# Настройка Bluetooth ESP32 в среде MicroPython.

И так ESP32 имеет возможность передачи данных по Bluetooth в одноканальном режиме, это значат что связь происходит только между устройством, подключённым на базе операционной системе android по технологии BLE что переводится как Bluetooth Low Energy.

Примечание: корректной работы требуется чтобы Bluetooth был 4 версий и выше, так как технология BLE использует 4 версию и для старых версий оно не доступно.

В качестве сервера выступает ESP32 и для подключения к его сети требуется программное обеспечение, так в обычном диспетчере сетей Bluetooth оно не доступно.

Но для начала требуется написать программу для ESP32 для активации BLE.

В MicroPython уже имеет библиотека с работой блитуз, но для корректной работы требуется прописать все необходимы настройки используя ООП (объекта ориентированное программирования).

Код программы ESP32.

from machine import Pin

from machine import Timer

from time import sleep\_ms

import ubluetooth

class ESP32\_BLE():

def \_\_init\_\_(self, name):

self.led = Pin(2, Pin.OUT)

self.timer1 = Timer(0)

self.name = name

self.ble = ubluetooth.BLE()

self.ble.active(True)

self.disconnected()

self.ble.irq(self.ble\_irq)

self.register()

self.advertiser()

def connected(self):

global is\_ble\_connected

is\_ble\_connected = True

self.led.value(1)

self.timer1.deinit()

def disconnected(self):

global is\_ble\_connected

is\_ble\_connected = False

self.timer1.init(period=100, mode=Timer.PERIODIC, callback=lambda t: self.led.value(not self.led.value()))

def ble\_irq(self, event, data):

global ble\_msg

if event == 1:

self.connected()

elif event == 2:

self.advertiser()

self.disconnected()

elif event == 3:

buffer = self.ble.gatts\_read(self.rx)

ble\_msg = buffer.decode('UTF-8').strip()

def register(self):

# Nordic UART Service (NUS)

NUS\_UUID = '6E400001-B5A3-F393-E0A9-E50E24DCCA9E'

RX\_UUID = '6E400002-B5A3-F393-E0A9-E50E24DCCA9E'

TX\_UUID = '6E400003-B5A3-F393-E0A9-E50E24DCCA9E'

BLE\_NUS = ubluetooth.UUID(NUS\_UUID)

BLE\_RX = (ubluetooth.UUID(RX\_UUID), ubluetooth.FLAG\_WRITE)

BLE\_TX = (ubluetooth.UUID(TX\_UUID), ubluetooth.FLAG\_NOTIFY)

BLE\_UART = (BLE\_NUS, (BLE\_TX, BLE\_RX,))

SERVICES = (BLE\_UART, )

((self.tx, self.rx,), ) = self.ble.gatts\_register\_services(SERVICES)

def send(self, data):

self.ble.gatts\_notify(0, self.tx, data + '\n')

def advertiser(self):

name = bytes(self.name, 'UTF-8')

adv\_data = bytearray('\x02\x01\x02') + bytearray((len(name) + 1, 0x09)) + name

self.ble.gap\_advertise(100, adv\_data)

print(adv\_data)

print("\r\n")

ble = ESP32\_BLE("ESP32BLE")

ble\_msg = ""

is\_ble\_connected = False

led = Pin(2, Pin.OUT)

while True:

if is\_ble\_connected: #Подключиноли устройиство

if ble\_msg: #Полчучаем сообщение

print(ble\_msg) #Выводим мообщение

ble.send('Hello') #Отправляем ответ

ble\_msg = ""

sleep\_ms(100)

В этой части программы происходит обрушение к созданному классу.

ble = ESP32\_BLE("ESP32BLE")

ble\_msg = ""

is\_ble\_connected = False

led = Pin(2, Pin.OUT)

while True:

if is\_ble\_connected:

if ble\_msg:

print(ble\_msg)

ble.send('Hello')

ble\_msg = ""

sleep\_ms(100)

Где:

ESP32\_BLE(“Название BLE сети”) – Создание точки Bluetooth сети с указанием его названия.

is\_ble\_connected – Обнаружение наличии подключения, если подключен то True, если нет то False.

ble\_msg – Получение сообщения из устройства.

ble.send – Отправка сообщения на устройство.

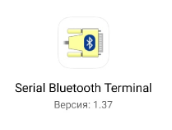
led – Указывает на наличий подключения к сети, если светодиод на плате мерцает то ESP32 ожидает наличия подключения, если светится то к нему подключено устройство.

Алгоритм работы программы:

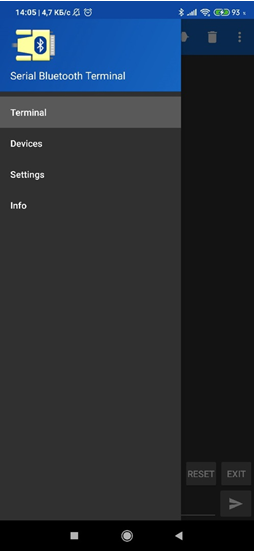
1. Создание сети Bluetooth с указанным именем
2. Ждем наличия подключения
3. Если устройство подключилось, то ожидаем сообщения от него.
4. Если сообщения пришло, то отправляем ответ “Hello”
5. После очищаем буфер полученного сообщения.

## Подключение к устройству Android.

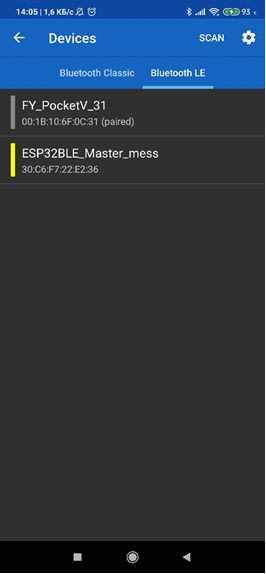
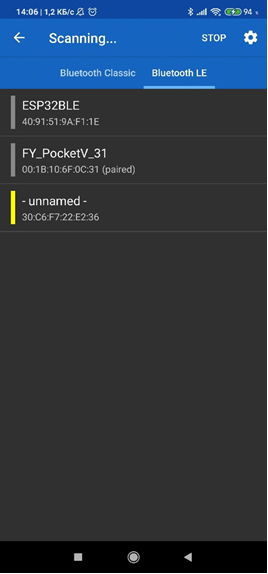
Для работы с Bluetooth требуется скачать приложение ‘Serial Bluetooth Terminal'



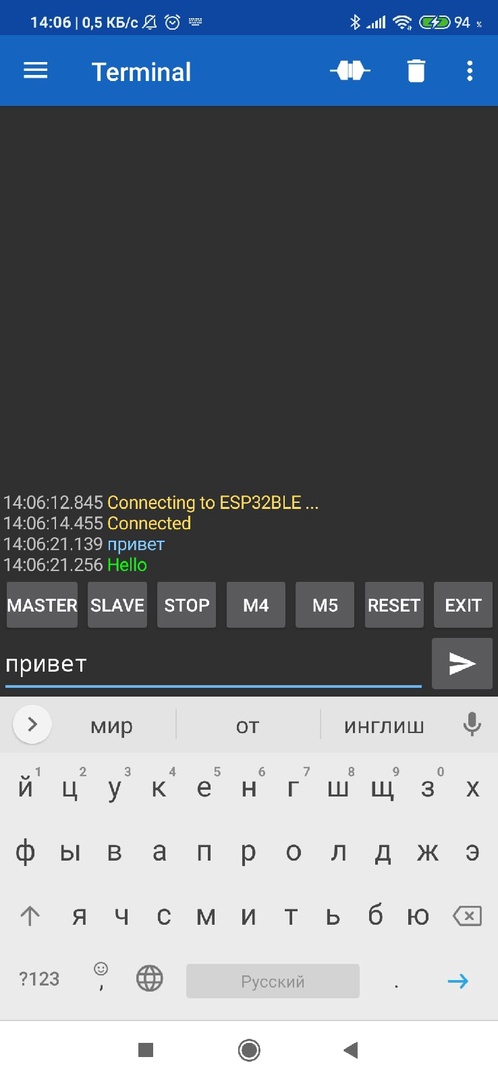
Далее запускаем приложение, и в открывшееся окне нажимаем на меню ‘≡’, далее ‘Devices’.

И в открывшемся окне выбираем “Bluetooth LE”, нажимаем “Scan”, и среди списков имен нажимаем на название созданной Bluetooth сети.

Далее в открывшемся окне пишем текс сообщения и отправляем, далее получаем ответ.



Отправленное сообщение в ESP32 выводится на терминал.

