Get started with MicroPython [W600 series]

Flash MicroPython firmware

The boards were already flashed micropython firmware. If they lost firmware or you need lastest firmware, you can flash MicroPython firmware by yourself.

1. Requirements

- CH340 Driver
- Python 3.7 or newer
- w600tool (for flash w600 firmware)
- `pip install w600tool`
- Micropython firmware v1.10

2. Flash firmware

```
w600tool.py -p PORT_NAME -u FIRMWARE.fls
```

Note

Don’t forget to change **PORT_NAME** and **FIRMWARE.fls**.

In Linux, **PORT_NAME** is like /dev/ttyUSB0. In windows, **PORT_NAME** is like COM4.

General board control

The machine module:

Turn on & off LED

```
from machine import Pin


led = Pin(Pin.PA_00, Pin.OUT, Pin.PULL_FLOATING)
led.value(1)
led.value(0)
```

Networking

The network module:

```
import network
```

```
sta_if = network.WLAN(network.STA_IF)
sta_if.active(True)
sta_if.scan()
sta_if.connect("WM2G","87654321")
sta_if.isconnected()
```

The socket module:

```
import socket

s = socket.socket()
addr = ('www.qq.com', 80)
s.connect(addr)
s.send("hello world!")
s.recv(64)
s.close()
```

I2C

Read SHT30:

```
from machine import Pin, I2C
import time

i2c = I2C(0, scl=Pin(Pin.PB_13), sda=Pin(Pin.PB_14), freq=100000)

buf = bytearray(2)
buf[0] = 0x30
buf[1] = 0xA2
i2c.writeto(0x44, buf)
time.sleep_ms(1000)

buf2 = bytearray(6)
buf[0] = 0x2c
buf[1] = 0x06
i2c.writeto(0x44, buf)
buf2 = i2c.readfrom(0x44, 6)

temp_raw = (buf2[0] << 8) + (buf2[1])
humi_raw = (buf2[3] << 8) + (buf2[4])
temp = 175 * temp_raw / 65535 - 45
humi = 100 * humi_raw / 65535
print("temp = {:.2f}, humi = {:.2f}".format(temp, humi))
```

Note

MicroPython supports both hardware I2C and software emulation I2C.

Using **Software I2C** when I2C device ID = -1.

Using **hardware I2C** when I2C device ID > -1.

Real time clock (RTC)

```
from machine import RTC

rtc = RTC()
rtc.init((2019, 9, 12, 3, 13, 0, 0, 0))
print(rtc.now())
```

Note

rtc.init((year, month, day[, hour[, minute[, second[, microsecond[, tzinfo]]]]]))

SPI

The w60x has 1x hardware SPI, up to 20Mhz.

Note

Using **software SPI** when the ID is -1.

Using **hardware SPI** when the ID is 0.

```
from machine import Pin, SPI

spi = SPI(0, baudrate=200000, polarity=1, phase=0, sck=Pin(Pin.PB_16), mosi=Pin(Pin.PB_18),
miso=Pin(Pin.PB_17), cs=Pin(Pin.PB_15))
spi.read(10)
spi.read(10, 0xff)

buf = bytearray(50)
spi.readinto(buf)
spi.readinto(buf, 0xff)

spi.write(b'12345')

buf2 = bytearray(4)
spi.write_readinto(b'1234', buf2)
spi.write_readinto(buf2, buf2)
```

SPI	W600x SPI supported IO
SCK	PA_01, PA_11, PB_16, PB_27
MOSI	PA_04, PA_09, PA_10, PB_02, PB_18
MISO	PA_03, PA_05, PA_10, PB_01, PB_17
CS	PA_02, PA_12, PB_00, PB_07, PB_15

PWM

```
from machine import Pin, PWM

pwm1 = PWM(Pin(Pin.PB_16), channel=2, freq=100, duty=0)
pwm1 = PWM(Pin(Pin.PB_16), channel=2, freq=100, duty=255)
pwm1.deinit()
```

```
pwm2 = PWM(Pin(Pin.PB_18))
pwm2.freq()
pwm2.freq(100)
pwm2.duty()
pwm2.duty(250)
```

PWM CH	W60x PWM supported IO
CH 0	PA_00, PA_05, PB_05, PB_18, PB_19, PB_30
CH 1	PA_01, PA_07, PB_04, PB_13, PB_17, PB_20
CH 2	PA_02, PA_08, PB_04, PB_03, PB_16, PB_21
CH 3	PA_03, PA_09, PB_02, PB_06, PB_15, PB_22
CH 4	PA_04, PA_10, PB_01, PB_08, PB_14, PB_23

Timer

W60X has 6x hardware timers (timer0 is used by WM_SDK, users only have tiemr1-tiemr5 available), Use software timer when the ID is -1, use hardware timer when ID is 1-5.

```
from machine import Timer

timer1 = Timer(-1)
timer1.init(period=5000, mode=Timer.ONE_SHOT, callback=lambda t:print(1))
timer3 = Timer(3)
timer3.init(period=2000, mode=Timer.PERIODIC, callback=lambda t:print(2))
```

UART

```
from machine import UART

uart = UART(1, 115200)
uart.init(115200, bits=8, parity=None, stop=1)

uart.write('hello world')

uart.readline()
print(uart.read(5))

buf = bytearray(6)
uart.readinto(buf)
print(buf)
```

WDT

```
from machine import WDT

wdt = WDT(0,5000000)
wdt.feed()
```

Note

WM_SDK has added the automatically feed function in lowest priority task. It can run without feeding watch dog in MicroPython.