How to use MicroPython for GR_ROSE

MicroPython for GR_ROSE

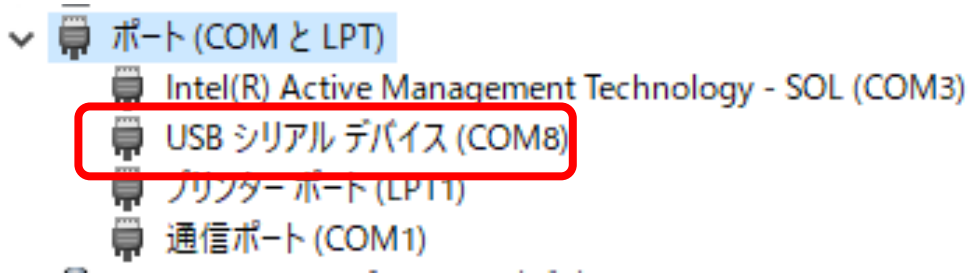
- This is a port of MicroPython's STM32 implementation to GR_ROSE (RX65N).
- The ported features are mostly compliant with the STM32 implementation (pyboard).
- See the pyboard manual (https://docs.micropython.org/en/latest/) for details on how to use it.
- However, USB, CAN, WDT… are not implemented.
- Some Pyboard module parameters are not implemented. See the source code for details.
- The module is different from MicroPython's implementation for ESP8266 and ESP32.

What to prepare

- PC running Windows 10
  - Terminal software (Tera Term or Putty is used here.)
- GR_ROSE
- Other gadgets
- This document covers the following sample gadgets:
  - OMRON 2JCIE board

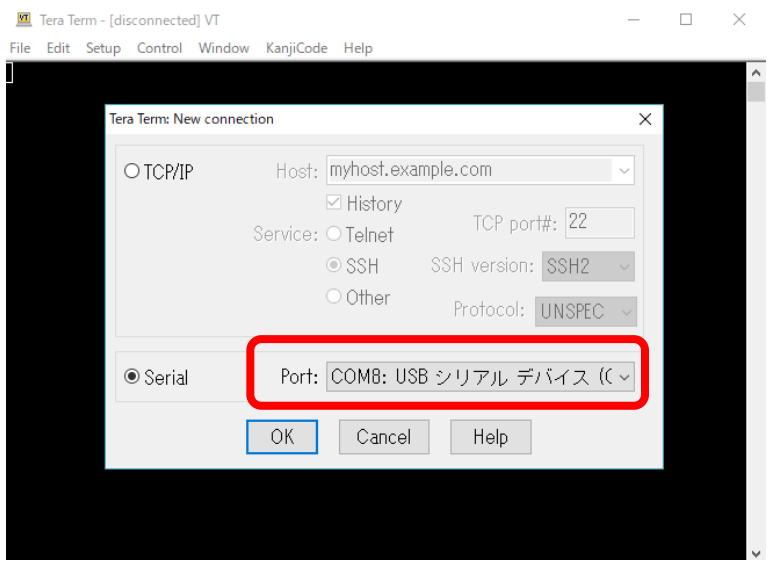
How to use

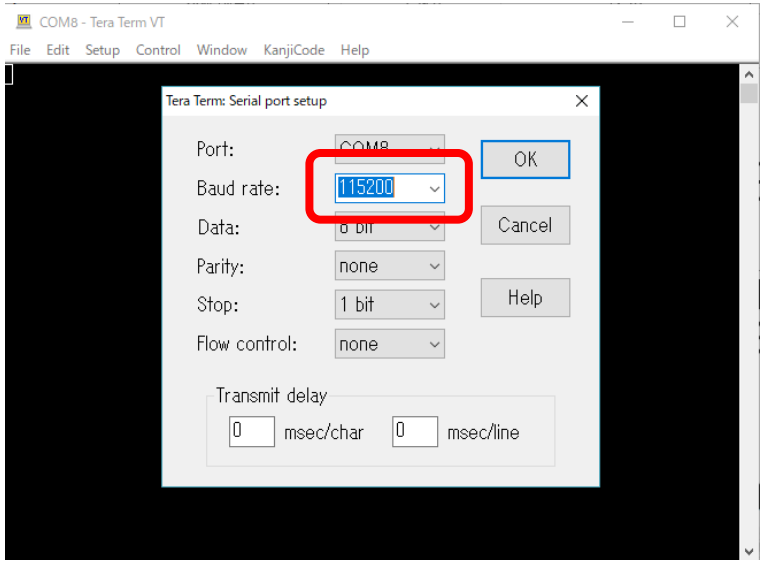
- Connect the USB micro connector of GR-ROSE and the USB connector of Window 10 PC.
- It is recognized as a USB serial device by the port of Device Manager.

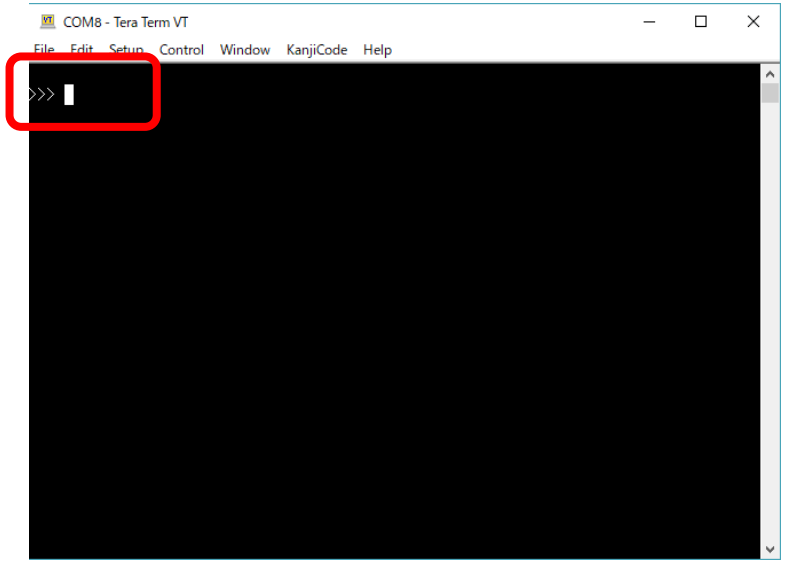


How to use

- Start Tera Term, select the recognized COM port, select Setup – Serial port menu, select 115200 for Baud rate, and press the Enter key. The >> MicroPython REPL prompt should appear.
- MicroPython programs can be executed from this console.



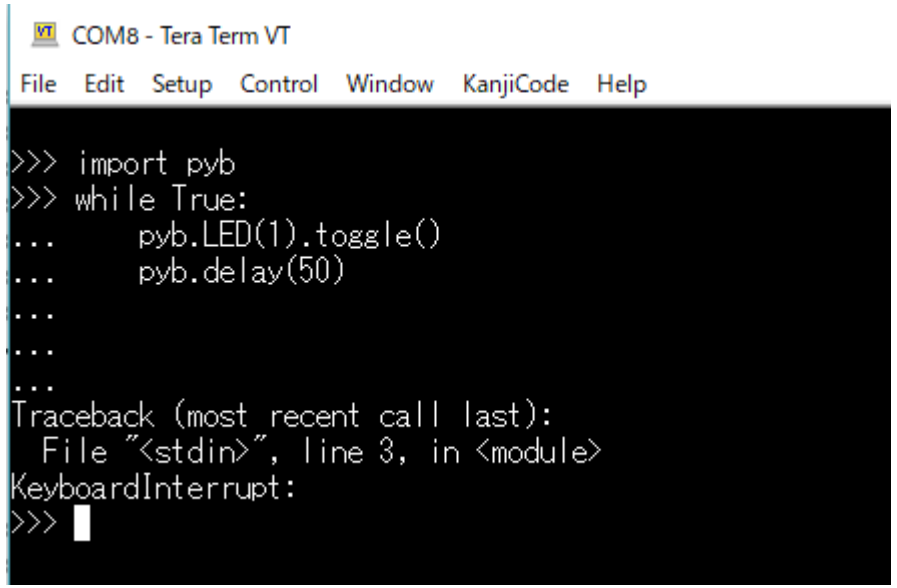




First Sample LED blink

- As the first sample, let's turn on the LED on the GR-ROSE board.
- Enter the program below.
- Enter the Enter key several times.
- Press Ctrl-C to end the program.

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

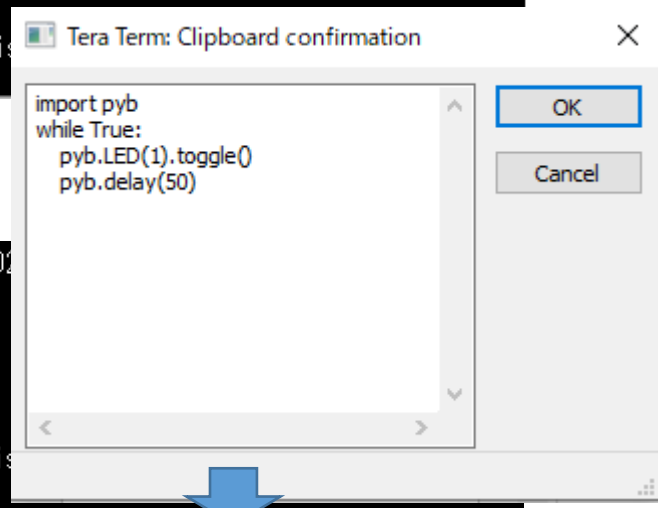


REPL notes

- To Cut & Paste an indented program, press Ctrl-E and then Paste.
- Run the program with Ctrl-D.
- Stop the program with Ctrl-C.
- Then use Ctrl-D to reboot the software.

```
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 finis
===
```

Ctrl-E to Paste mode




```
MicroPython v1.12-816-g4573864c0-dirty on 202
Type "help()" for more information.
>>>
>>>
>>>
>>>
paste mode; Ctrl-C to cancel, Ctrl-D to finis
=== import pyb
=== while True:
===     pyb.LED(1).toggle()
===     pyb.delay(50)
===
Traceback (most recent call last):
  File "<stdin>", line 4, in <module>
KeyboardInterrupt:
>>>
```

Ctrl-D to execute

Ctrl-C to stop

Ctrl-D to reboot

```
Traceback (most recent call last):
  File "<stdin>", line 4, in <module>
KeyboardInterrupt:
>>>
MPY: sync filesystems
MPY: soft reboot

MicroPython v1.12-816-g4573864c0-dirty on 2020-05-09; GR-MANGO with RZA2M
Type "help()" for more information.
>>>
```

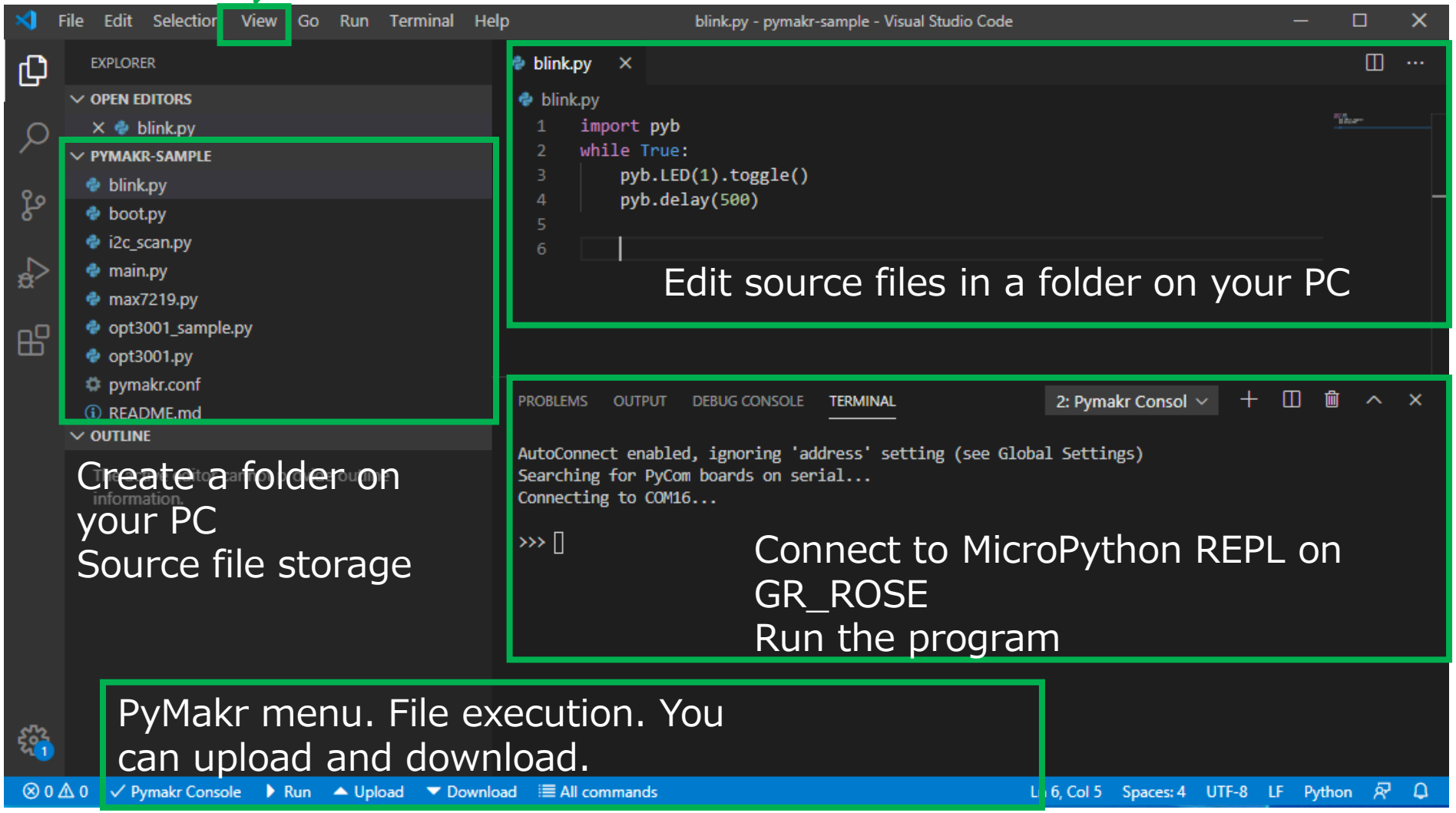
Storage

- With the GR_ROSE implementation, the built-in Flash and SD card can be used as storage.
- The built-in flash reserves 256KB from 0xfffa0000.
- At startup, it becomes the "/flash" default folder, and after initialization there are two files, boot.py and main.py.
- You can execute the program at startup by updating main.py.
- If the SD card is inserted at startup, the "/sd" folder will be the default folder.

```
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']
```

Program editing with Visual Studio Code + PyMakr

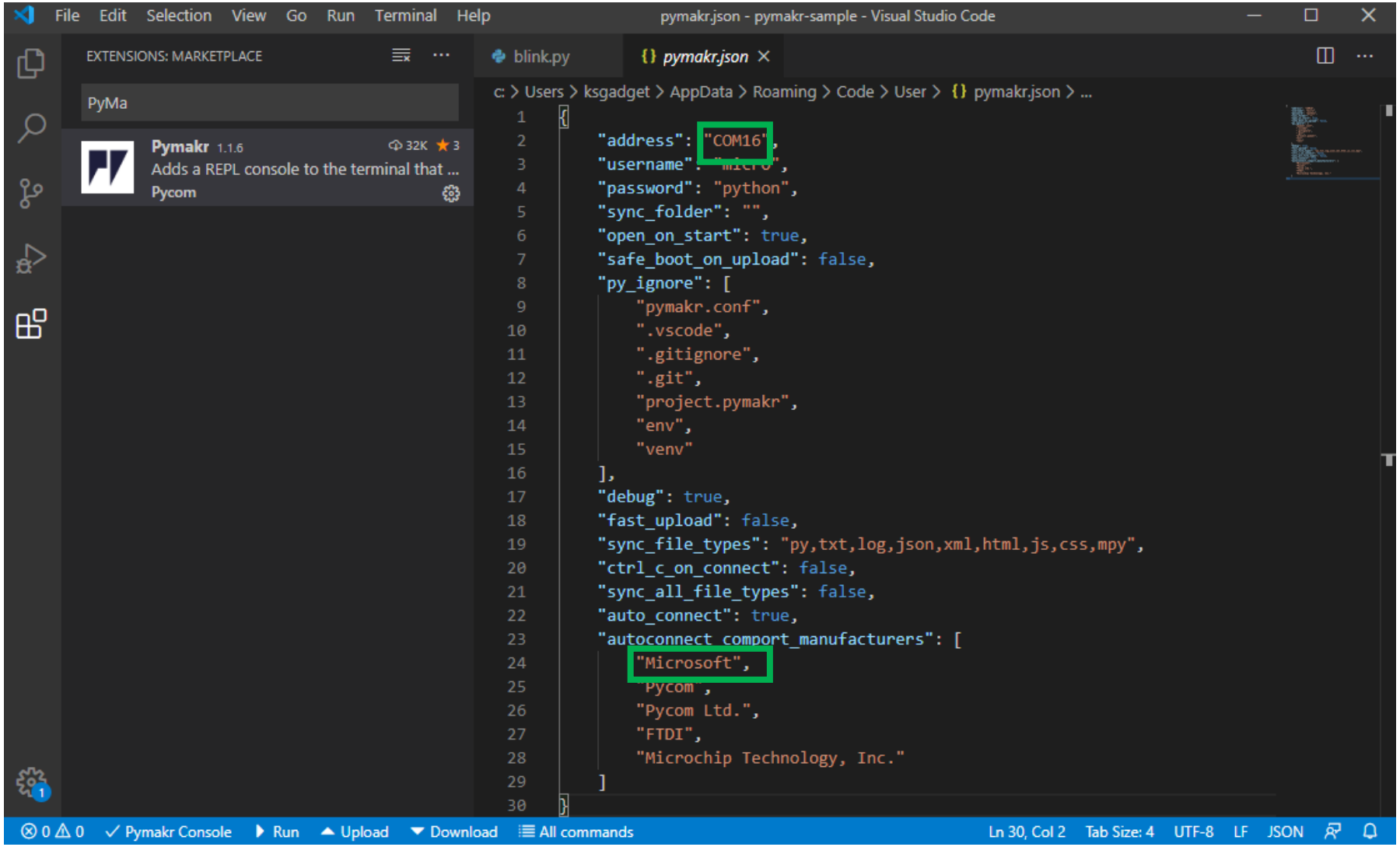
View – Access the PyMakr menu from the Command Palette



Install Visual Studio Code + PyMakr

- Install Visual Studio Code
  - https://visualstudio.microsoft.com/ja/
- Install Nodejs (6.9.5 or later) Maybe the latest version is fine
  - https://nodejs.org/en/
- Install PyMaker plugin in Visual Studio Code
  - Click the extension menu icon in the lower left corner, type PyMakr in the search box for EXTENSIONS: MARKET PLACE, and install.
- Register GR_ROSE COM port etc. in PyMaker plugin
  - From the View – Command Palette menu, select PyMakr – Global settings and set the COM port to xxxx of “address”: “xxxx” in the pymakr.json file.

PyMakr – Global settings



PyMakr Tips

- PyMakr files are stored in the following folders.
  - C: ¥Users ¥username ¥.vscode ¥extensions ¥pycom.pymakr-1.1.6
- PyMakr configuration file is placed in the following folder.
  - C: ¥Users ¥Username ¥.AppData ¥Roaming ¥Code ¥User
- If the COM port connection is intermittent in Windows environment, line 139 of C: ¥Users ¥username ¥.vscode ¥extensions ¥pycom.pymakr-1.1.6 ¥lib (and ¥src) ¥connections ¥pyserial.js Comment out from about.
- If Upload fails, set upload_chunk_size from 512 to 256 in config.js.

```
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;
  }
}
```

CN3_R	RESET
CN3_M	FINED
CN3_G	GND
CN3_V	3.3V

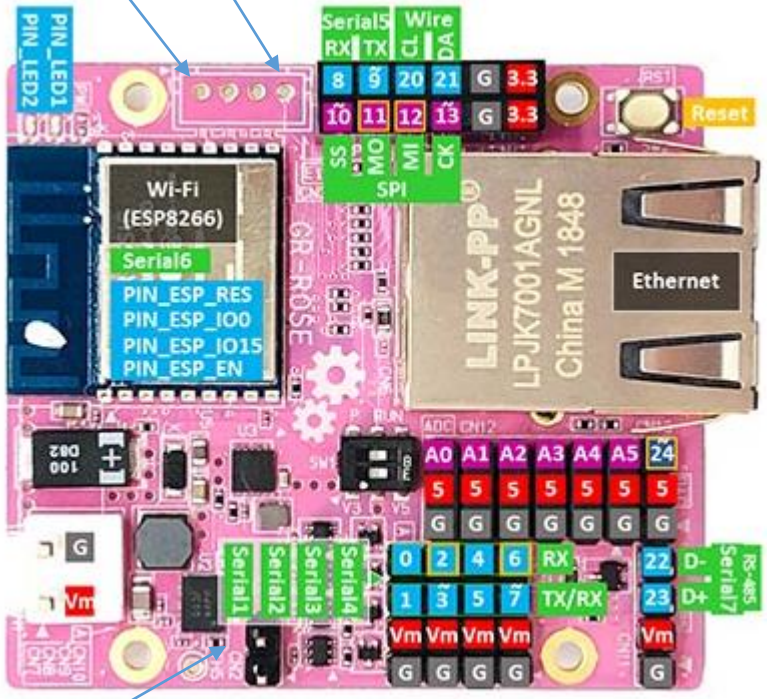
CN4_RX(8)	P30
CN4_TX(9)	P26
CN4_CL(20)	P52
CN4_DA(21)	P50
CN4_GND	GND
CN4_3V3	3.3V

CN4_SS(10)	PE4
CN4_MO(11)	PE6
CN4_MI(12)	PE7
CN4_CK(13)	PE5
CN4_GND	GND
CN4.3V3	3.3V

LED1	PA0
LED2	PA1

SER6_TX	P23
SER6_RX	P25
ESP_RES	P17
ESP_IO0	P27
ESP_IO15	P31
ESP_EN	P24

CN1_GND	GND
CN1_VM	VM



CN12C1_A5	PD7
CN12C2_A4	PD6
CN12C3_A3	PD5
CN12C4_A2	PD4
CN12C5_A1	PD3
CN12C6_A0	PD2
CN13_GND	GND
CN13_5V	5V
CN13_DA(24)	P05

CN11_D-	PC6
CN11_D+	PC7
CN11_VM	VM
CN11_GND	GND
DIR	PC5

CN2_5V	5V
CN2_VSYSA	VSYS

CN7_TX(0)	P21
CN7_RX(1)	P20
CN7_VM	VM
CN7_GND	GND
2 wire	P22 - 0

CN8_TX(2)	P12
CN8_RX(3)	P13
CN8_VM	VM
CN8_GND	GND
2 wire	P14-0

CN9_TX(4)	PC2
CN9_RX(5)	PC3
CN9_VM	VM
CN9_GND	GND
2 wire	PC4-0

CN10_TX(6)	P33
CN10_RX(7)	P32
CN10_VM	VM
CN10_GND	GND
2 wire	P34-0

Pin Assign Information

Available Pins

Pin Name	CPU Pin	Pin Name	CPU Pin	Pin Name	CPU Pin
SER1_TX	P20	DAC	P05	ETH_MDIO	PA3
SER1_RX	P21	A0	PD2	ETH_TXEN	PB4
SER1_SEL	P22	A1	PD3	ETH_TXD0	PB5
SER2_TX	P13	A2	PD4	ETH_TXD1	PB6
SER2_RX	P12	A3	PD5	ETH_RXD0	PB1
SER2_SEL	P14	A4	PD6	ETH_RXD1	PB0
SER3_TX	PC3	A5	PD7	ETH_RXER	PB3
SER3_RX	PC2	ACC_SCL	P52	ETH_CRS	PB7
SER3_SEL	PC4	ACC_SDA	P50	ETH_CLK	PB2
SER4_TX	P32	ACC_INT	P07	LED1	PA0
SER4_RX	P33	ESP_RES	P17	LED2	PA1
SER4_SEL	P34	ESP_IO0	P27	A_WIRE_SCL	PE2
SPI_SS	PE4	ESP_IO15	P31	A_WIRE_SDA	PE1
SPI_MO	PE6	ESP_EN	P24	A_ESP_CK	PC5
SPI_MI	PE7	SER6_RX	P25	A_ESP_MI	PC7
SPI_CK	PE5	SER6_TX	P23	A_ESP_MO	PC6
SER5_RX	P30	SER7_TX	PC7	A_ESP_SS	PC4
SER5_TX	P26	SER7_RX	PC6	A_ESP_IO0	P15
WIRE_CL	P52	SER7_DIR	PC5		
WIRE_DA	P50	ETH_MDC	PA4		

Pin interrupt

Pin Name	CPU Pin	INT
SER1_TX	P20	IRQ8
SER1_RX	P21	IRQ9
SER2_TX	P13	IRQ3
SER2_RX	P12	IRQ2
SPI_MO	PE6	IRQ6
SPI_MI	PE7	IRQ7
SPI_CK	PE5	IRQ5
SER5_RX	P30	IRQ0
DAC	P05	IRQ13
A0	PD7	IRQ7
A1	PD6	IRQ6
A2	PD5	IRQ5
A3	PD4	IRQ4
A4	PD3	IRQ3
A5	PD2	IRQ2
SER7_TX	PC7	IRQ14
SER7_RX	PC6	IRQ13

PWM pin

Pin Name	CPU Pin	PWM
SER1_TX	P20	MTIOC1A
SER4_TX	P32	MTIOC0C
SPI_CK	PE5	MTIOC4C
SER7_TX	PC7	MTIOC3A
SER7_RX	PC6	MTIOC3C
LED1	PA0	MTIOC4A

Module

Pyboard pyb modules	GR Rose pyb(rxb) modules	Description
Accel	Not implemented	Accelerometer
ADC	ADC	A/D Conversion
CAN	Not implemented	CAN (controller area network communication bus)
DAC	DAC	D/A Conversion
ExtInt	ExtInt	I/O pin interrupt
I2C	I2C	I2C (a two-wire serial protocol)
LCD	Not implemented	LCD
LED	LED	LED
Pin	Pin	I/O pin
PinAF	Not implemented	Pin Alternative Function
RTC	RTC	Real time timer
Servo	Servo	Servo (PWM)
SPI	SPI	SPI (a master-driven serial protocol)
Switch	Switch	Switch
Timer	Timer	Timer
TimerChannel	No same functionality	Timer channel
UART	UART	Serial communication
USB_HID	Not implemented	USB Human Interface Device (HID)
USB_VCP	Not implemented	USB virtual com port

Sample Code

- File access
- Pin interrupt
- Timer (software)
- I2C-Device Scan
- I2C-SH30 Temperature Humidity Sensor
- I2C – OMRON 2SMPB-02E MEMS Absolute Pressure Sensor
- network

File Access

- Display the contents of main.py in the default /flash folder.

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

```
MicroPython v1.12-571-gfaae6b130-dirty on 2020-05-24; GR-ROSE with RX65N
Type "help()" for more information.
>>> f = open('main.py', 'r')
>>> f.read()
'# main.py -- put your code here!\r\n'
>>> f.close()
```

Reading/writing to the /flash and /sd file systems is normally done by calling the file input/output library of MicroPython.

Pin Interrupt-Switch Detect

- Connect a switch to P32 pin and GND pin
- Press Switch (P32) to execute print (“intr”).
- IRQx can be executed only on pins that can be assigned.

```
from pyb import Pin, ExtInt
callback = lambda e: print("intr")
ext = ExtInt(Pin(Pin.cpu.P32, Pin.IN, Pin.PULL_UP),
ExtInt.IRQ_RISING, Pin.PULL_UP, callback)
```

Timer-timer interrupt

- The software timer function calls print (2) every 2 seconds.

```
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
```

I2C – SH30 temperature and humidity sensor

- Displays temperature and humidity.
  - After uploading the sh30.py module to /flash (via PyMakr etc.), execute the following code.

```
from sht30 import SHT30

sensor = SHT30(scl_pin=machine.Pin.cpu.P52,
sda_pin=machine.Pin.cpu.P50,
i2c_address=0x44)

temperature, humidity = sensor.measure()

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

The machine pins can be replaced with pyb.Pin definition.
machine.Pin.cpu.P52 -> pyb.Pin(‘WIRE_CL’)
machine.Pin.cpu.P50 -> pyb.Pin(‘WIRE_DA’)

```
[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 ℃, RH: 43.53094 %
>
```

The above picture is taken using GR-MANGO

Network-HTTP access

- Use the socket module to access http://micropython.org.
- It is unstable, but https access is also possible.

```
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¥nConnectio
n: keep-alive¥r¥nVary: Accept-Encoding¥r¥nX-Frame-Options: SAMEORIGIN¥r¥n¥r¥n<!D
OCTYPE html>¥n¥n¥n¥n<html lang="en">¥n  <head>¥n    <meta charset="utf-8">¥n
<meta http-equiv="X-UA-Compatible" content="IE=edge">¥n    <meta name="viewport"
 content="width=device-width, initial-scale=1">¥n    <!-- The above 3 meta tags
*must* come first in the head -->¥n¥n    <link rel="icon" href="/static/img/favi
con.ico'
```

Network-HTTPS access

- Use the socket module to access https://micropython.org.

```
import network
net=network.LAN()
net.active(True)
net.ifconfig("dhcp")
net.ifconfig()
import socket
import ussl
s = socket.socket()
addr = socket.getaddrinfo("micropython.org", 443)[0][-1]
s.connect(addr)
ss = ussl.wrap_socket(s)
ss.write(b"GET / HTTP/1.0¥r¥n¥r¥n")
data=ss.read(4096)
s.close()
print(data)
```

```
>>> import network
>>> net=network.LAN()
>>> net.active(True)
>>> net.ifconfig("dhcp")
>>> net.ifconfig()
('192.168.0.73', '255.255.255.0', '192.168.0.1', '192.168.0.1')
>>> import socket
>>> import ussl
>>> s = socket.socket()
>>> addr = socket.getaddrinfo("micropython.org", 443)[0][-1]
>>> s.connect(addr)
>>> ss = ussl.wrap_socket(s)
>>> ss.write(b"GET / HTTP/1.0\r\n\r\n")
18
>>> data=ss.read(4096)
>>> s.close()
>>> print(data)
b'HTTP/1.1 200 OK\r\nServer: nginx/1.10.3\r\nDate: Sun, 24 May 2020 04:09:00 GMT
\r\nContent-Type: text/html\r\nContent-Length: 11482\r\nLast-Modified: Fri, 20 M
ay 2016 09:54:37 GMT\r\nConnection: close\r\nVary: Accept-Encoding\r\nETag: "573
ededd-2cda"\r\nStrict-Transport-Security: max-age=15768000\r\nAccept-Ranges: byt
es\r\n\r\n<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"\n"http://www.
w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">\n\n<html>\n\n<head>\n<title>Damien P. G
eorge</title>\n<link rel="shortcut icon" type="image/x-icon" href="/favicon.ico"
```

Network-HTTP access via ESP8266

- Use the socket module to access http://micropython.org.

```
import network
esp = network.ESP8266()
esp.connect("xxxxxx", "xxxxxxxxxx")
esp.ifconfig()

import wsocket as socket
s = socket.socket()
addr = socket.getaddrinfo('www.micropython.org',
80)[0][-1]
s.connect(addr)
s.send(b"GET / HTTP/1.0¥r¥n¥r¥n")
data=s.recv(8192)
s.close()
print(data)
```

```
>>> import network
>>> esp = network.ESP8266()
AT ver=1.6.2.0(Apr 13 2018 11:10:59)
SDK ver=2.2.1(6ab97e9)
>>> esp.connect('[redacted]", '[redacted]')
>>> esp.ifconfig()
('192.168.0.75', '192.168.0.1', '255.255.255.0')
>>> import wsocket as socket
>>> s = socket.socket()
>>> addr = socket.getaddrinfo('www.micropython.org', 80)[0][-1]
>>> s.connect(addr)
>>> s.send(b"GET / HTTP/1.0\r\n\r\n")
18
>>> data=s.recv(8192)
>>> s.close()
>>> print(data)
b'HTTP/1.1 200 OK\r\nServer: nginx/1.10.3\r\nDate: Sun, 24 May 2020 04:00:12 GMT
\r\nContent-Type: text/html\r\nContent-Length: 54\r\nLast-Modified: Sat, 04 Oct
2014 21:54:13 GMT\r\nConnection: close\r\nVary: Accept-Encoding\r\nETag: "54306c
85-36"\r\nAccept-Ranges: bytes\r\n\r\nServer down for maintenance.\n\nPlease che
ck back soon.\n'
```

GC Memory Information

- Display GC Memory information.

```
import micropython
micropython.mem_info()
```

```
MicroPython v1.12-571-gfaae6b130-dirty on 2020-05-24; GR-ROSE with RX65N
Type "help()" for more information.
>>> import micropython
>>> micropython.mem_info()
stack: 960 out of 23552
GC: total: 250368, used: 1552, free: 248816
 No. of 1-blocks: 21, 2-blocks: 8, max blk sz: 40, max free sz: 15541
```

.

Notice

- See the book 「GR-ROSE」ではじめる電子工作 (Japanese)
  - http://www.kohgakusha.co.jp/books/detail/978-4-7775-2084-8
- See the MicroPython documentation for usage details.
  - https://docs.micropython.org/en/latest/
- The ported source code will be placed on the rx branch of Github below.
  - https://github.com/ksekimoto/micropython
  - git clone https://github.com/ksekimoto/micropython -b rx
  - The pre-built binary files will be stored under the rx_release folder.

See the book「GR-ROSE」ではじめる電子工作 (Japanese)

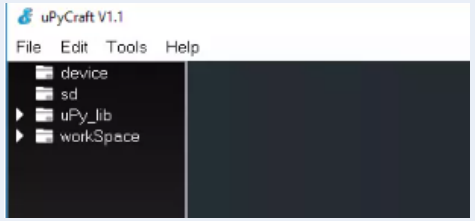
http://www.kohgakusha.co.jp/books/detail/978-4-7775-2084-8

Other restrictions

- The build method will be described in the readme.md file on Github.
  - Build definition file for USB drive copy to Boards ¥ GR_ROSE_DD
  - Boards ¥ GR_ROSE folder is build definition file for E1 debugging
- If the internal flash drive becomes inconsistent, connect a user-prepared switch to P32 and GND and press the switch for 3 seconds at startup to recreate boot.py and main.py.
- There is a bug in the processing of serial communication. Problems can occur with communication beyond the 4K byte receive buffer.
- The timer has a significantly different implementation of the MicroPython module from STM32. Please check the source file for details.
- The function of each class parameter may be largely omitted from the original.
- The print method of each class may have largely omitted the function from the original.

Backup Slide

Program editing, execution environment

Tool	Install	How to use
Visual Studio Code + PyMakr (Windows / Linux /Mac(??))	Install by typing PyMakr in the extension menu. Nodejs 6.9.5 or later installed. For the setting method, set the COM port to be used in the Pymakr> Global setting menu.	You can execute, upload, and download Python programs from the menu
uPyCraft v.1.1 (Windows) 	Download the executable file from the link below and execute it. https://randomnerdtutorials.com/uPyCraftWindows	You can execute, upload, and download Python programs from the menu https://randomnerdtutorials.com/install-upycraft-ide-windows-pc-instructions/
MU Editor (Windows/Linux) 	Download the source from Github and change it to recognize GR_ROSE USB (https://github.com/ksekimoto/mu/tree/pyboard)	Used as an editor for Microbit https://codewith.mu/

OMRON 2JCIE-EV sensor board

Sensor	Part Number	Model	Maker	Interface	
Temperature / Humidity	U1	SHT30-DIS-B	Sensirion	I2C (0x44)	
Ambient light sensor	U2	OPT3001DNP	Texas Instruments	I2C (0x45)	
MEMS Pressure	U3	2SMPB-02E	OMRON	I2C (0x56)	
MEMS Motion	U5	LIS2DW12	STMicroelectronics	SPI (SPI0 - CS:P84)	
MEMS Mic	U6	SPH0645LM4H-B	Knowles	I2S	