

GR_MANGO向けの MicroPython 使い方

MicroPython for GR_MANGO

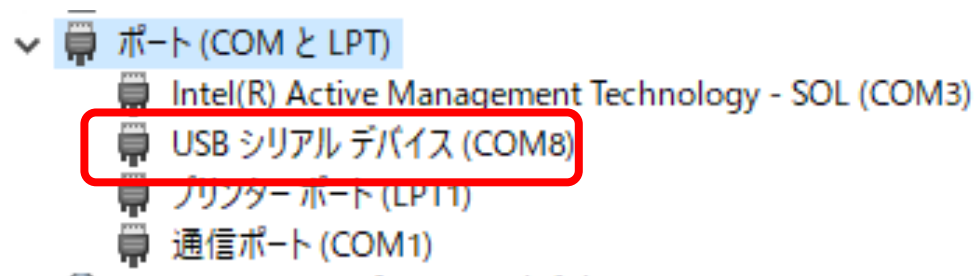
- MicroPythonのSTM32の実装をGR_MANGO(RZA2M)に移植したものです。
- 移植した機能はSTM32向けの実装(pyboard)にほぼ準拠しています。
 - 使い方の詳細はpyboardのマニュアル(<https://docs.micropython.org/en/latest/>)を参照してください。
 - ただし、Hardware Timer, PWM(Servo), USB, CAN, WDT...は未実装です。
 - Pyboardの一部のモジュールのパラメータが実装できていないものがあります。詳細はソースコードを参照ください。
 - MicroPythonのESP8266およびESP32向けの実装とはモジュールが異なります。

用意するもの

- Windows 10が動作するPC
 - ターミナルソフトウェア(ここではTera Termを使用します。)
- GR_MANGO
- その他ガジェット
 - このドキュメントでは、以下のガジェットのサンプルを取り上げます。
 - オムロン 2 JCIEボード
 - MAX7129 8x8 LED Matrixボード
 - NeoPixel Ring 12個

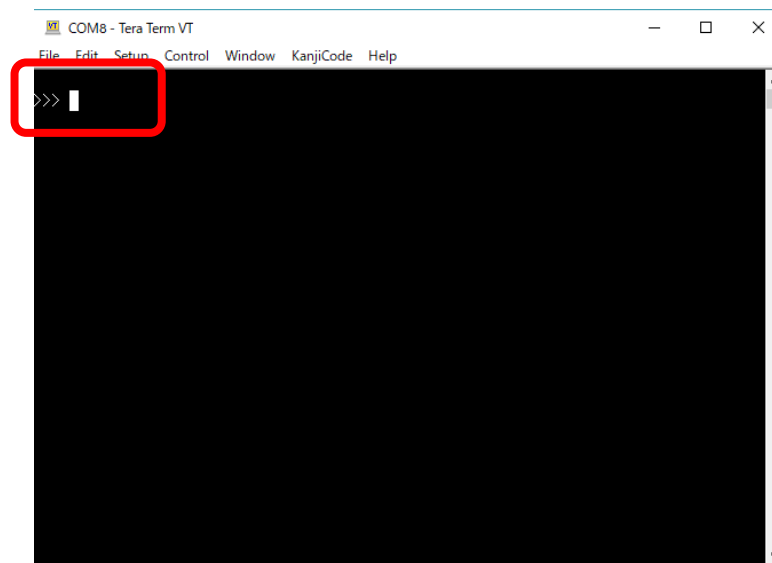
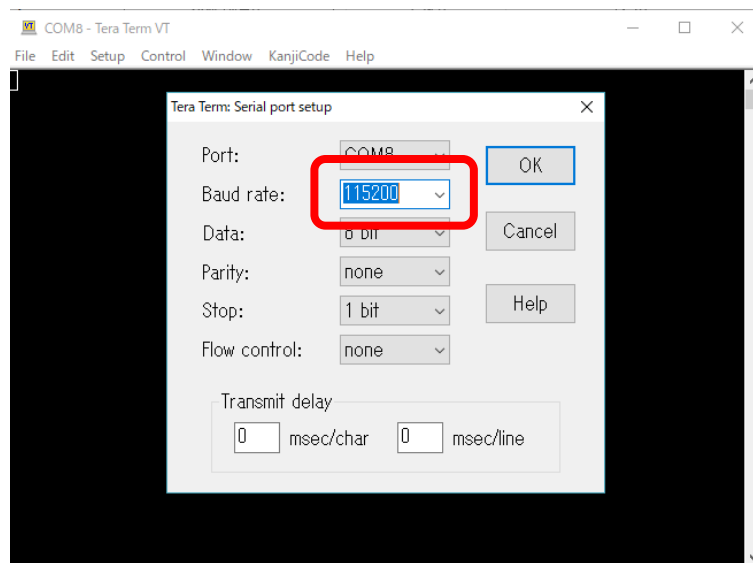
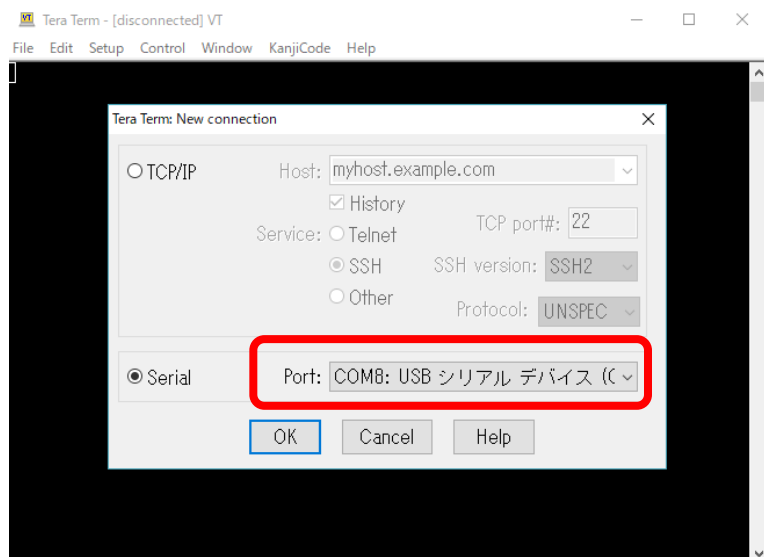
使い方

- GR-MANGOのUSBマイクロコネクタ(Type-Cではない方)とWindow 10 PCのUSBコネクタを接続します。
- デバイスマネージャのポートでUSBシリアルデバイスとして認識されます。
- GR_CITRUSではUSBディスクとして認識されましたが、GR_MANGOではUSBディスクの機能は実装されていません



使い方

- Tera Termを起動し、認識されたCOMポートを選択し、Setup – Serial portメニューを選択し、Baud rateで115200を選択し、Enterキーを押します。>>というMicroPython REPLプロンプトが表示されるはずです。
- このコンソールより、MicroPythonのプログラムが実行できます。



最初のサンプル Lチカ

- 最初のサンプルとして、GR-MANGOボード上のLEDを点灯してみます。
- 下記のプログラムを入力してみます。
- Enterキーを数回入力します。
- Ctrl-Cキーの入力でプログラムが終了します。

```
import pyb
while True:
    pyb.LED(1).toggle()
    pyb.delay(50)
```

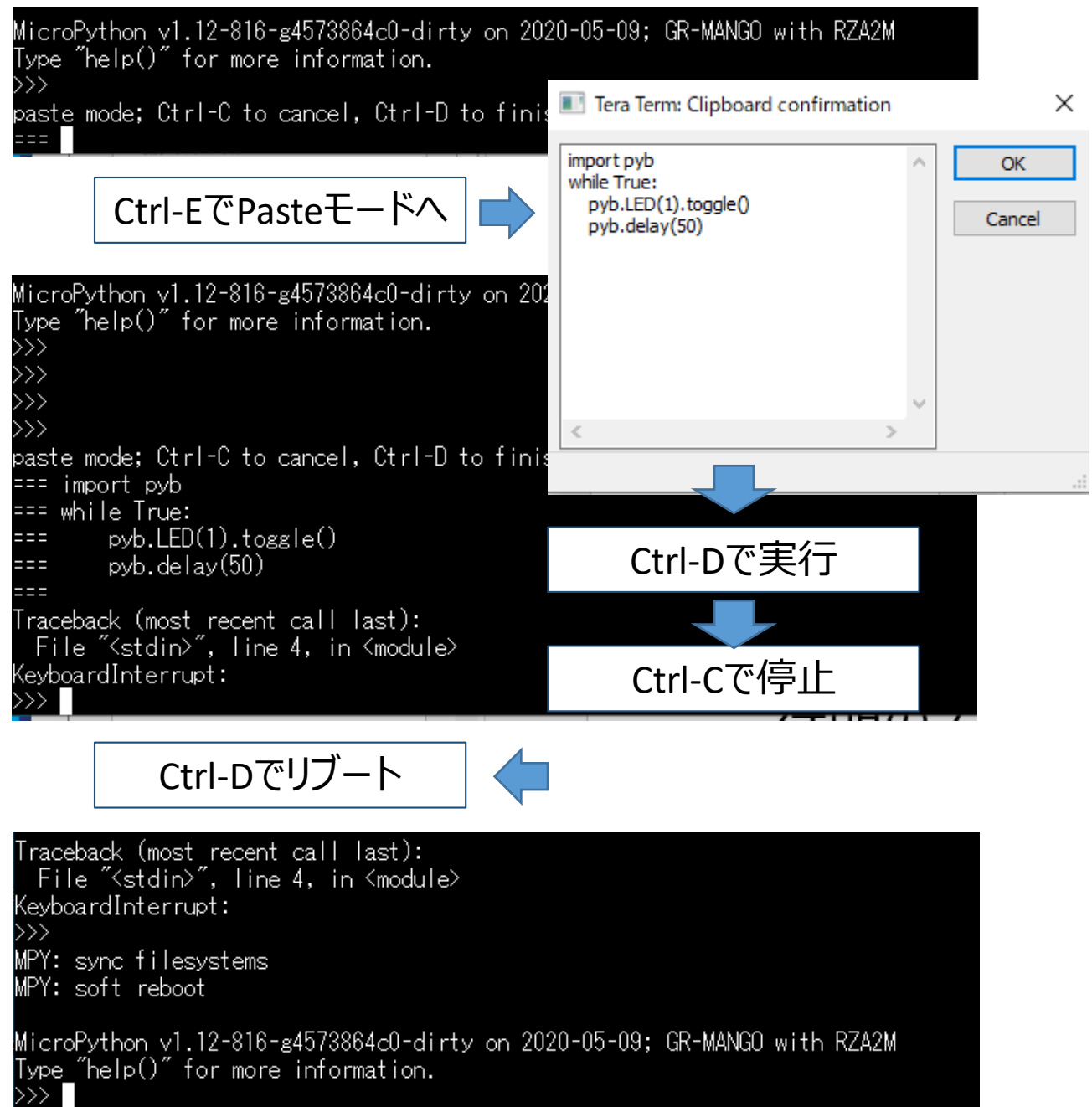


```
COM8 - Tera Term VT
File Edit Setup Control Window KanjiCode Help

>>> import pyb
>>> while True:
...     pyb.LED(1).toggle()
...     pyb.delay(50)
...
...
...
Traceback (most recent call last):
  File "<stdin>", line 3, in <module>
KeyboardInterrupt:
>>> █
```

REPLの補足

- インデントのあるプログラムをCut&Pasteする際は、Ctrl-Eを押してから、Pasteすると便利です。
- Ctrl-Dでプログラムを実行します。
- Ctrl-Cでプログラムを停止します。
- さらにCtrl-Dでソフトウェアリブートします。



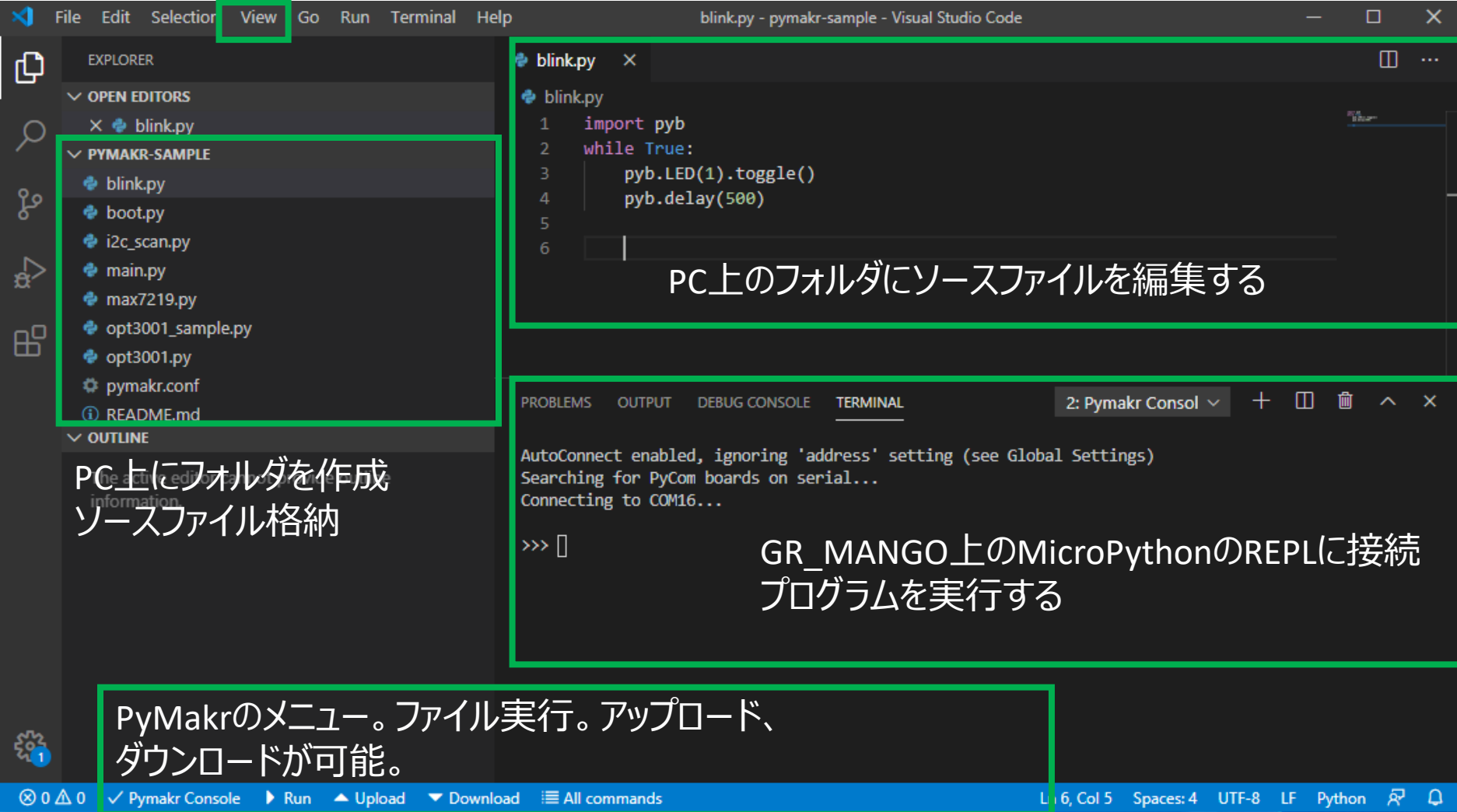
ストレージ

- GR_MANGOの実装では、ストレージとして、内蔵FlashとSDカードが使用できます。
- 内蔵フラッシュは、仮に0x50F00000より256KB確保しています。
- 起動時に、”/flash”デフォルトのフォルダとなり、初期化後はboot.pyとmain.pyの2つのファイルが置かれています。
 - main.pyを更新することで起動時にプログラムを実行することができます。
- 起動時にSDカードが挿入されている場合には”/sd”フォルダがデフォルトのフォルダとなります。

```
MicroPython v1.12-816-g4573864c0-dirty on 2020-05-09; GR-MANGO with RZA2M
Type "help()" for more information.
>>> import uos
>>> uos.listdir("")
['boot.py', 'main.py']
>>> uos.listdir("/")
['flash']
```


Visual Studio Code + PyMakrによるプログラム編集

View – Command PaletteからPyMakrのメニューにアクセス



The screenshot shows the Visual Studio Code interface with the following components:

- EXPLORER:** A list of files in the 'PYMAKR-SAMPLE' folder, including `blink.py`, `boot.py`, `i2c_scan.py`, `main.py`, `max7219.py`, `opt3001_sample.py`, `opt3001.py`, `pymakr.conf`, and `README.md`. A green box highlights this list with the text "PC上にフォルダを作成ソースファイル格納".
- EDITOR:** The `blink.py` file is open, showing the following code:

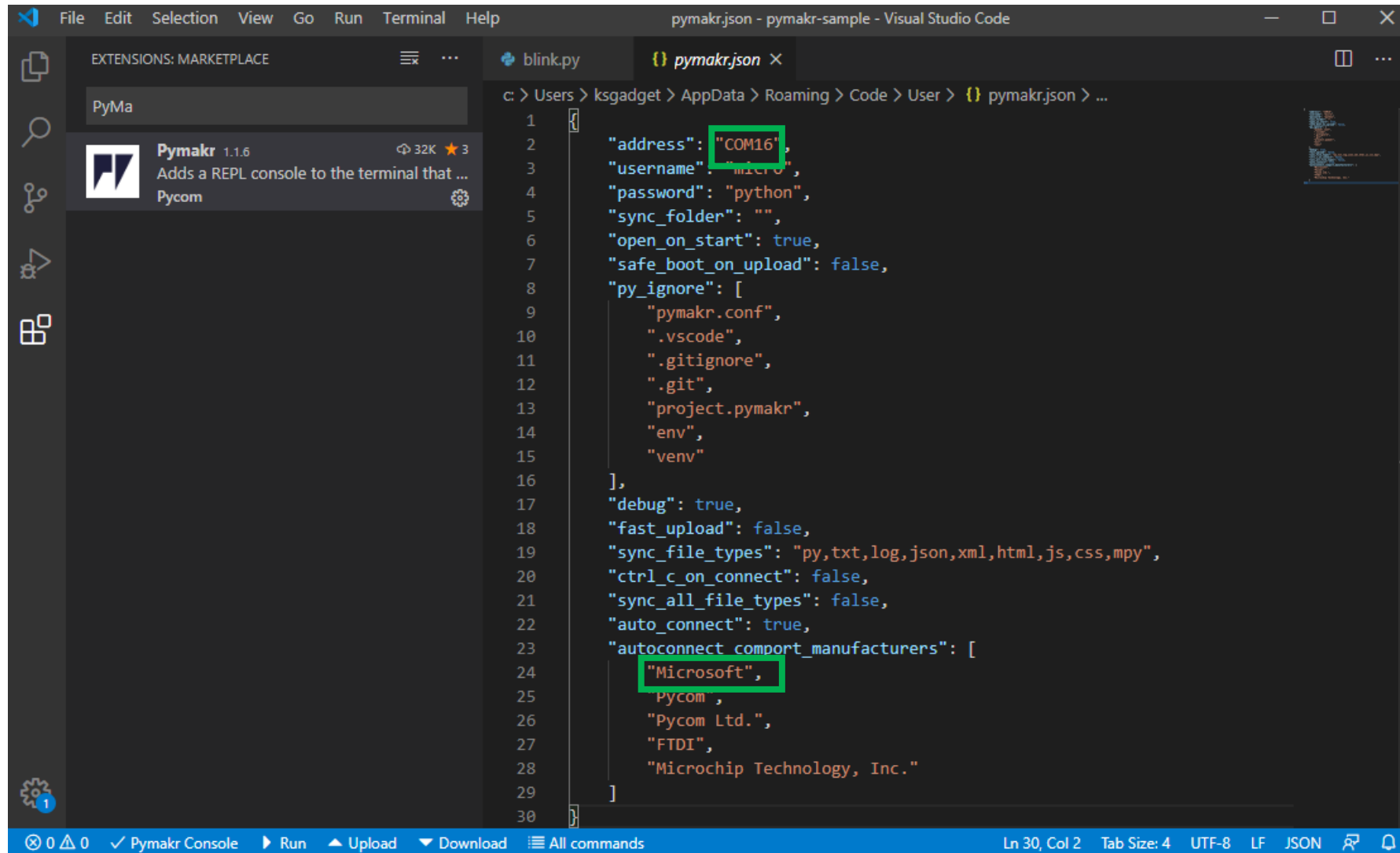
```
1 import pyb
2 while True:
3     pyb.LED(1).toggle()
4     pyb.delay(500)
5
6
```

A green box highlights the editor with the text "PC上のフォルダにソースファイルを編集する".
- TERMINAL:** The terminal shows the output of the PyMakr console, including "AutoConnect enabled, ignoring 'address' setting (see Global Settings)", "Searching for PyCom boards on serial...", and "Connecting to COM16...". A green box highlights the terminal with the text "GR_MANGO上のMicroPythonのREPLに接続プログラムを実行する".
- COMMAND PALETTE:** The 'View' menu is open, showing the 'Command Palette' option. A green box highlights this menu with the text "PyMakrのメニュー。ファイル実行。アップロード、ダウンロードが可能。".

Visual Studio Code + PyMakrのインストール

- Visual Studio Codeのインストール
 - <https://visualstudio.microsoft.com/ja/>
- Nodejsのインストール (6.9.5 以降) たぶん、最新版で問題ない
 - <https://nodejs.org/en/>
- Visual Studio CodeにPyMakerプラグインのインストール
 - 左下の拡張メニューアイコンをクリックし、EXTENSIONS:MARKETPLACEの検索欄にPyMakrと入力し、インストール
- PyMakerプラグインにGR_MANGOのCOMポートなどを登録
 - View – Command Paletteメニューで、PyMakr – Global settingsを選択し、pymakr.jsonファイル中の”address”: “xxxx”のxxxxにCOMポートを設定

PyMakr – Global settings



PyMakr Tips

- PyMakrのファイルは、以下のフォルダに置かれている。
 - C:¥Users¥ユーザ名¥.vscode¥extensions¥pycom.pymakr-1.1.6
- PyMakrの設定fileは、以下のフォルダに置かれている。
 - C:¥Users¥ユーザ名¥.AppData¥Roaming¥Code¥User
- Windows環境でCOMポート接続が断続する場合には、C:¥Users¥ユーザ名¥.vscode¥extensions¥pycom.pymakr-1.1.6¥lib(and ¥src)¥connections¥pyserial.jsの139行目ぐらいからのコメントアウトする。
- Uploadに失敗する場合、config.jsでupload_chunk_sizeを512から256に設定する。

```
sendPing(cb) {  
  //if (process.platform == 'win32') {  
  // avoid MCU waiting in bootloader on hardware restart by setting both dtr and rts high  
  // this.stream.set({ rts: true });  
  //}  
  // not implemented  
  if (this.dtr_supported) {  
    this.stream.set({ dtr: true }, function (err) {  
      if (cb) {  
        cb(err);  
        return err ? false : true;  
      }  
    });  
  } else {  
    cb();  
    return true;  
  }  
}
```

サンプルコード

- ファイルアクセス
- ピン割り込み
- Timer (ソフトウェア)
- I2C – デバイススキャン
- I2C – SH30 温度湿度センサー
- I2C – OMRON 2SMPB-02E MEMS絶対圧センサ
- ネットワーク

サンプル ファイルアクセス

- デフォルトの/flashフォルダのmain.pyの内容を表示します。

```
f = open('main.py', 'r')
f.read()
f.close()
```

```
MicroPython v1.12-816-g4573864c0-dirty on 2020-05-09; GR-MANGO with RZA2M
Type "help()" for more information.
>>> f = open('main.py', 'r')
>>> f.read()
'# main.py -- put your code here!¥r¥n'
>>> f.close()
```

/flash及び/sdファイルシステムへの読み書きは、標準的には、MicroPythonのファイル入出力ライブラリを呼び出すことで行っています。

サンプル SD Card

- SD Cardをマウントします

```
import pyb
import os
sd=pyb.SDCard()
os.mount(sd, "/sd")
os.listdir("/")
```

/flash及び/sdファイルシステムへの読み書きは、標準的には、MicroPythonのファイル入出力ライブラリを呼び出すことで行っています。

サンプルピン割り込み – スイッチ検出

- Switch(PD7)を押すと、print(“intr”)が実行されます。
- IRQxを割り当て可能なピンのみで実行できます。

```
from pyb import Pin, ExtInt
callback = lambda e: print("intr")
ext = ExtInt(Pin(Pin.cpu.PD7, Pin.IN, Pin.PULL_UP),
ExtInt.IRQ_RISING, Pin.PULL_UP, callback)
```


サンプル Timer – タイマー呼び出し

- ソフトウェアタイマ機能で、2 秒ごとに、print(2)を呼び出します。

```
from machine import Timer
tim = Timer(-1)
tim.init(period=2000, mode=Timer.PERIODIC,
callback=lambda t:print(2))
```

```
>>> from machine import Timer
>>> tim = Timer(-1)
>>> tim.init(period=2000, mode=Timer.PERIODIC, callback=lambda t:print(2))
>>> 2
2
2
2
2
2
```

サンプル I2C – デバイススキャン

- I2Cバスに接続されているデバイスをリストします。
- オムロンのセンサーボードを接続しています

```
import machine
i2c = machine.I2C(scl=machine.Pin.cpu.PD2,
sda=machine.Pin.cpu.PD3)

print('Scan i2c bus...')
devices = i2c.scan()

if len(devices) == 0:
    print("No i2c device !")
else:
    print('i2c devices found:',len(devices))

    for device in devices:
        print("Decimal address: ",device," | Hexa address: ",hex(device))
```

0x18	MAX9867ETJ+	Audio CODEC
0x3c	SII9022ACNU	HDMI Transmitter
0x44	SHT30-DIS-B	温度湿度センサー
0x45	OPT3001DNP	周辺光センサー
0x56	2SMPB-02E	MEMS絶対圧センサー
0x60	TUSB320IRWBR	USB Hub Controller

```
MicroPython v1.12-816-g4573864c0-dirty on 2020-05-09; GR-MANGO with RZA2M
Type "help()" for more information.
>>>
paste mode; Ctrl-C to cancel, Ctrl-D to finish
=== import machine
=== i2c = machine.I2C(scl=machine.Pin.cpu.PD2, sda=machine.Pin.cpu.PD3)
===
=== print('Scan i2c bus...')
=== devices = i2c.scan()
===
=== if len(devices) == 0:
===     print("No i2c device !")
=== else:
===     print('i2c devices found:',len(devices))
===
===     for device in devices:
===         print("Decimal address: ",device," | Hexa address: ",hex(device))
Scan i2c bus...
i2c devices found: 6
Decimal address: 24 | Hexa address: 0x18
Decimal address: 60 | Hexa address: 0x3c
Decimal address: 68 | Hexa address: 0x44
Decimal address: 69 | Hexa address: 0x45
Decimal address: 86 | Hexa address: 0x56
Decimal address: 96 | Hexa address: 0x60
>>>
```

サンプル I2C – SH30 温度湿度センサー

- 温度湿度を表示します。
 - sh30.pyモジュールを/flashに(PyMakr経由などで)Upload後、下記のコードを実行します。

```
from sht30 import SHT30

sensor = SHT30(scl_pin=machine.Pin.cpu.PD2,
sda_pin=machine.Pin.cpu.PD3, i2c_address=0x44)

temperature, humidity = sensor.measure()

print('Temperature:', temperature, '°C, RH:',
humidity, '%')
```

```
[1/9] Writing file blink.py (0kb)
[2/9] Writing file boot.py (0kb)
[3/9] Writing file i2c_scan.py (0kb)
[4/9] Writing file main.py (0kb)
[5/9] Writing file max7219.py (4kb)
[6/9] Writing file opt3001.py (0kb)
[7/9] Writing file opt3001_sample.py (0kb)
[8/9] Writing file sht30.py (7kb)
[9/9] Writing file sht30_sample.py (0kb)
Upload done, resetting board...
OK
MicroPython v1.12-816-g4573864c0-dirty on 2020-05-09; GR-MANGO with RZA2M
Type "help()" for more information.
>>> █
```

```
>>> Running selected lines
>>>
>>Temperature: 30.86672 °C, RH: 43.53094 %
>
```

サンプル I2C – OMRON 2SMPB MEMS絶対圧センサ

- 温度, 大気圧を表示します。
 - omron2smpb.pyモジュールを/flashに(PyMakr経由などで)Upload後、下記のコードを実行します。

```
from omron2smpb import OMRON2SMPB

sensor = OMRON2SMPB(scl_pin=machine.Pin.cpu.PD2,
sda_pin=machine.Pin.cpu.PD3, i2c_address=0x56)

sensor.init()
temperature = sensor.get_temperature()
pressure = sensor.get_pressure()

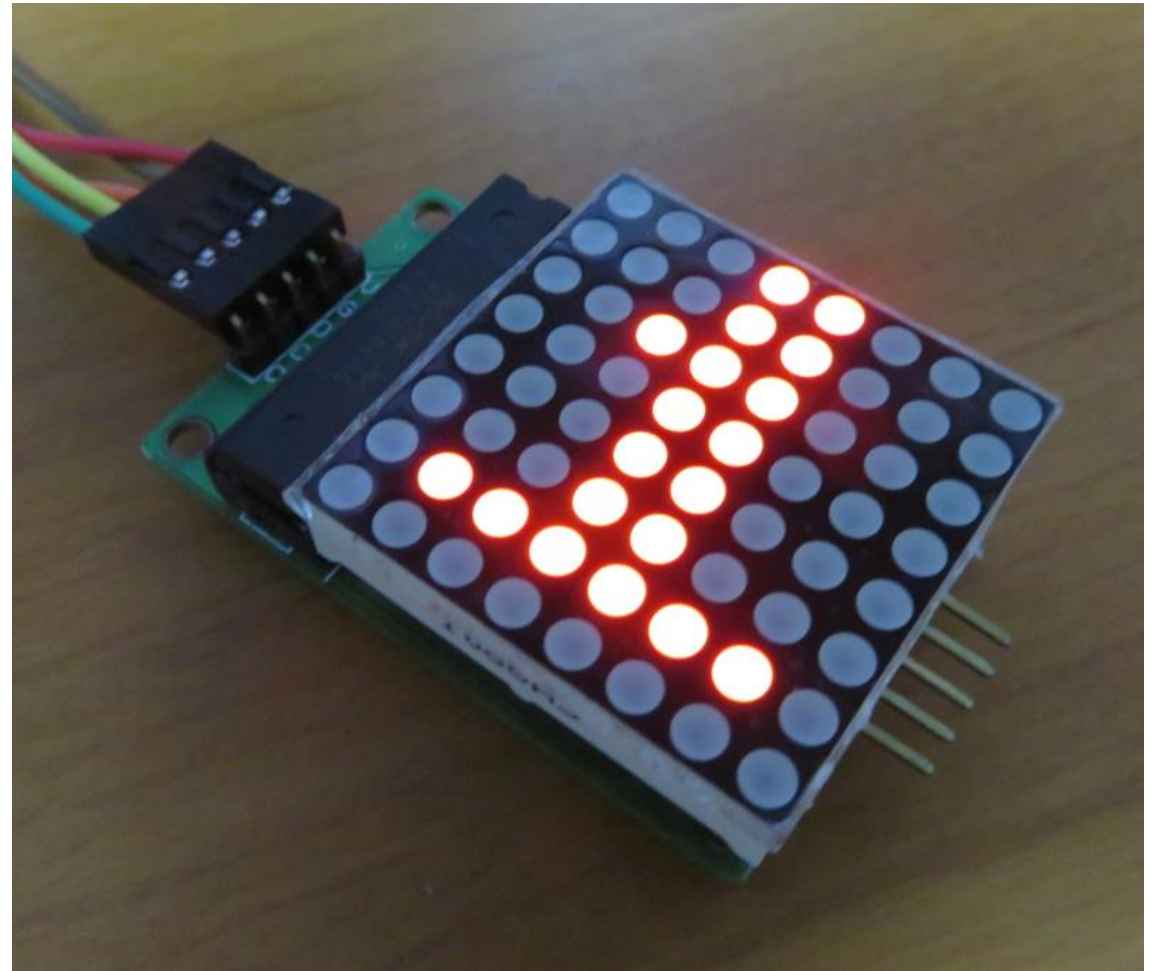
print('Temperature:', temperature, '°C, Pressure:',
pressure, 'Pa')
```

```
MicroPython v1.12-821-gdd1864a0b-dirty on 2020-05-09; GR-MANGO with RZA2M
Type "help()" for more information.
>>> from omron2smpb import OMRON2SMPB
>>>
>>> sensor = OMRON2SMPB(scl_pin=machine.Pin.cpu.PD2, sda_pin=machine.Pin.cpu.PD3, i2c_address=0x56)
>>>
>>> sensor.init()
id: 92
coe: bytearray(b'H\xc6\x02H\xf3V\xfe \x02\xcc\n\xc8\x07w\x01\x8d\xf2\xf8\xfc\xdc\xf0\xf6\xb1')
standby time: 4
temp_average: 3
press_average: 3
power_mode: 3
>>> temperature = sensor.get_temperature()
>>> pressure = sensor.get_pressure()
>>>
>>> print('Temperature:', temperature, 'C, Pressure:', pressure, 'Pa')
Temperature: 27.97554 C, Pressure: 99555.03 Pa
>>>
```

サンプル SPI - MAX7219 8x8 LED Matrix

- MAX7219 8x8 LED MatrixをSPIバス0 (Micropythonでは1)、CSをP84ピン (Header24)に接続. max7219.pyモジュールをUploadしておく。

```
import max7219
from machine import Pin, SPI
spi = SPI(1)
cs = Pin.cpu.P84
cs.init(cs.OUT, True)
display = max7219.Matrix8x8(spi, cs, 1)
# "1"を座標(0,0)でカラー1で描画
display.text("1",0,0,1)
display.show()
# 消去後、"A"を座標(0,0)にカラー1で描画
pyb.delay(1000)
display.fill(0)
display.show()
display.text("A",0,0,1)
display.show()
#消去後、縦線、横線をカラー1で描画
pyb.delay(1000)
display.fill(0)
display.hline(0,4,8,1)
display.vline(4,0,8,1)
display.show()
```



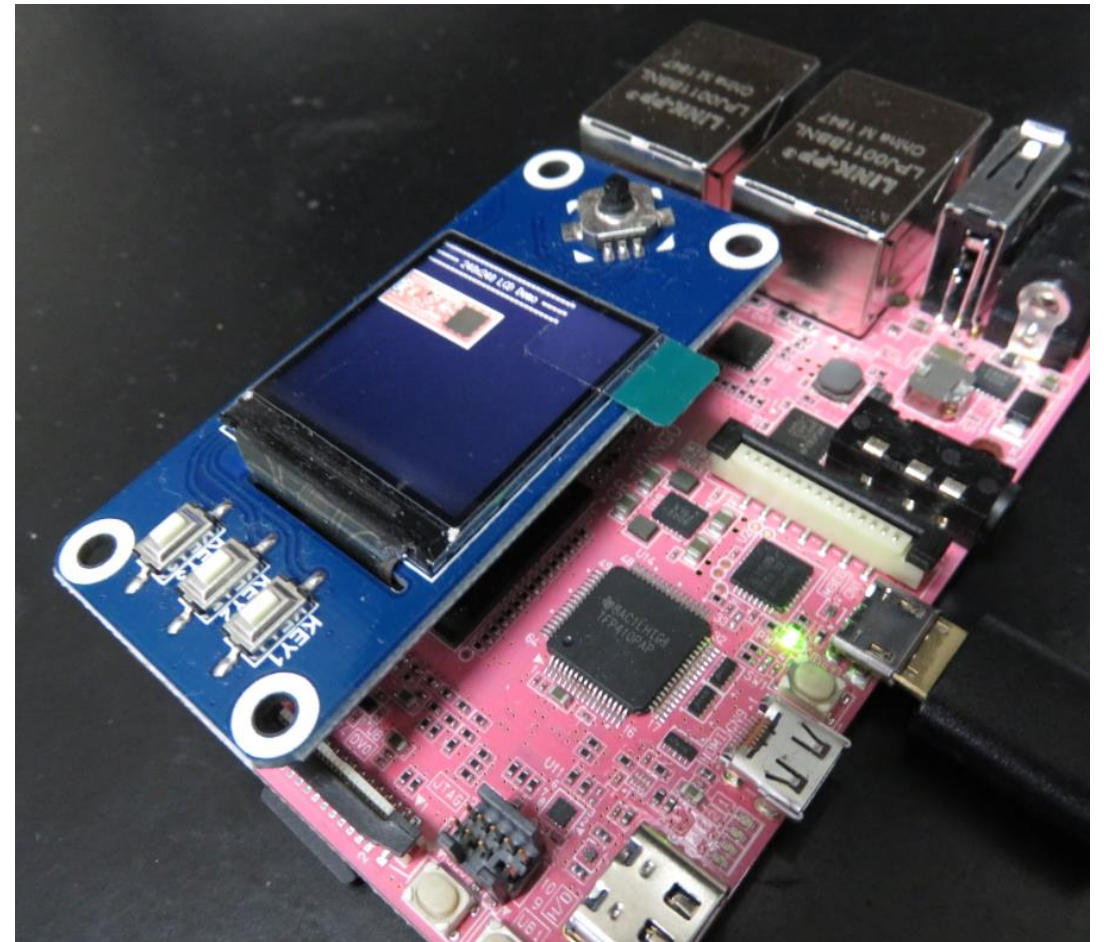
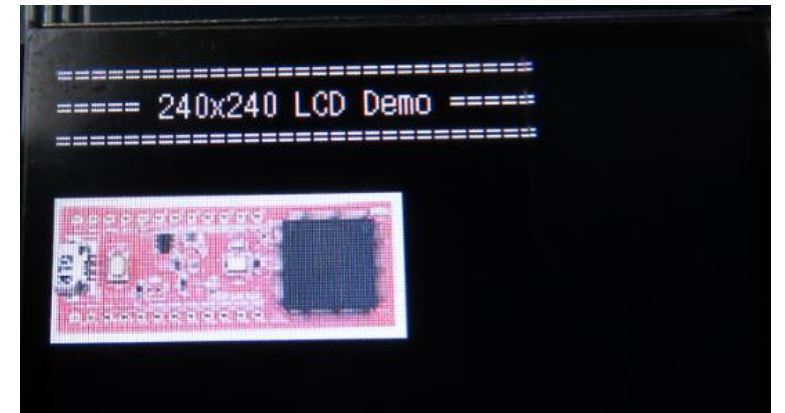
サンプル LCDSPI

- SPIインターフェースのLCDにビットマップを表示します

```
from pyb import LCDSPI, FONT, Pin
```

```
c=LCDSPI(lcd_id=LCDSPI.M_RASPI13LCDSPI,font_id=
FONT.MISAKIA_12,spi_id=1,baud=24000000,cs=Pin.
cpu.P84,clk=Pin.cpu.P87,dout=Pin.cpu.P86,
rs=Pin.cpu.PH6, reset=Pin.cpu.P45, din=Pin.cpu.P85)
c.puts("====¥r¥n")
c.puts("==== 240x240 LCD Demo ====¥r¥n")
c.puts("====¥r¥n")
```

```
import pyb
import os
sd=pyb.SDCard()
os.mount(sd, "/sd")
os.listdir("/")
c.disp_bmp_sd(0,50,'/sd/citrus24.bmp')
```



サンプル ネットワーク – HTTPアクセス

- ソケットモジュールを使用して、<http://micropython.org>にアクセスします。
- 不安定ですが、httpsアクセスも可能です。

```
import network
net=network.Ethernet()
net.ifconfig()
net.active(True)
net.ifconfig("dhcp")
net.ifconfig()
import usocket as socket
s = socket.socket()
addr = socket.getaddrinfo('micropython.org', 80)[0][-1]
s.connect(addr)
s.send(b'GET / HTTP/1.1\r\nHost: micropython.org\r\n\r\n')
data = s.recv(1000)
s.close()
data
```

```
>>> import network
>>> net=network.Ethernet()
>>> net.ifconfig()
('0.0.0.0', '0.0.0.0', '0.0.0.0', '0.0.0.0')
>>> net.active(True)
>>> net.ifconfig("dhcp")
>>> net.ifconfig()
('192.168.0.47', '255.255.255.0', '192.168.0.1', '192.168.0.1')
>>> import usocket as socket
>>> s = socket.socket()
>>> addr = socket.getaddrinfo('micropython.org', 80)[0][-1]
>>> s.connect(addr)
>>> s.send(b'GET / HTTP/1.1\r\nHost: micropython.org\r\n\r\n')
41
>>> data = s.recv(1000)
>>> s.close()
>>> data
b'HTTP/1.1 200 OK\r\nServer: nginx/1.12.2\r\nDate: Sat, 12 Jan 2019 03:27:43 GMT\r\nContent-Type: text/html; charset=utf-8\r\nContent-Length: 16839\r\nConnection: keep-alive\r\nVary: Accept-Encoding\r\nX-Frame-Options: SAMEORIGIN\r\n<!DOCTYPE html>\r\n\r\n<html lang="en">\r\n  <head>\r\n    <meta charset="utf-8">\r\n    <meta http-equiv="X-UA-Compatible" content="IE=edge">\r\n    <meta name="viewport" content="width=device-width, initial-scale=1">\r\n    <!-- The above 3 meta tags *must* come first in the head -->\r\n\r\n    <link rel="icon" href="/static/img/favico.ico'
```

おしまい

- 使い方の詳細は、MicroPythonのドキュメントを参照してください。
 - <https://docs.micropython.org/en/latest/>
- 移植したソースコードは以下のGithubのrzブランチに置く予定です。
 - <https://github.com/ksekimoto/micropython>
 - `git clone https://github.com/ksekimoto/micropython -b rz`
 - ビルド済のバイナリファイルは、rz_release フォルダ以下に格納する予定です。

その他、制限事項など。

- ビルド方法は、Github上のreadme.mdファイルに記載する予定です。
 - Boards¥GR_MANGO_DDにMBED USBドライブコピー用のビルド定義ファイル
 - Boards¥GR_MANGOフォルダがJ-linkデバッグ向けのビルド定義ファイル
- 内蔵フラッシュドライブに不整合が発生した場合には、起動時にSwitch(PinD7)を3秒押して、boot.py, main.pyを再作成してください。
- シリアル通信の処理にバグがあります。4Kバイトの受信バッファを超える通信で問題が発生する可能性があります。
- タイマーは、STM32とはMicroPythonのモジュールの実装が大幅に異なります。詳細はソースファイルをご確認ください。
- 各クラスのパラメータはオリジナルから機能を大幅に省略している場合があります。
- 各クラスのprintメソッドはオリジナルから機能を大幅に省略している場合があります。

Backupスライド

プログラム編集、実行環境

ツール	インストール	使い方など
Visual Studio Code + PyMakr (Windows / Linux / Mac(?))	拡張メニューでPyMakrと入力してインストール。Nodejs 6.9.5以降がインストールされていること。設定方法は、Pymakr > Global settingメニューで使用するCOMポートを設定する。	メニューから、Pythonのプログラムの実行、アップロード、ダウンロードが可能
uPyCraft v.1.1 (Windows環境) 	下記リンクより、実行ファイルをダウンロードして実行する。 https://randomnerdtutorials.com/uPyCraftWindows	メニューから、Pythonのプログラムの実行、アップロード、ダウンロードが可能 https://randomnerdtutorials.com/install-upycraft-ide-windows-pc-instructions/
MU エディタ (Windows/Linux) 	Githubからソースをダウンロードして、GR_MANGOのUSBを認識するように変更 (https://github.com/ksekimoto/mu/tree/pyboard)	Microbitのエディタとして利用されている。 https://codewith.mu/

オムロン2JCIE-EV センサーボード

センサ	部品番号	形式	メーカ	インターフェイス	
温湿度センサ	U1	SHT30-DIS-B	Sensirion	I2C (0x44)	
周辺光センサ	U2	OPT3001DNP	Texas Instruments	I2C (0x45)	
MEMS絶対圧センサ	U3	2SMPB-02E	オムロン	I2C (0x56)	
MEMSデジタル モーションセンサ	U5	LIS2DW12	STMicroelectronics	SPI (SPI0 -CS:P84)	
MEMSマイクロフォン	U6	SPH0645LM4H-B	Knowles	I2S	

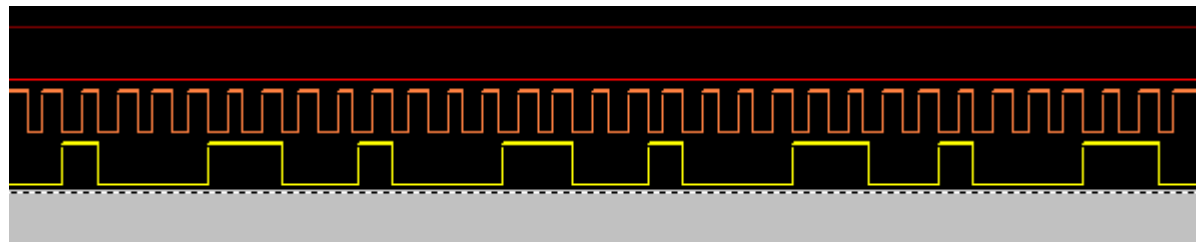
サンプル – NeoPixel 編

```
from ws2812 import WS2812
chain = WS2812(spi_bus=1, led_count=12)
data = [
    (255, 0, 0),  # red
    (0, 255, 0),  # green
    (0, 0, 255),  # blue
    (85, 85, 85), # white
    (255, 0, 0),  # red
    (0, 255, 0),  # green
    (0, 0, 255),  # blue
    (85, 85, 85), # white
    (255, 0, 0),  # red
    (0, 255, 0),  # green
    (0, 0, 255),  # blue
    (85, 85, 85), # white
]
chain.show(data)
```



NeoPixelのドライバの補足

- オリジナルは、<https://github.com/JanBednarik/micropython-ws2812>
- SPIの1クロックをNeoPixelのLowパルス、2クロックをHighパルスに割り当て、4クロックで1ビットを表現し、ソフト処理、割り込みで波形が乱れないように、DMAで送信しています。
- 本実装では、DMAは使用せず、エンコード方法を踏襲し、4クロックで1ビットを表現し、SPIの32ビット転送で1バイトデータを転送しています。



- 01010101の1バイトデータを送信する際の波形