RX63N/RX631向け MicroPython 使い方

2018/12/22

BY KSGADGET

MicroPython for RX63N/RX631

- MicroPythonのSTM32の実装をRX63N/RX631に移植したものです。
- ターゲットのボードは、GR-CITRUSとGR-SAKURAの2枚です。
- 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、CANおよび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。
  - ESP8266およびESP32向けの実装とはモジュールが異なります。

ボードの定義ファイルを用意することで、RX63NおよびRX631のボード向けに容易に移植できます。ただし、このドキュメントでは取り上げていません。

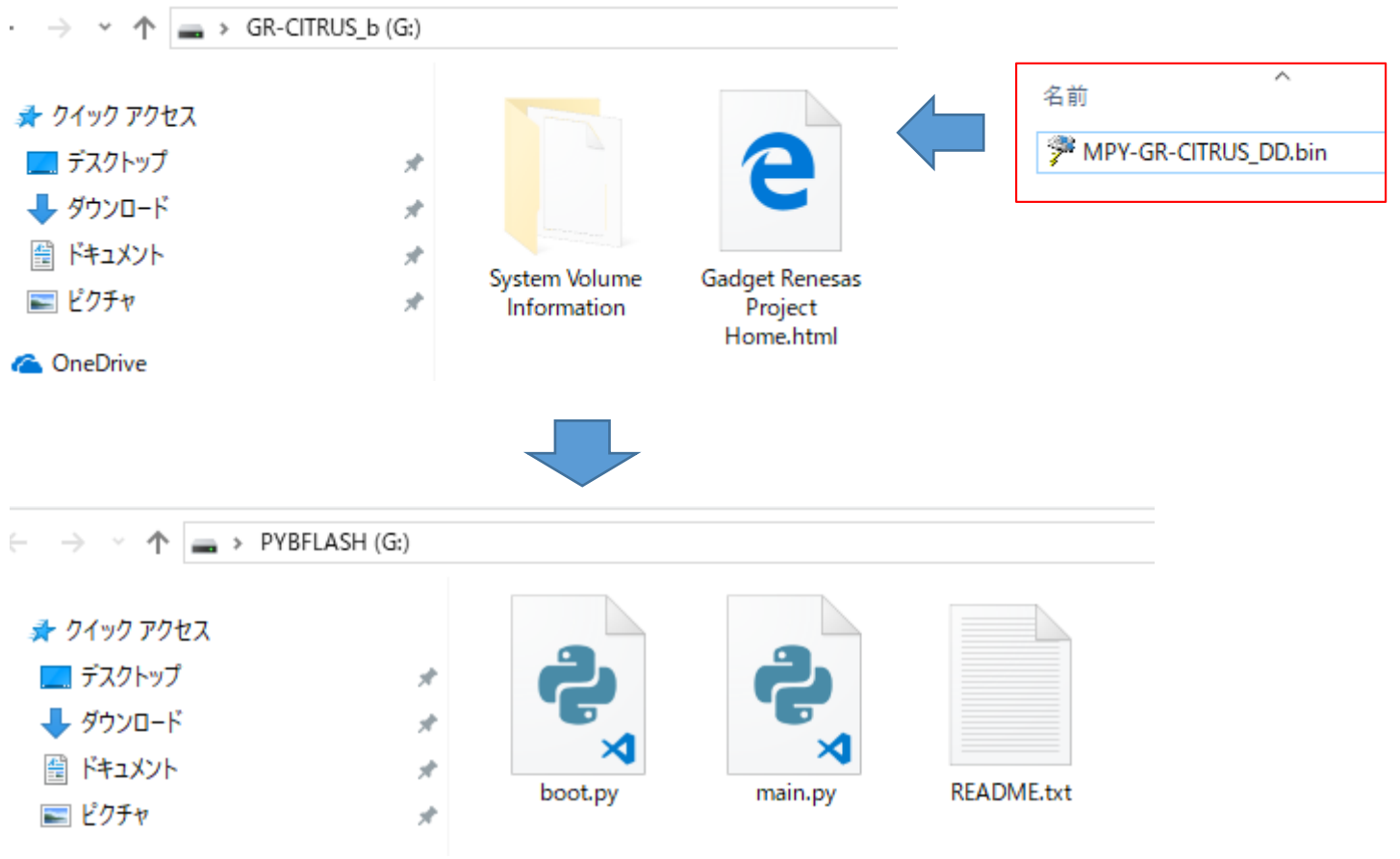
用意するもの

- 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-CITRUS (またはGR-SAKURA)
- 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。
  - MAX7129 8x8 LED Matrixボード
  - NeoPixel Ring 12個
  - SPI LCD 240x320 (ILI9340コントローラ)

インストール方法

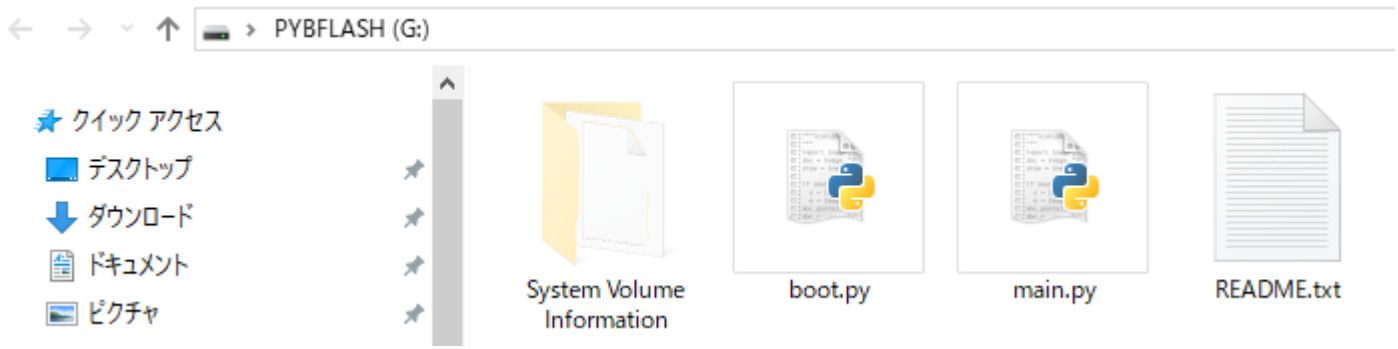
- 右記のURLよりGR-CITRUS用のMicroPythonバイナリファイル(MPY-GR_CITRUS_DD.binまたはMPY-GR_SAKURA_DD.bin) をダウンロードします。
- GR-CITRUSにUSBケーブルに接続し、Resetボタンを押すと、書き込みモードになります。
- そこで、バイナリファイルをGR-CITRUSドライブにDrag & Dropして書き込みます。
- 最初の起動時の初期化中はLEDが点灯し、5秒程度かかります。
- LEDが点灯中はキャッシュ中で内蔵フラッシュドライブには書き込まれていません。数秒後に内蔵フラッシュに書き込まれます。

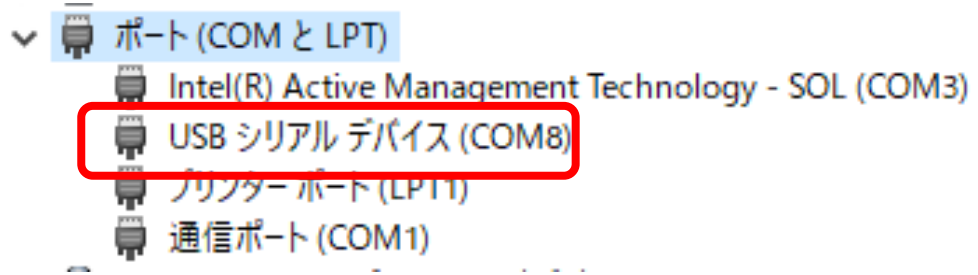
https://github.com/ksekimoto/micropython/raw/rx/rx_releases/gr_citrus/latest/MPY_GR_CITRUS_DD.bin
https://github.com/ksekimoto/micropython/raw/rx/rx_releases/gr_sakura/latest/MPY-GR_SAKURA_DD.bin



使い方 – GR-CITRUS編

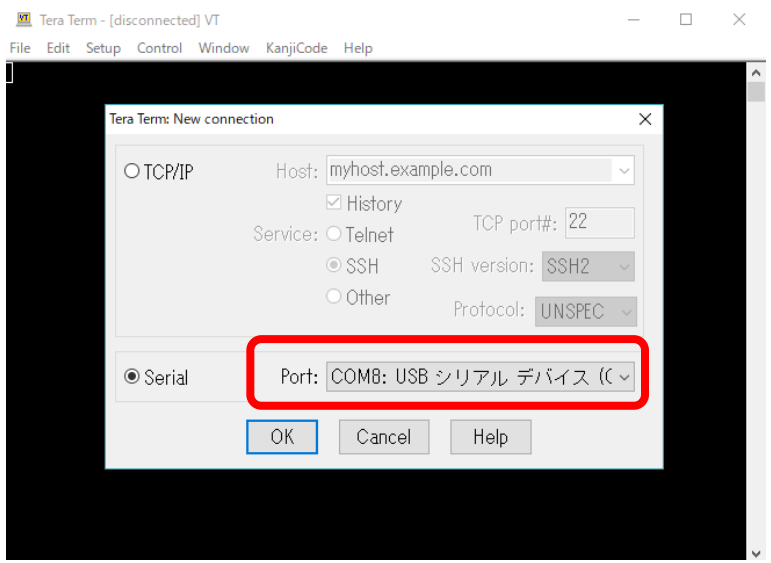
- GR-CITRUSのUSBコネクタとWindow 10 PCのUSBコネクタを接続します。
- GR-CITRUSのRX631 CPUのFlashメモリの一部が外部ドライブとして認識されます。(後で説明)
- 同時に、デバイスマネージャのポートでUSBシリアルデバイスとして認識されます。

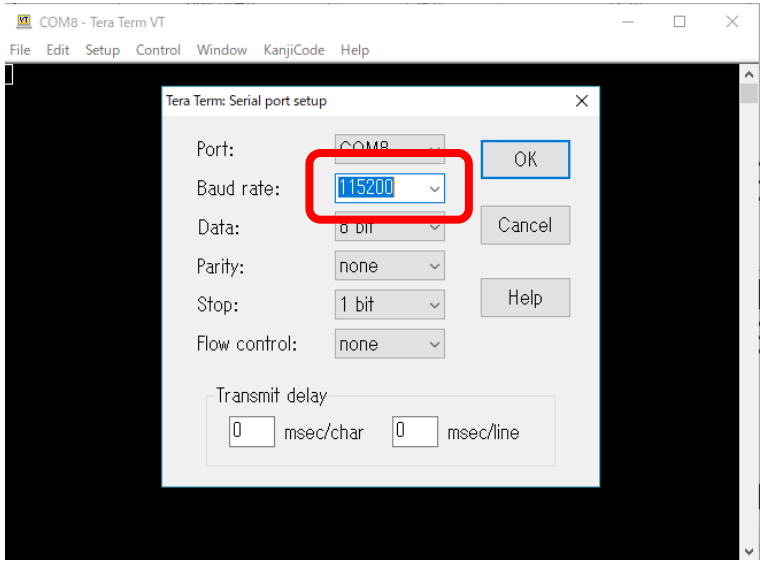


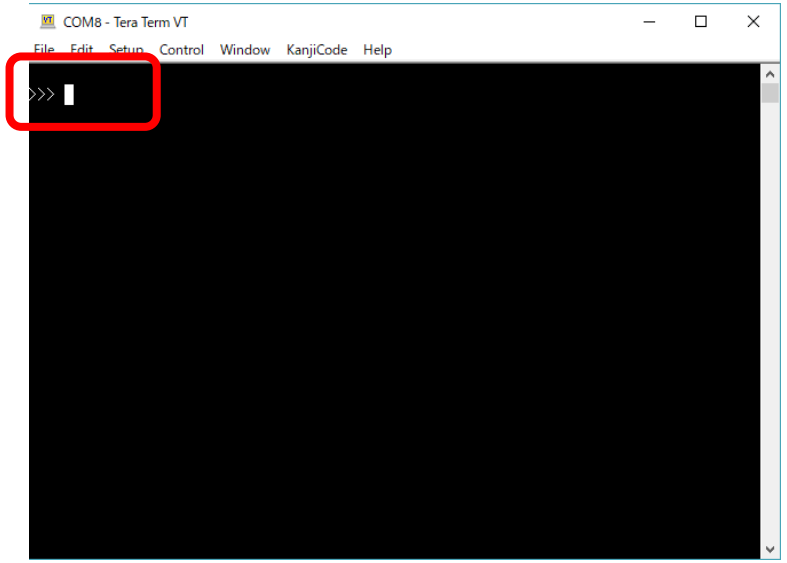


使い方 – GR-CITRUS編

- 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。



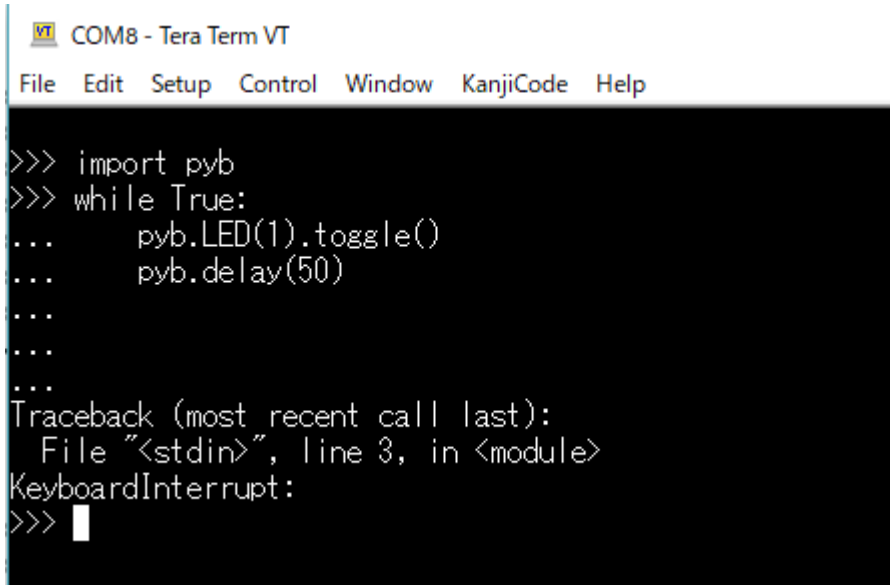




最初のサンプル Lチカ

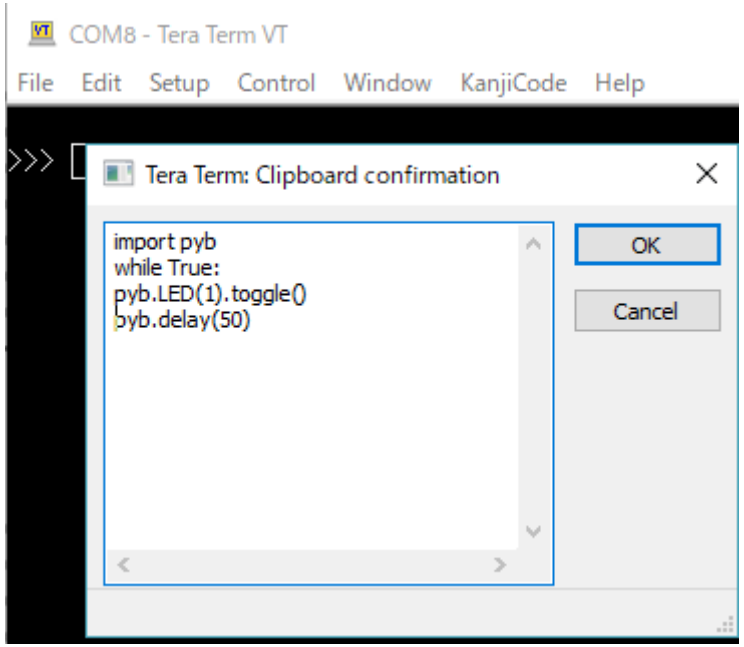
- 最初のサンプルとして、GR-CITRUSボード上のLEDを点灯してみます。
- 下記のプログラムを入力してみます。
- Enterキーを数回入力します。
- Ctrl-Cキーの入力でプログラムが終了します。

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



REPLの補足

- REPLではインデントが自動処理されますので、Cut&Pasteする際には行頭のスペースは入力しません。



- Ctrl-Dキーの入力でソフトウェアリブートします。

```
>>>
PYB: sync filesystems
PYB: soft reboot
icroPython v1.9.4-515-g5ee643622-dirty on 2018-12-02; GR-CITRUS with RX631
Type "help()" for more information.
>>>
```

内蔵フラシュメモリドライブの使い方

- 内蔵フラシュメモリドライブ(256KB)には初期起動時にはboot.py, main.pyおよびREADME.txtファイルが含まれています。
- MicroPython起動時には、(SD-CARDが未接続の場合)フラッシュメモリドライブ上のboot.pyが実行され、その後main.pyが実行されます。
- 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。新規ファイルはPCで編集したファイルをDrag&Dropでフラッシュメモリドライブにコピーしてください。

boot.py

```
1  # boot.py -- run on boot-up
2  # can run arbitrary Python, but best to keep it minimal
3
4  import machine
5  import pyb
6  #pyb.main('main.py') # main script to run after this one
7
```

main.py

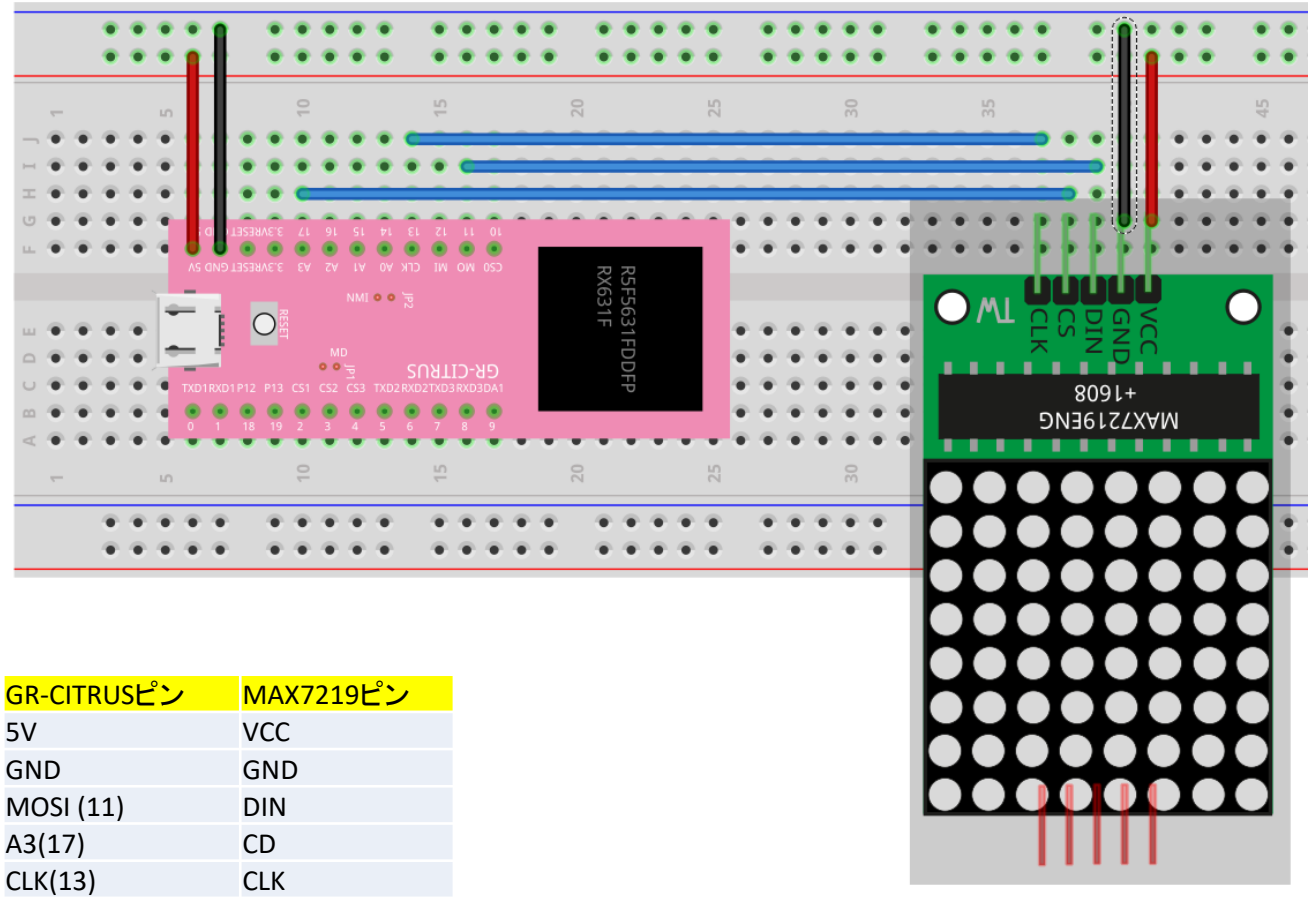
```
1  # main.py -- put your code here!
2
```

初期起動時にはmachineおよびpybモジュールのインポートだけ設定されています。
8KBを超えるサイズのファイルはコピーできません。

サンプル - MAX7219 8x8 LED Matrix

接続
- 右図のようにGR-CITRUSとMAX7219ボードのピンを接続します

プログラム
- MAX7219用のモジュールが入ったファイル(max7219.py)をフラッシュドライブにコピーします。モジュールはGithubのrx_releasesフォルダ以下にあります。
- オリジナルは、https://github.com/mcauser/micropython-max7219/blob/master/max7219.py
- 実行するプログラムは次のページの通りです。



GR-CITRUSピン	MAX7219ピン
5V	VCC
GND	GND
MOSI (11)	DIN
A3(17)	CD
CLK(13)	CLK

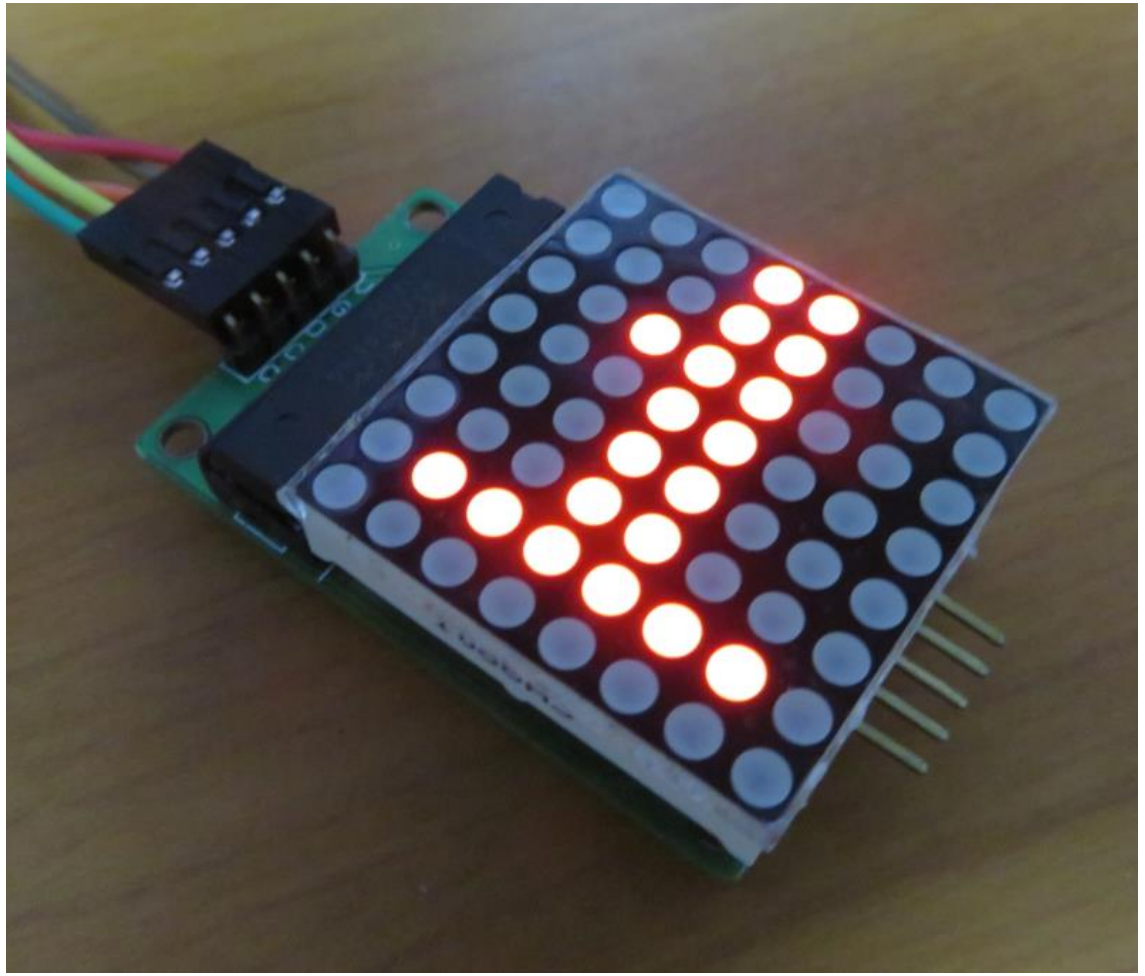
Aitendo: 8x8マトリックスモジュール（8x8）[M7SEGX1R-7219]
http://www.aitendo.com/product/10241
または
秋月電子:MAX7219使用赤色ドットマトリクスLEDモジュール[SKU-20-111-978]
http://akizukidenshi.com/catalog/g/gM-09984/

サンプル - MAX7219 8x8 LED Matrix

```
import max7219
from machine import Pin, SPI
spi = SPI(1)
cs = Pin.cpu.P43
cs.init(cs.OUT, True)
display = max7219.Matrix8x8(spi, cs, 1)
display.text('1',0,0,1)
display.show()

display.fill(0)
display.show()
display.text('A',0,0,0)
display.show()

display.pixel(0,0,1)
display.pixel(1,1,1)
display.hline(0,4,8,1)
display.vline(4,0,8,1)
display.show()
```

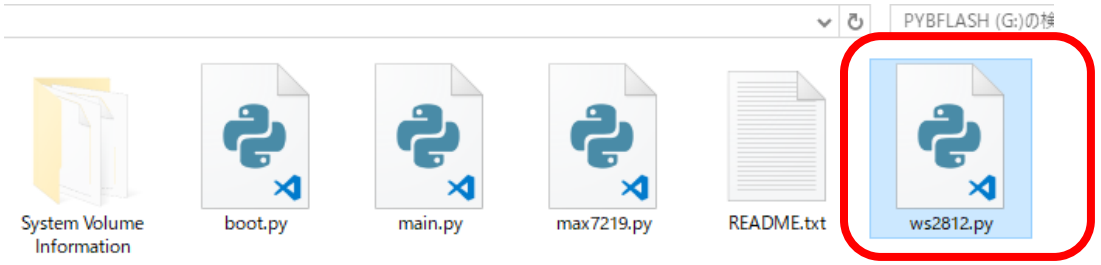


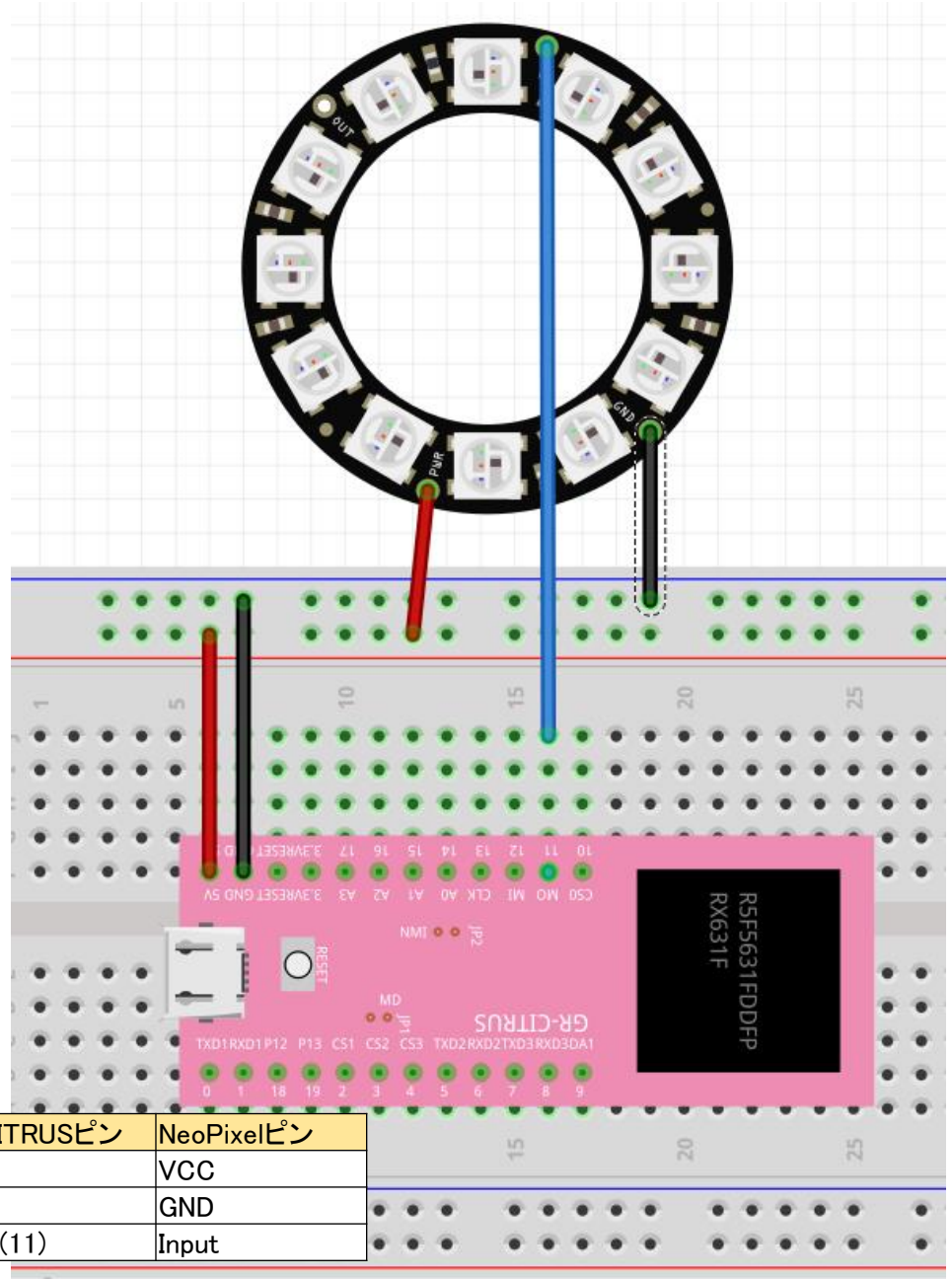
サンプル – NeoPixel 編

スイッチサイエンス: NeoPixel Ring - 12連フルカラーシリアルLED
https://www.switch-science.com/catalog/1593/

接続
- 右図のようにGR-CITRUSとNeoPixelリングのピンを接続します

プログラム
- NeoPixel用のモジュールが入ったファイル(ws2812.py)をフラッシュドライブにコピーします。モジュールはGithubのrx_releasesフォルダ以下にあります。
- 実行するプログラムは次のページの通りです。

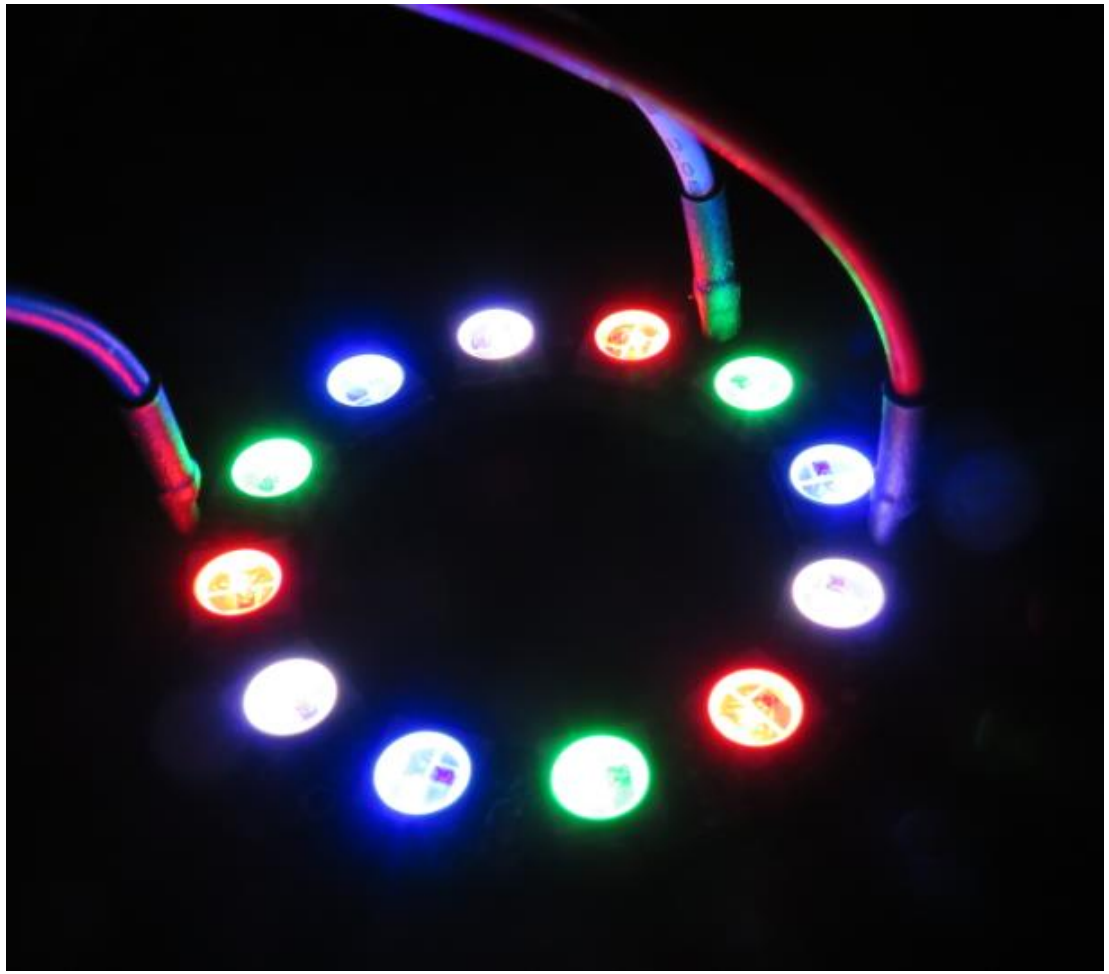




GR-CITRUSピン	NeoPixelピン
5V	VCC
GND	GND
MOSI (11)	Input

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)
```

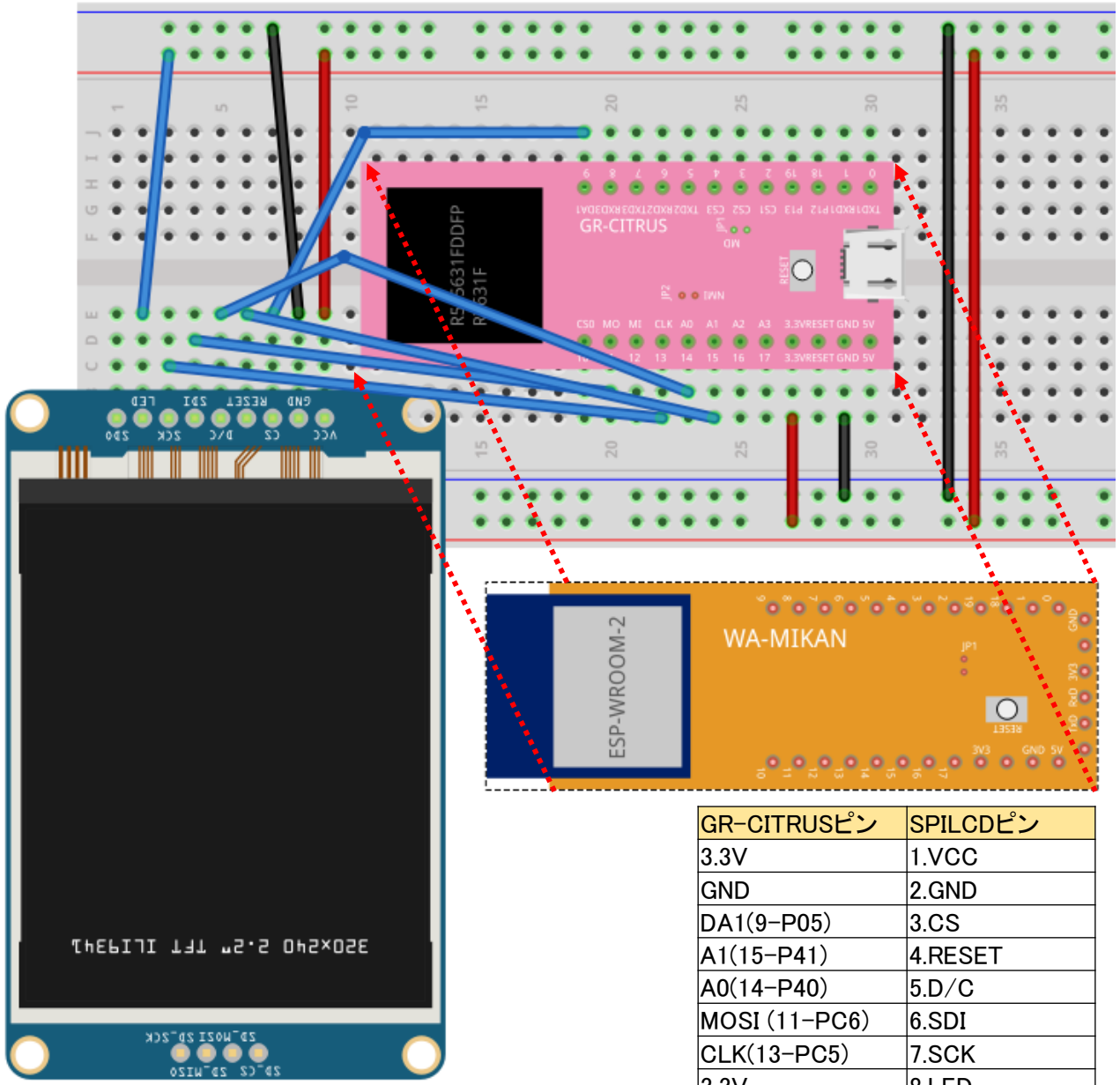


サンプル – SPI LCD編

接続
- 右図のようにGR-CITRUSとSPI LCD 240x320のピンを接続します
- Hardware SPIピンを利用します。

プログラム
- 実行するプログラムは次のページの通りです。

その他
- WA-MIKANを重ねた場合には、SD-CARDからBMPファイル、JPEGファイルを表示できます。

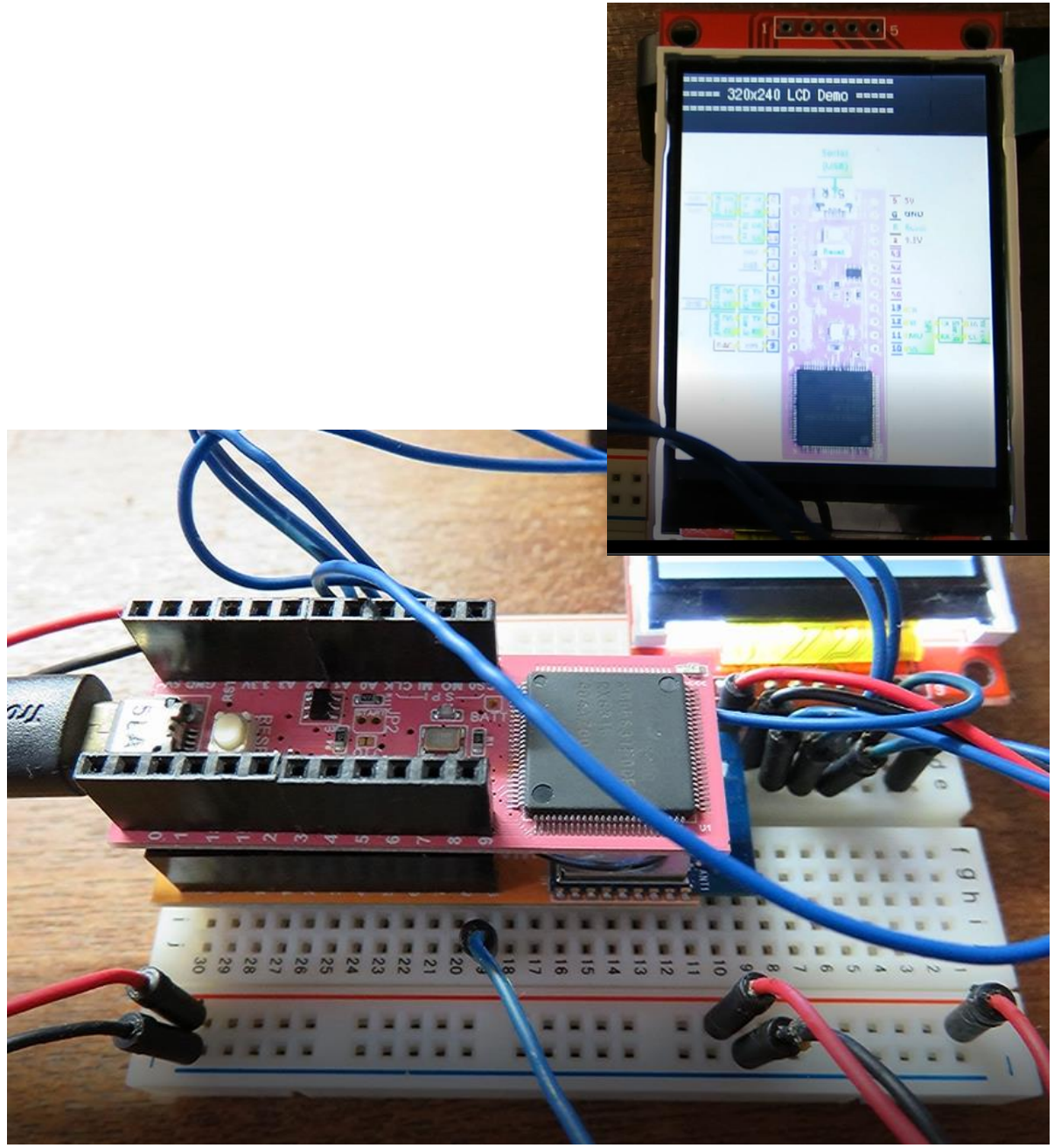


GR-CITRUSピン	SPILCDピン
3.3V	1.VCC
GND	2.GND
DA1(9-P05)	3.CS
A1(15-P41)	4.RESET
A0(14-P40)	5.D/C
MOSI (11-PC6)	6.SDI
CLK(13-PC5)	7.SCK
3.3V	8.LED
NC(未接続)	9.SDO

サンプル – SPI LCD編

```
from pyb import LCDSPI, Pin
c=LCDSPI(lcd_id=3,font_id=1,spi_id=0,baud=24000000,cs=Pin.cpu.P05,clk=Pin.cpu.PC5,dout=Pin.cpu.PC6, rs=Pin.cpu.P40, reset=Pin.cpu.P41, din=Pin.cpu.P42)
c
c.puts("============================¥r¥n")
c.puts("===== 320x240 LCD Demo =====¥r¥n")
c.puts("============================¥r¥n")

# WA-MIKANのSDカードを使用した場合
c.disp_jpeg_sd(0,50,'CITRUS00.JPG')
c.disp_bmp_sd(0,100,’CITRUS00.BMP’)
```



- Lcd_idは0-3まで、3がSPI LCD240x320です。
- Font_idは0-3まで、0: 8x8, 1:6x12, 2: 8x8 Unicode, 3:6x12 Unicode
- Spi_idは-1, 0-2まで、-1: Software SPI, 0-2: Hardware SPI CH

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rxブランチにあります
  - https://github.com/ksekimoto/micropython
  - git clone https://github.com/ksekimoto/micropython -b rx
  - ビルド済のバイナリファイルは、rx_release フォルダ以下に格納しました。

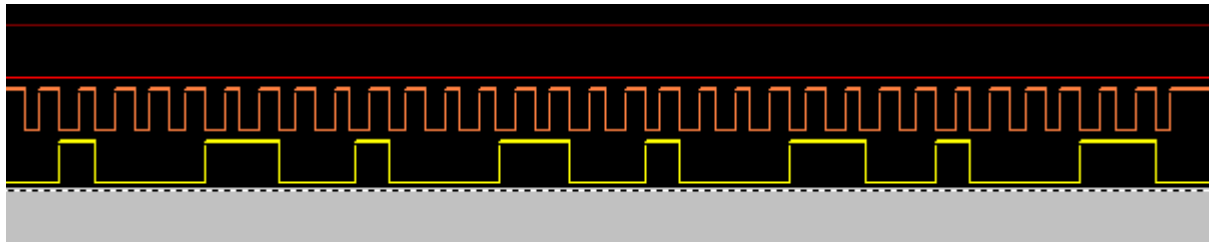
Backup

その他、制限事項など。

- ビルド方法は、Github上のreadme.mdファイルに記載されています。
  - Boards¥GR_CITRUS_DDおよびGR_SAKURA_DDフォルダにUSBドライブコピー用のビルド定義ファイル
  - Boards¥GR_CITRUSおよびGR_SAKURAフォルダがE1デバッグ向けのビルド定義ファイル
- 内蔵フラッシュを利用したドライブに、ファイルを書き込んだり、変更した際は、データはキャッシュされ、５秒後以降に、REPLのループプロセスでフラッシュされ、書き込まれます。
- 内蔵フラッシュドライブに不整合が発生した場合には、CPUのフラッシュメモリをイレーズし、USBマスストレージファームウェアから再度書き込んでください。フォーマットすると、boot.py, main.pyが正常に再作成されません。
- シリアルは、GR-CITRUSでは、0-5まで6チャンネルは送受信バッファは1024バイト、GR-SAKURAでは、0-3までの４チャンネルで送受信バッファは512バイトです。
- 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。

NeoPixelのドライバの補足

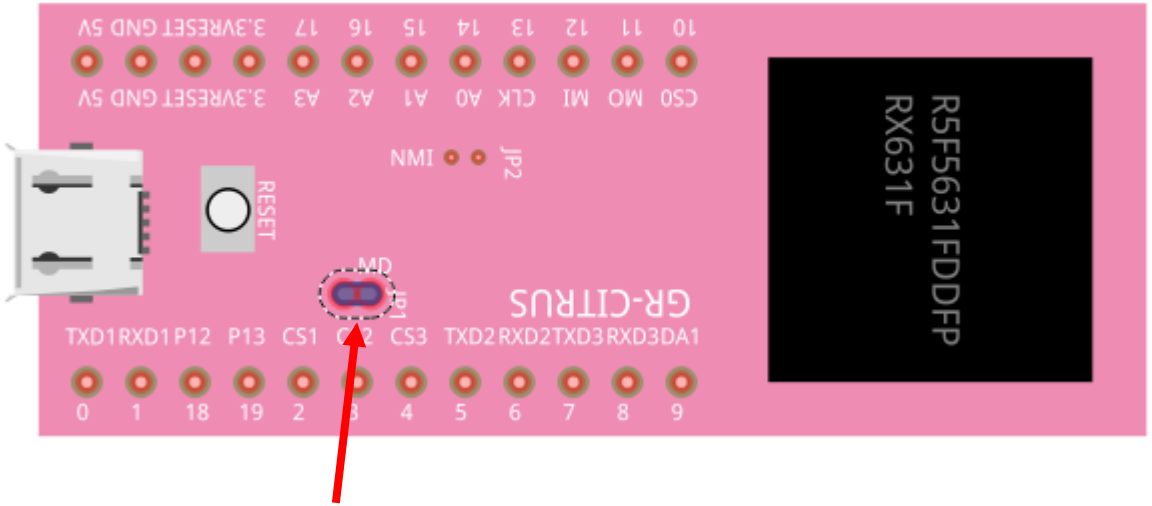
- オリジナルは、https://github.com/JanBednarik/micropython-ws2812
- SPIの1クロックをNeoPixelのLowパルス、2クロックをHighパルスに割り当て、4クロックで1ビットを表現し、ソフト処理、割り込みで波形が乱れないように、DMAで送信しています。
- 本実装では、DMAは使用せず、エンコード方法を踏襲し、４クロックで1ビットを表現し、SPIの３２ビット転送で1バイトデータを転送しています。



- 01010101の１バイトデータを送信する際の波形

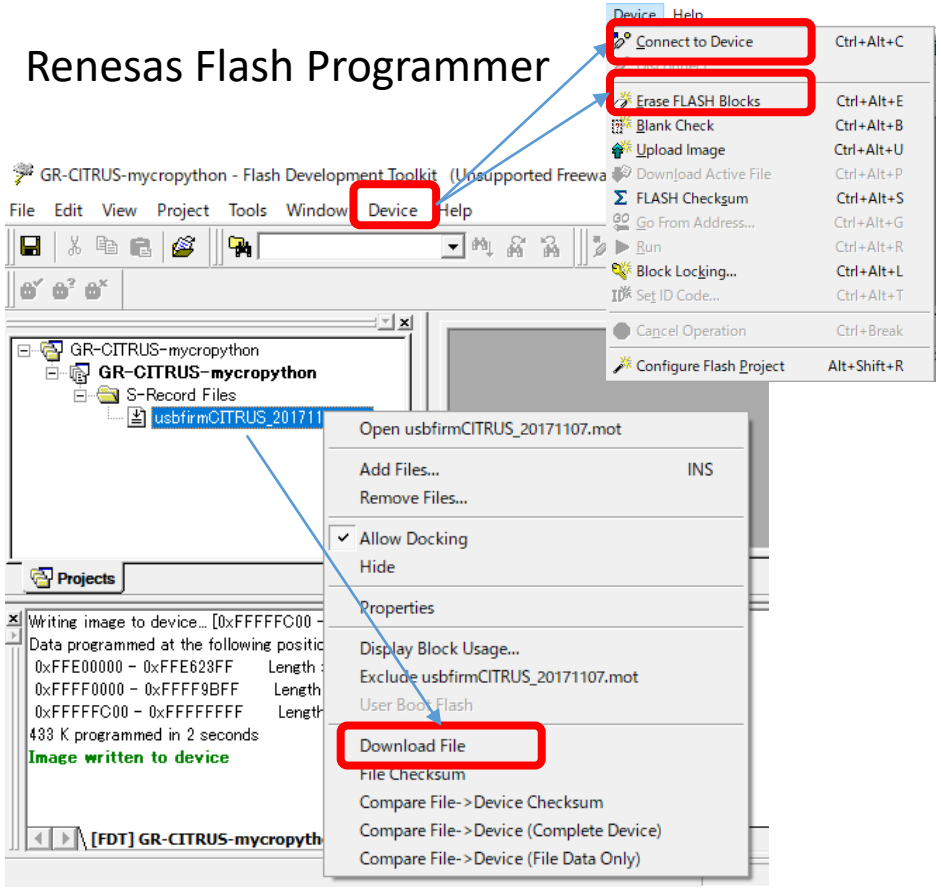
GR-CITRUS USBマスストレージファームウェア更新

- 更新方法は、下記のURLに記載されています。
  - http://gadget.renesas.com/ja/product/citrus.html



MDジャンパをクローズして起動
Renesas USB Development Toolsとして認識される

Renesas Flash Programmer



Device メニューでGR-CITRUSを認識させ、内蔵フラッシュをイレーズ後、書き込むファイルをS-Record Filesに登録し、Download Fileで書き込む。