W60X MicroPython 使用手册

V0.3

北京联盛德微电子有限责任公司（winner micro）

地址：北京市海淀区阜成路 67 号银都大厦 18 层

电话：+86-10-62161900

公司网址：www.winnermicro.com

文档修改记录

版本	修订时间	修订记录	作者	审核
V0.1	2018-11-13	创建	李利民	
V0.2	2019-09-12	增加使用实例	李利民	
V0.3	2019-11-08	增加 ADC 和 SSL 实例	李利民	

目录

1 引言

1.1 编写目的

指导如何在 W60X 上编译使用 MicroPython 项目；

1.2 预期读者

所有 W60X 相关的开发人员

1.3 术语定义

1.4 参考资料

2 MicroPython 简介

MicroPython 是 Python 3 编程语言的精简而有效的实现，其中包括 Python 标准库的一小部分，并针对微控制器和受限制的环境进行了优化。

MicroPython 包含了许多高级功能，如交互式提示，任意精度整数，闭包，列表理解，生成器，异常处理等等。但是它足够紧凑，可以在 256k 的代码空间和 16k 的 RAM 中运行和运行。

MicroPython 的目标是尽可能地与普通的 Python 兼容，使用者能够轻松地将代码从桌面传输到微控制器或嵌入式系统。

在 W60X 上运行的 MicroPython 具有的特性：

- 支持 UART0 进行 MicroPython 命令行交互
- 支持 16K 的任务栈和 40K 的堆空间用于 MicroPython 运行
- 支持了大多数的 MicroPython 特性和内部库（unicode、高精度整数、单精度浮点数、复数等）
- 支持硬件 GPIO、UART、SPI、I2C、PWM、WDT、ADC、RTC、Timer 模块
- 支持 WiFi 网络模块（包含一键配网功能）
- 支持 SSL 使用硬件加解密（仅 2M Flash 设备支持）
- 只用内部 Flash 文件系统（可用空间为 32K）
- 支持 FTP 上传脚本文件至模块

其启动运行时的界面如下图所示：

```
    __    __
    \ \  /\  / /
     \ \/  \/ /
      \  /\  /
       / /\ \
      / /\/\ \
     / /    \ \
    /_/      \_\

    WinnerMicro W600

MicroPython v1.10-284-g2eee4e2-dirty on 2019-11-08; WinnerMicro module with W600
Type "help()" for more information.
>>> help()
Welcome to MicroPython on the W600!

For generic online docs please visit http://docs.micropython.org/

For access to the hardware use the 'machine' module:

import machine
pb26 = machine.Pin(machine.Pin.PB_26, machine.Pin.OUT, machine.Pin.PULL_DOWN)
pb26.value(1)
pb27 = machine.Pin(machine.Pin.PB_27, machine.Pin.IN, machine.Pin.PULL_UP)
print(pb27.value())

Basic WiFi configuration:

import network
sta_if = network.WLAN(network.STA_IF)
sta_if.active(True)
sta_if.scan()                             # Scan for available access points
sta_if.connect("<AP_name>", "<password>") # Connect to an AP
sta_if.isconnected()                      # Check for successful connection

Control commands:
  CTRL-A        -- on a blank line, enter raw REPL mode
  CTRL-B        -- on a blank line, enter normal REPL mode
  CTRL-C        -- interrupt a running program
  CTRL-D        -- on a blank line, do a soft reset of the board
  CTRL-E        -- on a blank line, enter paste mode

For further help on a specific object, type help(obj)
For a list of available modules, type help('modules')
>>>
```

MicroPython 所支持的模块用法可以在 docs 目录下查看其使用方法，

此外官方还提供了很多的支持模块，官网的下载地址为：

https://github.com/micropython/micropython-lib

用户可以下载下来拷入模块，然后在脚本中导入使用即可。

3 快速上手 MicroPython

目前 W60X 在 MicroPython 1.10 版本上移植成功,既提供了源码包供有需要的用户重新编译，也提供了编译好的固件直接下载烧录使用。

3.1 直接下载固件使用

为了方便用户使用，省去编译固件的操作，我们在官网 http://www.winnermicro.com 提供了编译好的固件，只需要下载固件烧写到模块即可启动使用，如何烧写固件请参考官网提供的固件烧写文档。

3.2 重新编译 MicroPython

本操作基于 GCC 编译,在 Linux 系统下可以直接在 shell 中操作;在 Windows 系统下需要先安装 Cygwin，推荐下载我们官网提供的 W60X_IDE 集成包，里面带有 Cygwin 环境可直接使用。

3.2.1 下载交叉编译工具

W60X 使用的交叉编译工具中的 gcc 为 arm-none-eabi-gcc，下载地址为：
https://launchpad.net/gcc-arm-embedded/4.9/4.9-2014-q4-major。

解压之后，需要将交叉编译工具的路径加入到环境变量中，如放在/opt 目录下时：

```
export PATH=$PATH:/opt/tools/arm-none-eabi-gcc/bin
```

可以将此配置写入“.bashrc”文件中永久生效，避免每次都需要设置一遍的麻烦。

注意：因 WM_SDK 只支持 4.x 版本的 GCC 编译器进行编译，所以如果选择了其他版本的编译器，需要用户自行解决编译错误。

3.2.2 下载 WM_SDK 开发包

可以在 http://www.winnermicro.com 下载 SDK 包。
下载时需注意 WM_SDK 版本从 G3.01.00 才开始支持 MicroPython 编译。

解压之后需要设置环境变量“*WMSDK_PATH*”指明 WM_SDK 的路径，如：

```
export WMSDK_PATH=/home/w60x/WM_SDK
```

可以将此配置写入“.bashrc”文件中永久生效，避免每次都需要设置一遍的麻烦。

WM_SDK 包中存在一些在 MicroPython 项目中用不到的组件,可以设置在编译前裁剪掉以减少代码大小。具体需要打开 WM_SDK/Include/wm_config.h 文件进行修改，将需要裁剪掉的组件的宏开关“CFG_ON”改为“CFG_OFF”即可。
目前推荐关闭的组件有：

```
#define TLS_CONFIG_HOSTIF      CFG_OFF
#define TLS_CONFIG_RMMS        CFG_OFF
#define TLS_CONFIG_HTTP_CLIENT CFG_OFF
#define TLS_CONFIG_NTP         CFG_OFF
```

如果在编译时提示空间超过 ROM 限制，请务必执行此裁剪操作。

3.2.3 下载 MicroPython

请从官网 http://www.winnermicro.com 下载源码包，并解压。

3.2.4 编译

在 shell 中进入 MicroPython 工程的 ports/w60x 文件夹，其 Makefile 文件中可以修改 "MICROPY_USE_2M_FLASH"来指定是否编译 2M Flash 版本，默认编译 1M Flash 版本。

启动编译：

```
make V=s
```

编译完成之后，生成的固件位于 ports/w60x/build 目录下。

注意：只有具有 2M Flash 的设备才能使用 2M Flash 版本的固件，错刷固件设备将不能正常使用。

3.3 烧录 MicroPython

如果是第一次给 W60X 烧录 MicroPython 固件，可编译*.fls 文件进行烧录，命令如下：

```
make flash V=s
```

之后烧录固件，可编译*_gz.img 文件进行烧写，其烧写速度更快，命令如下：

```
make image V=s
```

烧写时需要配置模块串口号，可按照 shell 中的提示或者打开 w60x/tools/download_*.sh，修改“SERIAL_NAME”变量为实际使用的串口号，再执行烧录命令。

4 命令行使用实例

MicroPython 提供了一个叫做 REPL 的交互式命令行，可以敲入各种指令进行操作。

4.1 基本打印

```
print('hello world')
print(b'bytes 1234\x01')
print(123456789)
for i in range(4):
    print(i)
```

4.2 使用 WiFi

```
import network

sta_if = network.WLAN(network.STA_IF)
```

```
sta_if.active(True)
sta_if.scan()
sta_if.connect("WM2G","87654321")
sta_if.isconnected()
```

4.3 使用 OneShot

OneShot（一键配网）需要手机 APP 支持，可在官网或者应用商店自行安装。
也可使用微信 AirKiss 配网，可关注公众号或安装 AirKiss 测试 APP。

```
import network

sta_if = network.WLAN(network.STA_IF)
sta_if.active(True)
sta_if.oneshot(1)

# 在 OneShot APP 端启动配网

sta_if.isconnected()
```

4.4 使用 Socket

```
import socket

s = socket.socket()
addr = ('www.qq.com', 80)
s.connect(addr)

s.send("hello world!")
s.recv(64)

s.close()
```

注意：必须 WiFi 联网之后才能使用 socket 进行通信。

4.5 使用 SSL

```
import socket
import ussl

s = socket.socket()
addr = ('www.baidu.com', 443)
s.connect(addr)

sec = ussl.wrap_socket(s)
sec.write("GET / HTTP/1.1\r\n\r\n")
sec.read(1024)

sec.close()
s.close()
```

注意：必须 WiFi 联网之后才能使用 SSL 进行通信。

4.6 使用 Pin

```
from machine import Pin

led = Pin(Pin.PB_16, Pin.OUT, Pin.PULL_FLOATING)
led.value(1)
led.value(0)
```

4.7 使用 I2C

MicroPython 既支持硬件 I2C，也支持软件模拟 I2C，当 MicroPython 的 I2C 设备 ID 为-1 的时候为使用

软件 I2C 功能，非-1 则使用硬件 I2C 功能。

W60X 本身拥有硬件 I2C，这里以使用 SHT30 温湿度传感器举例：

```
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))
```

注意：W60X 可以用作硬件 I2C 的 IO 如下表所列：

I2C 功能	对应 IO
scl	PA_06，PA_08，PB_11，PB_13，PB_21
sda	PA_07，PA_15，PB_12，PB_14，PB_22

4.8 使用 RTC

```
from machine import RTC

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

注意：MicroPython 的星期取值范围为 0-6，0 表示星期一，其余依次类推。

4.9 使用 SPI

W60X 拥有硬件 SPI 功能，按时钟可分为高速 SPI(最高 50MHz 时钟)和低速 SPI(最高 20MHz 时钟)。使用高速 SPI 时，W60X 只能作为从设备；使用低速 SPI 时，W60X 可以作为主设备。

在 MicroPython 中，ID 为-1 时使用软件 SPI 功能，ID 为 0 时使用硬件 SPI 功能（暂不支持 W60X 作为从设备的高速 SPI 模式），下面使用硬件低速 SPI 举例：

```
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)
```

注意：W60X 可以用作硬件 SPI 的 IO 如下表所列：

SPI 功能	对应 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

4.10 使用 PWM

W60X 拥有 5 路硬件 PWM 功能分别用 0-4 表示，其频率范围为 1-156250，占空比范围为 0-255。

```
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)
```

注意：每路 PWM 只有部分 IO 可用，其对应关系为：

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

4.11 使用 Timer

W60X 拥有 6 组硬件定时器（Timer0 已被 WM_SDK 使用，用户只有 Tiemr1-Tiemr5 可供使用），当 ID 为-1 时使用软件定时器，当 ID 为 1-5 时使用硬件定时器。

```
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))
```

4.12 使用 ADC

W60X 拥有 12 路硬件 ADC 功能，分别用通道号 0-11 表示。

```
from machine import ADC

adc = ADC(0)
vcc = adc.read()
print("vcc = {:.3f}".format((vcc - 8192.0) / 8192 * 2.25 / 1.2 + 1.584))
```

注意：ADC 通道号和 IO 对应关系为：

ADC 通道号	IO
0	PB_19
1	PB_20
2	PB_21
3	PB_22
5	PB_23
6	PB_24
7	PB_25
8	PB_19 和 PB_20
9	PB_21 和 PB_22
10	PB_23 和 PB_24

11	PB_25 和 PB_26

4.13 使用 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)
```

注意：当所读字节数小于实际接收数时不会读取到任何数据。这是因为 WM_SDK 有这个限制，需要移除该限制的用户请自行修改 WM_SDK 中的 Platform\Drivers\uart\wm_uart.c 文件，按下图所示修改：

```
int tls_uart_read(u16 uart_no, u8 * buf, u16 readsize)
{
    ...

    recv = &port->recv;
    data_cnt = CIRC_CNT(recv->head, recv->tail, TLS_UART_RX_BUF_SIZE);
    if (data_cnt >= readsize)
    {
        buflen = readsize;
    }
    else
    {
        buflen = data_cnt;
    }

    ...
}
```

4.14 使用 WDT

```
from machine import WDT

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

注意:因为 WM_SDK 已自动在底层最低优先级任务中加入了喂狗功能,所以启用 WDT 之后,在 MicroPython 脚本中不用喂狗也可正常运行。

5 脚本文件使用指导

5.1 将脚本文件转为为字节码编入固件

MicroPython 提供了直接将脚本编入固件的功能，W60X 固件在上电之后，默认会执行 ports/w60x/scripts 目录下的_boot.py 脚本,用户可以将 Python 脚本代码写入该文件达到上电自动运行的目的。

任何放入 ports/w60x/scripts 目录下的脚本文件在编译时都会被编入固件，执行时需要在代码中调用“pyexec_frozen_module”来执行指定的脚本文件，如：

```
pyexec_frozen_module("_boot.py");
```

也可以在_boot.py 中再次调用别的脚本文件，两种方法都是可以的。

easyw600.py 是按此方式编入固件的脚本，其集成了部分常用功能，用户可以参考使用,该模块提供如下方法：

```
import easyw600
easyw600.scan() 方法扫描周边 WiFi 网络
easyw600.oneshot() 启动一键配网功能，直至联网后打印出 IP 地址
easyw600.connect(ssid="myssid", password=None) 方法启动模块连接 WiFi，联网后打印 IP
```

```
easyw600.disconnect() 方法断开网络连接
easyw600.createap(ssid="w60x_softap", password=None) 方法创建一个软 ap
easyw600.closeap() 方法关闭软 ap
easyw600.ftpserver() 方法启动内置的 FTP 服务器，端口号为 21，用户名 root，密码 root
```

注意：将脚本编入固件的好处是不占用 fs 文件系统空间，但是会增大固件镜像大小，请用户根据实际情况选择使用。

5.2 上传脚本文件到模块 Flash 中

5.2.1 脚本上传方法

W60X 内部 flash 提供了资源有限的文件系统，可供用户存储脚本使用。
为了方便使用，我们在 MicroPython 中集成了 FTP 服务器功能，配置模块联网之后，就可以在 PC 端使用 FTP 客户端将脚本文件拷贝到模块中。

模块联网之后，通过如下操作可以启动 FTP 服务器：

```
import w600
w600.run_ftpserver(port=21,username=None,password=None)
```

这些参数都具有默认值：端口不配时默认使用 21；用户和密码为带双引号的字符串格式，不配则使用匿名登陆。
匿名登陆之后只能查看和下载文件，不能上传、修改、删除文件，如果提示无权限操作，请设置用户名和密码后再尝试。

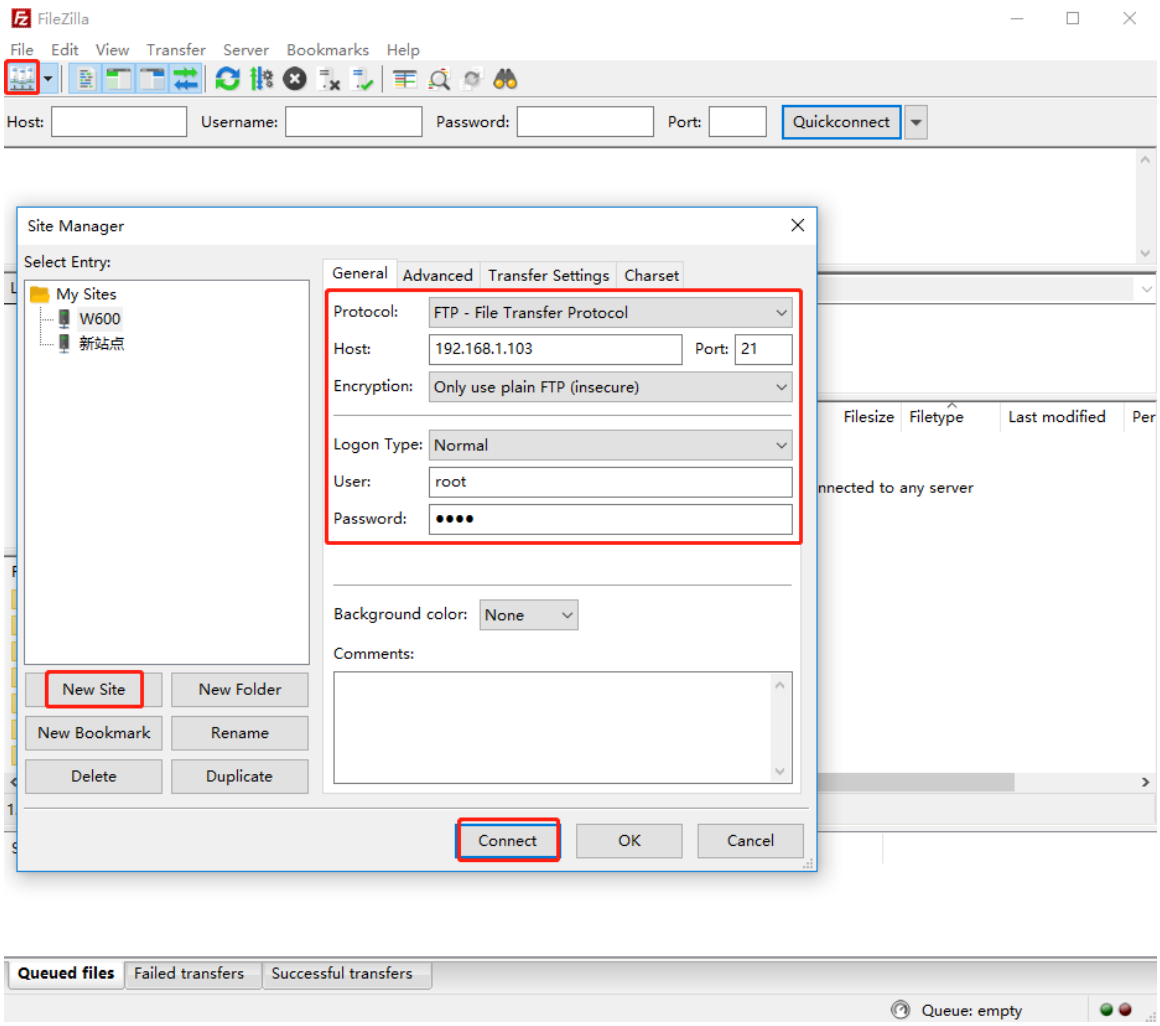
模块启动 FTP 服务器时会有如下提示：

```
>>> import w600
>>> w600.run_ftpserver(port=21,username="root",password="123456")
ftpserver is running.
>>>
```

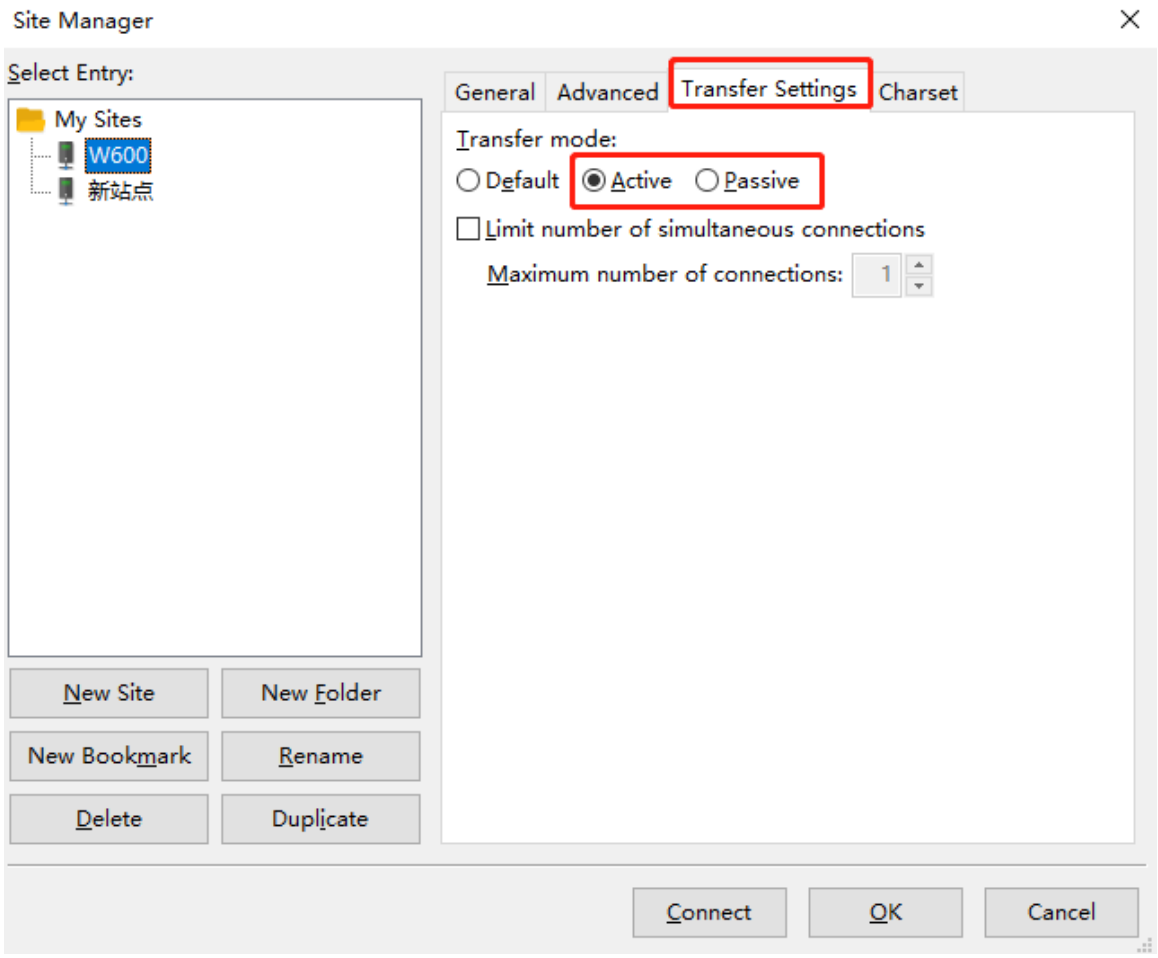
嵌入式 FTP 服务器所支持的功能比较有限，FTP 客户端有很多种类，可能存在各种兼容性差异，如果在使用时发现无法操作，可以尝试更改 FTP 连接的主动/被动模式来解决。

Linux 系统下使用时要注意可能需要关闭防火墙（iptables），否则有些主机不能连接 FTP。

下面以 FileZilla 为例简单介绍：



如果需要设置主动/被动模式，则如下操作即可：



5.2.2 Flash 文件系统结构说明

W60X 默认的文件系统中会有如下文件：

这些文件是系统默认创建的，用户可以直接修改他们直接使用。

其中，模块上电之后会先自动执行 boot.py 脚本，之后会执行 main.py 脚本，所以如果想上电自动运行某些代码，可以将代码写入这两个脚本即可。一般 boot.py 脚本中放入一些初始化的代码，main.py 脚本中放置功能代码，当然也可以再增加新的脚本文件，用户可以根据自己实际情况操作即可。

6 版本说明

目前 W60X 发布的 MicroPython 版本为 W60X_MicroPython_1.10_B1.5。

根据设备内置 Flash 大小分为 1M 版本和 2M 版本，只有 2M Flash 版本才支持 SSL 功能。

使用内部 Flash 文件系统，会用 FatFS 或 LittleFS 格式化分区，默认使用 LittleFS。
使用 FatFS 时 W60X 需使用部分 OTA 区域作为文件系统，所以*_gz.img 在 1M Flash 版本中最大不能超过 352KB，在 2M Flash 版本中最大不能超过 736KB。