходимо выполнить:

i2c.writeByte(0x02)

i2c.writeByte(0x00)

Чтобы считать данные из регистра модуля, необходимо сначала записать в модуль номер регистра и, сразу после этого, считать значение регистра. Поскольку данные о напряженности магнитного поля занимают 16 бит, необходимо считывать 2 байта подряд. Данные о напряженности магнитного поля Земли по оси X содержатся в регистре модуля с адресом 0x03. Данные о напряженности магнитного поля Земли по оси Z содержатся в регистре модуля с адресом 0x05. Данные о напряженности магнитного поля Земли по оси Y содержатся в регистре модуля с адресом 0x07.

Более подробно регистры датчика HMC5883L описаны в документации:

https://cdn-shop.adafruit.com/datasheets/HMC5883L_3-Axis_Digital_Compass_IC.pdf

Чтобы считать 2 байта, характеризующие компоненту напряженности магнитного поля Земли вдоль оси X необходимо выполнить:

```
i2c.writeByte(0x03)
```

```
x = i2c.read(2)
```

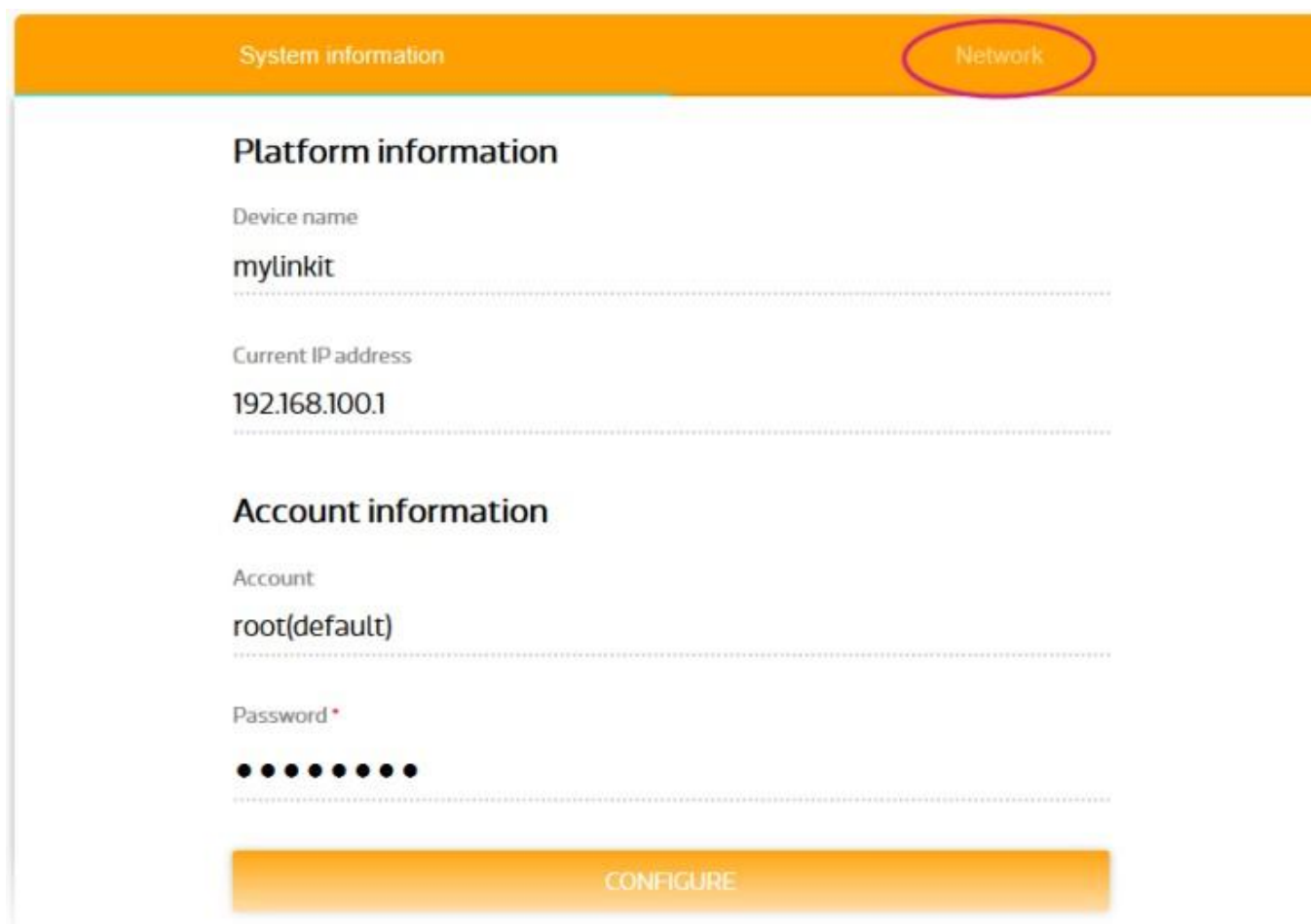
Загрузите программу **i2c_compass2.py** в модуль и запустите ее. Убедитесь, что она работает так же, как и предыдущая программа **i2c_compass.py**.

Подключение к WiFi роутеру и сети Internet

Пока вы работали с модулем в режиме точки доступа. Linkit Smart 7688 работал как точка доступа, к которой вы подключались со своего компьютера. Часто возникает ситуация, когда необходимо подключить модуль к существующему WiFi роутеру. Это может быть необходимо для обмена данными по WiFi между несколькими модулями и для подключения к Internet.

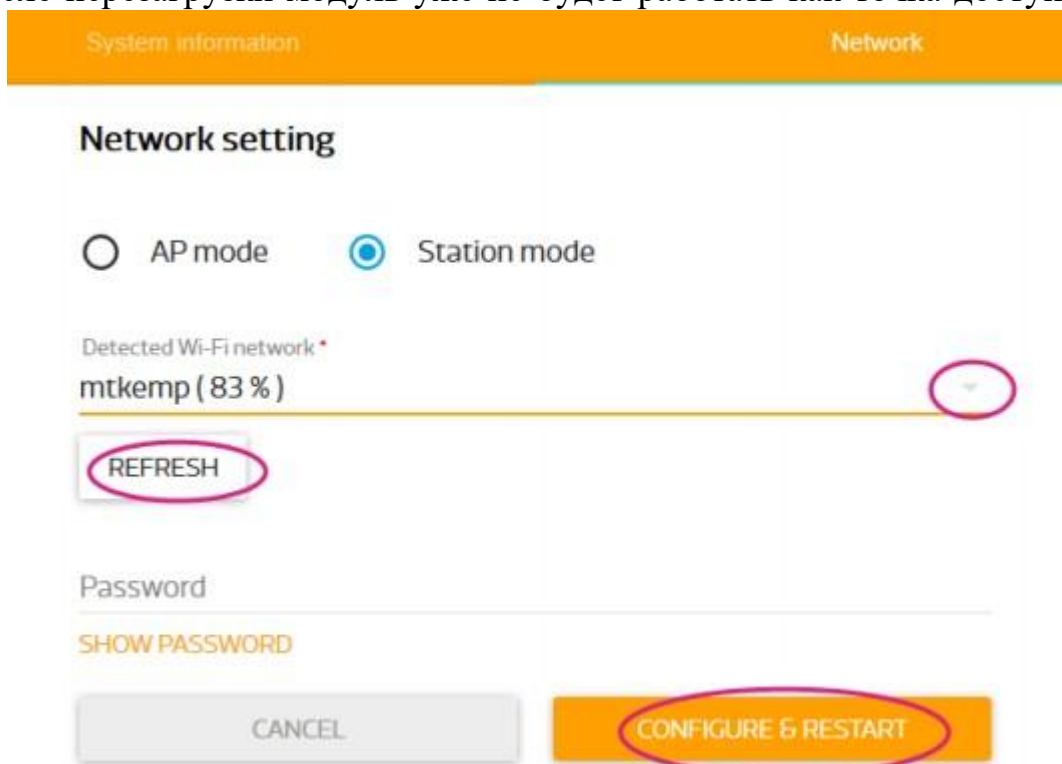
Для перевода модуля Linkit Smart 7688 из режима точки доступа в режим WiFi станции, в котором он может подключиться к WiFi роутеру, необходимо выполнить следующие действия:

1. Выполните WiFi подключение к модулю со своего компьютера, пока модуль еще работает в режиме точки доступа.
2. Зайдите в веб-интерфейс управления модулем (адрес в браузере: <http://mylinkit.local>) и нажмите кнопку **Network** в правом верхнем углу.



3. Выберите **Station Mode** и сеть **WiFi** к которой модуль Linkit Smart 7688 будет подключаться после загрузки (например, это может быть WiFi сеть Belka). Если сеть защищена паролем, установите пароль к WiFi сети (если сеть открытая, пароль устанавливать не нужно). Нажмите на кнопку **CONFIGURE & RESTART**. Это приведет к сохранению настроек и перезагрузка модуля

После перезагрузки модуль уже не будет работать как точка доступа. Вместо



этого Linkit Smart 7688 будет пытаться подключиться к WiFi сети, которую вы только что выбрали.

Если к сети нет доступа или вы указали неверный пароль, модуль не сможет подключиться к заданной WiFi сети. В таком случае для подключения к модулю по WiFi необходимо перевести его обратно в режим точки доступа. Как это сделать показано далее.

Если Linkit Smart 7688 успешно подключился к заданной WiFi сети, то на своем компьютере вы тоже можете подключиться к этой же WiFi сети и для доступа к модулю использовать уже знакомый адрес mylinkit.local

Перевод модуля в режим точки доступа

Предположим, вы настроили модуль Linkit Smart 7688 на подключение к определенной WiFi сети, которая сейчас недоступна. Чтобы подключиться к веб-интерфейсу управления модулем или к терминалу модуля через SSH вам необходимо перевести модуль в режим точки доступа и подключиться к этой точке доступа.

Для перевода модуля Linkit Smart 7688 в режим точки доступа вам необходимо после загрузки модуля нажать кнопку, около которой написано WiFi, и удерживать ее более 5 секунд, но менее 20 секунд (если ее удерживать 20 и более секунд, то будет выполнен сброс настроек модуля к стандартным и удаление созданных и загруженных вами файлов).

Видео инструкция по переводу модуля в режим точки доступа:

<https://www.youtube.com/watch?v=GZ48q2xxflw>

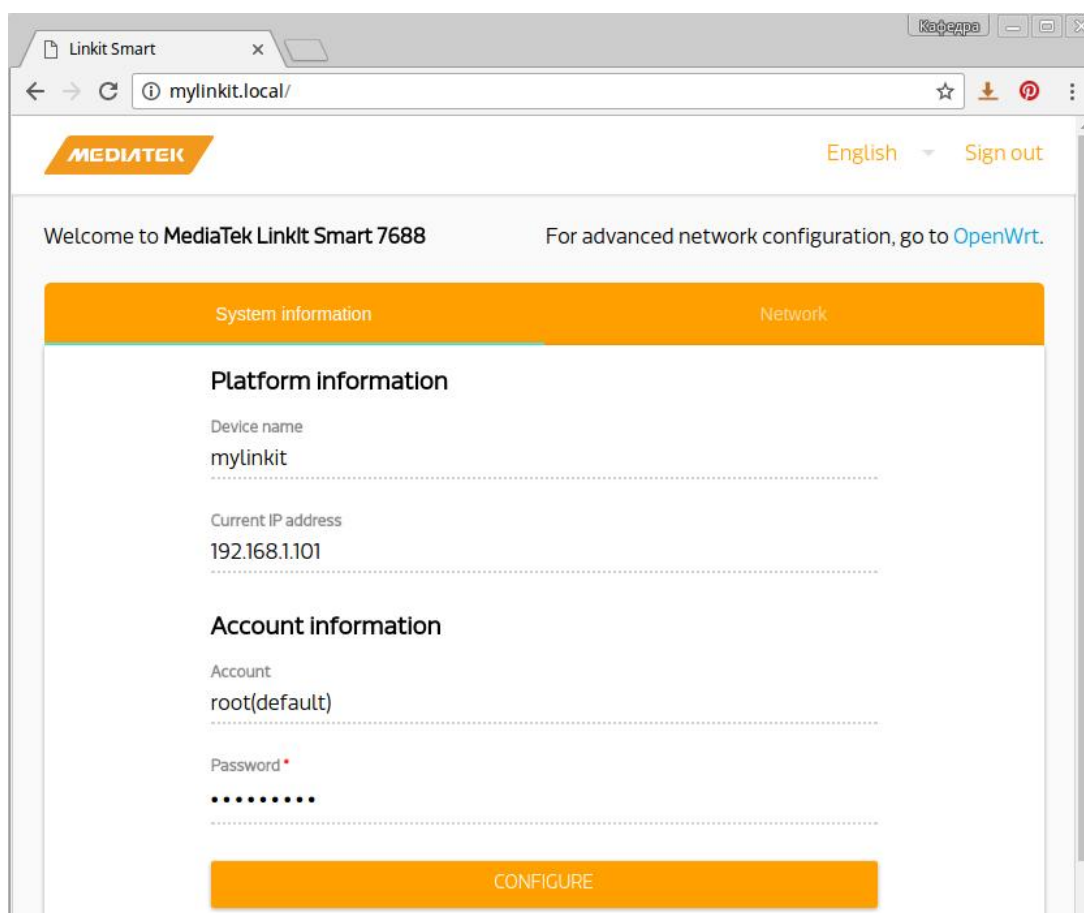
Установка нового имени (dns адреса) модуля

Когда модуль Linkit Smart 7688 работал в режиме точки доступа, для подключения к нему мы использовали адрес <http://mylinkit.local>

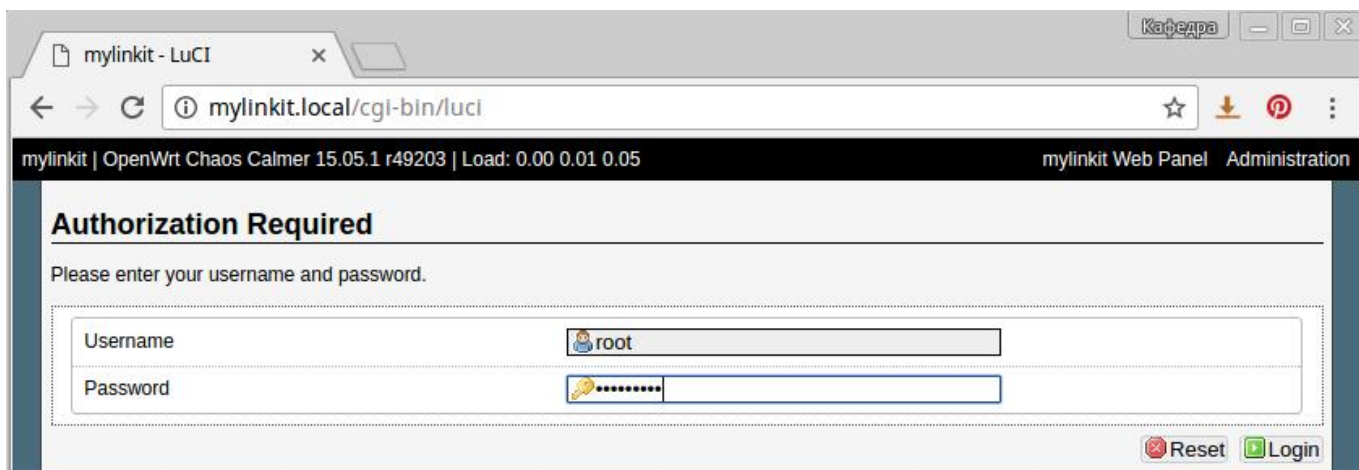
Если в одной сети будут присутствовать несколько модулей Linkit Smart 7688 подключенных к одному WiFi роутеру, то каждому модулю необходимо будет установить уникальное имя.

Для того, чтобы задать уникальное имя для Linkit Smart 7688 выполните следующие действия:

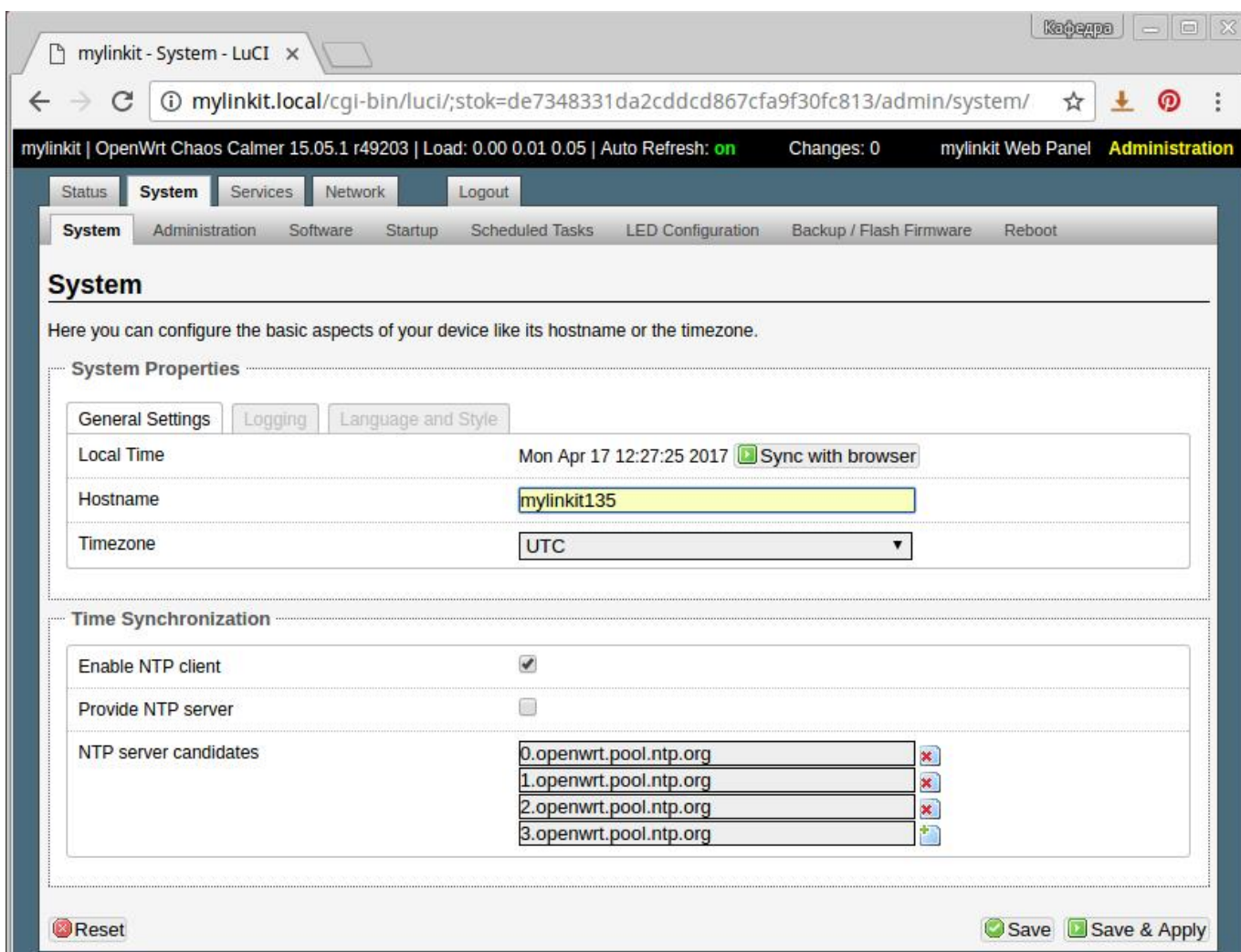
1. Пока модуль еще находится в режиме точки доступа, подключитесь к этой точке доступа по WiFi и зайдите в веб-интерфейс управления по адресу <http://mylinkit.local>
2. Для перехода к панели администрирования OpenWrt нажмите на ссылку OpenWrt в верхнем правом углу.



3. Введите пароль, который вы ранее использовали для доступа к веб-интерфейсу управления и нажмите кнопку **Login**



4. Нажмите на вкладку **System**. В поле **Hostname** введите новое имя модуля, например, mylinkit135 (старое имя было mylinkit). Для сохранения настроек нажмите кнопку **Save & Apply**.



5. Перезагрузите модуль (из веб-интерфейса управления, или отключив и снова подключив USB кабель питания). После перезагрузки новый адрес модуля будет <http://mylinkit135.local>

The screenshot shows a web browser window with the address bar containing `mylinkit135.local/`. The page header features the MediaTek logo on the left, and "English" and "Sign out" links on the right. The main content area is titled "Welcome to MediaTek Linkit Smart 7688" and includes a link to "OpenWrt." for advanced network configuration. Below this is a navigation bar with "System information" and "Network" tabs. The "System information" tab is active, displaying the following details:

- Platform information**
 - Device name: mylinkit135
 - Current IP address: 192.168.1.101
- Account information**
 - Account: root(default)
 - Password: [REDACTED]

At the bottom of the configuration section is a large orange button labeled "CONFIGURE".

Программный код очень прост. Сначала мы подключаем модуль веб-сервера Flask с помощью **from flask import Flask**. Далее с помощью **app = Flask(name)** создаем веб-сервер. Теперь переменная **app** описывает созданный веб-сервер. Обратите внимание на то, что имя веб сервера задаем с помощью встроенной константы **name_____**.

С помощью

```
@app.route('/hello')
```

```
def helloWorldHandler():
```

```
    return 'Hello World from Flask on LinkIt Smart!'
```

создаем функцию, которая вызывается при попытках доступа к веб-странице с именем **"/hello"**. Работает это следующим образом. Когда вы пишете адрес веб-страницы в браузере, браузер подключается к веб-серверу и просит предоставить html код этой веб-страницы (для ее отображения в окне браузера). В таком случае веб-сервер Flask вызывает функцию, которую мы задали для веб-страницы с адресом **"/hello"** и эта функция должна вернуть html код веб-страницы (в виде строки). Этот html код и передается браузеру. В нашем случае возвращается строка **Hello World from Flask on LinkIt Smart!**, которая и будет отображаться в окне браузера.

Наконец запускаем созданный веб-сервер:

```
app.run(host='0.0.0.0', port= 8090)
```

Для подключения к веб-серверу будет использоваться порт **8090**, поскольку стандартный порт **80** уже занят веб-сервером страницы настроек модуля.

Загрузите программу **python_web_server.py** в модуль и запустите ее. Откройте браузер у себя на компьютере, введите в поле адреса <http://mylinkit.local:8090/hello> и нажмите **Enter**.

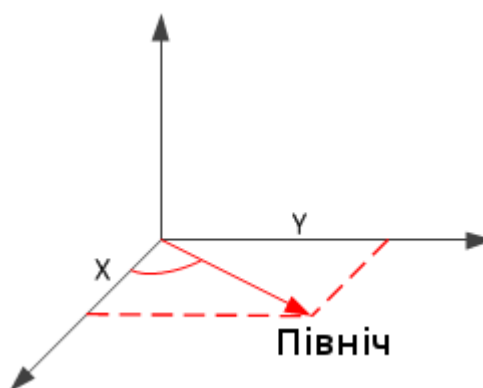
На консоль модуля сразу же выведется информация о том, что браузер с определенного IP адреса (в данном примере это адрес 192.168.1.100) считывает веб-страницу веб-сервера с адресом **"/hello"**

```
root@mylinkit:~# python python_web_server.py
* Running on http://0.0.0.0:8090/ (Press CTRL+C to quit)
192.168.1.100 - - [23/Apr/2017 12:14:21] "GET /hello HTTP/1.1" 200 -
192.168.1.100 - - [23/Apr/2017 12:14:21] "GET /favicon.ico HTTP/1.1" 404 -
```

При этом в окне браузера отобразится строка **Hello World from Flask on LinkIt Smart!**, которую мы вернули из функции **helloWorldHandler()**.



Теперь усложним задачу и создадим более серьезную веб-сайт, который будет отображать результаты, считанные из электронного компаса hmc58831, а именно - выводить угол между направлением на север и направлением на который указывает



ось X компаса.

На своем компьютере создайте текстовый файл с именем **"python_web_server2.py"**. Добавьте в файл следующий код на языке Python:

```
python_web_server2.py x
from flask import Flask
import pyupm_hmc58831 as hmc58831

hmc = hmc58831.Hmc58831(0)
hmc.set_declination(0.3)

app = Flask(__name__)

@app.route('/hello')
def helloWorldHandler():
    hmc.update()
    heading = int(hmc.heading())

    page = "<html> "
    page = page + "<head> "
    page = page + "<meta http-equiv='refresh' content='1' /> "
    page = page + "<title>Linkit 7688 Demo</title> "
    page = page + "<style> "
    page = page + "body { background-color: #cccccc; font-family: Arial, Helvetica, Sans-Serif; Color: #000088; } "
    page = page + "</style>"
    page = page + "</head>"
    page = page + "<body>"
    page = page + "<h1>Hello from Linkit 7688!</h1>"
    page = page + "<p>Compass heading: " + str(heading) + " deg </p>"
    page = page + "</body>"
    page = page + "</html>"

    return page

app.run(host='0.0.0.0', port= 8090)
```

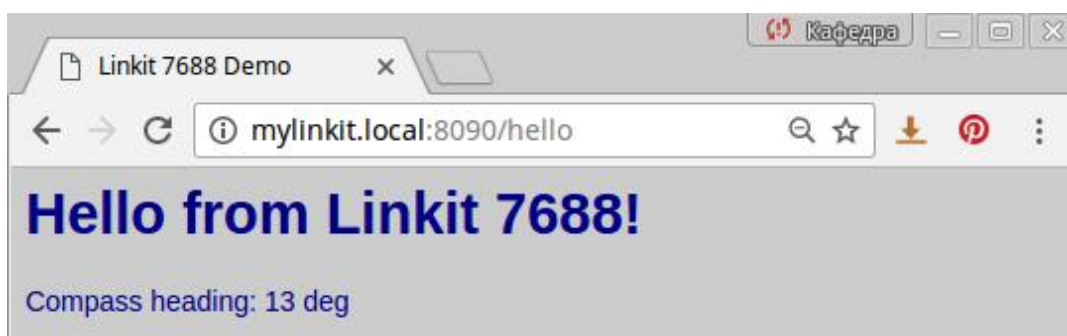

Подключите к модулю Linkit Smart 7688 электронный компас hmc5883l. Как это сделать, было показано в работе №7 данного документа. Программный код для работы с электронным компасом с помощью библиотеки UPM описан в работе №7 данного документа.

Только что созданная программа работает довольно просто. Когда браузер запрашивает от веб-сервера веб-страницу с адресом `"/hello"`, вызывается функция `helloWorldHandler`, которая возвращает веб-серверу html-код веб-страницы в виде строки. В данном примере мы уже используем полноценный html для описания веб-страницы. Для автоматической перезагрузки страницы каждую секунду используем `<meta http-equiv = 'refresh' content = '1'/>`. Каждый раз перед тем, как вернуть код веб-страницы, мы измеряем с помощью электронного компаса угол между направлением на север и направлением на который указывает ось x компаса и вставляем значение этого угла в HTML код веб-страницы:

```
page = page + "<p>Compass heading: " + str(heading) + " deg </p>".
```

Таким образом, вы можете выводить на веб-сайт информацию, полученную от любых датчиков.

Загрузите программу `python_web_server2.py` в модуль и запустите ее. Откройте браузер у себя на компьютере, напишите в поле адреса <http://mylinkit.local:8090/hello> и нажмите **Enter**. Убедитесь в том, что угол между направлением на север и направлением, на который указывает ось x компаса, выводится на веб-сайт и веб-страница обновляется каждую секунду.



Определение IP-адреса модуля

IP-адрес модуля Linkit Smart 7688 можно определить двумя способами:

1. Посмотреть в веб-интерфейсе управления:

Platform information

Device name

mylinkit135

Current IP address

192.168.1.101

2. Подключиться к терминалу модуля через ssh и написать в консоли ifconfig

IP адрес модуля находится справа около записи arcli0. Для данного примера IP адрес будет 192.168.1.101 .

```
root@mylinkit135:~# ifconfig
apcli0  Link encap:Ethernet  HWaddr 9E:65:F9:0E:76:88
        inet addr:192.168.1.101  Bcast:192.168.1.255  Mask:255.255.255.0
        inet6 addr: fe80::9c65:f9ff:fe0e:7688/64  Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:0 errors:0 dropped:0 overruns:0 frame:0
        TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

br-lan  Link encap:Ethernet  HWaddr 9C:65:F9:1E:68:DD
        inet addr:192.168.100.1  Bcast:192.168.100.255  Mask:255.255.255.0
        inet6 addr: fd28:7ece:7856::1/60  Scope:Global
        inet6 addr: fe80::9e65:f9ff:fe1e:68dd/64  Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:0 errors:0 dropped:0 overruns:0 frame:0
        TX packets:135 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:0
        RX bytes:0 (0.0 B)  TX bytes:20461 (19.9 KiB)

eth0    Link encap:Ethernet  HWaddr 9C:65:F9:1E:68:DD
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:0 errors:0 dropped:0 overruns:0 frame:0
        TX packets:135 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:0 (0.0 B)  TX bytes:23630 (23.0 KiB)
        Interrupt:5
```

Обновление прошивки модуля

Процессор модуля Linkit Smart 7688 выполняет операционную систему Linux OpenWrt. Периодически появляются новые версии этой операционной системы с новыми возможностями.

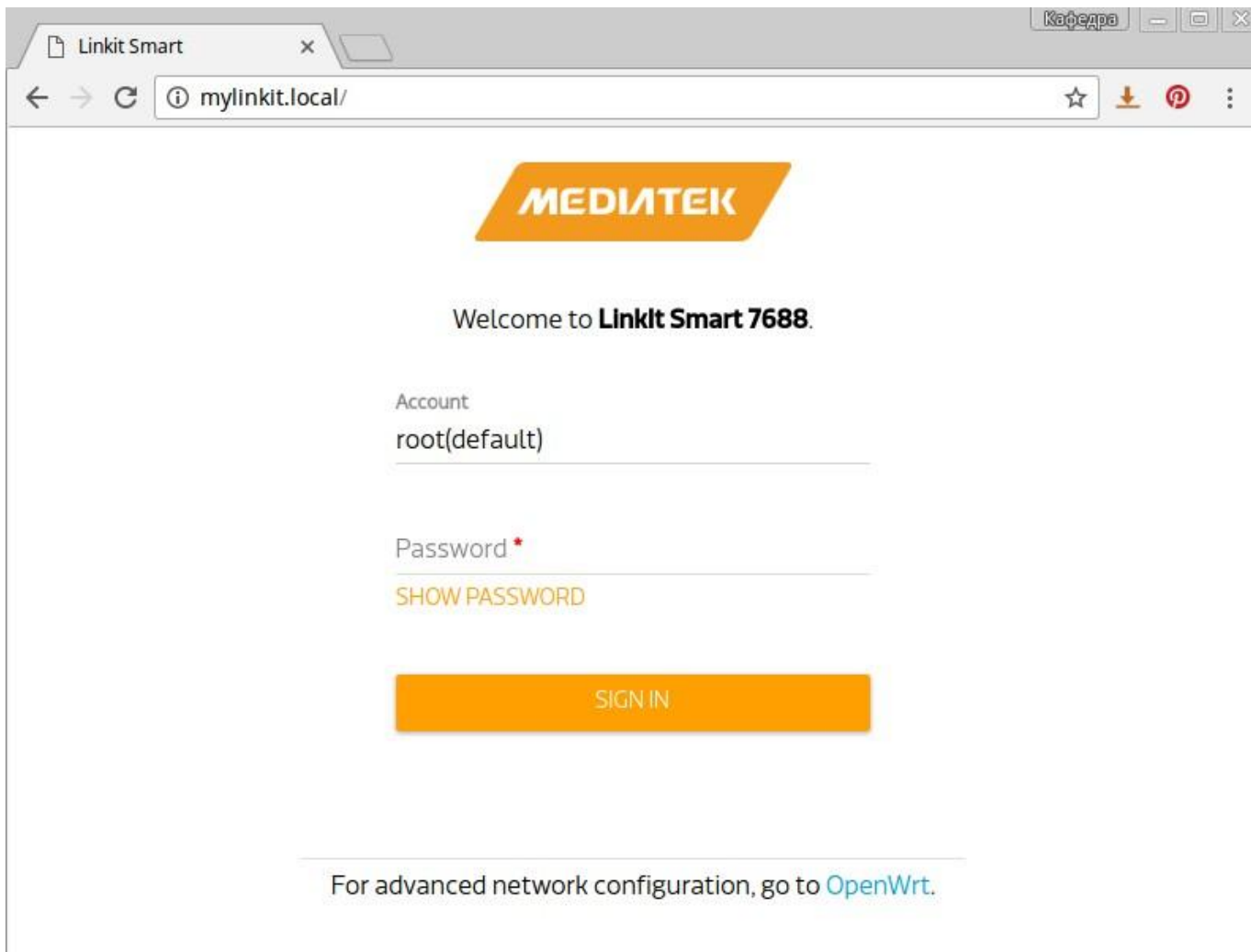
Для обновления версии Linux OpenWrt на модуле Linkit Smart 7688 выполните следующие действия:

1. Загрузите последнюю версию соответствующей прошивки с сайта производителя модуля (файл **lks7688.img**):

<https://docs.labs.mediatek.com/resource/linkit-smart-7688/en/downloads>

По состоянию на 17 апреля 2017 новейшей версией была 0.94

2. Подключите модуль к USB порту компьютера
3. Если модуль работает в режиме точки доступа, подключитесь к этой точке доступа (LinkIt_Smart_7688_1E7688). Если модуль работает в режиме WiFi станции, значит вы уже настроили его для подключения к определенной WiFi сети.
4. Перейдите к веб-странице управления модулем с помощью браузера (адрес: mylinkit.local)



5. Нажмите на кнопку **UPGRADE FIRMWARE**

Software information

Boot loader version

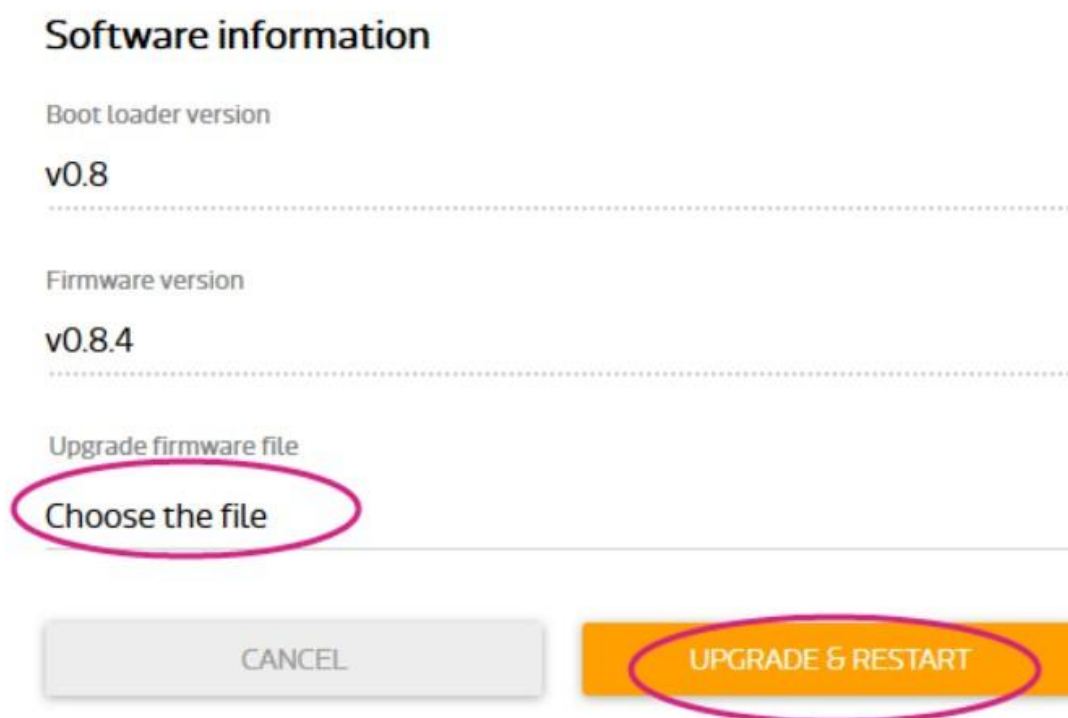
v0.8

Firmware version

v0.8.4

UPGRADE FIRMWARE

6. С помощью кнопки **Choose the file** выберите файл **lks7688.img** и нажмите на кнопку **UPGRADE & RESTART**. Новая прошивка начнет загружаться в модуль. Ни в коем случае не отключайте питание (USB кабель) от модуля до завершения процедуры прошивки! Прошивка продлится около 3-х минут. До завершения процедуры прошивки красный светодиод часто мигает. После прошивки начнет загружаться новая версия Linux OpenWrt. В течение загрузки (30 секунд) красный светодиод будет гореть, а после загрузки OpenWrt красный светодиод погаснет.



7. После завершения прошивки вы можете снова подключиться к веб-интерфейсу управления модулем и убедиться, что номер версии прошивки изменился на новый.

Видео инструкция по процессу обновления прошивки модуля находится по ссылке: <https://www.youtube.com/watch?v=GtmbgNOVbyM>

Восстановление стандартных настроек модуля

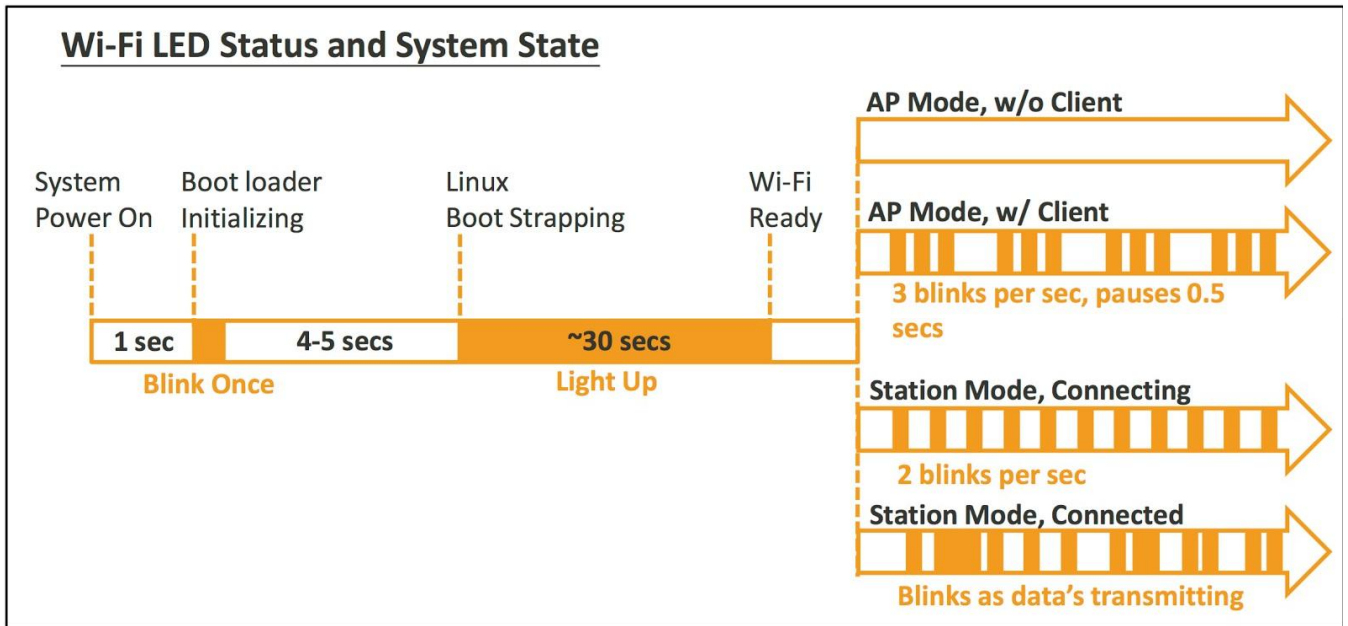
Иногда возможна ситуация, в которой вы забыли ранее установленный пароль или настроили модуль для подключения к определенной WiFi сети, которая сейчас недоступна. В таком случае может помочь сброс настроек модуля к стандартным. Важно понимать, что после такого процесса все настройки, установленные вами ранее и все загруженные/созданные файлы (программы и скрипты) будут автоматически удалены.

Для восстановления настроек модуля необходимо после его загрузки нажать кнопку на плате, которая обозначена как WiFi, и удерживать ее нажатой не менее 20 секунд, после чего кнопку необходимо отпустить. После этого начнется процесс восстановления настроек, в течение которого красный светодиод быстро мигает несколько секунд. После завершения восстановления модуль перезагрузится. Во время перезагрузки (30 секунд) красный светодиод будет гореть, а после загрузки OpenWrt он погаснет. Теперь параметры прошивки стандартные, модуль работает в режиме точки доступа, вы можете подключиться к нему и задать необходимые настройки (пароль и т.д.).

Видео-инструкция процесса сброса:

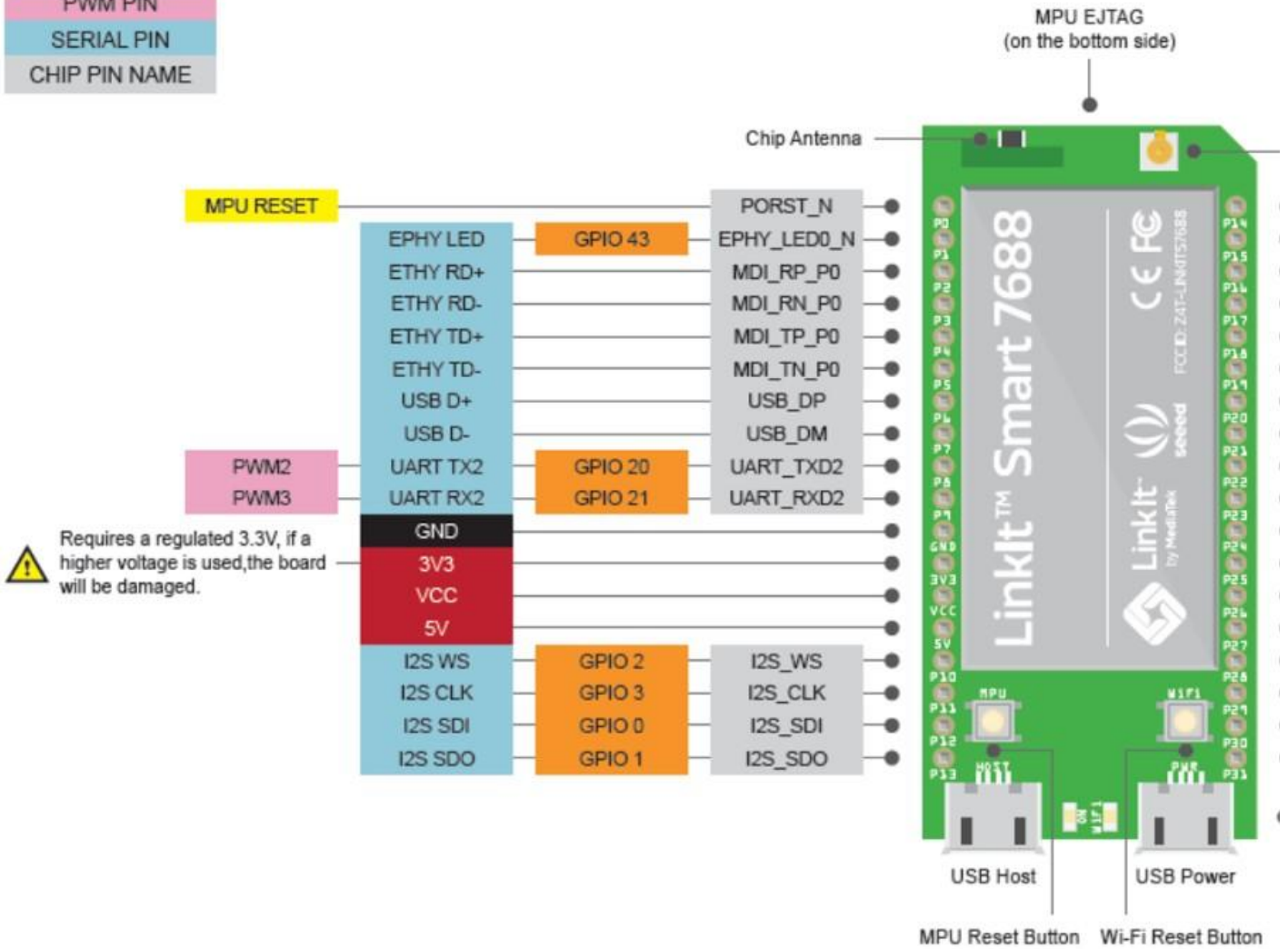
<https://www.youtube.com/watch?v=Y0r5c0mwm5Y>

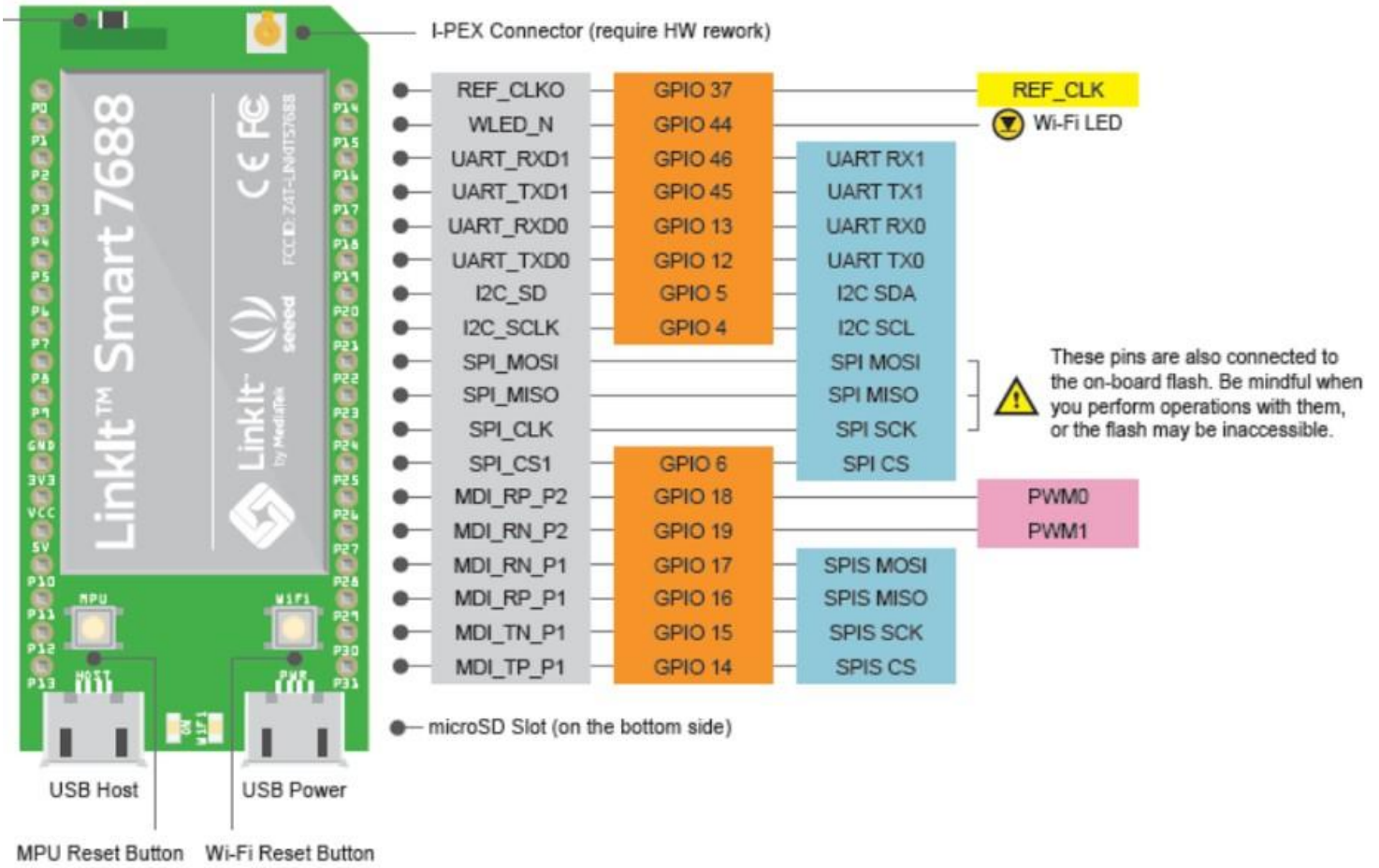
Индикация режимов работы модуля красным светодиодом



Описание линий ввода-вывода модуля

GND
POWER
CONTROL
DIGITAL PIN
PWM PIN
SERIAL PIN
CHIP PIN NAME





Ссылки

1. Описание и начало работы:

<https://www.kv.by/post/1048802-sobstvennoe-ustroystvo-dlya-inter-neta-veshchey-s-mediatek-linkit-smart-7688-duo>

2. Начало работы (англ.):

<https://goo.gl/yQDVVH>

3. Пример программирования на Python:

<https://docs.labs.mediatek.com/resource/linkit-smart-7688/en/tutorials/peripheral-support-on-linkit-smart-7688-development-board/using-mraa-in-python>

4. Пример работы с датчиками с использованием библиотеки UPM:

<https://docs.labs.mediatek.com/resource/linkit-smart-7688/en/tutorials/peripheral-support-on-linkit-smart-7688-development-board/using-upm-in-python>

5. Перечень датчиков, поддерживаемых библиотекой Python

UPM: <https://iotdk.intel.com/docs/master/upm/python/modules.html>

6. Пример программирования на NodeJS:

<https://docs.labs.mediatek.com/resource/linkit-smart-7688/en/tutorials/peripheral-support-on-linkit-smart-7688-development-board/using-mraa-in-node-js>

7. Указания по программированию на C:

<https://docs.labs.mediatek.com/resource/linkit-smart-7688/en/tutorials/c-c++-programming>

8. Developers guide:

<https://labs.mediatek.com/en/download/ih80Qtjo>

9. Запуск веб-сервера на Python:

<https://techtutorialx.wordpress.com/2016/12/27/linkit-smart-duo-running-a-flask-server/>

10. Примеры работы с облаком на NodeJS и Python:

<https://mcs.mediatek.com/7688/>

https://mcs.mediatek.com/resources/latest/tutorial/7688_led_tutorial

11. Сброс настроек модуля к стандартным (включая сброс пароля):

<https://www.youtube.com/watch?v=Y0r5c0mwm5Y>

12. Обновление прошивки модуля: _

<https://www.youtube.com/watch?v=GtmbgN0VbyM>

13. Подключение к терминалу OpenWrt с помощью преобразователя usb-uart: _

<https://docs.labs.mediatek.com/resource/linkit-smart-7688/en/tutorials/firmware-and-bootloader/bootloader-and-kernel-console>

14. Перевод в режим точки доступа: _

<https://www.youtube.com/watch?v=GZ48q2xxflw>

15. Линии ввода-вывода модуля:

http://www.cnx-software.com/wp-content/uploads/2015/12/Link_Smart_7688_Pinout.png

g