

W60X MicroPython 使用手册

V0.1

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	LiLm	

目录

文档修改记录.....	1
目录.....	2
1 引言.....	3
1.1 编写目的.....	3
1.2 预期读者.....	3
1.3 术语定义.....	3
1.4 参考资料.....	3
2 MicroPython 项目简介.....	4
3 快速构建 MicroPython.....	5
3.1 编译 MicroPython.....	5
3.1.1 下载交叉编译工具.....	5
3.1.2 下载 WM_SDK 开发包.....	6
3.1.3 下载 MicroPython.....	7
3.1.4 编译.....	7
3.2 烧录 MicroPython.....	7
4 命令行使用实例.....	7
4.1 基本打印.....	8
4.2 连接 WiFi.....	8
4.3 使用 socket.....	8
4.4 点亮 LED 灯.....	8
5 脚本文件使用指导.....	9
5.1 将脚本文件转为为字节码编入固件.....	9
5.2 上传脚本文件到模块 flash 中.....	9
5.2.1 脚本上传方法.....	9
5.2.2 flash 文件系统结构说明.....	12
6 版本说明.....	13

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 项目简介

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 45K 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、I2C、PWM、WDT、TIMER、RTC、PIN 和软件 SPI 模块
- 支持 WiFi 网络模块
- 只用内部 Flash 文件系统（可用空间为 27K）
- 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：

```
>>> hello w600 micropython
MicroPython v1.9.4 on 2018-12-19; 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
pin26 = machine.Pin(26, machine.Pin.OUT)
pin26.value(1)
pin27 = machine.Pin(27, machine.Pin.IN, machine.Pin.PULL_UP)
print(pin27.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 快速构建 MicroPython

目前 W60X 在 MicroPython 1.9.4 版本上移植成功，其基于 Linux 环境下的 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。

3.1 编译 MicroPython

3.1.1 下载交叉编译工具

W60X 使用的交叉编译工具中的 gdb 为 arm-none-eabi-gdb，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。

3.1.2 下载 WM_SDK 开发包

可以在 <http://www.winnermicro.com> 下载 SDK 包。

下载时需注意 SDK 版本从 G3.1 开始才支持 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=$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_HS_SPI      CFG_OFF
#define TLS_CONFIG_HOSTIF     CFG_OFF
#define TLS_CONFIG_RMMS       CFG_OFF
#define TLS_CONFIG_SOCKET_RAW 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。

3.1.3 下载 MicroPython

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

3.1.4 编译

在 shell 命令行中进入 MicroPython 工程的 ports/W60X 文件夹，然后执行编译命令：

```
Make V=s
```

等待编译完成之后，生成固件位于 ports/W60X/build 目录下。

3.2 烧录 MicroPython

在 MicroPython 工程的 ports/W60X/tools 目录下中附带一个 python 烧录脚本 “download.py”，在 shell 命令行中敲入 “python ./download.py --help” 查看其使用方法：

```
$ python ./download.py --help
USAGE:
win:python download.py [COM] [image]
    COM default: "COM1" image default: "../Bin/wm_w600_gz.img"
    eg: python download.py COM5 ../Bin/wm_w600_gz.img
Linux:python3 download.py [COM] [image]
    COM default: "ttyUSB0" image default: "../Bin/wm_w600_gz.img"
    eg: python3 download.py ttyUSB0 ../Bin/wm_w600_gz.img
```

使用者根据自己模块实际使用的端口进行设置，固件请选择 W60X_GZ.img 进行烧写。

注意：因为 W60X 只用了 Flash 部分空间作为文件系统，所以 W60X_GZ.img 最大不能超过 352KB，如果超过 352Kb 请裁剪掉部分代码以保证系统正常运行。

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 连接 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 socket

```
import socket
s=socket.socket()
addr=('192.168.18.92', 1234)
s.connect(addr)
s.send("hello world!")
s.close()
```

4.4 点亮 LED 灯

```
import machine
led=machine.Pin(26, machine.Pin.OUT, machine.Pin.PULL_FLOATING)
```

```
led.value(1)
```

```
led.value(0)
```

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/modules` 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/modules` 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。

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 W60X  
W60X.run_ftpsrvr(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。

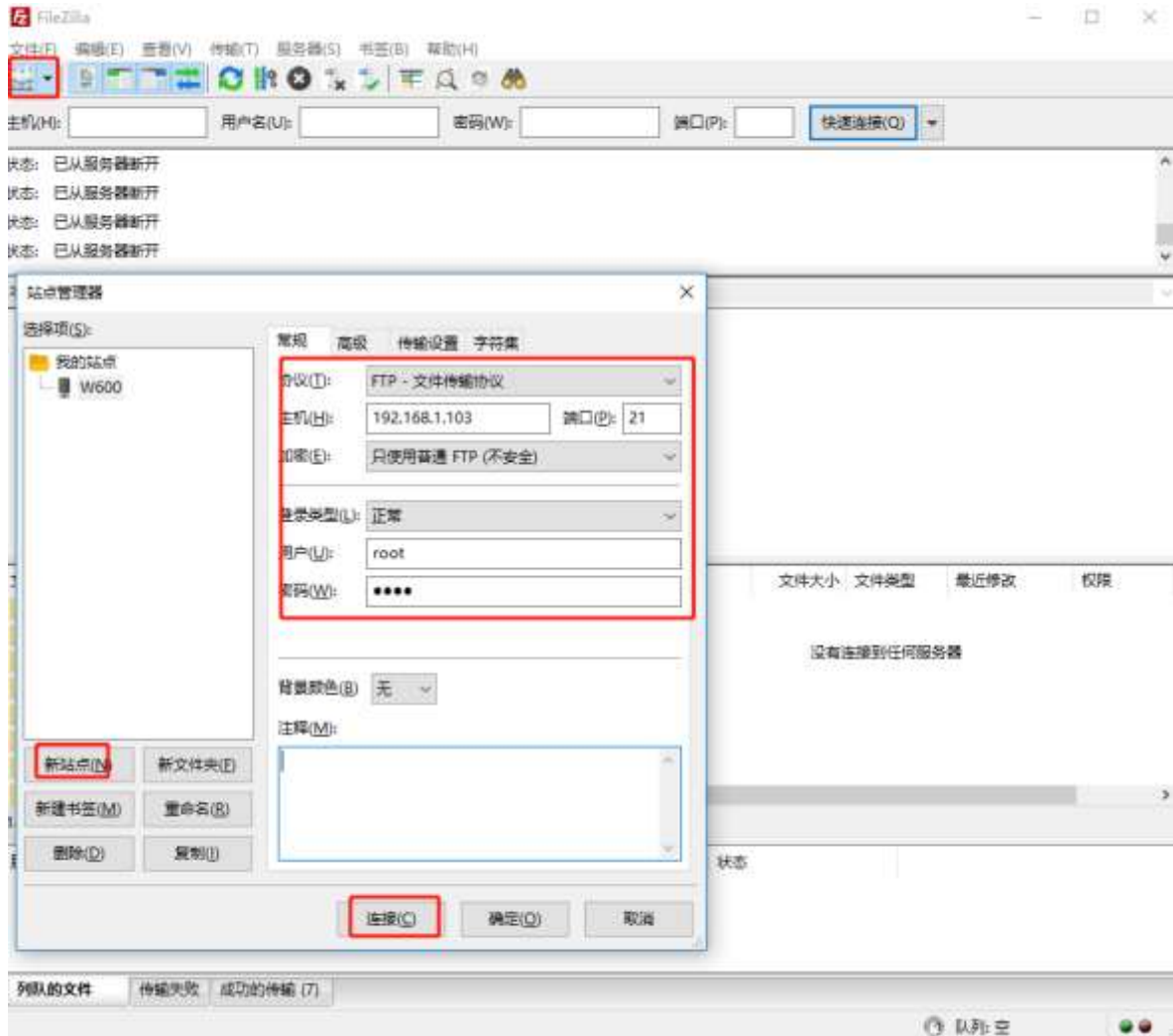
模块启动 FTP 服务器时会有如下提示：

```
>>> import w600  
>>> w600.run_ftpsrvr(port=21, username="root", password="123456")  
ftpsrvr 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：

文件名	文件大小	文件类型
..		
cert		文件夹
lib		文件夹
sys		文件夹
boot.py	86	Python File
easyw600.py	1,720	Python File
main.py	67	Python File

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。

easyW60X.py 脚本是 W60X 模块内置的模块，其具有常使用的部分功能，用户可以参考使用，该模块提供如下方法：

```
easyW60X.scan() 方法扫描周边 WiFi 网络  
easyW60X.oneshot() 启动一键配网功能，直至联网后打印出 IP 地址  
easyW60X.connect(ssid=" myssid" , password=None) 方法启动模块连接 WiFi，联网后打印 IP  
easyW60X.disconnect() 方法断开网络连接  
easyW60X.createap(ssid= "W60X_softap" , password=None) 方法创建一个软 ap  
easyW60X.closeap() 方法关闭软 ap  
easyW60X.ftpserver() 方法启动内置的 FTP 服务器，端口号为 21，用户名 root，密码 root
```



































cert、lib、sys 文件夹为扩充模块而留，当下载到新的模块之后，可以放置对应到这些目录下，之后就可以在脚本中导入直接使用。

6 版本说明

目前 W60X 发布的 MicroPython 版本为 W60X_MicroPython_1.9.4_B1，已经移植的模块如下图所示：

名称

^

-  _thread.rst
-  array.rst
-  builtins.rst
-  cmath.rst
-  framebuf.rst
-  gc.rst
-  machine.I2C.rst
-  machine.Pin.rst
-  machine.rst
-  machine.RTC.rst
-  machine.Signal.rst
-  machine.SPI.rst
-  machine.Timer.rst
-  machine.UART.rst
-  machine.WDT.rst
-  math.rst
-  micropython.rst
-  network.rst
-  sys.rst
-  ubinascii.rst
-  ucollections.rst
-  ctypes.rst
-  uerrno.rst
-  uhashlib.rst
-  uheapq.rst
-  uio.rst
-  ujson.rst
-  uos.rst
-  ure.rst
-  uselect.rst
-  usocket.rst
-  ustruct.rst
-  utime.rst
-  uzlib.rst

暂不支持 ssl 模块，待后续更新版本解决，请期待。