# MCUSH开发测试环境说明

说明：此文档描述如何搭建MCUSH开发测试环境，包括固件开发、串口调试和自动化测试工具的开发。

作者：彭树林

更新日期：2019-9-25

## 1、固件开发环境

### 操作系统

MCUSH最早是在Ubuntu Linux环境下开发的，Debian/Ubuntu下已得到最全面的测试验证。后续用到的编译测试工具大部分已经预编译打包，仅需一条apt-get命令就能快速安装。

### 编译器

目前仅支持ARM CORTEX-M3/4系列的单片机，用的是交叉编译器arm-none-eabi-gcc，Ubuntu的软件库已经集成，也可以自行编译。

编译链接还需要用到的Newlib C库，也已经集成在系统软件库中。

### 项目管理

与众多开源项目不同，MCUSH的项目管理没有选用传统的make，而是改用scons，原因主要有：

1、作者熟悉Python，使用Python语法的配置文件更得心应手。

2、需要搭配自行开发的scons扩展库，才能灵活配置MCUSH的定制选项。

扩展库下载地址：http://github.com/pengshulin/site\_scons.git

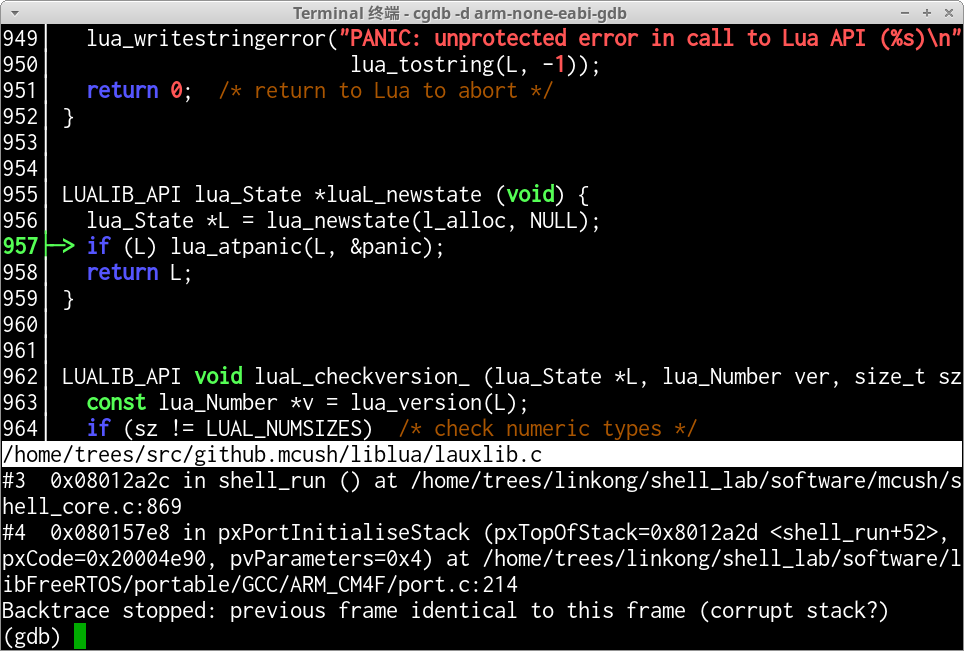
### 版本管理

首选git，并且整个项目都托管在github.com。

### 调试器

使用GCC的配套调试器GDB，交叉编译版arm-none-eabi-gdb。

其缺点是公认的交互界面不太友好，有众多的前端工具可以弥补这一点，比如作者正在使用的折中方案：cgdb，一个终端环境下用的GDB封装，支持语法着色。



GDB远程调试需要代理服务，目前使用的是OpenOCD，调试ARM CORTEX-M比较顺利。

### 集成开发环境

作者开发时避免使用大而全的IDE（如Eclipse），而是用众多小而精的工具组合。

如果你想使用自己熟悉的IDE开发，需要些额外工作，将上述工具链组合进去。

### 远程开发测试环境

由于上述开发环境都是基于命令行终端，很方便就能搭建一套远程开发测试环境：

1、在一台基于LINUX的开发机上安装全部的编译、调试、测试工具。

2、远程登录该机器，远程编辑、编译、调试和测试。

若局域网环境较好，首选这种方案，这样工程师电脑和开发测试电脑分离，相互不影响，如：

1、不会因为意外的断电重启影响当前的调试测试。

2、不会因为USB串口的插拔打乱分配到的串口编号（在LINUX下，USB串口号是动态顺序分配的）。

3、不会因为要下班了而中断还在进行的测试环境。

如能实现外网远程登录（如搭建VPN），开发测试人员的灵活性就更大了，甚至可以工程师在家里远程调试在公司的开发环境，或者出差在外远程检查一下测试机的设备日志。

远程登录容易遇到网络异常断开的情况，需要远端具有这种容错性，作者推荐使用tmux，不仅解决上述问题，还支持终端分割、窗口管理，方便调试测试。

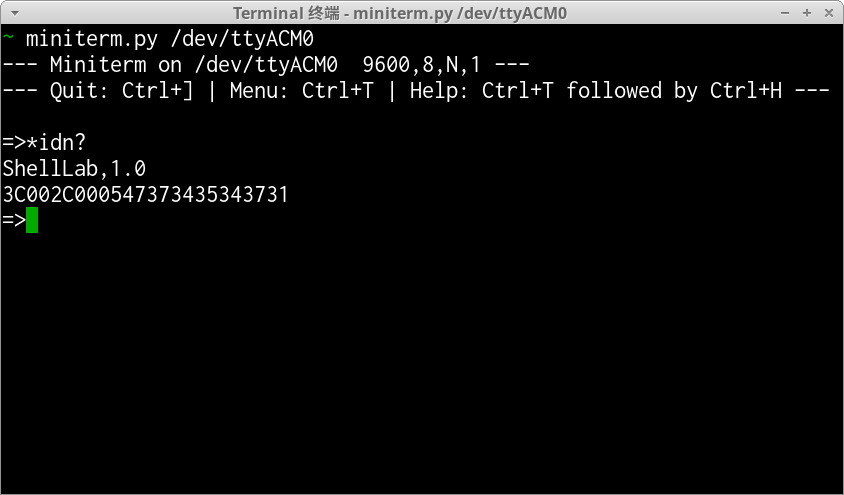
## 2、串口调试环境

### Linux

终端环境下首推python-pyserial自带的调试器miniterm.py，这是一个pyserial库的演示脚本，功能单一但已经够用了。

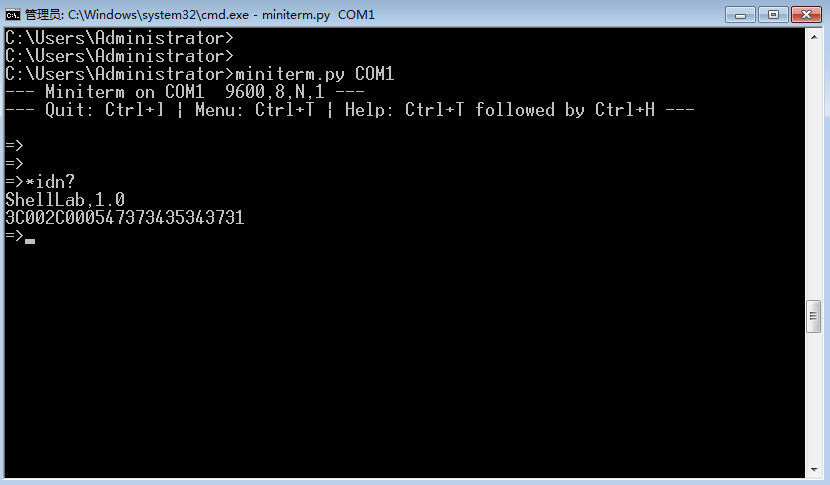
安装：sudo pip install pyserial，注意不要与serial库混淆。

安装完成后执行miniterm.py <device\_name>



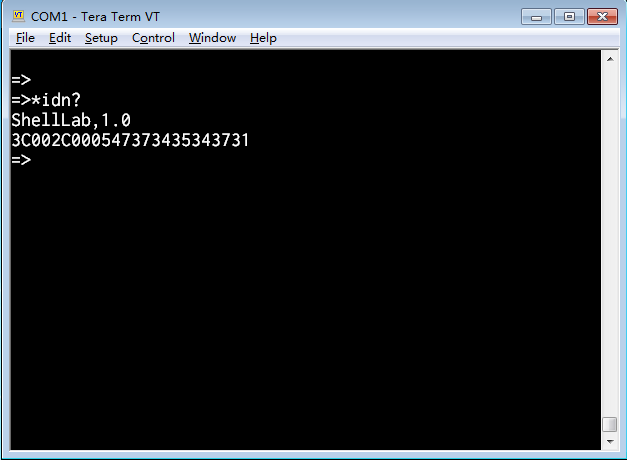
### Windows

上述的miniterm.py依然可用，但是受限于Windows自带的命令行终端，不太灵活。

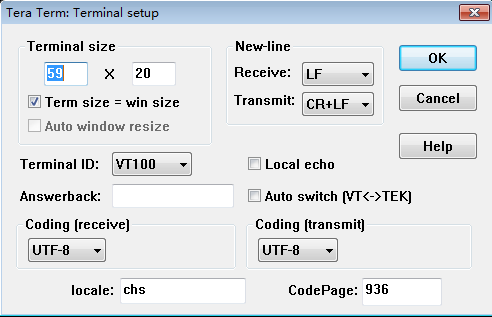


首选的替代方案是Tera Term，配置灵活、字体可调、支持行编辑的图形工具。

下载：https://ttssh2.osdn.jp/



需要注意的是：设置新行结束符LF（收）和CR+LF（发），如下图。



## 3、自动测试环境

### Python库

MCUSH提供封装好的模块库，通过调用串口，模拟手动调试指令，读取分析返回结果，并提供更多API，帮助搭建测试脚本和工具。

安装方式：sudo pip install mcush

### 图形化工具

有时需要构建图形化应用，提供更方便直观的工具，降低使用门槛。

过程大致都是先用命令行测试脚本将需求基本验证完成后，再用图形库封装一下；如果需要部署到Windows机器上运行，又有代码保密性要求，就还需要发布成可执行的EXE和并打包成安装程序。

作者常用的图形封装库是wxPython，绘制对话框的设计工具是wxGlade。

