Bluetooth

ECE 476 Advanced Embedded Systems

Jake Glower - Lecture #30

Please visit Bison Academy for corresponding
lecture notes, homework sets, and solutions

Introduction:

Bluetooth is a way for your Pico board to send and receive data from your cell phone (as well as other devices). With bluetooth, you can

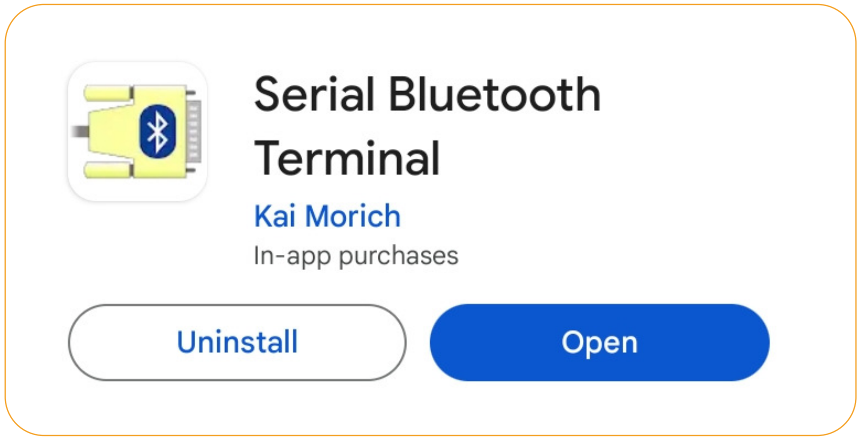
- Send sensor data to you cell phone, such as temperature, pressure, or acceleration readings, or
- Receive data from your cell phone, allowing you to turn on or off lights, set the speed of a motor, and so on.

This lecture presents methods for connecting your Pi-Pico to your cell phone to send and receive data.

Cell Phone App

Before sending and receiving data, you need to install a serial bluetooth terminal app on your cell phone.

- Several exist.
- Serial Bluetooth Terminal by Kai Morich
  - What's used with the following code

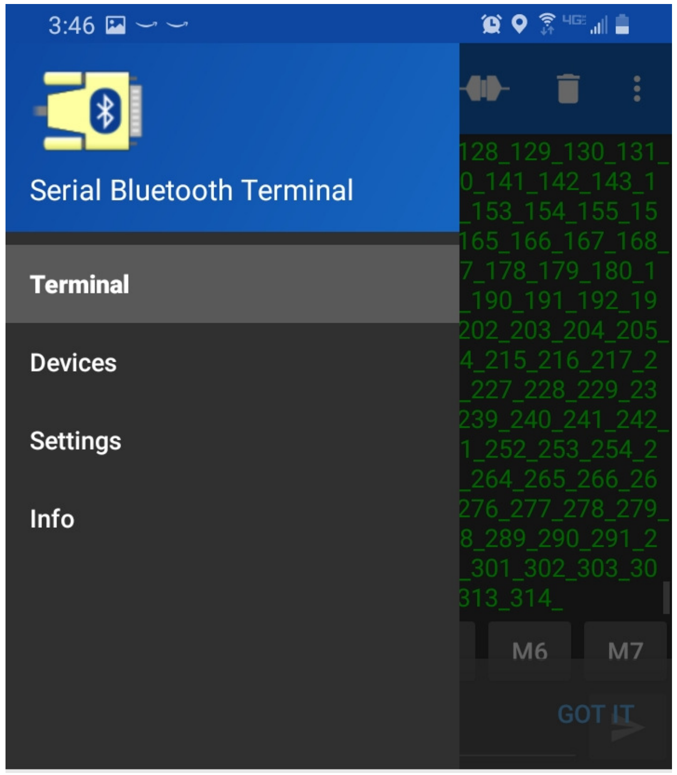


Serial Bluetooth Terminal app by Kau Morich

Serial Bluetooth Terminal (app)

Once you install this app and open it, you will get several options.

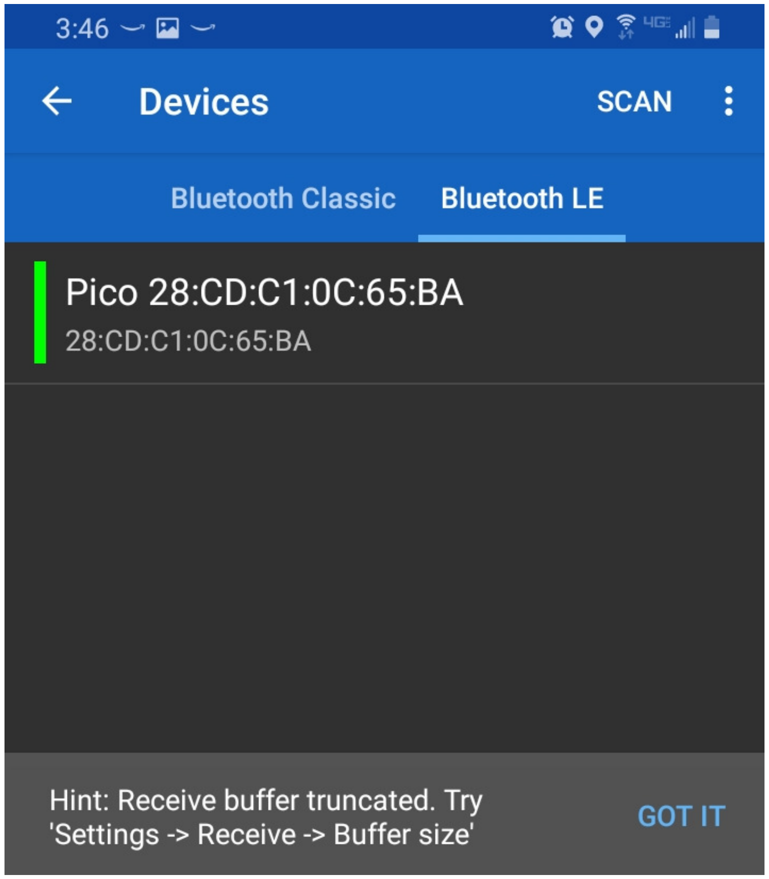
- Terminal allows you to send and receive serial data to your Pico board
- Devices lets you connect (pair) with your Pico board
- Settings allow you to adjust the display
- Info tells you the version you're using and other information



Pairing with your Cell Phone

Once you run the following programs, you first need to connect to your Pico board.

- Selecting Devices
- Select your Pico board



Pico 28:CD:C1:0C:65:BA
28:CD:C1:0C:65:BA

Hint: Receive buffer truncated. Try 'Settings -> Receive -> Buffer size'

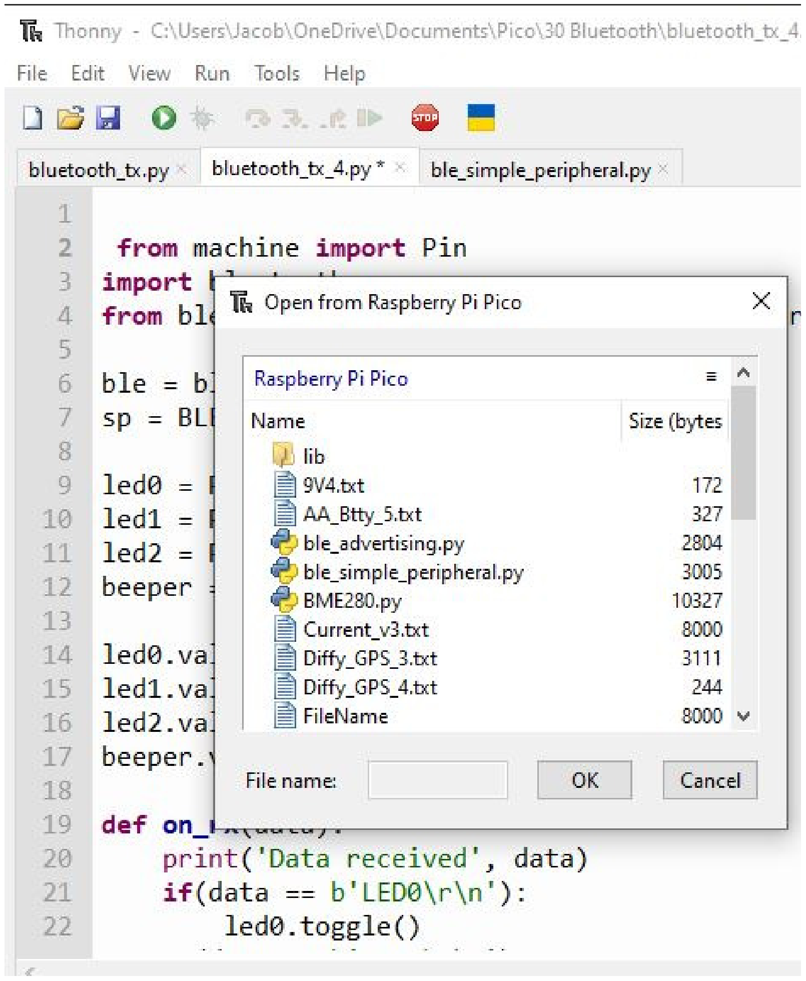
Driver Files for your Pico Board

In order to use BlueTooth on your Pico board, two files need to be in the root directory.

- *ble_advertising.py*
- *ble_simple_peripheral.py*

Open these files and save them to your Pico board using Thonny.

Once these files are on your Pico board, you're ready to send and receive serial data.


Thonny - C:\Users\Jacob\OneDrive\Documents\Pico\30 Bluetooth\bluetooth_tx_4.
File Edit View Run Tools Help
bluetooth_tx.py
bluetooth_tx_4.py *
ble_simple_peripheral.py
Open from Raspberry Pi Pico
Raspberry Pi Pico
File name:
OK
Cancel

```

 from machine import Pin
import 
from ble

ble = b
sp = BLE

led0 = 
led1 = 
led2 = 
beeper 

led0.va
led1.va
led2.va
beeper.

def on_
    print('Data received', data)
    if(data == b'LED0\r\n'):
        led0.toggle()
```

Name	Size (bytes
lib	
9V4.txt	172
AA_Btty_5.txt	327
ble_advertising.py	2804
ble_simple_peripheral.py	3005
BME280.py	10327
Current_v3.txt	8000
Diffy_GPS_3.txt	3111
Diffy_GPS_4.txt	244
FileName	8000

Bluetooth Transmit

The following program

- Sets up a bluetooth connection to your cell phone
- Once connected
  - p.is_connected() == true
  - a count is sent
  - along with a carriage return and linefeed
  - once per second.

Note that the data is an ascii string

- Strings are easier to understand
- Binary data is OK if you select HEX when receiving data

```
import bluetooth
from time import sleep
from ble_advertising import
advertising_payload
from ble_simple_peripheral import
BLESimplePeripheral

ble = bluetooth.BLE()
p = BLESimplePeripheral(ble)

i = 0
while(1):
    if p.is_connected():
        i += 1
        data = str(i)
        p.send(data + "\r\n")
        print("tx ", data)
    sleep(1)
```

shell

```
tx 1
tx 2
tx 3
```

Receiving the Data as Text

From the terminal emulator, go to

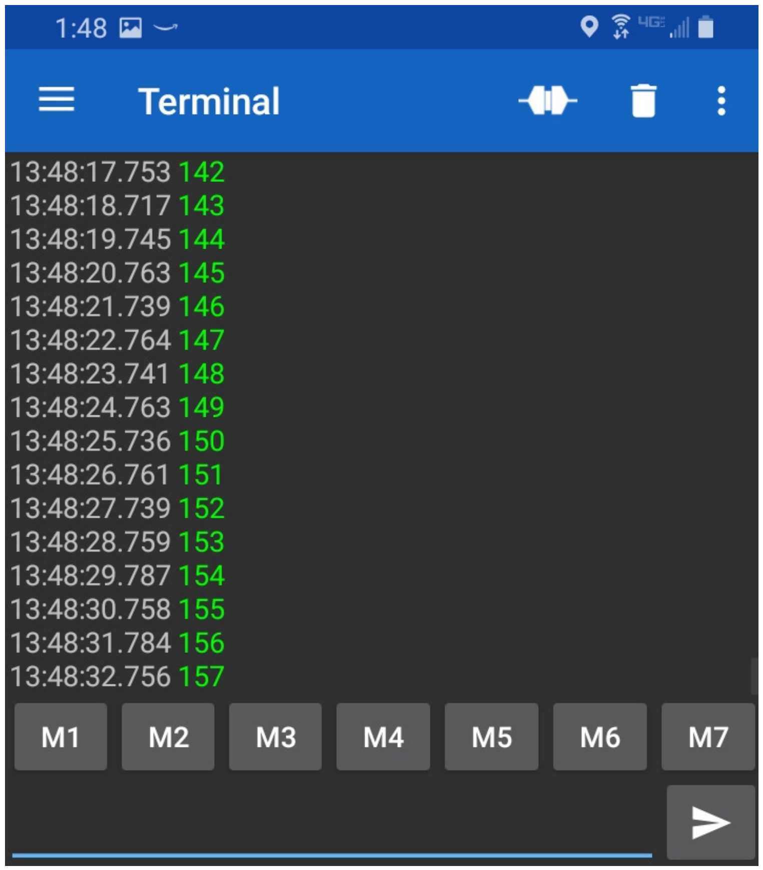
- Settings
- Display mode
- Text

The ascii data is displayed

- (numbers here)

This doesn't work well if the data is binary

- Many non-printable ascii characters



```
13:48:17.753 142
13:48:18.717 143
13:48:19.745 144
13:48:20.763 145
13:48:21.739 146
13:48:22.764 147
13:48:23.741 148
13:48:24.763 149
13:48:25.736 150
13:48:26.761 151
13:48:27.739 152
13:48:28.759 153
13:48:29.787 154
13:48:30.758 155
13:48:31.784 156
13:48:32.756 157
```

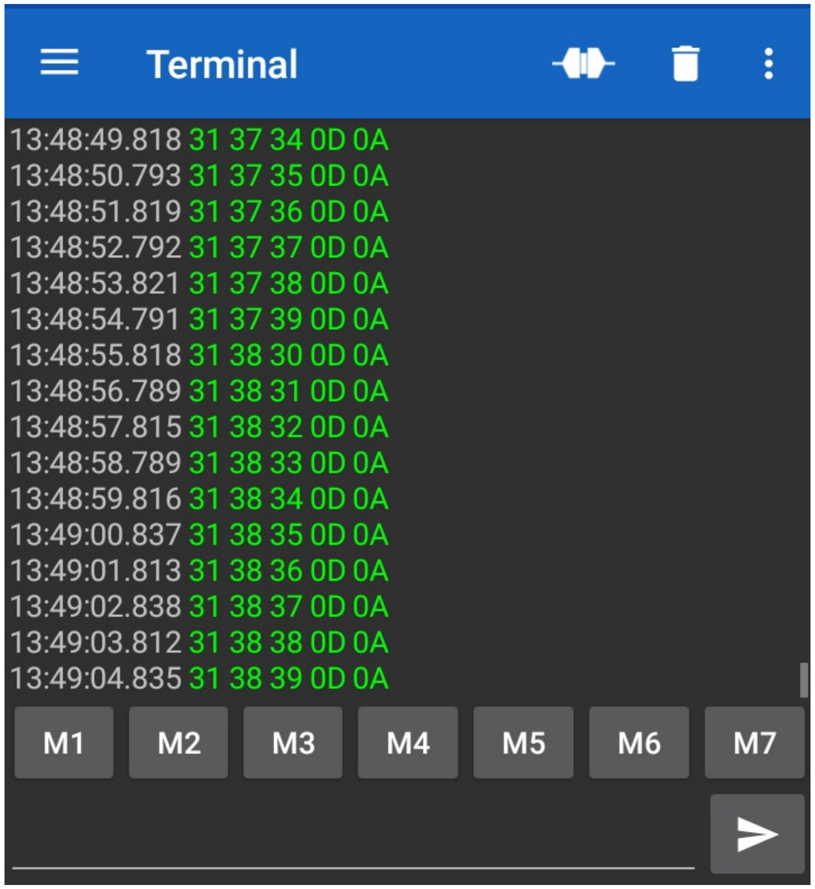
Receiving Binary Data

Select

- Settings
- Display mode
- Hex

and you see each byte in hex format

Useful if receiving binary data



```
13:48:49.818 31 37 34 0D 0A
13:48:50.793 31 37 35 0D 0A
13:48:51.819 31 37 36 0D 0A
13:48:52.792 31 37 37 0D 0A
13:48:53.821 31 37 38 0D 0A
13:48:54.791 31 37 39 0D 0A
13:48:55.818 31 38 30 0D 0A
13:48:56.789 31 38 31 0D 0A
13:48:57.815 31 38 32 0D 0A
13:48:58.789 31 38 33 0D 0A
13:48:59.816 31 38 34 0D 0A
13:49:00.837 31 38 35 0D 0A
13:49:01.813 31 38 36 0D 0A
13:49:02.838 31 38 37 0D 0A
13:49:03.812 31 38 38 0D 0A
13:49:04.835 31 38 39 0D 0A
```

What you can do with bluetooth transmit

Once you are transmitting data from the Pico and receiving it from your cell phone, you can monitor...

- Temperature in the room
- Acceleration of the Pi-Pico (has it been moved?)
- Door open or closed (button pressed, not pressed),
- etc.

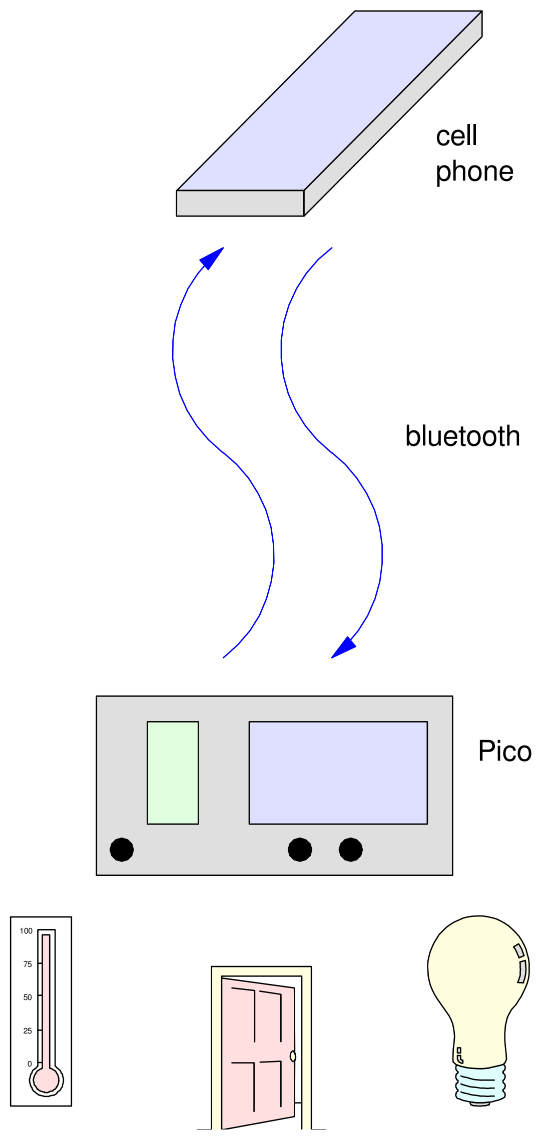
Kind of only limited by your creativity

BlueTooth Receive

The Pico can also receive data from your cell phone.

With this, you can

- Turn on and off an LED
- Set the color of a NeoPixel
- Turn off an alarm, etc.



Bluetooth Receive Example

Receive serial messages

- Display each message received

Look for a message

- b'toggle\r\n'

If found

- toggle the LED on GP16

```
from machine import Pin
import bluetooth
from ble_simple_peripheral import BLESimplePeripheral

ble = bluetooth.BLE()
sp = BLESimplePeripheral(ble)

led = Pin(16, Pin.OUT)
led.value(0)


def on_rx(data):
    print("Data received: ", data)
    if data == b'toggle\r\n':
        led.toggle()

while True:
    if sp.is_connected():
        sp.on_write(on_rx)
```

shell

```
Data received:  1234\r\n
Data received:  toggle\r\n
```

Turning On/Off Lights from your Cell Phone

From your cell phone

- Running Serial Bluetooth Terminal

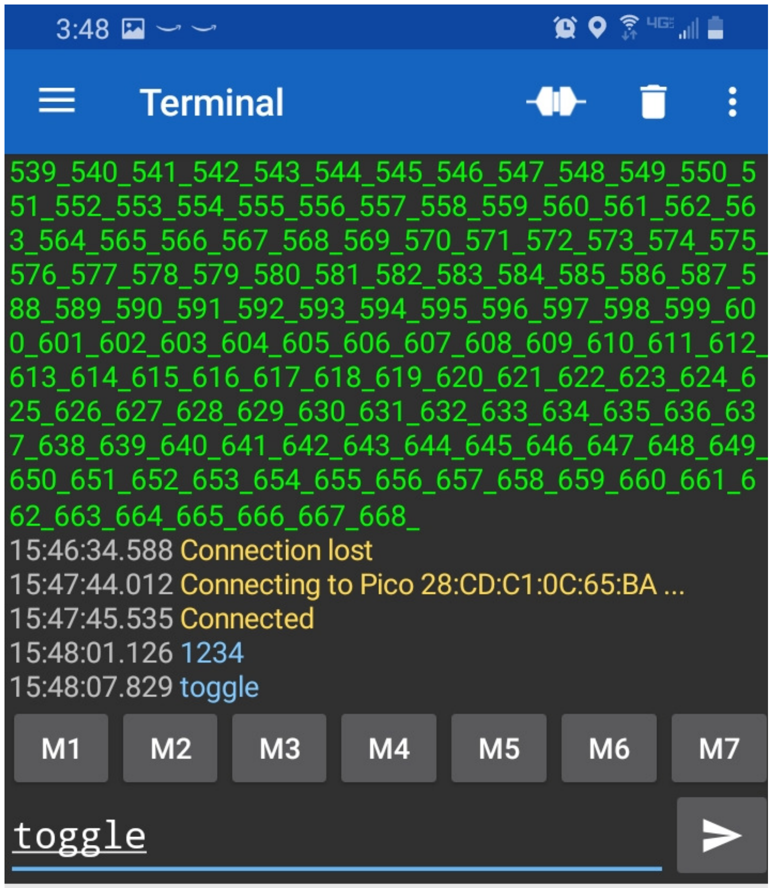
Type in your message

- Hit send (arrow on lower right)

This sends your message to your Pico

- messages terminated with carriage return and linefeed (\r\n)

When you send 'toggle', the LED should toggle on and off



```
539_540_541_542_543_544_545_546_547_548_549_550_5
51_552_553_554_555_556_557_558_559_560_561_562_56
3_564_565_566_567_568_569_570_571_572_573_574_575_
576_577_578_579_580_581_582_583_584_585_586_587_5
88_589_590_591_592_593_594_595_596_597_598_599_60
0_601_602_603_604_605_606_607_608_609_610_611_612_
613_614_615_616_617_618_619_620_621_622_623_624_6
25_626_627_628_629_630_631_632_633_634_635_636_63
7_638_639_640_641_642_643_644_645_646_647_648_649_
650_651_652_653_654_655_656_657_658_659_660_661_6
62_663_664_665_666_667_668_
15:46:34.588 Connection lost
15:47:44.012 Connecting to Pico 28:CD:C1:0C:65:BA ...
15:47:45.535 Connected
15:48:01.126 1234
15:48:07.829 toggle
```

Short-Cuts

M1 through M7 allow you to store messages

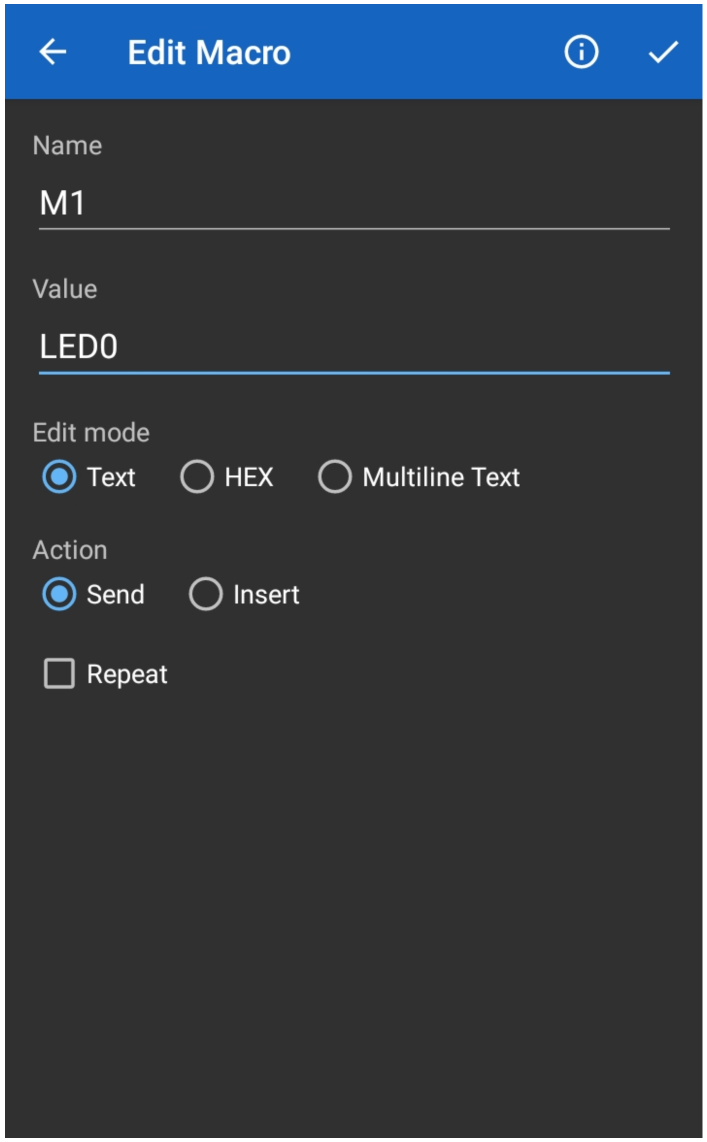
- Press and hold M1.
- Type in the message
- Click the check mark (upper right corner)
- Ditto for M2 .. M7

Each time you press

- M1 message is sent out along with a carriage return and line feed.
- M2 message is sent out along with a carriage return and line feed

This allows you to output seven messages at the push of a button

- toggle seven devices
- turn some on, turn some off, etc



Name: M1

Value: LED0

Edit mode: [✓ Text] HEX Multiline Text

Action: [✓ Send] Insert

☐ Repeat

Toggle 4 Devices

Pico LED

- LED0

GP16 LED

- LED1

GP17 LED

- LED2

Beeper

- Beep

Note:

- These are binary strings
- Terminated with cr/lf
- Case sensitive

```
from machine import Pin
import bluetooth
from ble_simple_peripheral import BLESimplePeripheral

ble = bluetooth.BLE()
sp = BLESimplePeripheral(ble)

led0 = Pin("LED", Pin.OUT)
led1 = Pin(16, Pin.OUT)
led2 = Pin(17, Pin.OUT)
beeper = Pin(13, Pin.OUT)

def on_rx(data):
    print("Data received: ", data)
    if data == b'LED0\r\n':
        led0.toggle()
    if data == b'LED1\r\n':
        led1.toggle()
    if data == b'LED2\r\n':
        led2.toggle()
    if data == b'Beep\r\n':
        beeper.toggle()

while True:
    if sp.is_connected():
        sp.on_write(on_rx)
```

Summary

In order to connect your Pico board to your cell phone using a bluetooth connection, two files need to be added to the Pico board:

- *ble_advertising.py*
- *ble_simple_peripheral.py*

Once added, you can send and receive serial data to your cell phone. This allows you to monitor and control a device through your cell phone.

References

- https://electrocredible.com/raspberry-pi-pico-w-bluetooth-ble-micropython