MediaTek LinkIt Smart 7688
开发者指南（中文版）

版本：1.1

发布日期：2016 年 2 月 22 日

版本修订历史

版本	日期	简介
1.0	2015 年 12 月 1 日	初次发布
1.1	2016 年 2 月 22 日	更新第 6.6.5“PyMata 方案”中关于安装 Arduino PyMata 程序的说明

目　　录

图表目录

译者序

2015 年 12 月 3 日，联发科技创意实验室（MediaTek Labs）宣布推出 LinkIt™ Smart 7688 开发平台。该平台是联发科技 LinkIt 开发平台系列的最新成员，可提供多种开发环境选项来加速各种先进的 Wi-Fi 无线连接设备的开发进程，比如利用云服务的 IP 摄像头、监控设备、智能电器和无线网关。

该平台现可提供两种版本的硬件开发套件（HDK）：LinkIt Smart 7688，包含一个基于 MT7688AN SOC 的微处理器单元（MPU）；LinkIt Smart 7688 Duo，除 MPU 外，还包含一个微控制器单元（MCU），并且与 Arduino 兼容。这两套开发板均支持内置 Wi-Fi、128MB RAM 和 32MB 闪存，以及具备连接多种多样的周边设备的能力。

笔者作为物联网（IoT）DIY 爱好者，发现市面上有多家厂商提供物联网应用方案，如 AR9331、基于全志系列 CPU 的友善之臂开发板、树莓派等。笔者没有对它们逐一尝试、无法对这些开发平台做过多评价。笔者在对比了多种 IoT 开发平台之后，发现 LinkIt Smart 7688 作为一款入门级的产品，对 IoT 小白来说，具有亲民的价格、专门的技术论坛，同时丰富的外设和稳定的 OS 能满足大部分的 IoT 应用。笔者在学习该开发平台后，本着互相交流学习的心态，为降低广大 DIY 爱好者的入门门槛，把《MediaTek LinkIt Smart 7688 Developers Guide V1_1》翻译为中文版，供大家学习。

本文档中涉及大量 Linux 环境下的基础知识，由于笔者水平有限，翻译过程中难免会有所错漏，仅作为广大 DIY 爱好者学习参考之用，一切以原版为准。如对本文档中的部分内容有疑惑或不同见解的，可参考官方网站提供的相关资料，或与笔者联系（qzrzq@126.com），作进一步的探讨。

译者

2017 年 5 月 16 日

1.概述

这一章是有关 MediaTek LinkIt 开发平台的概述，介绍 MediaTek LinkIt Smart 7688 开发平台，同时也作为本文档的内容向导。

1.1 什么是 MediaTek LinkIt

MediaTek LinkIt 是为可穿戴和物联网（IoT）设备原型化而设计的一系列开发平台，每个开发平台都提供了一系列工具、硬件和相关资源，使开发人员能够涉足各种可穿戴和物联网领域。

1.2 什么是 MidiaTek LinkIt Smart 7688 开发平台

MediaTek LinkIt Smart 7688 开发平台由一个基于 Linux Wi-Fi 芯片的开发板组成，旨在实现物联网设备的原型设计。例如应用于家庭或办公室的 Wi-Fi 安防网络摄像头和传感器，以及应用于幼儿或长者的网络监视器以及云应用。

LinkIt Smart 7688 是一款基于 OpenWrt Linux 发行版的开放式开发平台。它提供了大量的内存和储存空间，以满足各类应用程序开发。 该平台还可以使用 Python、Node.js 和 C 语言进行设备应用程序开发。

在原型化过程中，平台可以利用免费的联发科云沙盒服务来存储云中的数据。联发科实验室合作伙伴联系计划可帮助设备进入市场。

1.3 硬件开发包

LinkIt Smart 7688 开发平台提供两种开发板：

- LinkIt Smart 7688：仅 MPU，有 MediaTek MT7688 驱动。
- LinkIt Smart 7688 Duo：MPU 和 MCU，由 MT7688 和 Atmeag32U4 驱动。

有关开发板的详细信息，请参阅第 2 章“硬件开发包”。

1.4 程序设计环境

LinkIt Smart 7688 开发平台支持高级语言（Python 与 Node.js）和本地应用程序开发（C 语言）。另外，LinkIt Smart 7688 Duo 上的 MCU 还能使用 Arduino API 和工具进行程序设计。

有关软件开发的详细信息，请参阅第 3 章“程序设计环境向导”。

1.5 软件开发工具

LinkIt Smart 7688 为多种任务提供了软件开发工具，如配置开发板、更新主板固件、管理开发板板的 Arduino 支持和软件安装等。

有关软件开发工具和辅助程序的详细信息，请参阅第 4 章“软件和工具”。

1.6 开始向导

您可以在 MediaTek Lab 网站上找到 LinkIt Smart 7688 的开始向导，并学习如何运行一个闪烁灯例程，LinkIt Smart 7688 和 LinkIt Smart 7688 Duo 的应用步骤包含以下方面：

- 建立开发环境
- 连接到 LinkIt Smart 7688 开发板
- 升级开发板固件
- 访问并使用系统控制台
- 在 LinkIt Smart 7688 开发板上运行闪烁灯例程
- 连接到 Internet

1.6.1 文档、代码例程和相关信息

以下是几个有助于 LinkIt Smart 7688 软件开发的参考资料：

- 开发者向导，最新版本在 MediaTek Labs 网站可找到。
- LinkIt Smart 7688 开发板引脚图：提供开发板管脚引出的详细信息。
- LinkIt Smart 7688 Duo 开发板引脚图：提供开发板管脚引出的详细信息。

另外还有几个有助于设计最终的设备硬件的参考资料：

- LinkIt Smart 7688 硬件参考设计：包含：
  - LinkIt Smart 7688 开发板原理图和布局
  - LinkIt Smart 7688 开发板引脚图
  - MediaTek MT7688AN 芯片数据手册
- LinkIt Smart 7688 Duo 硬件参考设计：包含：
  - LinkIt Smart 7688 开发板原理图和布局
  - LinkIt Smart 7688 开发板引脚图
  - MediaTek MT7688AN 芯片数据手册

其他文档可能会不时出现，您可以在 MediaTek Labs 网站的开发平台文档页面上找到。

1.7 更多信息

LinkIt Smart 7688 开发平台基于开源 Linux 发行版，支持各种高级编程软件，您可以在下面找到有关软件开发的更多信息：

- OpenWrt
- C
- Python
- Node.js(jacaScript)
- Arduino

1.8 加入我们

可穿戴和物联网设备将带领下一波消费电子革命。MediaTek 在这个领域中饰演关键角色，结合了现有的电话制造商、电子设备制造商、和电信运营商，创造开放、充满活力的开发商和制造商社区。

无论你是制造商、设备厂商、学生、DIY 爱好者、还是程序员，你可以使用这个强大而简单的平台创造一些创新的东西。您可以注册 labs.mediatek.com，加入联发科 LinkIt 生态，我们期待您加入一起创造一些更好的东西。

2.硬件开发包

LinkIt Smart 7688 硬件开发包分为两种开发板：LinkIt Smart 7688（仅 MPU）和 LinkIt Smart 7688 Duo（具备 MPU 和 MCU）。其中 MPU 是 MediaTek 的 MT7688AN 芯片，MCU 则为 Atmega32U4。

MPU 支持 OpenWrt Linux 发行版，它集中处理各类逻辑任务并提供 Wi-Fi 连接，它支持 Python、Node.js 和 C 语言开发。MCU 负责处理实时的外设控制及运行 Arduino 程序。

2.1 MediaTek MT7688AN 芯片规格概述

MT7688AN 芯片规格如表 1 所示。

MT7688 芯片规格	
CPU	MIPS24Kec（580MHz）
总 DMIPs	560×1.6DMIPs
I-Cache,D-Cache	62KB，32KB
L2Cache	N/A
Memory	DDR1/DDR2 16bits Max 2Gb，193MHz
SPI Flash	3B addr mode(max 127Mbit) 4B addr mode (max 512　Mbit)
SD	SD-XC(class 10)
RF	1T1R 802.11n 2.4GHz
封装	DR-QFN156-12 mm ×12 mm

接口	数量
PCIe	1
USB2.0	1
Fash Ethernet Switch	5
I2S	1
PCM	1
PWM	4
SPI	1
I2C	1
UARTLite	3
JTAG	1

表 1 MT7688AN 芯片规格

2.2 LinkIt Smart 7688

LinkIt Smart 7688 是一块可用于物联网设备的高集成度、紧凑的硬件开发板。

2.2.1 关键特性

LinkIt Smart 7688 的关键特性如下：

- 1T1R Wi-Fi 802.11b/g/n (2.4G)
- GPIO、I2C、I2S、SPI、UART、PWM 和 Ethernet 端口引出
- 580MHz MIPS CPU
- 32MB flash 和 128MB DDR2 RAM
- USB host
- Micro SD 插槽

LinkIt Smart 7688 开发板如图 1 所示。

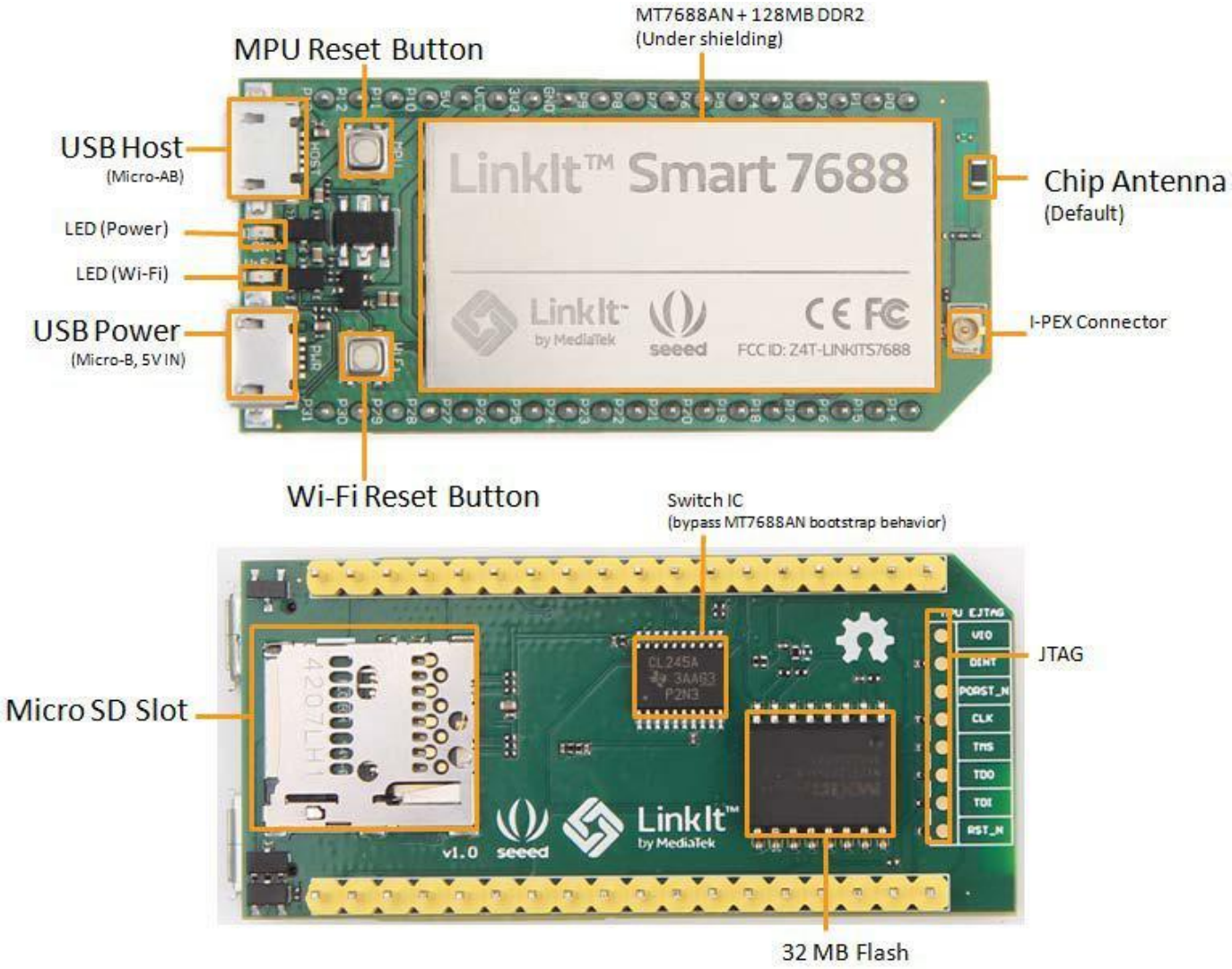


图 1 LinkIt Smart 7688 开发板（仅 MPU）

2.2.2 按钮

这一小节介绍 LinkIt Smart 7688 开发板上的按钮的功能，如表 2 所示。

情景	按钮	用法
复位 MPU	MPU 复位按钮	按一下（开发板已启动的情况下）
把 Wi-Fi 复位到 AP 模式	Wi-Fi 复位按钮	按下并保持至少 5 秒后释放（开发板已启动的情况下）
恢复出厂设置并进入 AP 模式（警告：开发板会恢复到默认设置并删除所有用户数据）	Wi-Fi 复位按钮	按下并保持至少 20 秒后释放（开发板已启动的情况下）
使用 U 盘升级固件	Wi-Fi 复位按钮	按下并保持至少 5 秒后释放（开发板正在上电启动时）
使用 U 盘升级启动引导程序（bootloader）	Wi-Fi 复位按钮	按下并保持至少 20 秒后释放（开发板正在上电启动时）

表 2 LinkIt Smart 7688 开发板的按钮

2.2.3 LEDs

这一小节介绍开发板上的 LEDs 的功能。

- 电源

  当开发板上电时，绿色的电源 LED 保持常亮。

- Wi-Fi

  Wi-Fi LED 呈橙色，其闪烁模式如表 3 所示。

模式	状态	LED 闪烁模式
AP 模式	有客户端设备	每秒闪烁 3 次，然后是 0.5s 的暂停，不断重复
	无客户端设备	熄灭
STA 模式	未连接	熄灭
	正在连接	每秒闪烁两次
	已连接	随着数据包的传输而闪烁

表 3 LinkIt Smart 7688 的 Wi-Fi LED 闪烁模式

有关 Wi-Fi 的 AP 模式和 STA 模式的详细信息，请参阅第 3.4 节“网络环境”。

2.2.4 天线

LinkIt Smart 7688 开发板支持两种天线：

1) 内置的 Wi-Fi 芯片天线，这是默认天线
2) 外部天线：I-PEX 接口

若要使用 I-PEX 接口，请把 I-PEX 接口左上角的电阻 R233 拆除，如图 2 所示。

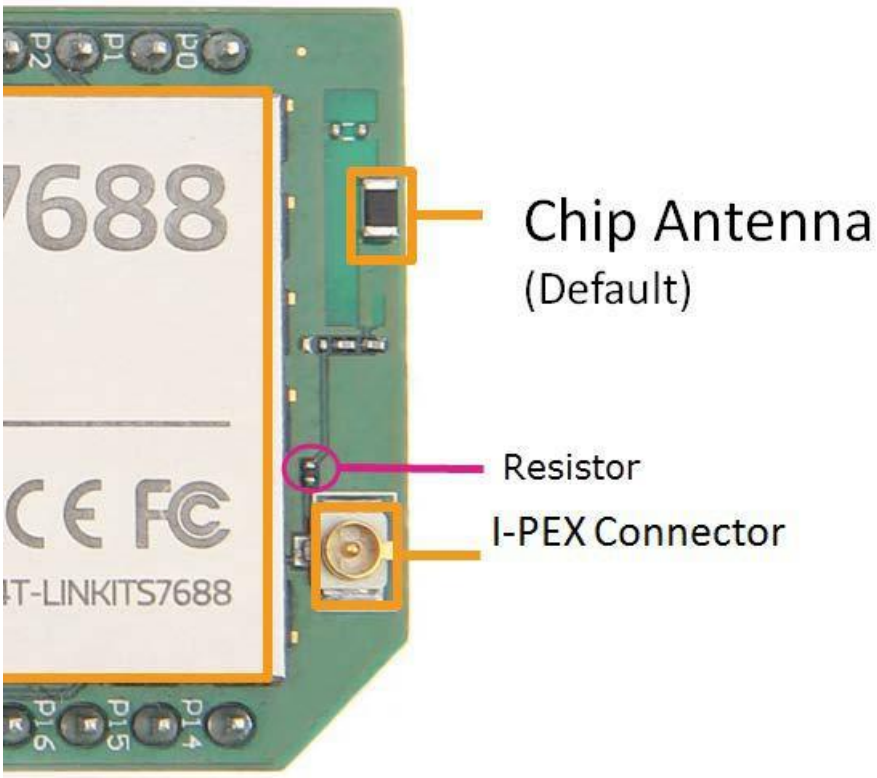


图 2 拆除电阻才能使用 I-PEX 接口

2.2.5 USB Host

LinkIt Smart 7688 有 USB Host 功能，用户可以用它连接各种 USB 设备，如网络摄像头、U 盘、键盘、摇杆等等。开发板上使用的是 USB micro-AB 型接口，其位置请参阅图 1。

2.2.6 USB 电源

开发板所需的 5V 电源由连接到计算机或其他电源的 USB 线提供。当您添加其他的外设，如 SD 卡、U 盘或其他 USB 设备，可能需要提供额外的电源。请使用高质量的 USB 线以降低功率损耗，如果您使用的外设需要大量的电源供应，建议您最好使用外部电源。

各种情景下 LinkIt Smart 7688 的近视功耗如表 4 所示。

注意：建议您使用 5V/1A 的电源。

情景		近似功耗
建立 Wi-Fi 连接	峰值	475.3 mA
	平均值	255.6 mA
设备启动	峰值	605.4 mA
	平均值	195.1 mA
通过 Wi-Fi 下载文件到 SD 卡	峰值	540.4 mA
	平均值	275.8 mA
通过 Wi-Fi 下载文件到 U 盘	峰值	569.5 mA
	平均值	304.9 mA
通过 Wi-Fi 下载文件到 Flash	峰值	522.4 mA
	平均值	271.3 mA

表 4 各种情景下的典型功耗

2.2.7 其他附件

所售的 LinkIt Smart 7688 开发平台不包含附件，因此您可以需要以下工具：

1) USB 电源线（必须）：USB A 型接口转 micro-AB 型接口的 USB 线，用于给开发板供电。
2) USB OTG 线（可选）：用来连接 A 型接口的 USB 设备，如 U 盘、USB 摄像头等，如图 3 所示。

图 3 USB OTG 线

USB 转串口线（可选）：用来与 Linux 系统控制台进行通讯。

SD 卡（可选）：作为外部存储空间，用于存放应用程序代码和数据。

U 盘（可选）：外部存储，也可用于存储待升级的 LinkIt Smart 7688 启动引导程序和固件。

2.2.8 JTAG

您可以使用 JTAG 接口调试 MT7688AN。要访问 JTAG 接口，您需要把开发板上的电阻 R95 拆下来，并重新把它焊接到 R3 的位置。完成这一步后重启开发板即可激活 JTAG 功能。具体的硬件操作步骤如下：

1) 找到的 LinkIt Smart 7688 开发板背面右上角的一组电阻，如图 4 红色圆圈所示。

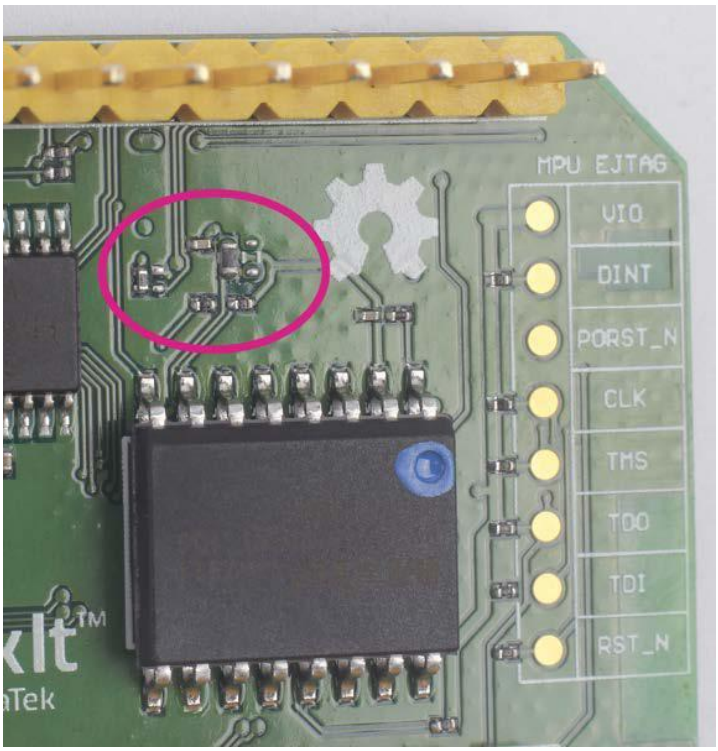


图 4 LinkIt Smart 7688 开发板背面的 JTAG 电阻

2) 把电阻拆下来并重新焊接到右边的位置，如图 5 所示。完成这一步之后，重启开发板即可激活 JTAG 功能。

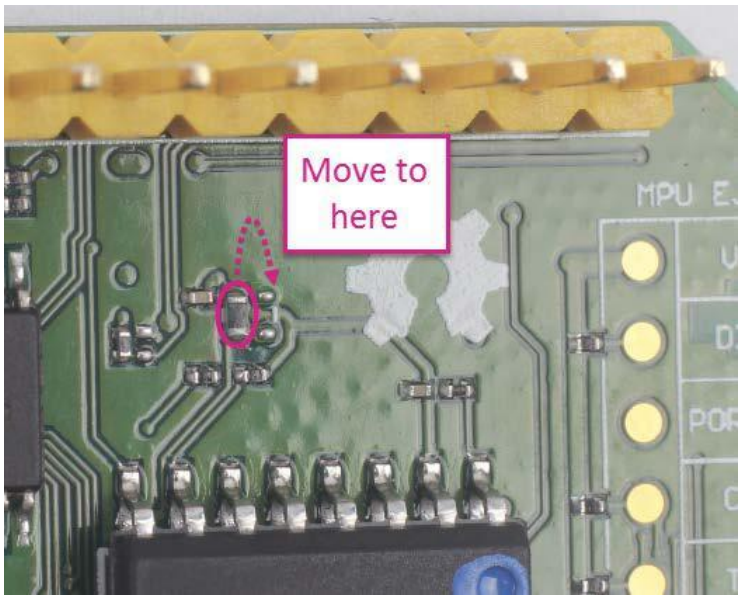


图 5 移动电阻以范围 JTAG 模式

2.2.9 规格

LinkIt Smart 7688 的主要规格参数如表 5 所示。

类别	特性	规格参数	
MPU	芯片	MT7688AN	
	内核	MIPS24KEc	
	时钟速度	580MHz	
	工作电压	3.3V	
PCB 大小	尺寸	55.7×26mm	
内存	Flash	32MB	
	RAM	128MB DDR2	
电源	USB 电源	5V(USB micro-B)	
	VCC	3.3V (管脚引出)	
GPIO	引脚数	22	
	引脚号	P1,P8,P9,P10,P11,P12,P13,P14,P15,P16,P17,P18,P19, P20,P21,P25,P26,P2,P7,P28,P29,P30,P31	
	电压	3.3V	
PWM	引脚数	4	
	引脚号	P8,P9,P26,P27	
	电压	3.3V	
	最大分辨率	7 位（可自定义）	
	最大频率@分辨率	标准模式	100kHz@1-bit 50kHz@2-bit 25kHz@3-bit 12.5kHz@4-bit 6.25kHz@5-bit 3.125kHz@6-bit 1.5625kHz@7-bit
		快速模式	40MHz@1-bit 20MHz@2-bit 10MHz@3-bit 5MHz@4-bit 2.5MHz@5-bit 1.25Mhz@6-bit 625kHz@7-bit

类别	特性	规格参数
外部中断	引脚数	22
	引脚号	P1,P8,P9,P10,P11,P12,P13,P14,P15,P16,P17,P18,P19, P20,P21,P25,P26,P2,P7,P28,P29,P30,P31
SPI	数量	1
	引脚号	P22,P23,P24(与板载 Flash 共用),P25
	最高速度	25MHz
SPI 从机	数量	1
	引脚号	P28,P29,P30,P31
	最高频率	25MHz
I2S	数量	1
	引脚号	P10,P11,P12,P13
I2C	数量	1
	引脚号	P20,P21
	速度	120K/400K
串口	数量	3
	引脚号	P8,P9,P16,P17,P18,P19
	最高速度	115200bps
USB Host	数量	1
	引脚号	P6,P7
	接口类型	Micro-AB
通讯	Wi-Fi	1T1R 802.11b/g/n(2.4G)
	以太网	1-port 10/100 FE PHY
	引脚号	P2,P3,P4,P5
用户存储	SD 卡	Micro SD SDXC

表 5 LinkIt Smart 7688 开发板规格参数

2.2.10 引脚图

引脚图有助于您为LinkIt Smart 7688开发板的引脚和对应外设（如GPIO、PWM、I2C、I2S、UART、SPI 等）之间建立映射关系。LinkIt Smart 7688 的引脚图如图 6 所示，您也可以在 MediaTek 网站下载。

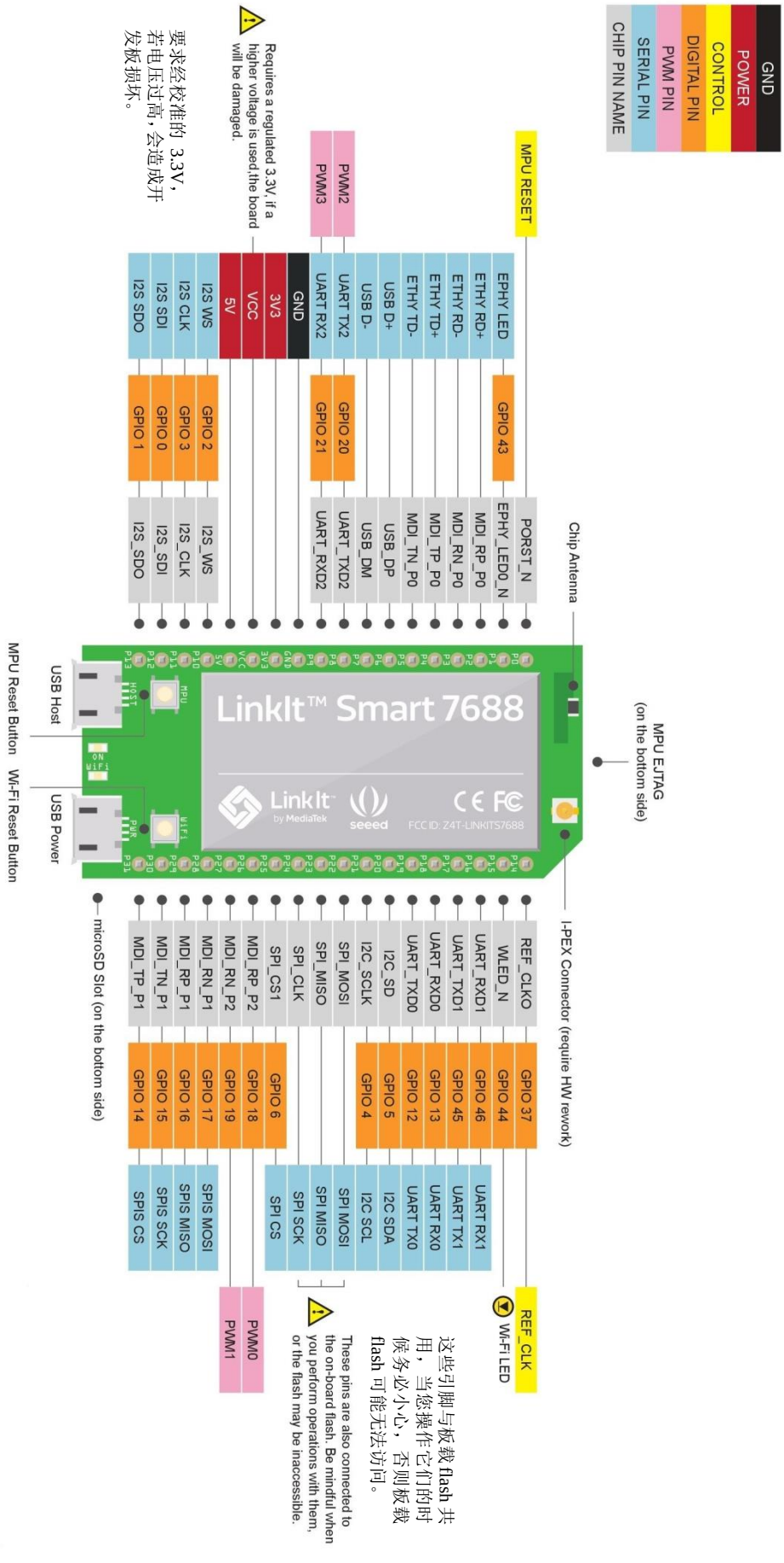


图 6 LinkIt Smart 7688 引脚图

2.3 LinkIt Smart 7688 Duo

LinkIt Smart 7688 Duo 开发板拥使用一款和 LinkIt Smart 7688 一样的 MT7688AN 芯片，但它还有一颗 Atmega32U4 MCU。因此，它支持一些额外的功能，如模拟 I/O、支持 Arduino IDE 等。这款开发板的功能是两款芯片的组合：搭载 OpenWrt Linux 系统的 MT7688AN 芯片负责 Wi-Fi 和以太网连接，使用 Arduino 程序的 Atmega32U4 负责各类外设控制。

2.3.1 关键特性

LinkIt Smart 7688 Duo 的关键特性如下：

- 1T1R Wi-Fi 802.11b/g/n (2.4G)
- GPIO、I2C、I2S、SPI、UART、PWM、ADC 和 Ethernet 端口引出
- 580MHz MIPS CPU
- 32MB flash 和 128MB DDR2 RAM
- USB host
- Micro SD 插槽
- 支持 Arduino（Atmega32U4）

LinkIt Smart 7688 Duo 开发板如图 7 所示。

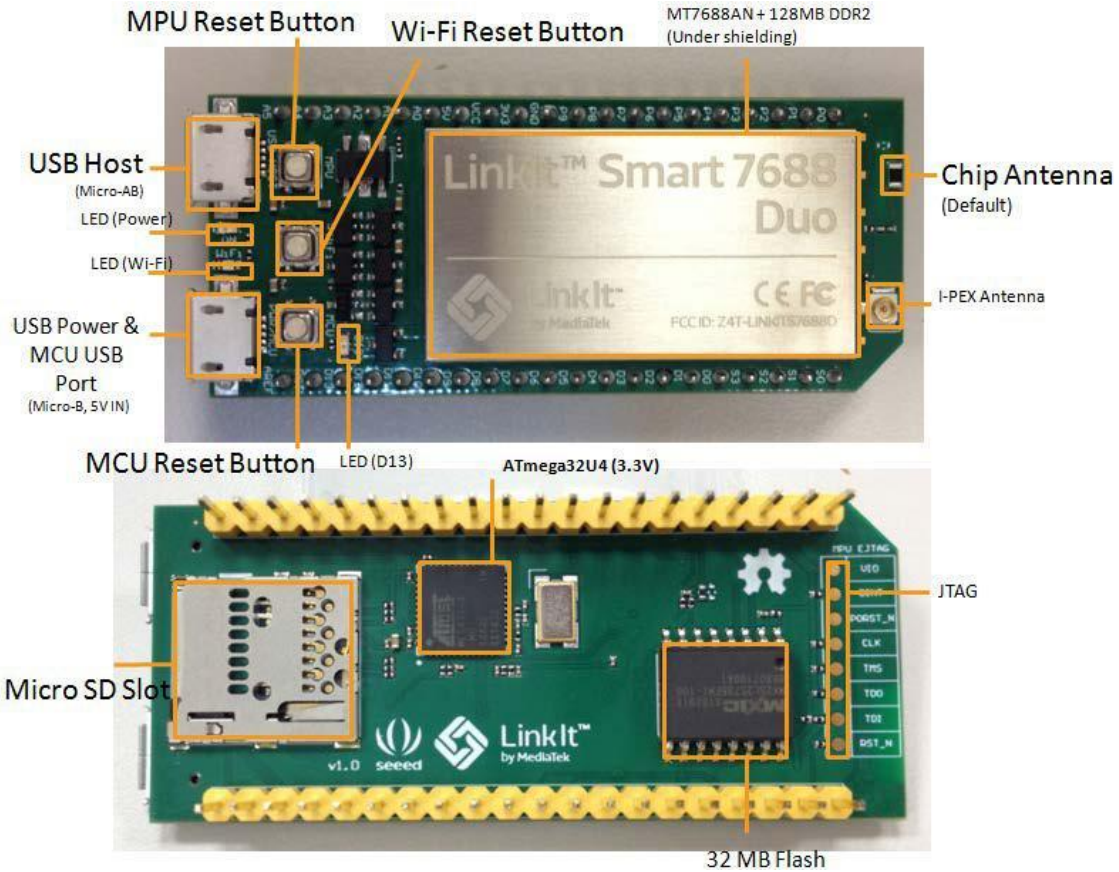


图 7 LinkIt Smart 7688 Duo 开发板（MPU+MCU）

2.3.2 按钮

这一小节介绍 LinkIt Smart 7688 Duo 开发板上的按钮的功能，如表 6 所示。

情景	按钮	用法
复位 MPU	MPU 复位按钮	按一下
复位 MCU	MCU 复位按钮	按一下
进入 MCU 的 bootloader 模式（8 秒后超时）	MCU 复位按钮	750ms 内连续按两次
把 Wi-Fi 复位到 AP 模式	Wi-Fi 复位按钮	按下并保持至少 5 秒后释放（开发板已启动的情况下）
恢复出厂设置并进入 AP 模式（警告：开发板会恢复到默认设置并删除所有用户数据）	Wi-Fi 复位按钮	按下并保持至少 20 秒后释放（开发板已启动的情况下）
使用 U 盘升级固件	Wi-Fi 复位按钮	按下并保持至少 5 秒后释放（开发板正在上电启动时）
使用 U 盘升级启动引导程序（bootloader）	Wi-Fi 复位按钮	按下并保持至少 20 秒后释放（开发板正在上电启动时）

表 6 LinkIt Smart 7688 开发板的按钮

2.3.3 LEDs

LinkIt Smart 7688 Duo 和 LinkIt Smart 7688 一样，有电源 LED 和 Wi-Fi LED，请参阅第 2.2.3 节“LEDs”查看更多有关信息。除了电源和 Wi-Fi LEDs，LinkIt Smart 7688 Duo 还有一个连接 D13 引脚的 LED，这颗 LED 是由用户的 Arduino 控制的。

2.3.4 天线

LinkIt Smart 7688 Duo 的天线和 LinkIt Smart 7688 一样，请参阅第 2.2.4 节“天线”查看更多信息。

2.3.5 USB Host

LinkIt Smart 7688 Duo 有 USB Host 功能，用户可以用它连接各种 USB 设备，如网络摄像头、U 盘、键盘、摇杆等等。开发板上使用的是 USB micro-AB 型接口，其位置请参阅图 7。

2.3.6 USB 电源

开发板所需的 5V 电源由连接到计算机或其他电源的 USB 线提供。当您添加其他的外设，如 SD 卡、U 盘或其他 USB 设备，可能需要提供额外的电源。请使用高质量的 USB 线以降低功率损耗，如

果您使用的外设需要大量的电源供应，建议您最好使用外部电源。USB 电源接口的位置请参阅图 7。

各种情景下 LinkIt Smart 7688 Duo 的近似功耗如表 7 所示。

情景		近似功耗
建立 Wi-Fi 连接	峰值	596.4 mA
	平均值	273.5 mA
设备启动	峰值	672.6 mA
	平均值	248.9 mA
通过 Wi-Fi 下载文件到 SD 卡	峰值	605.4 mA
	平均值	300.4 mA
通过 Wi-Fi 下载文件到 U 盘	峰值	616.6 mA
	平均值	347.5 mA
通过 Wi-Fi 下载文件到 Flash	峰值	578.5 mA
	平均值	336.3 mA

表 7 典型功耗

2.3.7 其他附件

所售的 LinkIt Smart 7688 Duo 开发平台不包含附件，因此，和 LinkIt Smart 7688 一样，您可以需要一些其他附件，请查阅第 2.2.7 小节“附件”。

2.3.8 转接板

Seed Studio 为 LinkIt Smart 7688/7688 Duo 提供了转接板，这块转接板提供了所有的 MT7688AN 和 Atmega32U4 管脚引出，您可以使用它轻松连接各类传感器和外设。

2.3.9 JTAG

您可以使用 JTAG 接口调试 MT7688AN。要访问 JTAG 接口，您需要把开发板上的电阻 R95 拆下来，并重新把它焊接到 R3 的位置。完成这一步后重启开发板即可激活 JTAG 功能。具体的硬件操作步骤如下：

1) 找到的 LinkIt Smart 7688 Duo 开发板背面右上角的一组电阻，如图 8 红色圆圈所示。

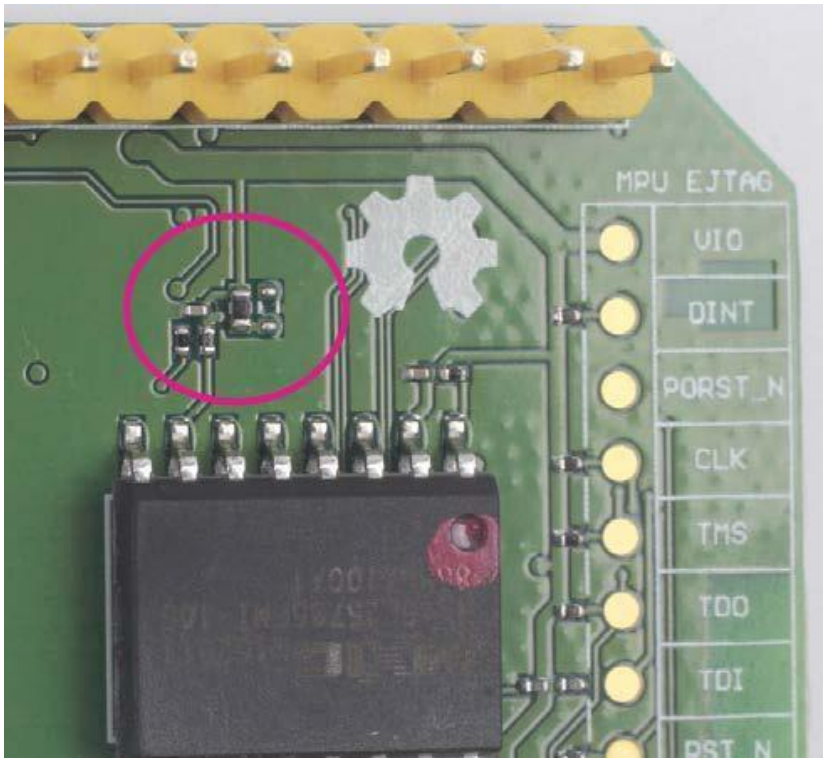


图 8 LinkIt Smart 7688 Duo 开发板背面的 JTAG 电阻

2) 把电阻拆下来并重新焊接到右边的位置，如图 9 所示。完成这一步之后，重启开发板即可激活 JTAG 功能。

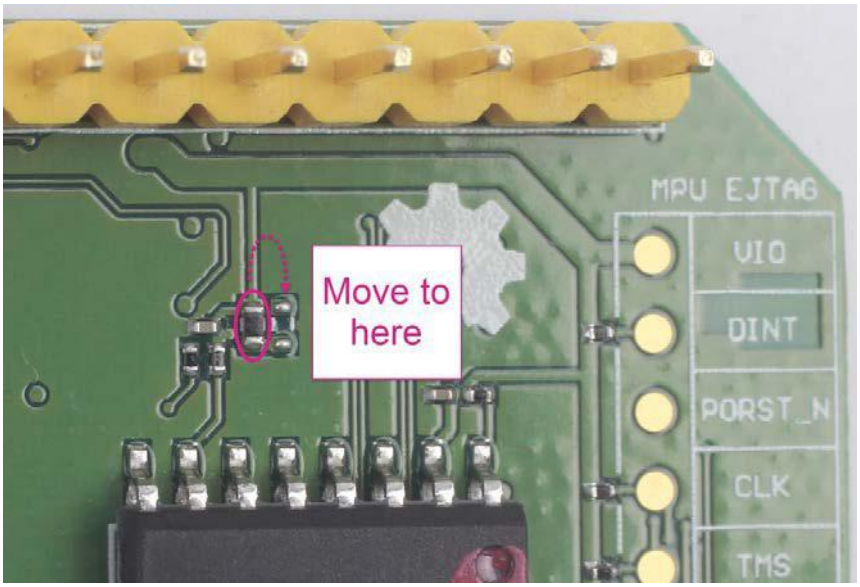


图 9 移动电阻以范围 JTAG 模式

2.3.10 规格

LinkIt Smart 7688 Duo 的主要规格参数如表 8 所示。

类别	特性	规格参数
MPU	芯片	MT7688AN
	内核	MIPS24KEc
	时钟速度	580MHz
	工作电压	3.3V
MCU	芯片	ATmega32U4
	内核	Atmel AVR
	时钟速度	8MHz
	工作电压	3.3V
PCB 大小	尺寸	60.8×26mm
内存	Flash	32MB
	RAM	128MB DDR2
电源	USB 电源	5V(USB micro-B)
	VCC	3.3V (管脚引出)
GPIO	引脚数	3（MT7688AN） 24（Atmega32U4）
	引脚号	P1,P8,P9,A0,A1,A2,A3,A4,A5,S0,S1,S2,S3,D0,D1,D2,D3,D4,D5,D6,D7,D8,D9,D10,D11,D12,D13,
	电压	3.3V
PWM	引脚数	8（Atmega32U4）
	引脚号	D3, D5, D6, D9, D10, D11, D12, D13
	电压	3.3V
	最大分辨率	16 位（可自定义）
	最大频率@分辨率	31.25kHz@8-bit Timer 0 (4 sets)
		2MHz@2-bit 122Hz@16 -bit Timer 1 &3 (4 sets)
		187.5kHz@8-bit 46.875kHz@10-bit Timer 4 (6 sets)

类别	特性	规格参数
ADC	引脚数	12 (ATmega32U4)
	引脚号	A0,A1,A2,A3,A4,A5,D4,D6,D8,D9,D10,D12
	电压	3.3V
外部中断	引脚数	8 (ATmega32U4)
	引脚号	S0,S1,S2,S3,D8,D9,D10,D11
SPI	数量	1 (ATmega32U4)
	引脚号	S0, S1, S2, S3
	最高速度	4MHz
I2C	数量	1
	引脚号	D2,D3
	速度	400K
串口	数量	1 (MT7688AN)
		1 (ATmega32U4)
	引脚号	P8,P9(MT7688AN)
		D0, D1(ATmega32U4)
	最高速度	115200bps(MT7688AN)
		0.5 Mbps(ATmega32U4)
USB Host	数量	1 (MT7688AN)
	引脚号	P6,P7
	接口类型	Micro-AB
通讯	Wi-Fi	1T1R 802.11b/g/n(2.4G)
	以太网	1-port 10/100 FE PHY
	引脚号	P2,P3,P4,P5
用户存储	SD 卡	Micro SD SDXC

表 8 LinkIt Smart 7688 Duo 开发板规格参数

2.3.11 引脚图

引脚图有助于您为 LinkIt Smart 7688 Duo 开发板的引脚和对应外设（如 GPIO、PWM、I2C、I2S、UARTI 等）建立映射关系。LinkIt Smart 7688 的引脚图如图 10 所示，您也可以在 MediaTek 网站下载。

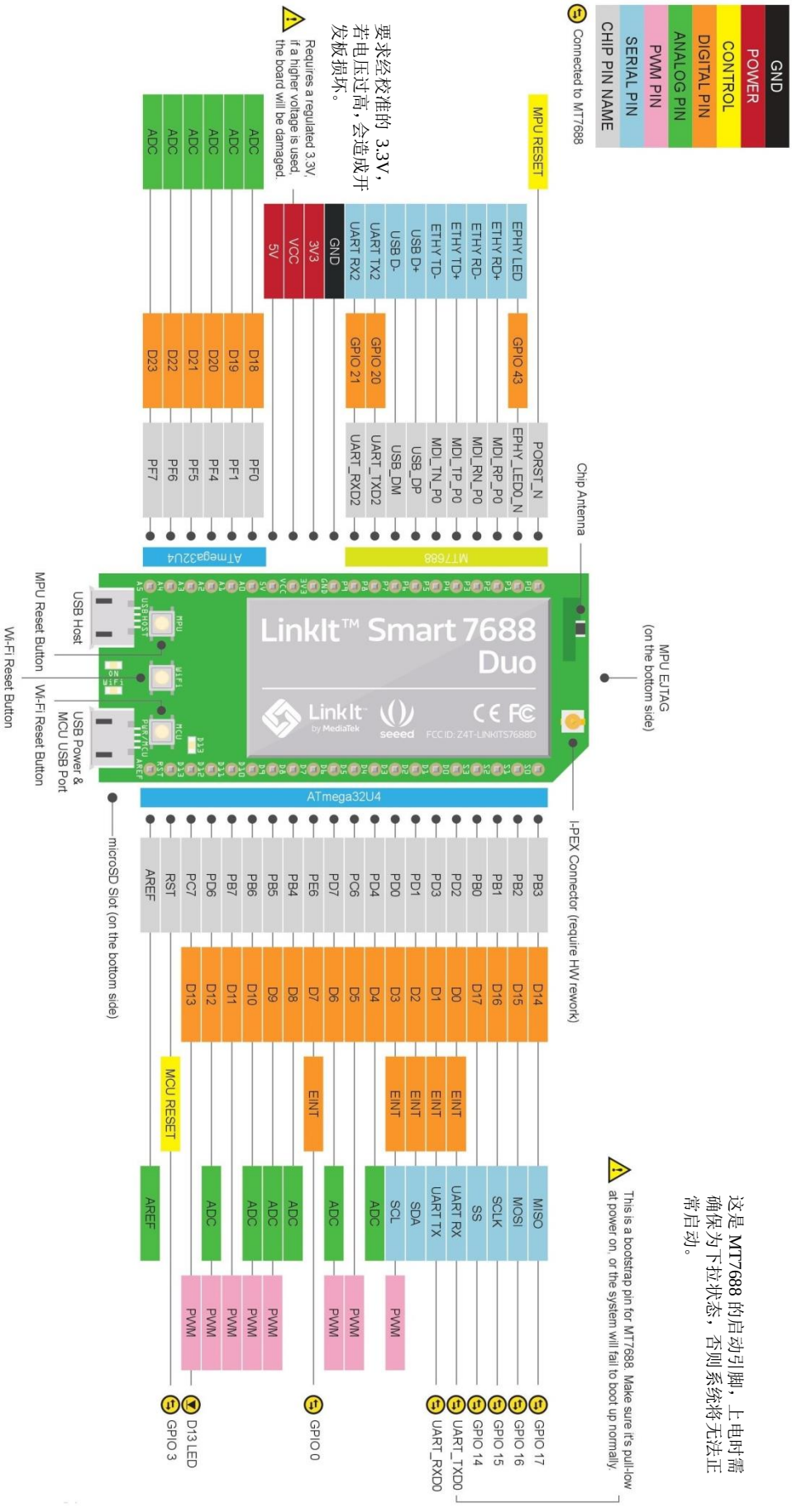


图 10 LinkIt Smart 7688 Duo 引脚图

2.4 FCC,CE 和 NCC 认证

LinkIt Smart 7688 和 LinkIt Smart 7688 Duo 开发板通过 FCC,CE 和 NCC 认证，详细信息参阅附录 A。

3.程序设计环境向导

这一章介绍：

- 本平台使用的操作系统
- LinkIt Smart 7688 开发平台提供的程序设计环境
- 各种程序设计模型以及如何在开发板上使用
- 用于与开发板进行 Wi-Fi 通信的网络环境
- 使用 C/C++、Python 和 Node.js 创建应用程序的方法

3.1 平台操作系统

LinkIt Smart 7688 开发平台使用 OpenWrt 开源嵌入式 Linux 操作系统，这个操作系统最初是针对嵌入式设备 (如无线路由器) 而开发的。OpenWrt 的主要特性如下：

1) 大量的网络控制功能
2) 完全可写的文件系统，同时具备包管理
3) 丰富和可扩展的功能集，超过 3400 种软件包且数量还在不断增长。

3.2 程序设计环境概述

LinkIt Smart 7688 开发平台运行在 OpenWrt Linux 环境，它支持本地 C/C++、高级语言 Python 和 JavaScript（使用 Node.js）开发。

通过本地应用程序，您可以为需要更高性能的设备创建驱动程序、框架和系统应用程序。而通过高级语言开发则使您能够快速构建设备原型。

因为 LinkIt Smart 7688 没有显示器，所以您需要在一台独立的计算机上远程开发高级程序，通常把这台计算机称为主机平台。大部分的编辑和开发工作都在主机平台上完成，最终生成的程序再传送到 LinkIt Smart 7688 这个目标平台上执行

本节稍后将简要地说明整个开发过程的程序设计方法。LinkIt Smart 7688 使用的程序设计语言及其相关的开发环境概览如表 9 所示。

编程语言	工具和库	应用	主机平台
C/C++	交叉编译工具链	系统编程	● OS X ● Linux
Python	LinkIt Smart 7688 的 Python 运行时库	● 设备原型化 ● 网络 ● 物联网应用	● OS X ● Linux ● Windows
Node.js	LinkIt Smart 7688 的 Node.js 运行时库	● 设备原型化 ● 网络 ● 物联网应用	● OS X ● Linux ● Windows

表 9 LinkIt Smart 7688 程序设计环境概览

3.3 不同开发板的程序设计模型

LinkIt Smart 7688 和 LinkIt Smart 7688 Duo 开发板使用同样的内核程序设计（编程）环境（译者注：后续出现的“程序设计”和“编程”是同一个意思），主要的差异是两款开发板可用的接口不同，以及 LinkIt Smart 7688 Duo 多出来的微控制器（译者注：后续“微控制器”使用“MCU”代替）。

图 11 显示了 LinkIt Smart 7688 和 LinkIt Smart 7688 Duo 的程序设计模型以及有关的用于访问传感器的软件栈。

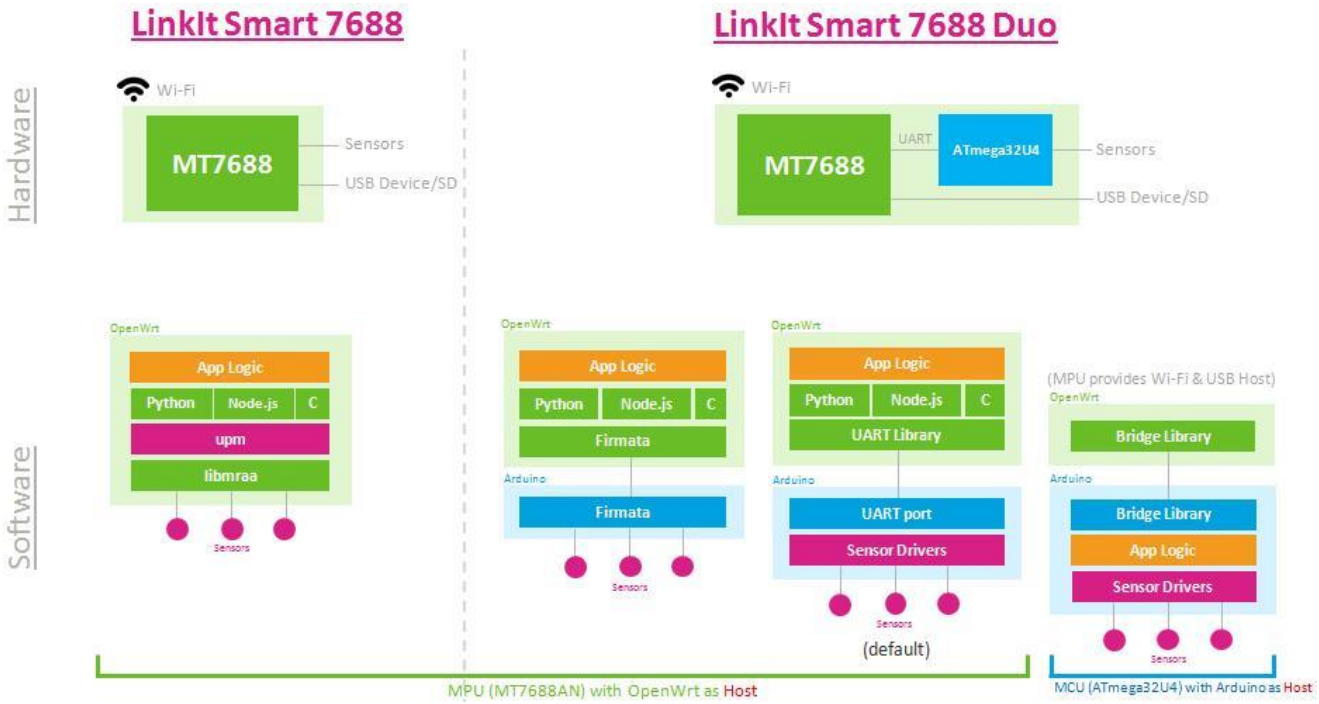


图 11 LinkIt Smart 7688 开发平台的程序设计模型

在 LinkIt Smart 7688 开发板中，各类外部设备和外设都直接连接到 MT7688AN MPU 上并由 Linux 系统控制。设备应用程序也是在 MT7688AN MPU 的 Linux 系统中执行。

而在 LinkIt Smart 7688 Duo 开发板中，各类外部设备和外设连接到 Atmega32U4，并受 MCU 的控制。但是，由于 MPU 和 MCU 可以通过 UART 接口进行通讯，所以您既可以选择使用 C/C++、Python 和 Node.js 为 MPU 开发应用程序，也可以使用 Arduino 为 MCU 开发 Arduino 程序。

根据不同层次的用户情景和可编程性，可以使用不同的方案和软件栈，如图 11 所示。 如果您使用的是 LinkIt Smart 7688 开发板，请参阅第 5 章“LinkIt Smart 7688 的外设编程”，了解如何连接不同的设备和外设。如果您使用的是 LinkIt Smart 7688 Duo 开发板，请参阅第 6 章“LinkIt Smart 7688 Duo 的外设编程”，了解如何建立 Arduino IDE，以对 MCU 进行编程，以及如何实现 MT7688AN 和 ATmega32U4 微控制器之间的通信。

3.4 网络环境

LinkIt Smart 7688 开发平台的 Wi-Fi 通讯共有两种模式：AP 模式和 STA 模式，这一节将介绍这两种模式。

3.4.1 AP 模式

在 AP 模式下，LinkIt Smart 7688 开发板作为一个热点组建一个局域网，如图 12 所示。AP 模式主要用于配置开发板设置。

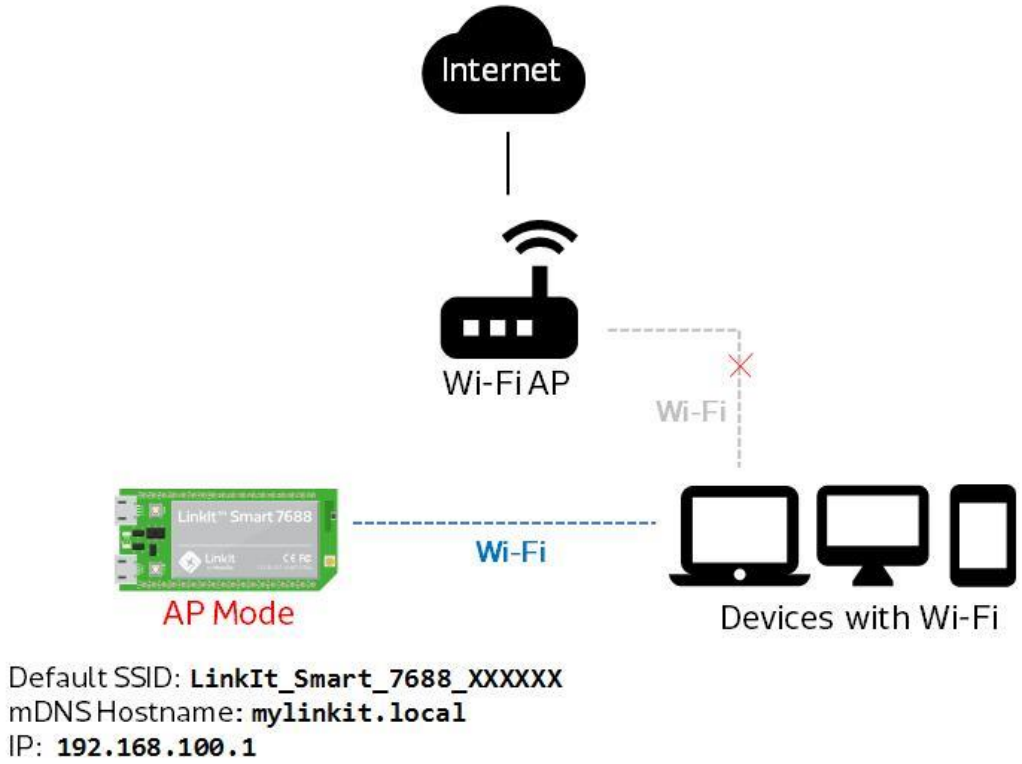


图 12 LinkIt Smart 7688 运行在 AP 模式

3.4.2 STA 模式

在 STA 模式下，LinkIt Smart 7688 开发板可以加入一个已知的 Wi-Fi 网络并访问 Internet，如图 13 所示。

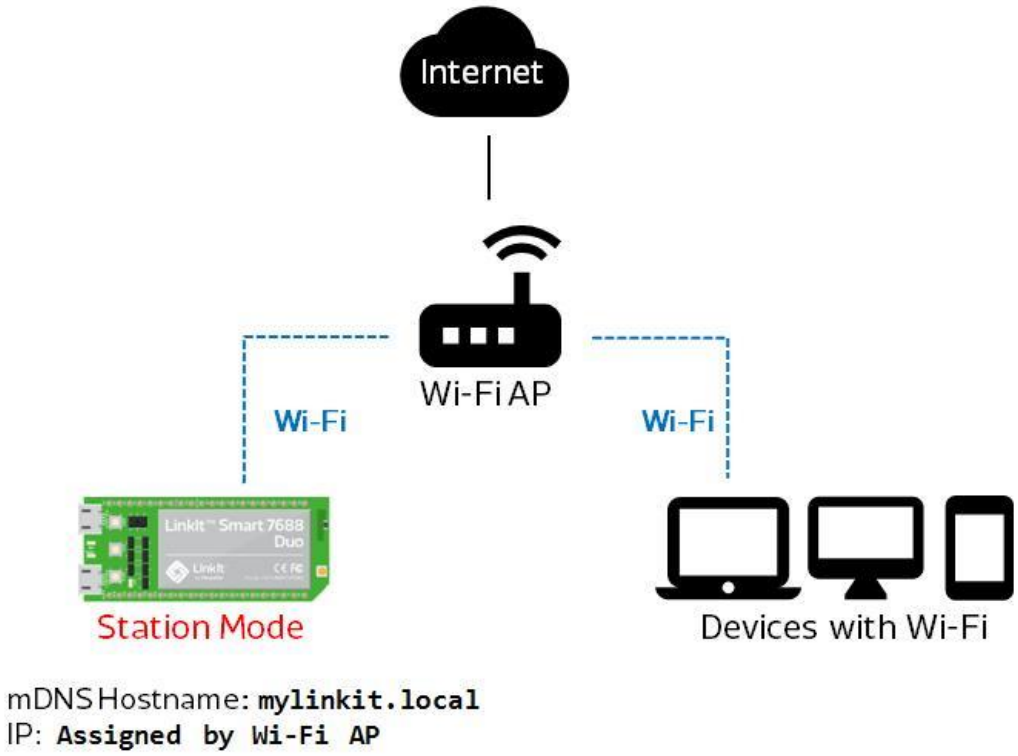


图 13 LinkIt Smart 7688 运行在 STA 模式

这个模式除了可以让应用程序在 Internet 上访问远程系统或云服务之外，还可以让用户使用 opkg 软件包管理器将软件从 OpenWrt 安装到开发板上。这时，您的计算机需加入到与开发板相同的 Wi-Fi 网中，并通过 SSH 协议连接到开发板。

3.5 使用 C/C++编程

本地应用程序开发需要使用工具链将 C/C++源文件编译和链接为可执行的二进制文件。虽然您可以直接在 LinkIt Smart 7688 Linux 系统中安装开发工具，但板载的 128MB RAM 可能会不够用，这无疑会限制本地应用程序开发。

为了避免在本地应用程序开发过程中发生内存溢出，建议您在一个更加强大的主机平台上建立本地应用程序开发环境，使您能够将程序源码交叉编译为 LinkIt Smart 7688 目标板的可执行文件。

3.5.1 建立 C/C++编程环境

交叉编译工具链包含在 LinkIt Smart 7688 SDK 的软件包中，它支持 Mac OS X 和 Linux 系统，在编写这个文档时还不支持 Windows 系统。要使用这个工具链，请把它下载并解压到您选择的目录，并把工具链目录命名为 CC_TOOLS。

3.5.2 C 语言的 Hello World 例程

1) 打开文本编辑器并创建一个名为 helloworld.c 的文件

2) 复制并粘贴下方的示例代码，保存。

```
# include <stdio.h>
int main(int  argc, char** argv)
{
  printf( "Hello, World!\n ");
  return 0;
}
```

3) 在主机 PC，输入以下命令对代码进行交叉编译

```
CC_TOOLS/bin/mipsel-openwrt-linux-g++ helloworld.c -o helloworld
```

4) 假设主机环境已连接到 LinkIt Smart 7688 的 Wi-Fi 网络，使用 SCP 工具将生成的名为 helloworld 的二进制文件传送到 LinkIt Smart 7688。 例如：

```
scp ./helloworld root@mylinkit.local:helloworld
```

5) 最后，在 LinkIt Smart 7688 的 SSH 终端执行这个程序：

```
# ./helloworld
```

现在，您应该能看到屏幕输出了字符串“Hello,World!”。

3.6 使用 Python 编程

高级编程语言由 LinkIt Smart 7688 中的相应语言解释器执行。您可以远程编程，然后将代码发送到 LinkIt Smart 7688 中执行。

3.6.1 建立 Python 编程环境

高级编程语言环境很简单，您只需要安装一个文本编辑器和一个用来在计算机和 LinkIt Smart 7688 之间传输程序文件的工具软件即可，请参阅第 4.7 节“文件编辑器及传输”。

3.6.2 Python 语言的 Hello World 例程

1) 打开文本编辑器，复制并粘贴以下示例代码，保存为 helloworld.py。

```
print "Hello World!"
```

2) 假设主机环境已连接到 LinkIt Smart 7688 的 Wi-Fi 网络，使用 SCP 工具将生成的名为 helloworld 的二进制文件传送到 LinkIt Smart 7688（译者注：这里貌似有误）。 例如：

```
scp ./helloworld root@ mylinkit.local:helloworld
```

3) 执行示例代码。在 LinkIt Smart 7688 系统控制台中进行此操作，输入以下命令调用 Python 解释器：

```
# python helloworld.py
```

现在，您应该能看到程序输出了字符串“Hello World!”。

3.6.3 为 Python 安装额外的模块

Python 有一个名为 **pip** 的默认包管理器，您可以使用它安装一些额外的 Python 模块，要想使用它，首先要确保您的开发板已经连接到 Internet，如 4.6.5 小节“将 LinkIt Smart 7688 连接到 Wi-Fi 热点以访问 Internet”所述，点击这里查看可用的包列表，然后在系统控制台中使用 **pip install** 命令安装软件包，例如：

```
# pip install  requests
```

上面这个示例是安装了一个常用的 requests 模块，它可以帮助您轻松地生成 HTTP 请求。

需要注意的是，某些依赖于本机 C/C++实现的 Python 软件包可能会回退到 Python 实现，或者根本无法安装，这是因为开发板上没有可用的本地编译工具链环境。例如，在安装 simplejson 模块时，您将在安装过程中看到如下警告：

```
warning: install_lib: byte- compiling is disabled, skipping.

**********************************************************************
WARNING: The C extension could not be compiled, speedups are not enabled.
Plain-Python installation succeeded.
**********************************************************************
Successfully installed simplejson
```

安装最终成功，是因为 simplejson 模块提供了纯 Python 实现作为备用——这个模块仍然能够运行，但可能会比那些有本地编译工具链环境的运行得慢。

3.7 使用 Node.js 编程

LinkIt Smart 7688 的 Node.js 程序设计环境和 Pyhon 一样，使用一个文本编辑器和一个文件传输工具来执行程序代码，或者使用 SSH 协议来创建您的 Node.js 程序。

在开始之前，请确保您已经通过 SSH 连接到 LinkIt Smart 7688。详细信息请参阅第 4.5.4.1 小节“使用 SSH（安全外壳协议）”。

3.7.1 Node.js 的 Hello World 例程

这个示例直接在 LinkIt Smart 7688 控制台中执行。

1) 通过 SSH 打开系统控制台，创建一个名为 app 的文件夹。

```
# mkdir  example
# cd example
```

2) 使用 vi 编辑器创建一个名为 app.js 的文件。

```
# vim app.js
```

3) 在 vi 编辑器中，输入字母 **i** 来插入代码，请输入以下代码：

```
console.log('Hello World');
```

4) 保存文件并退出 vi 编辑器
   a) 按下 **Esc** 键
   b) 按下 **Esc** 键并输入**:wq!**
5) 输入以下命令，执行示例代码

```
# node app.js
```

现在，您应该能看到程序输出了字符串“Hello World”。

3.7.2 为 Node.js 安装额外的软件包

Node.js 有一个名为 **npm** 的默认包管理器，您可以使用它安装一些额外的 Node.js 模块，要想使用它，首先要确保您的开发板已经连接到 Internet，如 4.6.5 小节“将 LinkIt Smart 7688 连接到 Wi-Fi 热点以访问 Internet”所述，点击这里查看可用的包列表，然后在系统控制台中使用 **npm install** 命令安装软件包，例如：

```
# npm install request
```

上面这个示例是安装了一个常用的 requests 模块，它可以帮助您轻松地生成 HTTP 请求。

需要注意的是，某些依赖于本机 C/C++实现的 NPM 软件包可能会回退到纯 JavaScript 实现，或者根本无法安装，这是因为开发板上没有可用的本地编译工具链环境，特别是 node-gyp。例如，在安装 socket.io 模块时，您将在安装过程中看到如下警告：

```
sh: node-gyp: not found
npm WARN optional dep failed, continuing  utf-8-validate@1.2.1
```

这是因为 socket.io 依赖于其他的模块，如 utf-8-validate 和 bufferutil，他们需要 node-gyp 建立其本地实现。对于 socket.io 模块来说，因为他们是可选的依赖，所以最终这个模块能安装成功，而您也可以使用其有限的功能。而某些需要本地实现的模块，会完全无法安装。

4.软件和工具

这一章介绍了 Linkle Smart 7688 开发板用于创建、测试和运行应用程序的软件和工具。

4.1 软件和工具

Linkle Smart 7688 的软件和工具包括:

- 最新的开发板启动引导程序
- 最新的开发板固件
- 本地应用程序开发所需的用于 Libmraa 和 OpenWrt SDK 的工具链

需要注意的是上述软件和工具不包括 Python、Node.js 和 Arduino 的编辑器软件（译者注：即不包括 Python、Node.js 和 Arduino 的 IDE），对于 Python 和 Node.js，要使用标准开发环境建创建程序代码，然后使用系统控制台工具在开发板上启动应用程序。对于 Arduino 开发，使用标准的 Arduino IDE 来编写代码并在开发板上运行。

4.2 支持的主机环境

根据您采用的开发模式，对应使用的主机环境如表 10 所示。

开发模式	Windows	Ubuntu	Mac OS X
本地（OpenWrt）C/C++应用程序	不支持(1)	支持	支持
Python 或 Node.js 应用程序	支持	支持	支持
Arduino 程序（仅 Linkle Smart 7688 Duo）	支持	支持	支持

表 10 操作系统支持

(1)您可以在 Windows 计算机上使用 Ubuntu 的虚拟机进行开发。

Linkle Smart 7688 软件和工具支持以下操作系统版本：

- Windows XP、7、8 和 10
- Mac OS X 10.9 和 10.0
- Ubuntu 14.04 LTS

4.3 默认的 OpenWrt 包

两种开发板都预装了常用的软件包，详见表 11。

软件包	简介
Dropbear	一个轻量级 SSH 服务器
cURL	利用 URL 语法在命令行方式下工作的文件传输工具
stty	打印或更改 terminal(终端)的设置
UVC USB 摄像头支持	USB 摄像头的内核驱动
Python	Python 语言支持
pySerial	提供串口访问功能的 Python 库
Node.js	JavaScript 语言支持
node-serialport	提供串口访问功能的 JavaScript 库
Bridge library	Arduino Yun's Bridge library
libmraa	Linux 系统下，用于 IO 接口的 C/C++库，绑定了 JacaScript 和 Python 的 C/C++库，
UPM	一个基于 libmraa 的传感器驱动库
OpenSSL	TLS/SSl 协议和密码库工具包
AVAHI	通过 mDNS/DNS-SD 协议在本地网络上发现设备和服务
AVRDUDE	Linux 系统下，用于烧写代码到 MCU 的命令行工具

表 11 Linkle Smart 7688 开发平台内含的软件包

除此之外，还有其他可用的 OpenWrt 软件包供您升级或安装以扩展您的工具集。OPKG 程序可用于升级和管理这些软件包。有关 OPKG 的更多信息，请参见第 4.4 节“OPKG 包管理器”或 OpenWrt 网站。

4.4 OPKG 包管理器

OPKG 包管理器是一个用于从本地或 Internet 安装 OpenWrt 软件包的工具。您可以使用此工具在 Linkle Smart 7688 上安装和更新软件包。

命令行中常用的 opkg 参数如下：

- **List**

获取当前安装的软件包列表，例如：

```
# opkg list-installed
```

- **Update**

更新可以获取的软件包列表。在安装新的软件包之前，请首先更新可以获取的软件包列表。例如：

```
# opkg update
```

- **Install**

安装软件包。这个参数后面必须跟随包名或全称域名（FQDN）。

```
# opkg install <pkgs| FQDN>
```

例如要安装名为“nano”的文本编辑器，你既可以用包名也可以用 FQDN。

```
# opkg install nano

# opkg install
http://mirror2.openwrt.org/mt7688/packages/nano_2.4.1-1_ramips_24kec.ipk
```

- **Remove**

卸载已经安装的指定的软件包。例如：

```
# opkg remove <pkgs| globp>
```

- **Upgrade**

对已经安装的软件包升级。例如：

```
# opkg upgrade <pkgs| globp>
```

注意：使用 opkg 进行安装或升级，您需要将 Linkle Smart 7688 切换到 STA 模式，即将设备连接到 Internet。有关的内容，请参阅第 3.4.2 小节“STA 模式”。同时，如果您将软件包升级到较新的版本，可能造成 API 或行为发生变化，这可能导致软件包无法与当前项目一起使用。

有关 opkg 功能的详细信息，请参阅 OpeWrt 网站。

4.5 系统配置

这一节介绍配置 Linkle Smart 7688 开发板的工具和方法。

4.5.1 系统配置工具

配置 Linkle Smart 7688 开发板有两种方法：Web UI 和系统控制台。

Web UI 可以完成常见的相关开发任务，但如果要获得对系统的完全访问权限，您需要使用系统

控制台。表 12 列出了这两种方案的特点。

配置	接口	连接方法	参考
Wi-Fi 简单配置	Web UI	Wi-Fi 网络	连接到 Web UI
系统配置	系统控制台	SSH（Wi-fi）	使用 SSH
		内核控制台（UART2 接口）	使用 USB 转串口线

表 12 Web UI 和系统控制台的配置功能

此外，某些功能（如更新启动引导程序和固件）可以使用 USB 驱动器（译者注：U 盘）完成。

4.5.2 本地域

本开发板使用本地域名 **mylinkit.local**，你的计算机需要支持 mDNS 才能使用这个本地域。Windows 8 及后续版本、Mac OS X 和 Linux 均支持 mDNS。然而，如果你使用的是 Windows 7，你需要安装 Bonjour 打印服务，才能使你的计算机在本地域名中发现 Linkle Smart 7688 的 IP 地址。

如果您使用的是虚拟机，请注意，mDNS 到达客户机操作系统网络时可能会有问题（译者注：假设 XP 系统上安装了 Linux 虚拟机，则 XP 称为宿主操作系统，Linux 称为客户机操作系统。这一句话译者水平有限，可自行斟酌，原文：If you are using a virtual machine, please note that mDNS may have problems reaching the guest OS network）。在这种情况下，下一节（连接到 Web UI）的内容请使用宿主操作系统的浏览器完成。您还可以在宿主操作系统中使用 ping 程序查询 mylinkit.local 的 IP 地址并在客户机操作系统中是使用这个 IP 地址。

4.5.3 连接到 Web UI

Linkle Smart 7688 Web UI 可以配置系统信息、升级固件、完成设备复位、切换 AP 模式与 STA 模式等。以下步骤适用于 Windows、Mac OS X 和 Linux。

1) 使用任意 USB 电源（如计算机和 USB 线）对开发板上电。如图 14 所示，使用的是 Linkle Smart 7688 Duo 开发板。

图 14 连接 Linkle Smart 7688 开发板和计算机

请确保将 USB 线连接到电源接口，而不是 MPU 复位按钮旁边的 USB 主机接口。绿色 LED（电源指示灯）会点亮并保持常亮，紧接着是橙色 LED（Wi-Fi 指示灯）闪烁一次（启动引导初始化）。然后，大约 5 秒后，设备开始启动，橙色 LED 将保持常亮大约 30 秒。

设备启动之后，Wi-Fi LED 熄灭。这意味着系统已准备就绪，可以接受 Wi-Fi 连接，接下来的步骤下面会继续介绍。图 15 显示了 Wi-Fi LED 状态和系统对应状态阶段。

2) 打开计算机上的 Wi-Fi 连接面板并连接到名为 LinkIt_Smart_7688_1B09F3（1B09F3 是 MAC 地址，你的可能不一样）的热点，如图 16 所示。

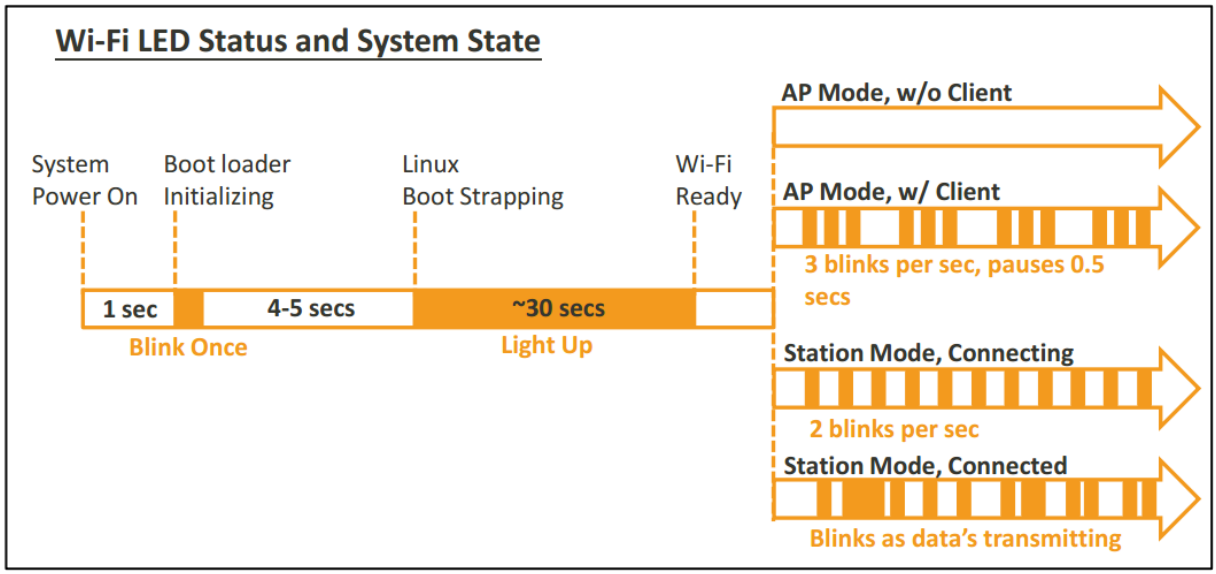


图 15 Wi-Fi LED 状态

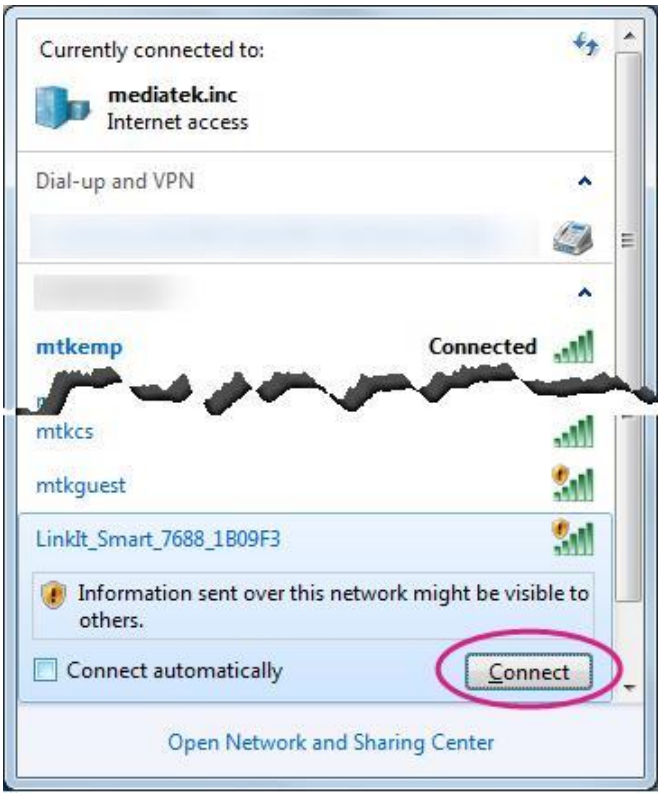


图 16 连接到 LinkIt_Smart_7688 AP

当您连接到连接到 LinkIt_Smart_7688 热点后，Wi-Fi LED 将每秒闪烁三次。

请记住，一旦您连接到 LinkIt Smart 7688，您的计算机将不能访问 Internet——它现在已加入 LinkIt Smart 7688 组建的局域网，如图 17 所示。在之后的步骤中您将了解如何将 LinkIt Smart 7688 连接到 Internet。但首先您需要配置 LinkIt Smart 7688，这在将在下一步中介绍。

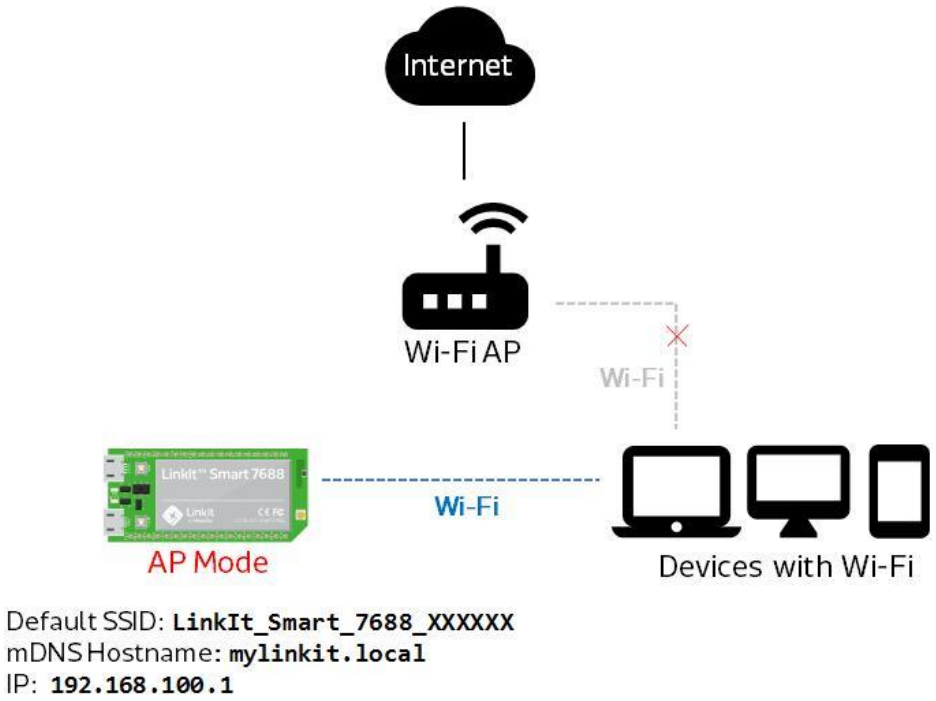


图 17 LinkIt Smart 7688 工作在 AP 模式

3) 在您的 Web 浏览器打开 http:// mylinkit.local，如图 18 所示。

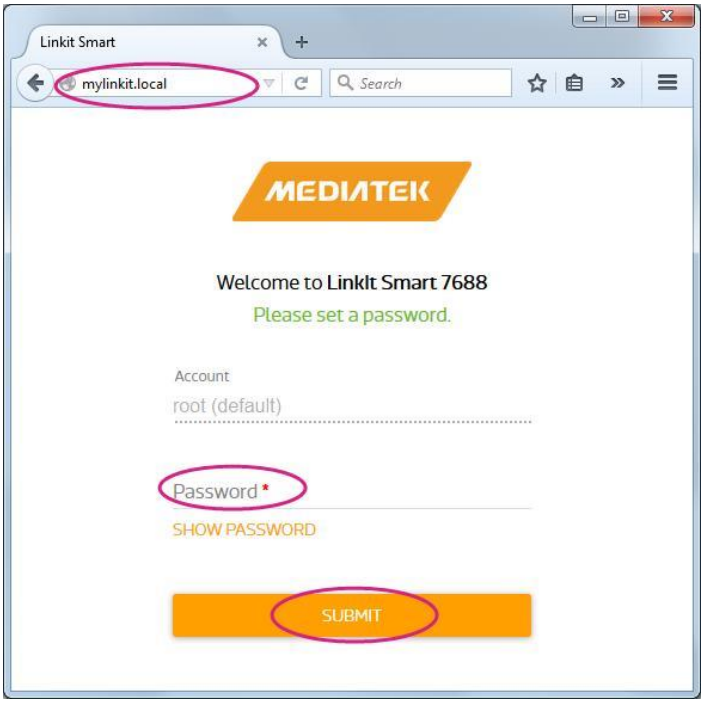


图 18 LinkIt Smart 7688 Web UI 登录

如果开发板已经有密码，而且您没有使用过或丢失，请使用 USB 驱动器升级固件或按住 Wi-Fi 按钮至少 20 秒钟并释放。请记住，上述任一方法，将恢复开发板为默认设置并且删除所有用户数据。有关如何使用 USB 驱动器升级固件或使用按钮的详细信息，请参阅第 4.6.1 小节“升级固件”或第 4.6.4 小节“恢复出厂设置”。

4) 点击 **Password** 栏并设置密码（至少 6 位字母或数字）。

点击 **Submit** 并再次输入密码登录。

4.5.4 连接到系统控制台

访问 LinkIt Smart 7688 开发板系统控制台的的方法有两种，如下所述：

4.5.4.1 使用 SSH（Secure Socket Shell 安全外壳协议）

在开始之前，首先要确保您已经如 4.5.3 小节“连接到 Web UI”所述，通过 Web UI 设置了密码，同时，LinkIt Smart 7688 与您的计算机在同一个网络中。

- **Windows 系统下的操作：**

1) 从这里下载并安装 PuTTY

2) 在 Host Name box 中输入 **mylinkit.local**，选择 SSH 单选按钮，点击 Open，如图 19 所示。

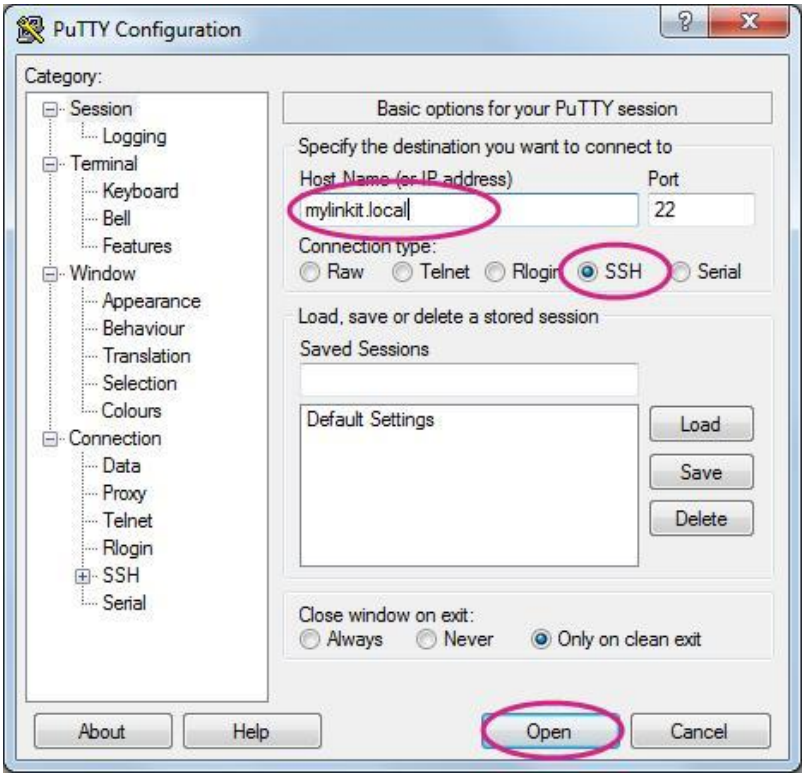


图 19 使用 Windows PuTTY 的 SSH

3) 当您第一次使用 PuTTY、更新固件后或用了不同的开发板，会如图 20 所示弹出一个安全警告，点击 Yes。

图 20 PuTTY 安全警告

4) 在打开的 PuTTY 终端窗口中，使用用户名 **root** 登录并输入您先前在 Web UI 中设置的密码，登录后应该显示类似图 21 所示。

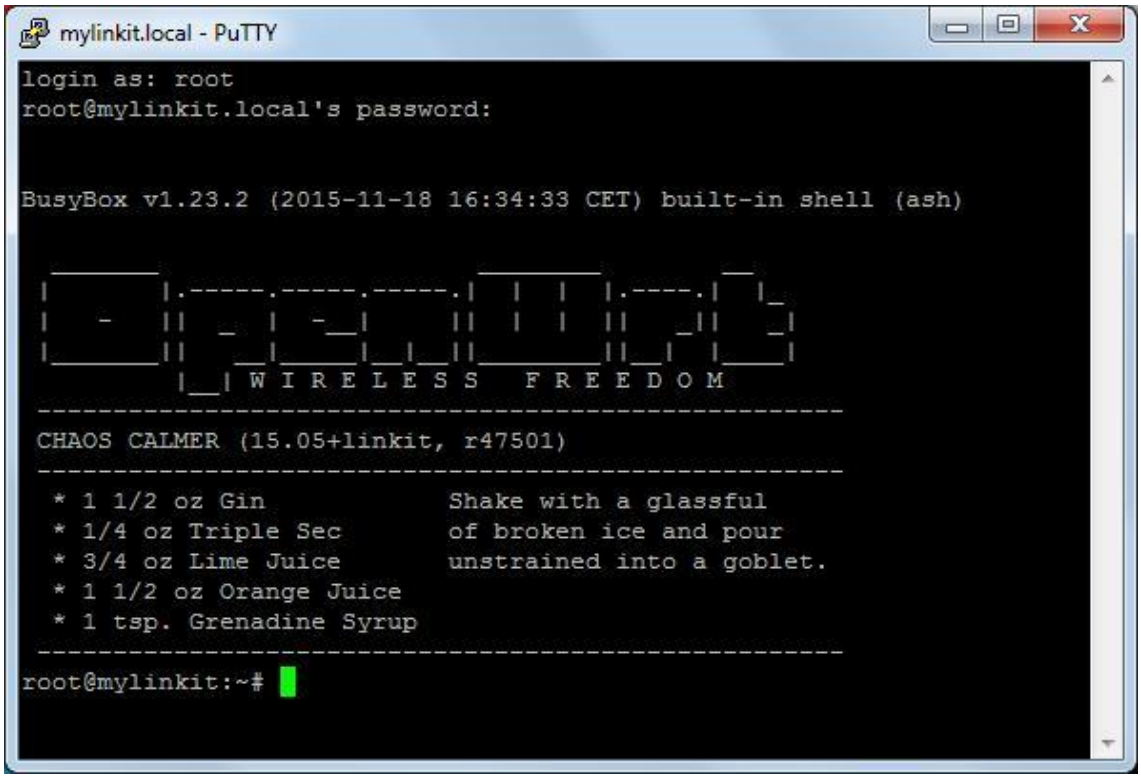


图 21 系统控制台

- **Mac 下的操作：**

打开终端。在终端的命令行中输入 **ssh root@mylinkit.local**，点击返回（译者注：这里应该点击 Enter？没用过 Mac，不懂），提示输入 root 密码。输入先前在 Web UI 中设置的密码。

- **Linux 下的操作**

打开终端。在终端的命令行中输入 **ssh root@mylinkit.local**，点击返回（译者注：这里应该点击 Enter？没用过 Linux，不懂），提示输入 root 密码。输入先前设置的密码。

如果您看到一个提示主机 ID 问题的错误，请参阅联发科实验室网站上的常见问题，了解更多有关使用 SSH 的信息，请参考这里。

4.5.4.2 使用 USB 转串口线

您可以使用 USB 转串口线连接 LinkIt Smart 7688 系统控制台。首先，您要检查一下您的 USB 转串口线是否需要安装驱动：因为这有可能需要，也有可能不需要。请根据您的操作系统检查以下内容。（注意：从**当前支持的 VCP 驱动**列表（仅针对 FTDI 芯片）下载您操作系统对应的驱动程序版本。安装完毕后，才能成功识别 COM 端口。）

- Windows：

1) 安装驱动。如果你使用的 USB 转串口线是基于 FTDI 芯片的，请从这里下载驱动程序。如果使用最新版本的驱动出现问题的话，可以尝试一下旧版本的驱动。
2) 如表 13 所示，使用 USB 串口线连接到 LinkIt Smart 7688 的 UART 引脚。

USB 转串口线	LinkIt Smart 7688 UART 引脚
RX	P8
TX	P9
GND	GND

表 13 LinkIt Smart 7688 UART 引脚

3) 连接好 USB 转串口线后，打开设备管理器查看如图 22 所示的 COM 端口号，这个端口号在不同的电脑上可能会不一样。

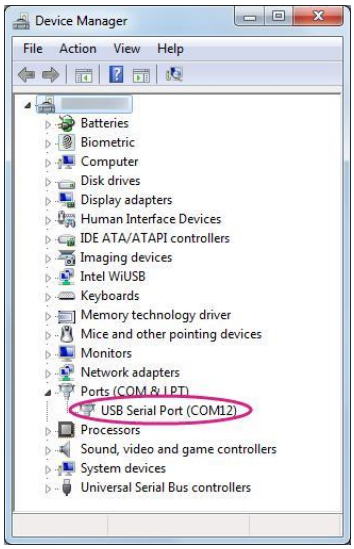


图 22 使用 USB 转串口线的 LinkIt Smart 7688 端口号

4) 打开 PuTTY 终端并输入上述设备管理器中的 **COM 端口**号，点击 **Serial** 单选按钮，在 Speed 框输入 **57600** 然后点击 **Open**，如图 23 所示。

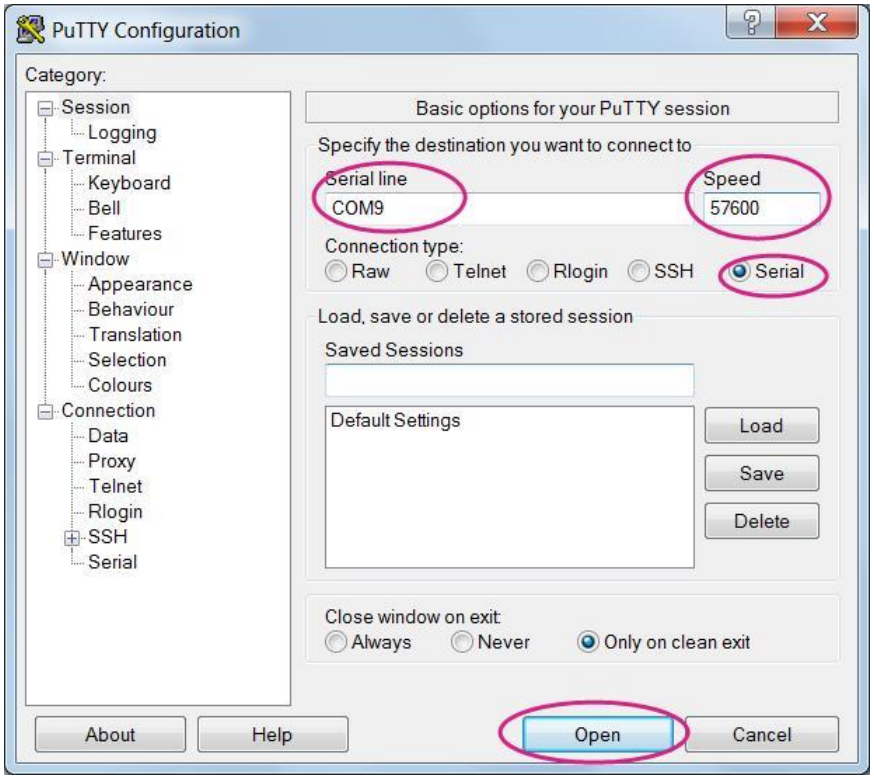


图 23 使用 USB 转串口线和 Windows 终端访问系统控制台

5) 点击 PuTTY 窗口右上角的“×”即可退出系统控制台。

- Mac：

1) 如有必要则安装驱动程序。在串口线生产厂家的网站上查看 Mac 系统下的驱动要求及安装说明。

2) 插入串口线并连接到 LinkIt Smart 7688

3) 打开 Terminal session。您可以通过应用程序、辅助工具或 Terminal 打开它。

4) 在 Terminal 中输入 **ls/dev/cu***，您将看到一个设备列表，找到一个类似 cu.usbserial–XXXXXXX 的名字（XXXXXXXX 是一个随机 ID），这就是访问系统控制台的串口设备。例如：

```
$ls /dev/cu*
/dev/cu.Bluetooth-Incoming- Port
/dev/cu.Bluetooth-Modem
/dev/cu.pablop-WirelessiAP
/dev/cu.usbserial-A6YMCQBR
```

5) 使用 screen 工具连接到串口并设置波特率为 57600，这是因为系统控制台通讯的波特率默认为 57600，例如：

```
$screen /dev/cu.usbserial-XXXXXXXX 57600
```

6) 现在，您应该已经连接到系统控制台。在 Terminal 中按 ENTER 键，弹出提示，您会注意到这时的提示已与您的操系统上原来的 Terminal 不同，它是 LinkIt Smart 7688 的提示，类似如下所示：

```
root@myLinkIt:/#
```

7) 现在已准备好通过这个系统控制台对 LinkIt Smart 7688 系统做出更改。

8) 要退出系统控制台，请输入**<CTRL>a+k**，和 **y**。（译者注：没用过 Mac，不懂这步如何操作）

- Linux

1) 如有必要则安装驱动程序。在串口线生产厂家的网站上查看 Linux 系统下的驱动要求及安装说明。

2) 插入串口线并连接到 LinkIt Smart 7688

3) 打开一个 Terminal 程序

4) 在 Terminal 中输入 **ls/dev/ttyUSB*,**您将看到一个设备列表，找到一个类似 ttyUSB0 的名字（0 是一个随机 ID），这就是访问系统控制台的串口设备。例如：

```
$ls /dev/ttyUSB*
/dev/ttyUSB0
```

5) 使用 screen 工具连接到串口并设置波特率为 57600，这是因为系统控制台通讯的波特率默认为 57600，有一点需要注意的是，在 Ubuntu 系统中，访问串口需要 dialout 组权限。可以使用 sudo 命令来提升权限,例如：

```
$sudo screen /dev/ttyUSB0 57600
```

6) 现在，您应该已经连接到系统控制台。在 Terminal 中按 ENTER 键，弹出提示，您会注意到这时的提示已与您的操系统上原来的 Terminal 不同，它是 LinkIt Smart 7688 的提示，类似如下所示：

```
root@myLinkIt:/#
```

7) 现在已准备好通过这个系统控制台对 LinkIt Smart 7688 系统做出更改。

8) 要退出系统控制台，请输入**<CTRL>a+k**，和 **y**。（译者注：没用过 Linux，不懂这步如何操作）

4.6 系统配置工作

这一节介绍了有哪些你需要完成的系统配置工作以及如何通过 Web UI、系统控制台或 U 盘完成这些系统配置工作。

4.6.1 升级固件

这一小节介绍了如何升级固件，你可以选择以下两种方式升级 LinkIt Smart 7688 的固件：

（注意：升级固件后，所有的用户数据都会被删除，LinkIt Smart 7688 和 LinkIt Smart 7688 Duo 开发板的固件都是一样的。）

- 使用 Web UI

1) 登录 LinkIt Smart 7688 Web UI，如果您之前没有登录过的话，请参阅第 4.5.3 小节“连接到 Web UI”。

2) 登录之后，如图 24 所示，点击 **UPGRADE FIRMWARE**。

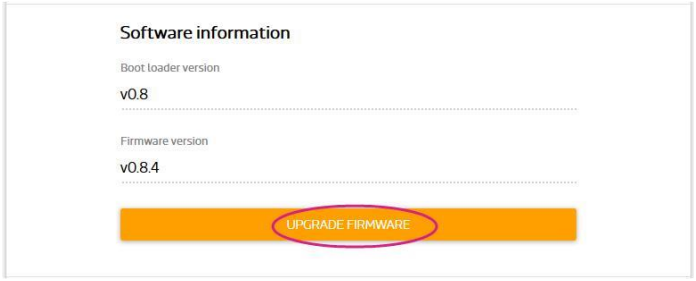


图 24 固件升级按钮

3) 点击 **Choose the file** 并选择 lks7688.img 文件。然后点击 **UPGRADE & RESTART**，如图 25 所示。

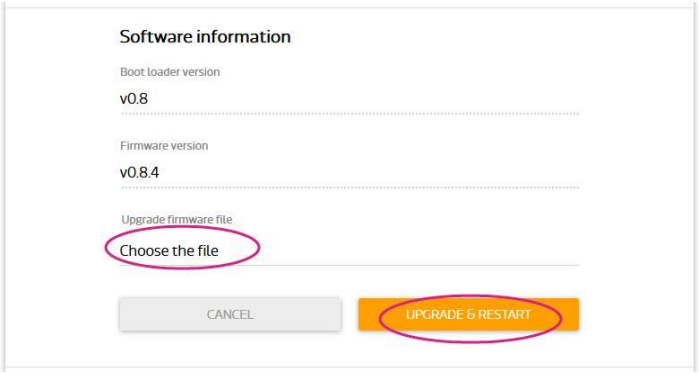


图 25 选择固件文件

4) 固件将被上传到 LinkIt Smart 7688，注意固件升级过程中，开发板不能断电。

5) 橙色的 Wi-Fi LED 将会持续闪烁约 3 分钟，当升级完成的时候，这个 LED 会变成常亮，开发板将重启。

6) 几秒钟之后，Wi-Fi LED 熄灭，打开 Wi-Fi 连接面板，搜索并连接名为 LinkIt_Smart_7688_XXXXXX 的热点，重新加载 mylinkit.local 网页，设置新的密码并登录，在软件信息栏查看新的固件版本号，如图 26 所示，固件升级完成。

图 26 查看固件版本

- 使用 U 盘

  U 盘必须是 FAT 文件系统，否则 LinkIt Smart 7688 开发平台将无法识别。

1) 把固件文件（lks7688.img）保存在 U 盘根目录，您可以从 MediaTek Labs 网站下载最新版本固件。

2) 把 U 盘插入 LinkIt Smart 7688 开发板。

3) 按下 Wi-Fi 按钮和 MPU 复位按钮，然后仅松开 MPU 复位按钮，保持 Wi-Fi 按钮 5 秒后再松开（5 秒后 Wi-Fi LED 熄灭），LED 状态如图 27 所示，注意，按住 Wi-Fi 按钮不能超过 20 秒，否则开发板会升级启动引导程序而不是固件。

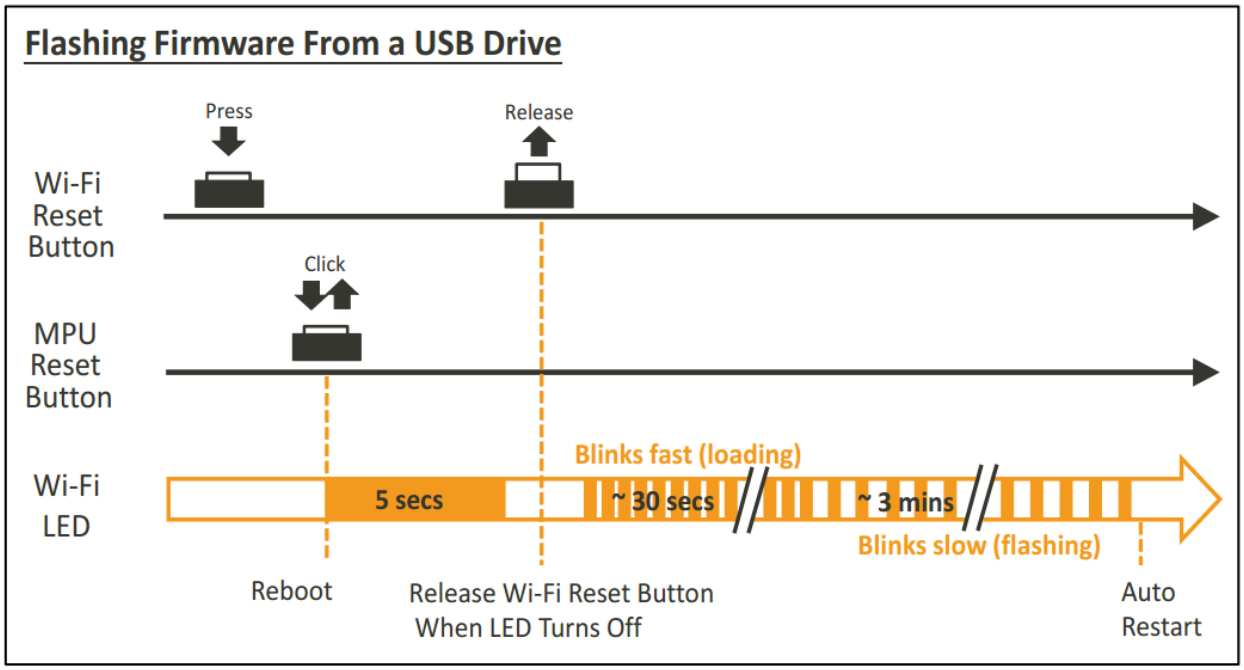


图 27 固件升级过程中的 Wi-Fi LED 状态

4) LinkIt Smart 7688 将开始读取固件映像（此时 Wi-Fi LED 快速闪烁）并完成固件升级过程（此时 Wi-Fi LED 慢速闪烁），这个过程大约持续 3 分钟。

4.6.2 升级启动引导程序（bootloader）

警告：在升级启动引导程序的过程中，切勿断电或拔出 U 盘，否则 Flash 中的数据将损坏，进而导致开发板损坏而无法再次启动。

这一小节介绍了如何升级启动引导程序，注意 U 盘必须是 FAT 文件系统，否则 LinkIt Smart 7688 开发平台将无法识别。

请按照以下步骤操作：

1) 将启动引导程序（lks7688.ldr）保存在 U 盘根目录并命名为“lks7688.ldr”，您可以从 MediaTek Labs 网站下载最新版本的启动引导程序。
2) 把 U 盘插入 LinkIt Smart 7688 开发板。
3) 同时按下 Wi-Fi 按钮和 MPU 复位按钮，然后仅松开 MPU 复位按钮，保持 Wi-Fi 按钮 20 秒后再松开（20 秒后 Wi-Fi LED 会点亮）。
4) LinkIt Smart 7688 将开始读取启动引导程序（此时 Wi-Fi LED 快速闪烁）并完成启动引导程序的升级（此时 Wi-Fi LED 慢速闪烁），这个过程大约持续 2 秒钟。

4.6.3 Wi-Fi 复位

按住 Wi-Fi 复位按钮至少 5 秒并释放，设备将进入 AP 模式。

4.6.4 恢复出厂设置

这一小节介绍如何对 LinkIt Smart 7688 开发板恢复出厂设置。请谨慎操作，恢复出厂设置会删除所有的用户数据，有两种方法恢复出厂设置：

- 使用 Web UI

登录 Web UI，详细信息请参考 4.5.3 小节“连接到 Web UI”，在 Factory reset 下方，点击 **RESET**，如图 28 所示。

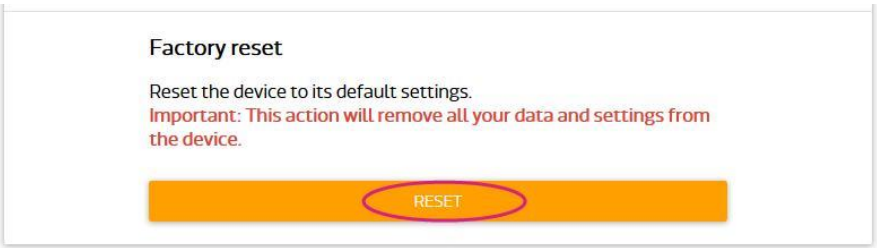


图 28 使用 LinkIt Smart 7688 Web UI 恢复出厂设置

- 使用 Wi-Fi 复位按钮

LinkIt Smart 7688 启动之后，按下 **Wi-Fi 复位按钮**并保持 20 秒后释放。Wi-Fi 复位按钮的位置见

图 1 与图 7。LinkIt Smart 7688 开发板会重启到默认设置，擦除所有用户数据，所以请谨慎操作。

4.6.5 将 LinkIt Smart 7688 连接到 Wi-Fi 热点以访问 Internet

这一小节介绍如何将 LinkIt Smart 7688 开发板连接到一个 Wi-Fi 网络热点以访问 Internet，要想连接到 Wi-Fi 热点，LinkIt Smart 7688 要设置为 STA 模式，详细信息可参阅第 3.4 节“网络环境”。（译者注：意思就是如何把 7688 的 AP 模式切换为 STA 模式，并连接路由器联网）

4.6.5.1 使用 Web UI

打开浏览器并输入链接 mylinkit.local，如果你之前没有登录过的话，请参阅第 4.5.3 小节“连接到 Web UI”。

1) 如图 29 所示，点击由右上角的 **Network** 按钮。

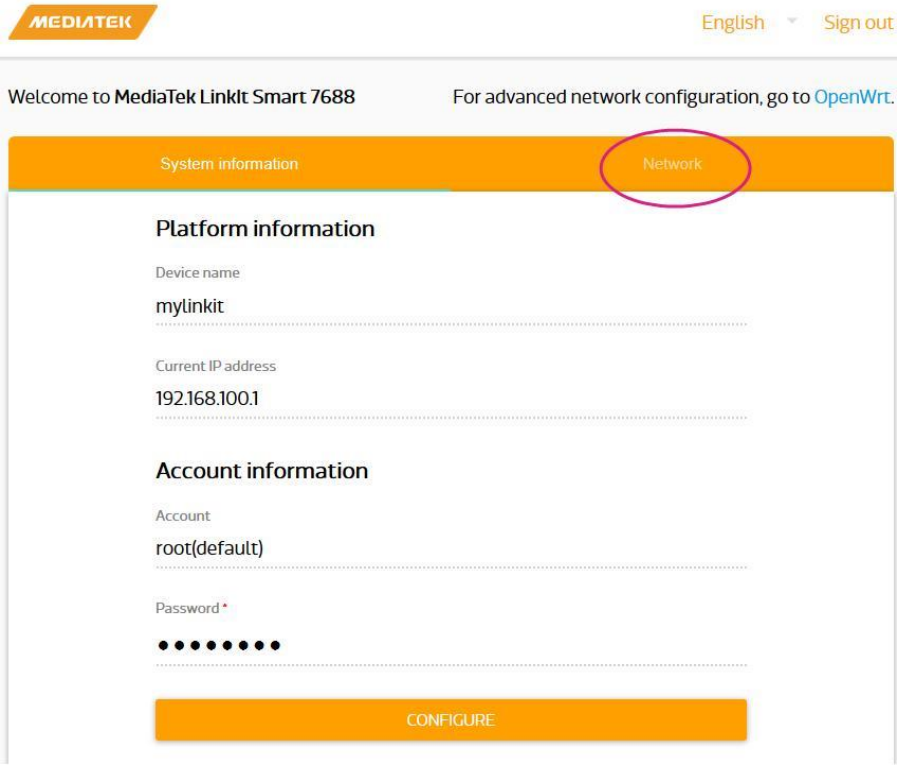


图 29 使用 Web UI 改变网络设置

2) 选择 **Station mode**，点击 **REFRESH** 或右边的**向下箭头**查找可连接的网络热点，选择好网络热点后，按要求输入密码。然后点击 **CONFIGUER & RESTART** 完成。如图 30 所示。

注意：如果您输入的 Wi-Fi 密码不正确，您可以按下 Wi-Fi 按钮至少 5 秒，把开发板复位到 AP 模式，再重新设置为 STA 模式。

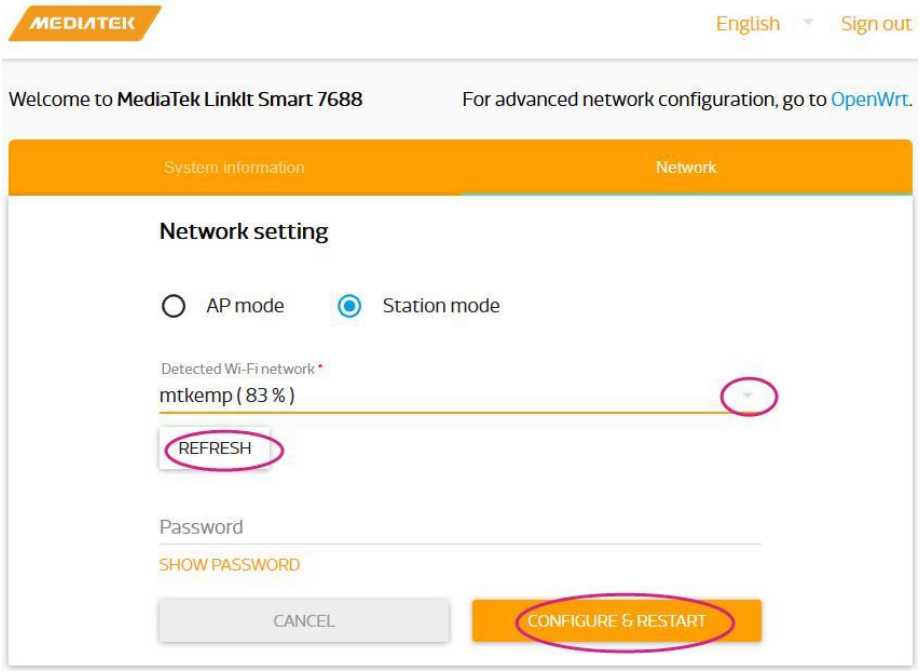


图 30 使用 Web UI 设置为 STA 模式

3) 接着会弹出一个消息窗口，告诉你开发板正在连接到你所选择的 Wi-Fi 热点，然后重新加载 Web UI 并登录。
4) 登录之后，查看 Wi-Fi LED，它现在应该每 2 秒闪烁一次，这表明开发板已经成功进入 STA 模式。
5) 要想把 LinkIt Smart 7688 改回 AP 模式，请按下 Wi-Fi 复位按钮并保持至少 5 秒后释放，详细内容请参阅 4.5.3 小节“连接到 Web UI”。重新加载 Web UI 并登录，当您看到 Wi-Fi LED 3 次短闪烁之后，表明开发板已进入 AP 模式。

   您也可以使用系统控制台将 LinkIt Smart 7688 改为 AP 模式，详细内容请参阅 4.6.5.2 小节“使用系统控制台”

4.6.5.2 使用系统控制台

要想让 LinkIt Smart 7688 开发板访问 Internet，您要把开发板设置成 STA 模式，然后把它连接到一个已经联网的 Wi-Fi 热点上。

当 LinkIt Smart 7688 运行在 STA 模式时，您可以使用 opkg 命令，从 OpenWrt 安装软件到开发板上。

这一小节介绍如何使用集中式配置信息管理接口（UCI）在系统控制台中将 LinkIt Smart 7688 改为 STA 模式。（译者注：UCI 是 Unified Configuration Interfcae 的缩写，详细信息请自行百度）

1) 确保主机电脑连接到了一个 Wi-Fi 热点。（译者注：这个 Wi-Fi 热点指的是最终要让 7688 连接的热点）

2) 从这个 Wi-Fi 热点获取以下信息：

- SSID 加密方式，参考表 14：

加密方式	UCI 命令字符串
Open network	none
WPA-PSK	psk
WPA2-PSK	psk2
WEP	不支持

表 14 Wi-Fi 热点加密方式

有关加密方式的更多内容，请查看这里。

- 密码

3) 打开系统控制台并输入以下指令将 LinkIt Smart 7688 改为 STA 模式：

假设使用的是如下的热点：

- SSID：**MyAP**
- 加密方式：**WPA2-PSK**（psk2）
- 密码：**12345678**

```
# uci set wireless.sta.ssid=MyAP
# uci set wireless.sta.encryption=psk2
# uci set wireless.sta.key=12345678
# uci set wireless.sta.disabled=0
# uci commit
# wifi
```

4) 尝试在 terminal 窗口中键入以下命令来检查是否已成功建立了网络连接。

```
# ping –c 5 www.mediatek.com
```

如果屏幕显示类似图 31 所示，恭喜，这表示您已经成功连接到 Wi-Fi 热点。此时，Wi-Fi LED 会每 2 秒闪烁一次，表明现在正运行在 STA 模式。

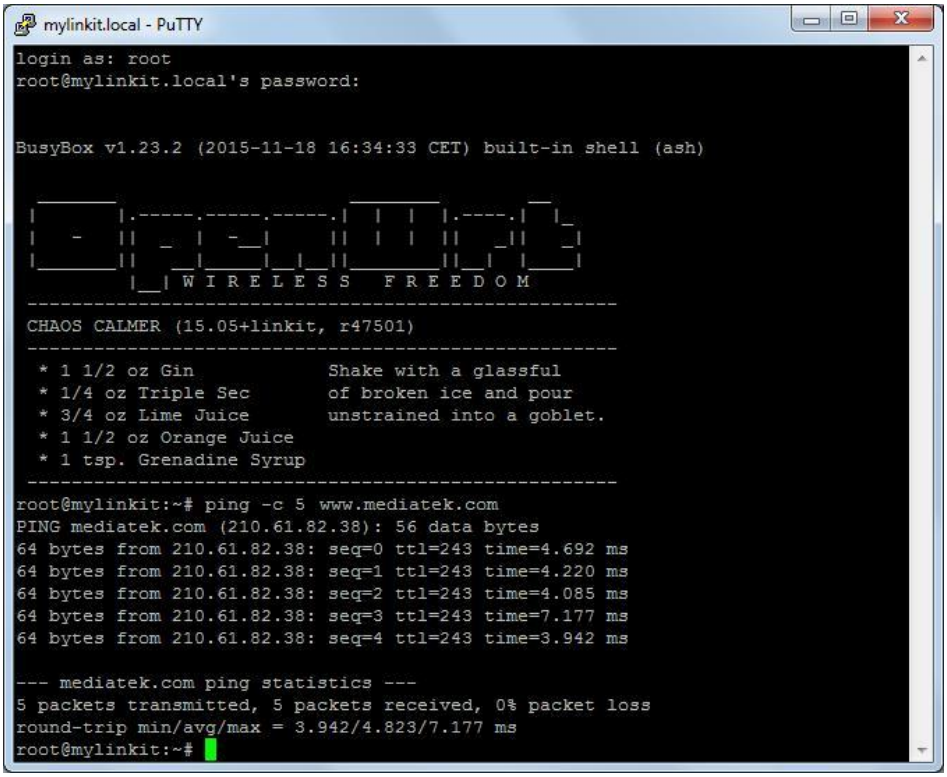


图 31 LinkIt Smart 7688 STA 模式连接 Wi-Fi 热点

4.6.6 通过 Web UI 查看系统信息

您可以通过任何浏览器查看 LinkIt Smart 7688 的系统信息，当您把 LinkIt Smart 7688 连接到一个网络后，打开浏览器，在地址栏输入 **http://mylinkit.local** 并登录，点击 System information 栏下的 **Configure** 按钮，将会显示如下信息，如图 32 所示。

- 设备名称
- 当前 IP 地址
- 账户
- 密码

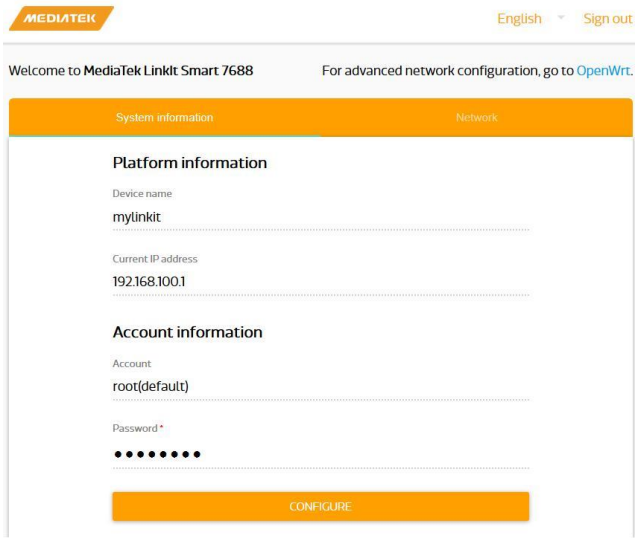


图 32 LinkIt Smart 7688 Web 上的系统信息

4.6.7 访问 U 盘和 SD 卡

LinkIt Smart 7688 插入的 U 盘和 SD 卡，可以在**/Media/SD***或**/Media/USB***目录下访问（所显示的设备名称因您使用的驱动器数量、U 盘或 SD 卡上可用的分区数量而有所不同）。

以下命令可用来查看 U 盘和 SD 卡上的内容：

- 假设 U 盘名称为 USB-A1：

```
> ls /Media/USB-A1
```

- 假设 SD 卡名称为 SD-P1：

```
> ls /Media/SD-P1
```

4.7 文件编辑器和传输

作为开发人员，您当然希望自由地使用任何文本编辑器编写程序代码。您可以在系统控制台中启动内置的 vim 编辑器，直接在 LinkIt Smart 7688 中编辑源代码文件，或者通过远程计算机使用其他编辑器，但是在开发过程中，对于程序执行这类工作，一般来说，会通过在 LinkIt Smart 7688 和远程计算机之间传输文件来完成。LinkIt Smart 7688 的默认系统映像提供了多种通过 Wi-Fi 传输文件的工具。如表 15 所示（译者注：Windows 和 OS X 自带的资源管理器均支持 SMB 协议的文件传输）。

文件传输服务	协议	主机平台	工具
SCP（Secure Copy）	SCP	Windows	WinSCP
		OS X	scp command
		Linux	scp command
Samba	SMB/CIFS	Windows	内置的文件资源管理器支持
		OS X	内置的 Finder 支持
		Linux	smbclient

表 15 文件传输工具

4.7.1 在 Windows 系统下使用 SCP 复制文件

使用 SCP 工具软件是通过 SCP 协议传输文件的最可靠的方法。您可以自行下载支持 SCP 协议的工具软件。本指南使用 WinSCP。它提供 GUI 和命令行两种界面：

a) 要使用 GUI 界面，请启动 WinSCP，当您看到一个如图 33 所示的警告窗口时，点击 **Yes**。

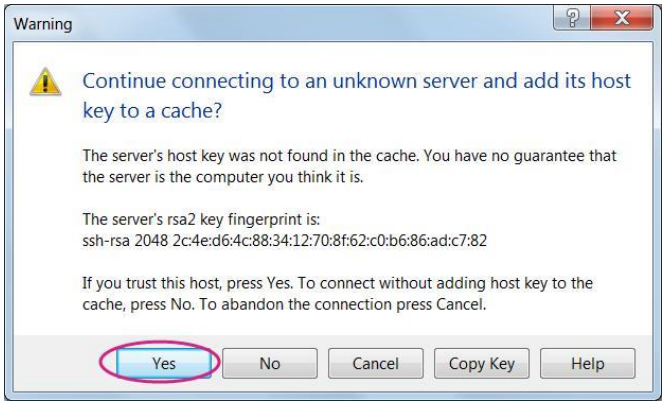


图 33 SCP 安全警告

b) 在 File protocol 栏选择 SCP，在 Host name 栏输入 **mylinkit.local**，在 User name 栏输入 **root**，再输入您在 Web UI 中设置的密码，点击登录，如图 34 所示。

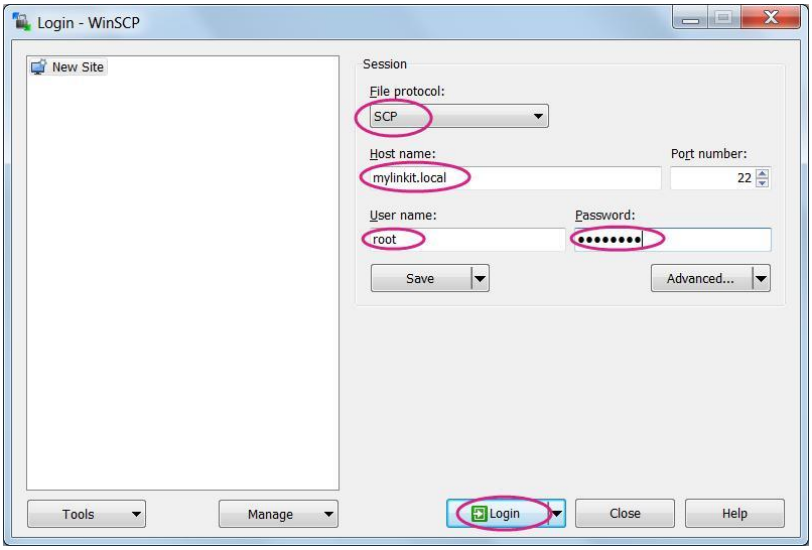


图 34 WinSCP 登录窗口

c) 在左侧窗格（您的计算机）上找到要传输的文件，并将其拖动到右侧，如图 35 所示。

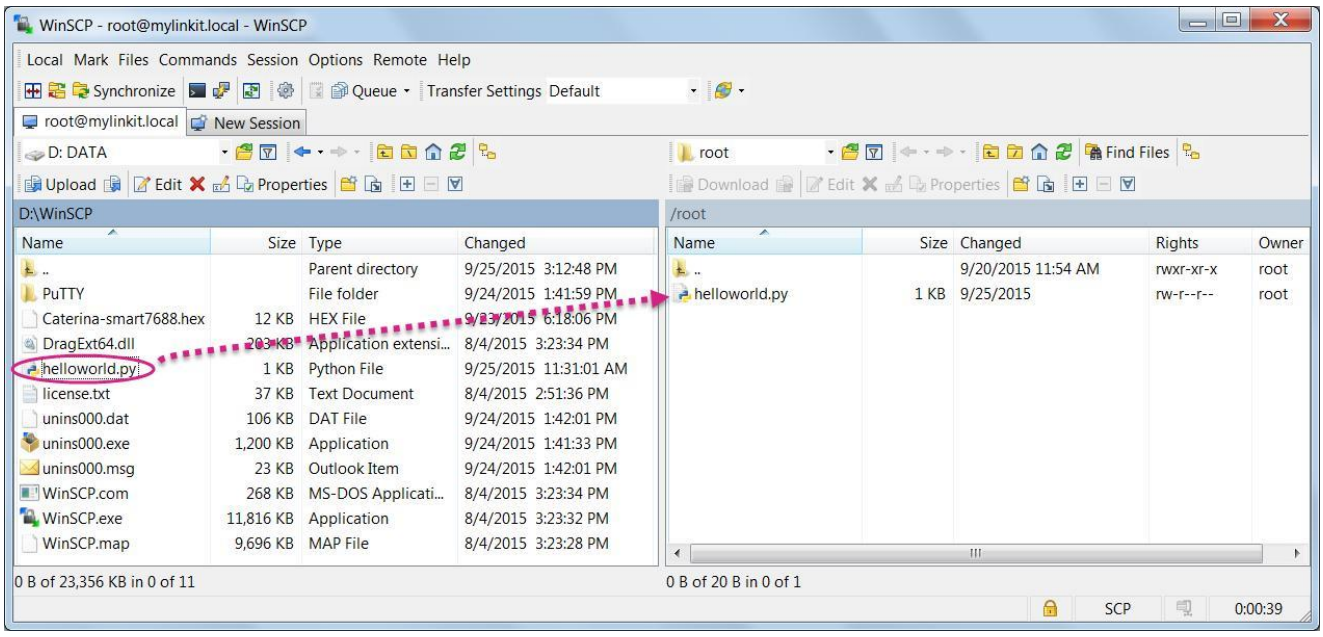


图 35 使用 WinSCP 传输文件

d) 拖动文件后，将出现一个上传窗口，如图 36 所示，要求您确认上传。单击 "确定"。

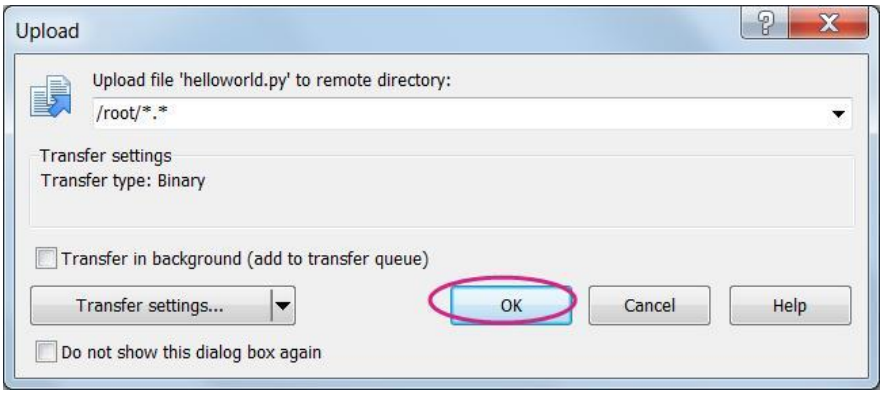


图 36 文件传输确认

查看 LinkIt Smart 7688 系统控制台，helloworld.py 文件应该已经上传到那里。

或者，您也可以使用命令行界面，在 Windows 命令行控制台中输入以下命令：

```
winscp.com -hostkey=”*” scp://root@myLinkt.local helloworld.py
```

请注意，默认情况下，它要求用户明确地指定远程服务器的主机密钥，因此我们需要通过添加-hostkey="*"这个参数明确地允许未知主机密钥。

4.7.2 在 OS X 系统下使用 SCP 复制文件

在 OS X 中，SCP 命令行工具应该已经安装好可以使用。如果未安装，可以使用软件包管理器（如 MacPorts 或 Homebrew）安装。要使用 SCP，需打开 Terminal 并输入以下命令：

```
scp ./helloworld root@mylinkit.local:/example/helloworld
```

在上面的例子中，一个名为 helloworld 的文件从当前目录被复制到 LinkIt Smart 7688 的/example/helloworld 路径中。SCP 工具将指导您输入 root 帐户的密码（4.5.1 小节“系统配置工具”中所设置的账户和密码）。

4.7.3 为 LinkIt Smart 7688 安装 Samba 服务

Samba 是一个文件传输工具，它内置在了 LinkIt Smart 7688 中，它提供一个共享文件夹**/Media** 用作文件传输。下面的例子您将学习如何使用 UCI 命令把共享文件夹改为**/IoT** 并为该文件夹设置适当的访问权限。

1) 把共享文件夹路径改为**/IoT**。在 LinkIt Smart 7688 控制台中输入以下命令。

```
# uci set samba.media.path=’/IoT’
```

2) 把共享文件夹命名为 MySharedFloder，例如：

```
# uci set samba.media.name='MySharedFolder'
```

3) 把共享文件夹的访问权限改为可读可写，例如：

```
# chmod o+rwx /IoT
```

4) 保存并重启 LinkIt Smart 7688 开发平台，例如：

```
# uci commit
# reboot
```

5) LinkIt Smart 7688 重启并将其连接到与您的计算机相同的本地网络后，您可以开始使用 samba 传输工具。接下来，根据您使用的操作系统选择对应的步骤。

4.7.4 Windows 系统下是使用 Samba 复制文件

打开资源管理器并输入\\mylinkit.local，看到 MySharedFolder。再打开另一个资源管理器，拖动文件到 MySharedFolder，如图 37 所示。

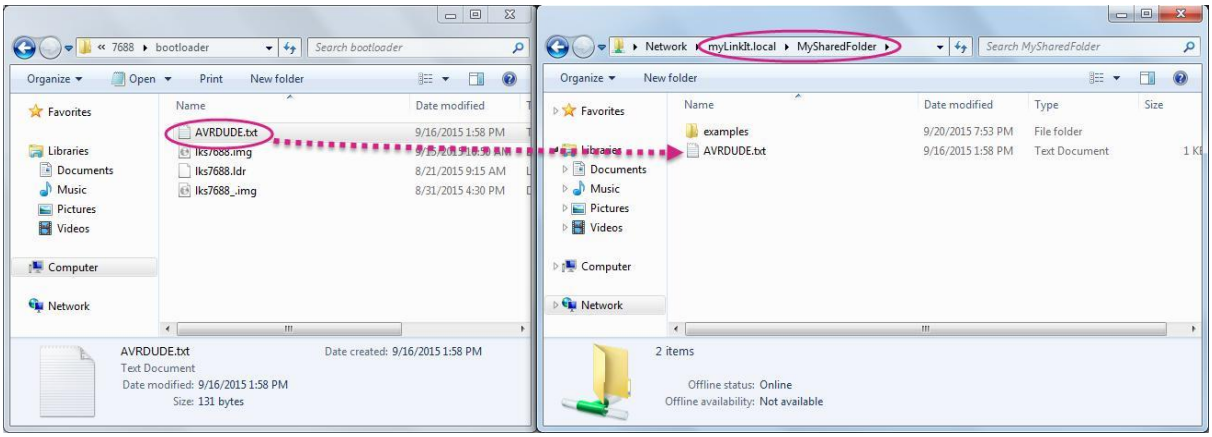


图 37 Windows 下使用 Samba 传输文件

4.7.5 OS X 系统下使用 Samba 复制文件

Mac 系统下使用 Samba 传输文件的步骤如下：

1) 打开 Finder，点击顶部菜单的 Go>Connect to server，如图 38 所示。

图 38 使用 Finder 连接 LinkIt Smart 7688

2) 在 server address 栏，输入 smb://mylinkit.local 并点击连接，如图 39 所示。

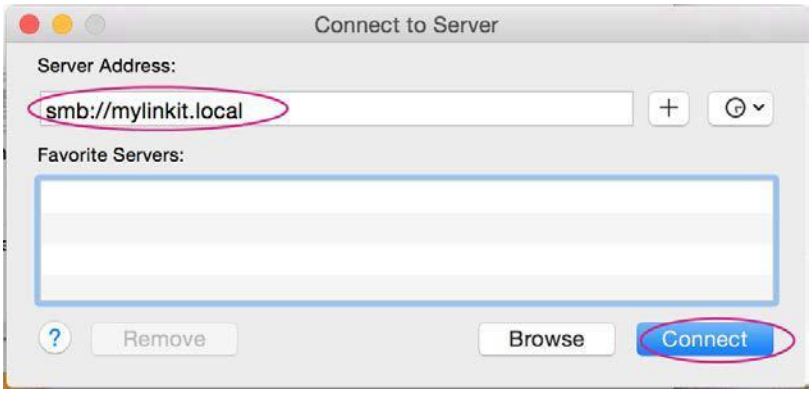


图 39 Mac 系统下连接 mylinkit.local 服务器

3) 如图 40 所示，以 guest 用户连接。

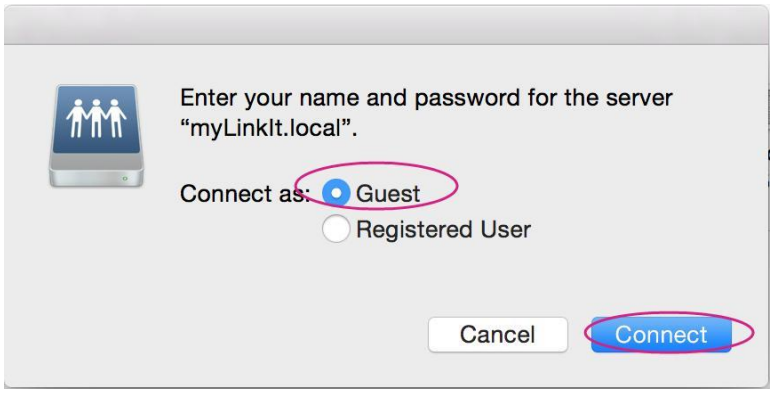


图 40 Mac 系统下以 guest 用户连接到 mylinkit.local

4) 查看 Finder，你将能看到 MySharedFolder 如图 41 所示。现在您可以传输文件到这个文件夹了。

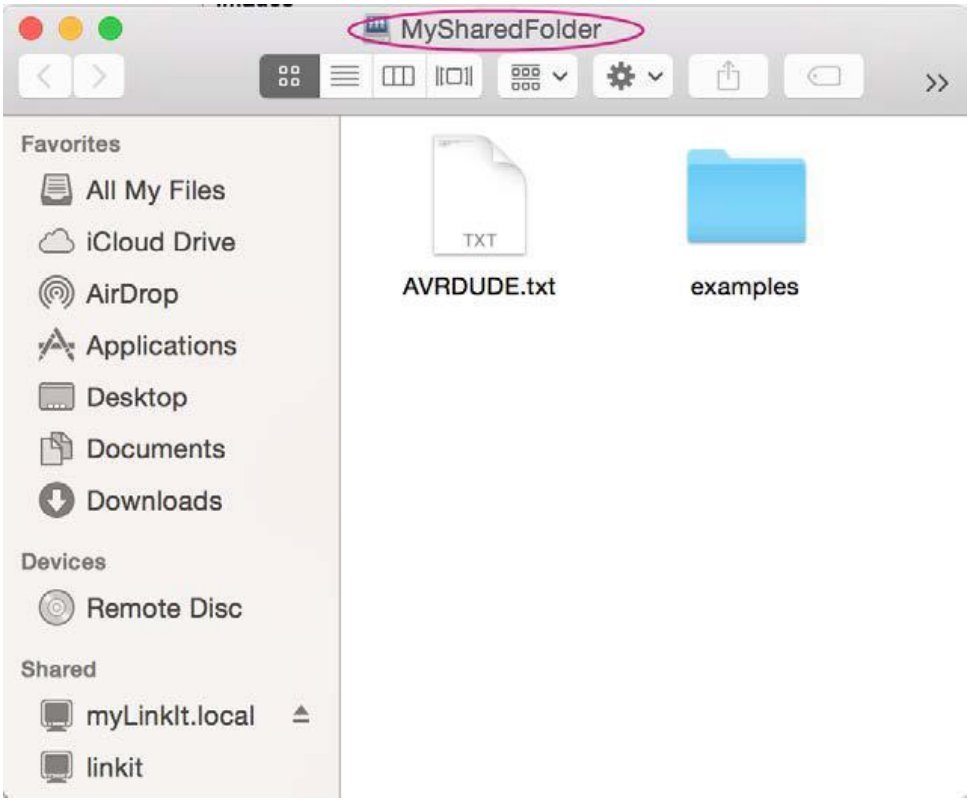


图 41 Mac Finder 中的 MyShareFloder

5.LinkIt Smart 7688 的外设编程

这一章介绍了如何对 LinkIt Smart 7688 的外设传感器进行编程。

硬件方面，在 LinkIt Smart 7688 开发板上的 MediaTek MT688AN 芯片负责处理所有的 Wi-Fi 通讯、USB 设备控制、SD 卡访问和传感器连接，如图 42 所示。同时，给开发者提供相关的软件栈用来访问那些连接到 MT7688AN 上的传感器。

UPM 是用 libmraa 编写的传感器驱动库，它为 Python、Node.js 和 C 语言提供了 API 封装。所以对于开发者来说，很容易通过 UPM 和自己喜欢的编程语言访问外设传感器和模块。LinkIt Smart 7688 有内置的 UPM 支持，UPM 支持的详细的传感器列表，请参阅 [UPM project page](#)。

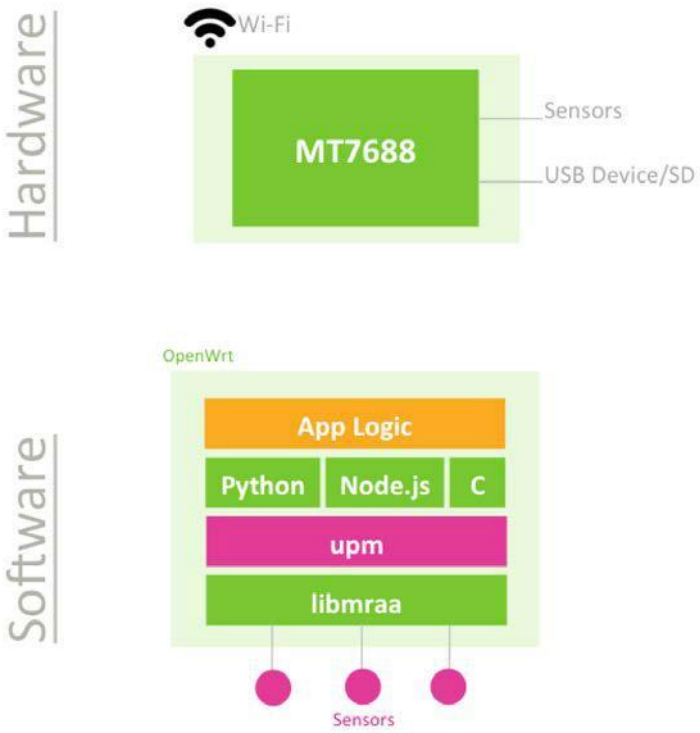


图 42 LinkIt Smart 7688 软件架构

5.1 如何使用 MRAA 访问 LinkIt Smart 7688 的外设

Libmraa 是 LinkIt Smart 7688 外设接口的 C/C++库。它在系统映像中已经预先安装，它支持 Python、Javascript 和 Node.js 语言。这一节将介绍如何通过 libmraa 控制 LinkIt Smart 7688 的 GPIO 等外设。

5.1.1 安装 MRAA

Libmraa 已经在系统映像中预先安装，所以您无须再安装一次，然而，如果你想更新或升级这个库，请参考 [libmraa](#)获取更多信息。

要在 Python 中使用 libmraa，首先要导入这个库，下面例子是导入 libmraa 并输出库版本。

```
import mraa
print (mraa.getVersion())
```

有关 libmra 的 API 信息，请参阅这里。

5.1.2 libmraa 的基本概述

大多数的硬件模块（如 GPIO、UART、SPI 和 PWM）都表示为一个由 mraa 创建的对象。这些模块被初始化在特定的引脚上。LinkIt Smart 7688 开发板 libmraa 库中的引脚号和数据手册、Linux GPIO 子系统里面的 GPIO 号是一样的。

下面的例子是在 GPIO2 上创建一个 GPIO 对象，对应的是 LinkIt Smart 7688 的 P10 引脚，在数据手册中，它同时也是 IS2_WS 引脚，如表 16 所示。开发板上的丝印和 GPIO 号的对应映射关系请参考 LinkIt Smart 7688 引脚图。在 Linux 系统上对应这个 P10 引脚的是/sys/class/gpio/2 目录。

GPIO	引脚映射
LinkIt Smart 7688	P10
Datasheet	I2S_WS

表 16 LinkIt Smart 7688 GPIO 引脚映射

```
import mraa
pin = mraa.Gpio( 2 )     # Initialize GPIO2 (P10 on LinkIt Smart 7688 board)
```

5.1.3 Libmraa 主要功能

这一小节介绍了 libmraa 的主要模块，如何使用 GPIOs 以及 LinkIt Smart 7688 上其他可用的接口。

- **GPIO 和中断**

  要控制 GPIO 引脚，需把引脚初始化为 GPIO 引脚，并设置它的模式。最简单的是**输出**模式——设置引脚为高电平或低电平，用以使能外部开关或形成信号流。

```
import mraa
pin = mraa.Gpio(2)     # Initialize GPIO2 (P10 on LinkIt Smart 7688 board)
pin.dir (mraa.DIR_OUT) # set as OUTPUT pin
```

然后，调用 pin.write(0)设置引脚输出低电平，或者调用 pinwrite(1)设置引脚输出高电平，为了使 Wi-Fi LED 周期性地闪烁，需设置 GPIO44（WLED_N）为 GPIO 模式并执行以下代码：

```
import mraa
import time

# Refer to the pinout digram for the GPIO number to silk print mapping
# in this example the number 44 maps to Wi-Fi LED.
pin = mraa.Gpio(44)
pin.dir (mraa.DIR_OUT)
while True:
  pin.write(1)
  time.sleep(1)
  pin.write(0)
  time.sleep(1)
```

另一种 GPIO 模式是输入模式。这种模式下它从引脚读取数字信号输入并将其解释为 1 和 0。以下例子将不断地输出从开发板 P10 引脚读取的值。你可以短接 3V3 和 P10 来观察值的变化。

```
import mraa
import time
# Refer to the pinout digram for the GPIO number to silk print mapping
# in this example the number 2 maps to P10 on LinkIt Smart 7688 board
pin = mraa.Gpio(2)
pin.dir (mraa.DIR_IN)
while True:
  print "P10 state:", pin.read()
  time.sleep(0.3)
```

最后，您还可以将中断服务例程安置到引脚上，当输入引脚 P10 (GPIO2) 的值发生变化时调用。使用 isr API 函数并指定触发类型和回调函数即可。请注意，这个功能会在另一个线程中运行。

```
import mraa
import time
def callback(userdata):
    print "interrupt triggered with userdata=", userdata
pin = mraa.Gpio(2)
pin.dir (mraa.DIR_IN)
pin.isr(mraa.EDGE_BOTH,callback,None)
while (True):
  time.sleep(1)
  # simply wait for interrupt
```

- **PWM**

使用 PWM 模块可以生产脉冲宽度调制信号。这对于控制一些伺服电机之类的执行部件是非常有用的。要使用 PWM 模块，首先要在某个引脚上进行初始化，需要注意的是，LinkIt Smart 7688 开发板只有 GPIO18、GPIO19、GPIO20、GPIO21 支持 PWM 功能。控制 PWM 模式需要以下几个参数：

- 周期（Period）

  定义调制的载波频率。它由 period、period_ms 和 period_us 这几个 API 指定。

- 占空比或脉冲宽度

  这两个参数是相互关联的，只需要设置其中一个即可。占空比由 write 函数指定。它的值介于 0.0 到 1.0 之间，其中 0.0 是 0%的占空比，1.0 是 100%的占空比。脉冲宽度根据单位定义了不同的函数：以正脉冲持续的时间为单位。它由 pulsewidth、pulsewidth_ms 和 pulsewidth_us 这几个 API 定义的。

```
import  mraa
pin = mraa.Pwm(18)   # initialize on GPIO18 (pin P26)
pin.period_ms(2)     # set PWM frequency to 500Hz (2ms period)
pin.enable(True)     # enable PWM output
pin.write(0.25)      # set duty cycle to 25%
```

- **I2C**

I2C（iner-Integrated Circuit）是外设中广泛使用的通讯协议。他有两个信号引脚（通常称为 SDA 和 SCL）。LinkIt Smart 7688 有一路 I2C 接口，分别以 GPIO4（P21）和 GPIO5（P20）作为 SCL 和 SDA 引脚。

I2Cs 的初始化方式与 GPIO 模块稍有不同，它不使用引脚号，而是根据设备索引来初始化 I2Cs。因为 LinkIt Smart 7688 内只有 1 个 I2C 主机，所以您可以简单地把索引值设为 0——始终使用 GPIO4 和 GPIO5 引脚。

```
import mraa
i2c = mraa.I2c(0)
```

I2C 能够将多个从机连接到单个 I2C 主机上。每个从机都由 7 位地址标识。下面的示例通过从寄存器中读取设备 ID 来扫描连接到 LinkIt Smart 7688 的 Seeed Studio 3 轴数字加速度传感器。

```
import mraa
i2c = mraa.I2c(0)
# Grove - 3-Axis Digital Accelerometer(+-16g)
# is a ADXL345 configured to I2C address 0x53.
i2c.address(0x53)
# The device ID should be
if 0xE5 == i2c.readReg(0x00):
    print "Grove - 3-Axis Digital Accelerometer found on I2C Bus"
  else:
    print "Grove - 3-Axis Digital Accelerometer not found"
```

关于 I2C API 的详细信息，请查看这里。

要驱动 I2C 设备，您需要操作多个 I2C 序列。LinkIt Smart 7688 预先安装了 libUPM，这是一套基于 libmraa 的驱动程序库。请参阅这里的 Seeed Studio 3 轴数字加速度传感器驱动例程。

- **SPI**

SPI（Serial Peripheral Inerface）也可以用来控制外设，LinkIt Smart 7688 有四个引脚：SPI_MOSI(P22)、SPI_MISO(P23)、SPI_CLK(P24)和 SPI_CS1(P25)。

重点注意的是 SPI 接口亦被用来与内部的 Flash 存储器通讯，因此，开发者只能通过 SPI 模块来使用 SPI 功能，避免把 SPI 引脚当作普通 GPIO 使用。否则，Flash 存储器将无法正常工作。

libmraa 中 SPI 模块初始化使用设备索引，而不使用引脚号。

```
import mraa
spi = mraa.Spi(0)
```

SPI 协议允许对信号模式进行灵活的配置。根据外设备的不同，您可能需要使用诸如 bitPerWord 和 mode 等 API 来配置模式。有关 API 详细信息，请参阅这里。

5.1.4 基于 mraa 和 Python 语言的 LinkIt Smart 7688 LED 闪烁例程

这个例子讲述如何使用 mraa 库和 Python 语言使 LinkIt Smart 7688 上的 Wi-Fi LED 闪烁。

1) 使用 LinkIt Smart 7688 SSH 系统控制台，创建一个名为 app 的目录并进入这个目录，例如：

```
# mkdir app
# cd app
```

2) 在 root@mylinkit:~/app#提示符处，输入以下命令创建一个名为 blink.py 的文件。

```
# vim blink.py
```

3) 按 **i**，在编辑器中插入以下代码，例如：

```
import mraa
import time
# Refer to the pinout digram for the GPIO number to silk print mapping
# in this example the number 44 maps to Wi- Fi LED
led = mraa.Gpio(44)
led.dir(mraa.DIR_OUT)  # set direction to output

while True:
  led.write( 1 )      # turn on LED
  time.sleep(1 )
  led.write( 0 )      # turn off LED
  time.sleep(1 )
```

4) 按 **Esc** 键和**:wq!**保存 blink.py 文件。

5) 现在你可以运行这个 Python 程序了，在 app 目录中，输入以下命令运行程序：

```
# python ./blink.py
```

LinkIt Smart 7688 上的 Wi-Fi LED 将每两秒闪烁一次。

按**<Ctrl>C** 即可结束这个 Python 程序。

5.1.5 基于 mraa 和 Node.js 语言的 LinkIt Smart 7688 LED 闪烁例程

这个例子讲述如何使用 mraa 库和 Node.js 语言使 LinkIt Smart 7688 上的 Wi-Fi LED 闪烁。

1) 使用 LinkIt Smart 7688 SSH 系统控制台，创建一个名为 app 的目录并进入这个目录，例如：

```
# mkdir app
# cd app
```

2) 使用 NPM 初始化文件，NPM 是 Node.js 的包管理器，它预先安装在了 LinkIt Smart 7688 中。以防万一，您也可以从这里安装。

```
#npm init
```

3) 大约 10 秒之后，会有提示要你输入多个参数，例如 app 名称、版本号、描述等等，如图 43 所示。这是 Node.js 应用程序的建立过程，在本教程中，对所有问题均输入 Enter 即可。

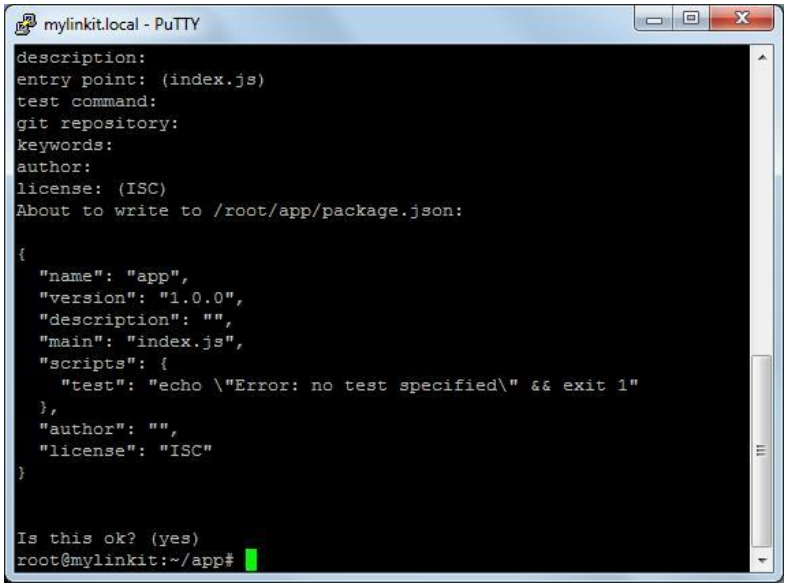


图 43 建立 Node.js 应用程序的提示

4) 在 root@mylinkit:~/app#提示符处，输入以下命令创建一个名为 app.js 的文件。

```
# vim app.js
```

5) 按 **i**，在编辑器中插入以下代码，例如：

```
var m = require('mraa');
var ledState = true;
var myLed = new m.Gpio(44);   // GPIO44 is the Wi- Fi LED

myLed.dir(m.DIR_OUT);

function periodicActivity() {
  myLed.write(ledState ? 1 : 0);
  ledState = !ledState;
  setTimeout(periodicActivity,  1000);
}
periodicActivity();
```

按 **Esc** 键和**:wq!**保存 app.js 文件。

6) 现在你可以运行这个 Node.js 程序了，在 app 目录中，输入以下命令运行程序：

```
# node app
```

LinkIt Smart 7688 上的 Wi-Fi LED 将每两秒闪烁一次。按**<Ctrl>C** 即可结束这个程序。

5.2 如何使用 UPM 访问传感器和外设

UPM 是一个基于 libmraa APIs 的开源传感器和外设驱动库，可应用在诸如 I2C 加速度传感器和很多其他流行传感器的驱动上。

LinkIt Smart 7688 系统映像预先安装了 UPM 库，您可以立即开始对传感器进行编程。但是，如果默认的 UPM 库不可用的话，你可以使用 opkg 包管理器对其进行升级。

UPM 和 libmraa 一样支持 C++、Python 和 Node.js 语言。下面以一个例子学习如何使用 UPM 和 Python 读取一个 I2C 加速度传感器的数据——Grove 3 轴数字加速度传感器。

1) 把加速度传感器连接到您的开发板。如果您有转接板的话，您可以直接把它接在 I2C grove 接口上。否则，您只能把传感器的引脚连接到对应开发板上的 GND、3V3、SDA（P20）和 SCL（P21）引脚。
2) 在你的程序中，从 UPM 库导入 pyupm_adxl345 模块，这一步稍后再完成。之所以使用这个模块，是因为 Grove 3 轴数字加速度传感器使用 ADXL345 芯片
3) 输入以下内容创建一个 Python 脚本 adxl.py

```
import pyupm_adxl345 as adxl
import time

device = adxl.Adxl345(0)

while True:
  device.update()
  a = device.getAcceleration()
  print "(x,y,z)=%5.1f, %5.1f, %5.1f" % (a[0], a[1], a[2])
  time.sleep(0.3)
```

在系统控制台输入以下命令，执行这个 Python 脚本

```
# python  adxl.py
```

您将看到加速度数据不断重复的输出。尝试移动一下传感器观察数据的变化。现在您已经可以使用这个传感请来检测加速度了。

要想查询 UPM 库中其他可用的 Python 模块，请查看这里。UPM 库也附带了一些示例供您参考。

6.LinkIt Smart 7688 Duo 的外设编程

这一章介绍如何对连接到 Atmega32U4 微控制器的传感器进行编程，以及如何实现微控制器和 MT7688AN 之间的通讯，如图 44 所示。

有几点值得重点关注：大多数外设，如 I2C、SPI 接口的各类传感器，都是直接连接到 LinkIt Smart 7688 Duo 的微控制器上，同时，这个微控制器已经预装载了支持 Arduino IDE 编程的启动引导程序。因此，以下内容首先会介绍如何建立 Arduino IDE 开发环境，再介绍 MT7688AN 和 Atmaga32U4 的基本通讯方法，然后才介绍一些不同的应用程序开发模式的例子。

图 44 LinkIt Smart 7688 硬件架构

6.1 安装 Arduino IDE

LinkIt Smart 7688 Duo 的 MCU（Atmega32U4）能使用 Arduino IDE 1.6.5 及其硬件支持包进行程序开发，你可以从这里安装 Arduino IDE 1.6.5。

6.2 安装硬件支持包

Board Manager 工具让 Arduino IDE 1.6.5 能支持第三方的集成硬件平台。对于 Arduino IDE 来说，LinkIt 7688 Duo 开发板是个插件，您需要安装对应硬件支持包才能让 Arduino 支持 LinkIt 开发板，具体步骤如下：

1) 在 Arduino IDE 里，在 **File** 菜单中点击 **Perferences**，然后在 **Additional Boards Manager URLs** 框输入 http://download.labs.mediatek.com/package_mtk_linkit_smart_7688_index.json，如图 45 所示。

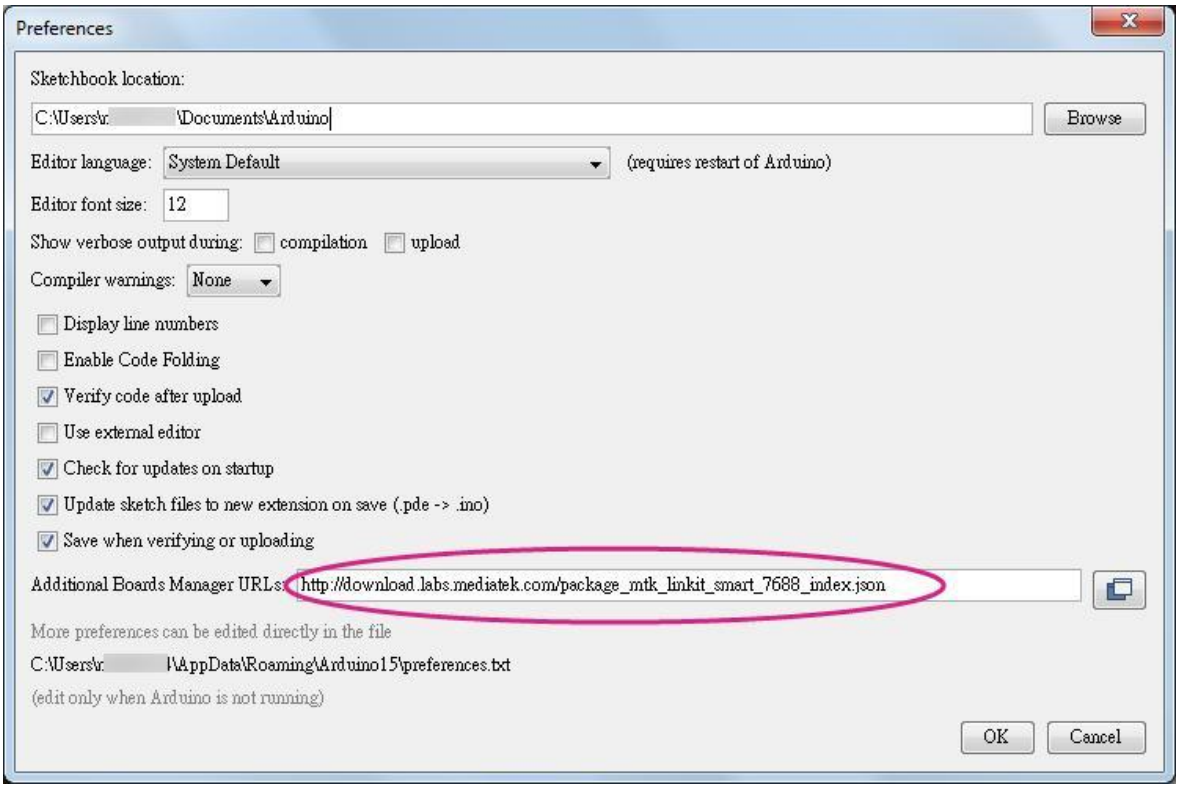


图 45 在 Arduino IDE 中自定义安装硬件支持包时的 LinkIt Smart 7688 Duo 包链接

2) 确保您的电脑已经连接到 Internet。

3) 在 Arduino 的 Toos 菜单中，点击 Board>Boards Manager，如图 46 所示。

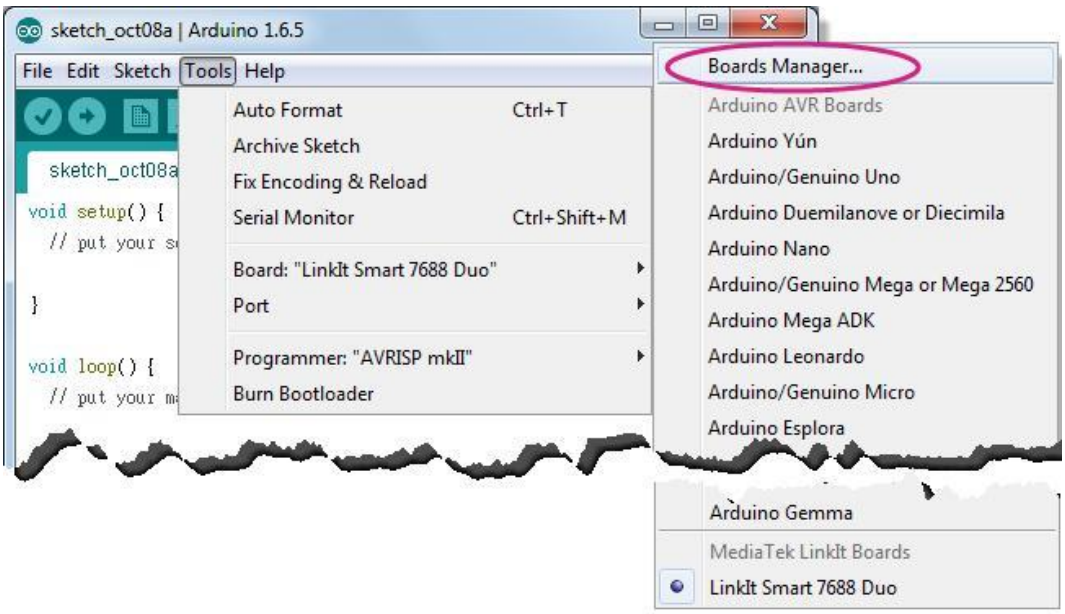
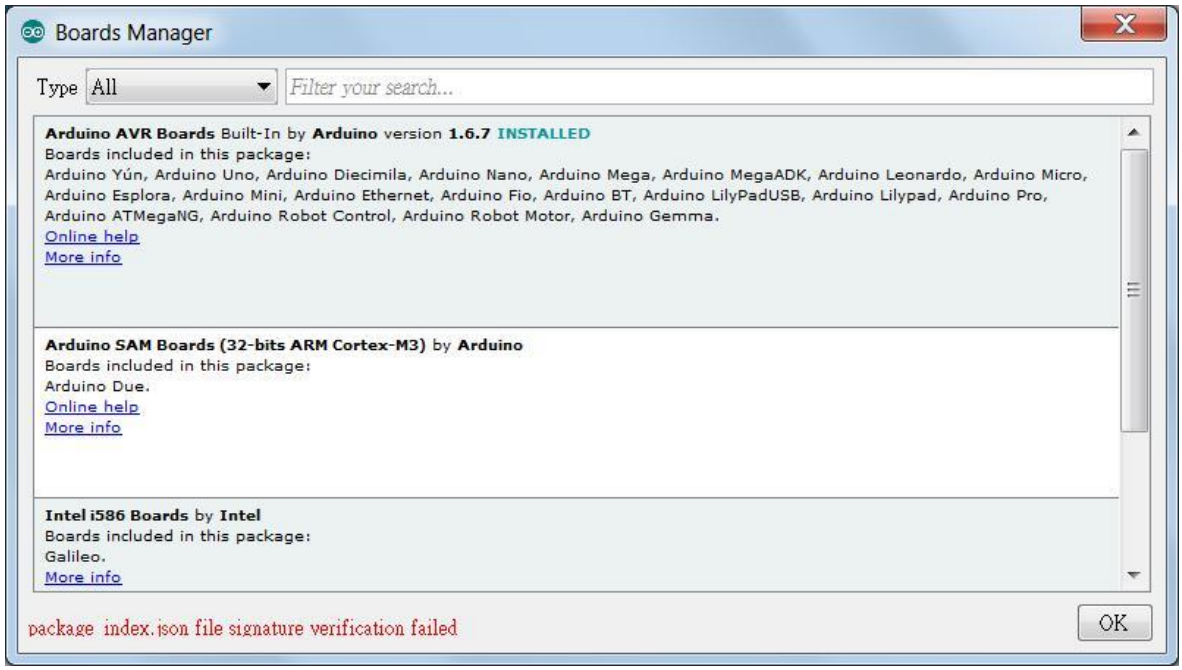


图 46 Arduino IDE 的 Board Manager 菜单

4) LinkIt Smart 7688 开发板支持包会自动开始下载，几秒钟后 Boards Manager 即可完成下载。

如果出现下载错误（如下图），请删除缓存的.json 文件。它所在的目录和 preferences.txt 一样，通过点击 **File** 菜单下的 **Preferences** 可以找到。



LinkIt ONE SDK 包下载错误

5) 接下来在 Boards Manager 的开发板列表里会出现一个 LinkIt Smart 开发板，如图 47 所示。点击“版本”按钮，选择最新版本安装。

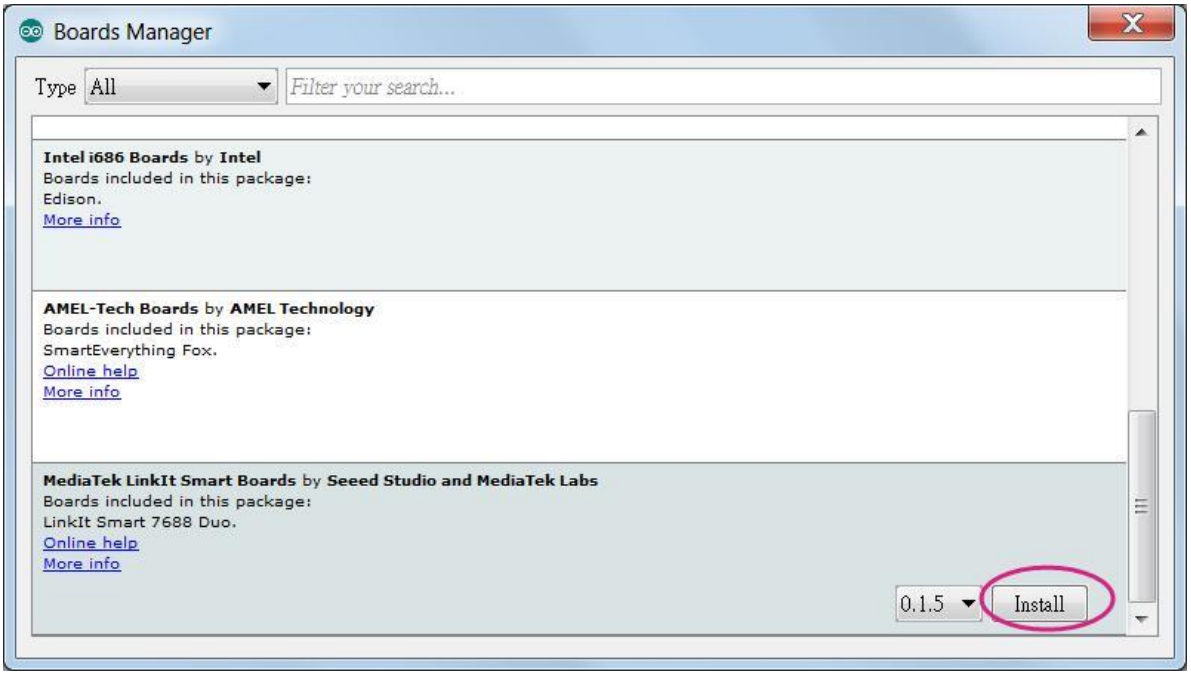


图 47 LinkIt Smart 7688 硬件支持包菜单

6) 如图 48 所示，安装完成。

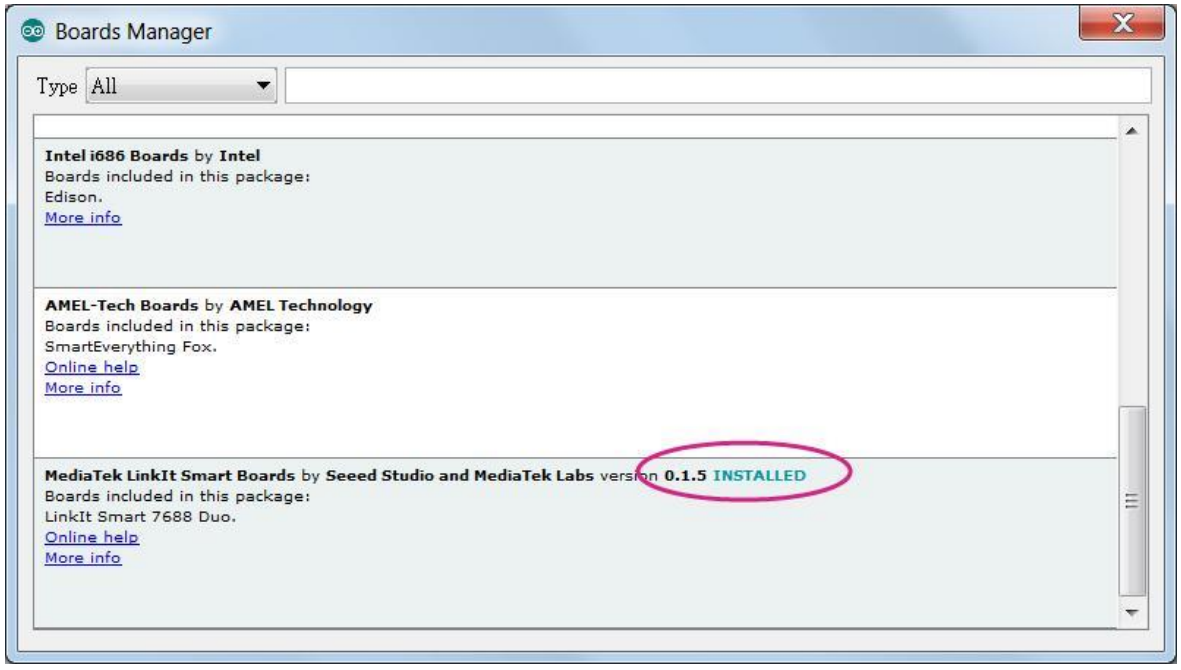
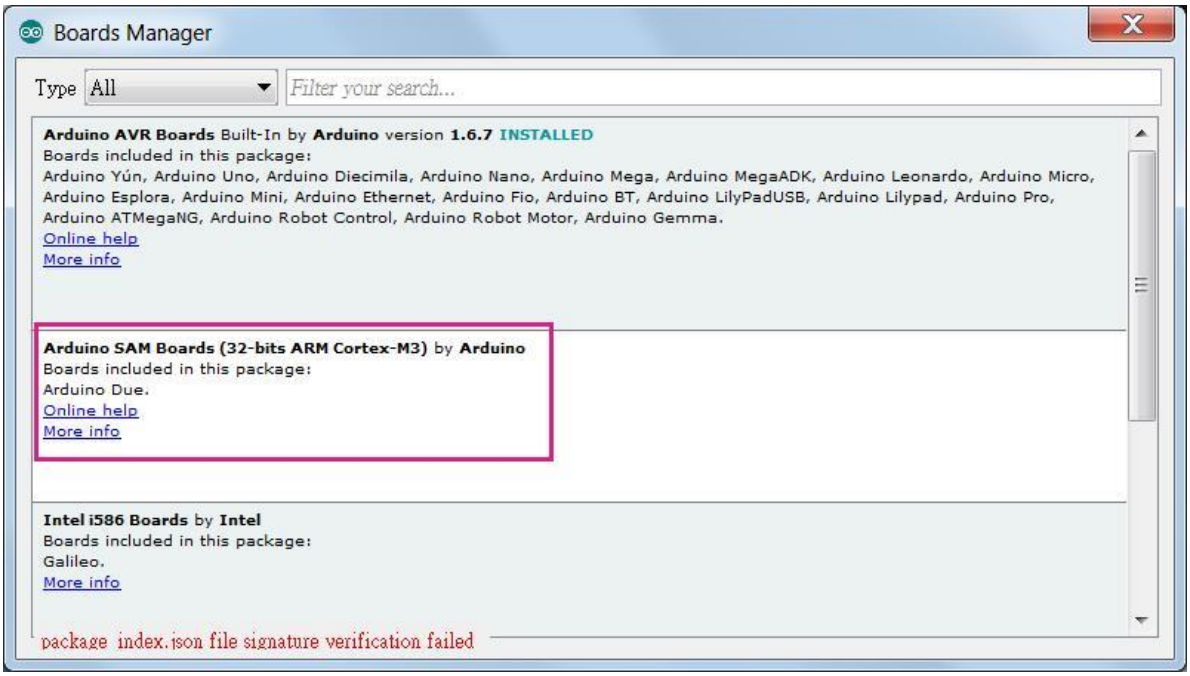


图 48 在 Arduino IDE 中安装 LinkIt Smart 7688 Duo 硬件支持包

如果安装过程中出现问题，请尝试先安装 Arduino SAM 支持包，如下图所示，然后再次安装 LinkIt Smart 7688 Duo 支持包。



删除缓存的.json 文件后的下载 SDK 时出现问题

7) 如图 49 所示，现在您已经成功在 Arduino IDE 中安装 LinkIt Smart 7688 Duo 的硬件支持包。

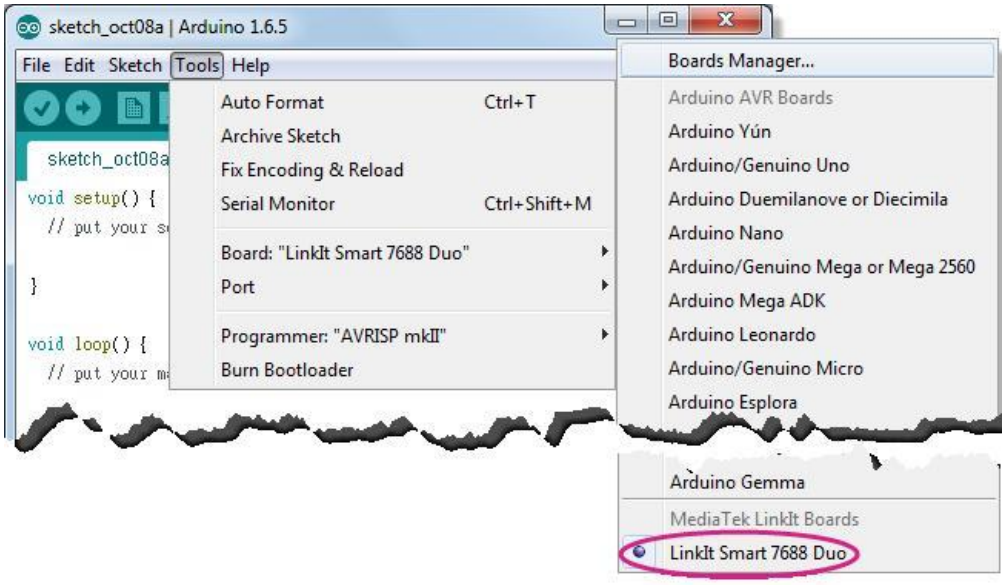


图 49 Arduino IDE 中安装的 LinkIt Smart 7688 Duo

6.3 安装 LinkIt Smart 7688 Duo 串口驱动

安装了硬件支持包之后，把 LinkIt Smart 7688 连接到您的计算机，您将在设备管理器中看到一个 USB 串行接口，它的端口 ID 如下：

- Bootloader COM port: VID=0x0E8D,PID=0xAB00
- Arduino Sketch COM port: VID=0x0E8D,PID=0xAB01

接下来，根据您使用的操作系统，需要为它们安装驱动程序，步骤如下：

- 对于 Windows10 系统，无须安装驱动，但是，为了让系统能识别这块开发板，仍然需要执行以下步骤。将 LinkIt Smart 7688 Duo 连接到您的 Windows 10 计算机，然后在 700 毫秒内快速按 MCU 复位按钮**两次**。这样，系统就会把 LinkIt Smart 7688 Duo 识别为 USB 串行设备(COM5)。在不同的计算机上，这个端口号可能有所不同。您只需要在首次把 LinkIt Smart 7688 连接到您的 Windows 机器时执行此操作。
- 对于 Windwos 8，系统可能会阻止驱动程序安装。请按照此链接了解如何在 Windows8 上禁用驱动程序强制签名。在禁用强制签名后，请按照下面的 Windwos7 的步骤安装驱动程序。
- 对于 Windows7，您可以在以下路径查找串行口的 INF 驱动，您也可以从这里安装。{ARDUINO_IDE_PREFERENCE_LOCATION}Arduino15/packages/ LinkIt/ hardware/avr/0.1.5/ driver/linkit_smart_7688.inf

您可以在 File>Preferences 中找到 Arduino 首选项，如图 50 所示的 preference.txt 文件路径。

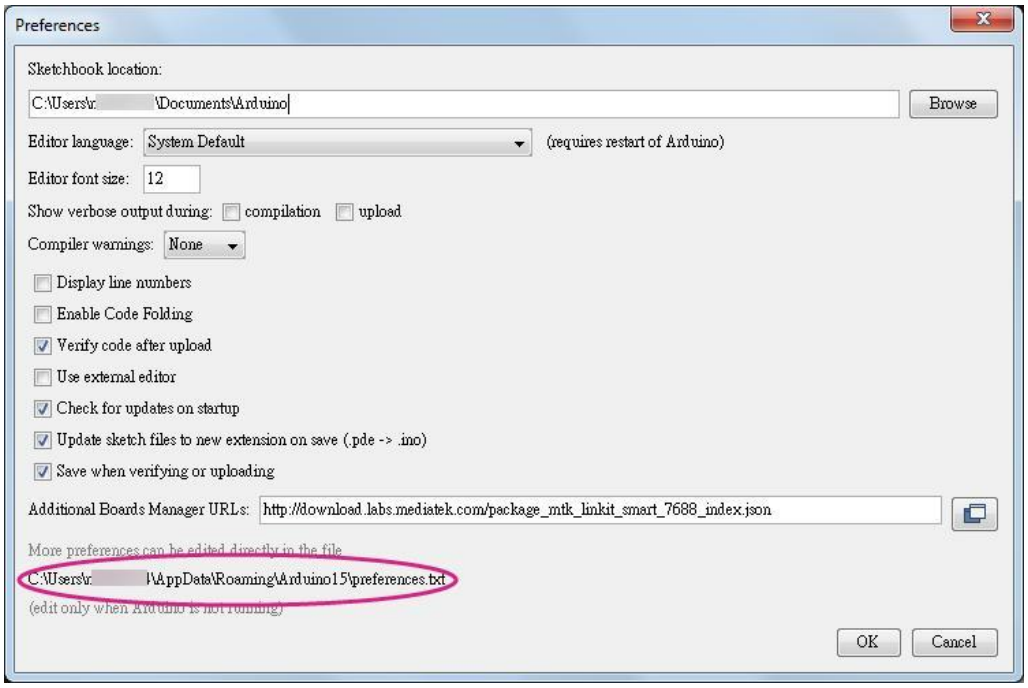


图 50 Arduino Preference 的位置

右击 linkIt_smart_7688.inf 并选择安装，会弹出一个安全警告窗口，点击**始终安装此驱动程序**，完成驱动程序安装，如图 51 所示。

图 51 安装驱动

- 对于 Ubuntu Linux，无须安装驱动程序，LinkIt Smart 7688 Duo 在目录/dev 中，挂载为 ttyusb0，这个数字 0 在不同 Ubuntu 计算机上可能会不一样。
- 对于 OS X，同样不需要安装驱动，LinkIt Smart 7688 Duo 在目录/dev/tty.usbmodem1413 中挂载为一个串行设备，这个数字 1413 在不同 OS X 计算机上可能会不一样。

6.4 LinkIt Smart 7688 的编程模式

在 LinkIt Smart 7688 Duo 开发板中，MT7688AN 负责 Wi-Fi 通信，USB 设备控制和 SD 卡访问。对于传感器和外设连接，为了能实现更好的实时控制响应，使用了 ATmega32U4。MCU 编程控制使用的是 Arduino IDE，如图 52 所示。针对不同的应用情景和编程方法，为构建软件系统，以下列举了

三种方案供参考。

图 52 Smart 7688 Duo 硬件架构

方案 1：通过简易的 UART 接口

由于 MT7688AN 和 ATmega32U4 是通过 UART 接口连接的，因此您可以使用 Python、Node.js 或 C 语言编写应用程序并发出命令，通过 UART 接口传送到 MCU 一侧，如图 53 所示。在 MUC 一侧，编写代码解释这些命令，并通过访问传感器驱动完成相应的操作。

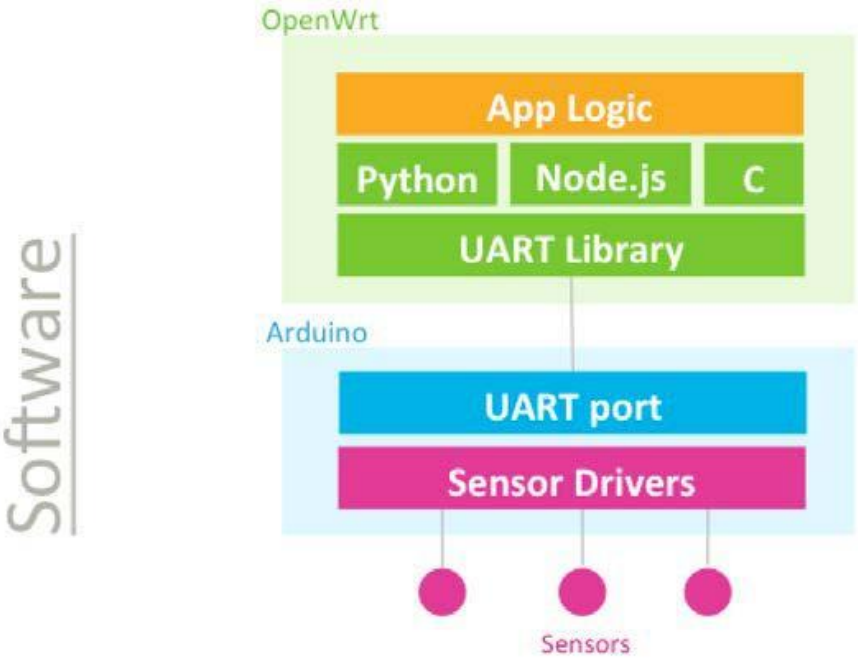


图 53 软件架构

方案二：通过 Firmata 协议

在上面的方案 1 中，您需要在 UART 通讯中实现命令的编制和解释；这为实现 MPU 与 MCU 之间的交互提供了最通用、最灵活、最有力的方法。但是，这也意味着您需要在 MPU 和 MCU 两方面都进行编程，这可能会增加软件开发的复杂性。

为了降低 UART 命令实现的复杂性，Firmart 协议被提出作为一种解决方法。它是 MCU 和 MPU 之间的一种通讯协议，目前已有很多成功应用案例。因此，对于那些熟悉使用 Python、Node.js 和 C 进行 MPU 编程的开发者来说，他们可以选择自己喜欢的开发语言来使用 Firmata 协议实现 MPU 和 MCU 之间的通讯，而不用处理具体的 UART 命令，也不用在 MCU 侧进行编程，如图 54 所示。举个

例子，Cylon.js 是一个流行的用于机器人应用的 Node.js 框架，它就是使用 Firmata 协议实现 MPU 和 MCU 之间的通讯，从而减少了 Node.js 开发者从事底层的 UART 和 MCU 编程工作。

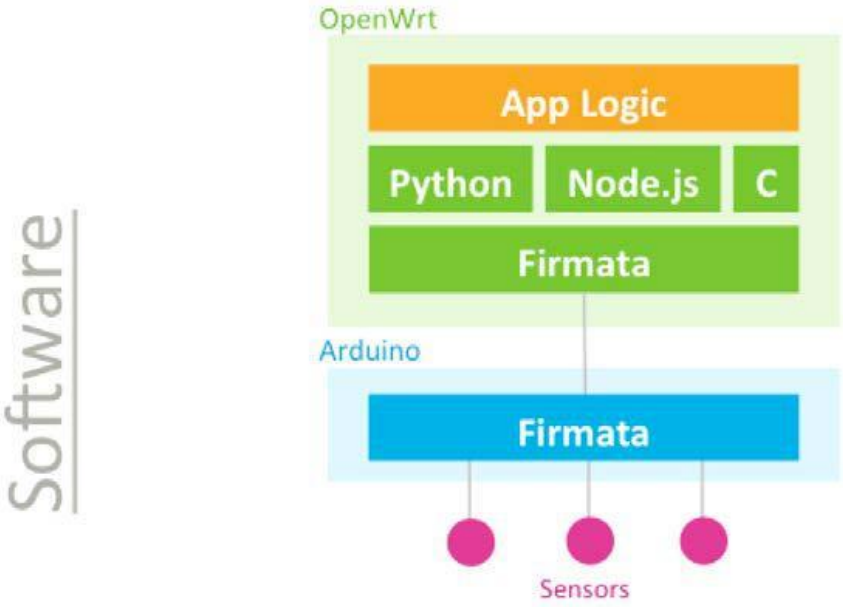


图 54 软件架构

方案三：通过 Arduino Yun Bridge 库

对于 Arduino Yun 开发者来说，MPU Linux 侧可以理解为一个简单地提供 Wi-Fi 连接、USB/SD 卡外设访问的黑盒子。LinkIt Smart 7688 提供 Arduino Yun Bridge 库支持，开发者可以使用它们熟悉的方式进行项目开发，如图 55 所示。这给开发人员不仅是一种熟悉的开发方式，而且还可以获得一个由 MT7688AN 提供的更稳定、更强大、更高效的平台。

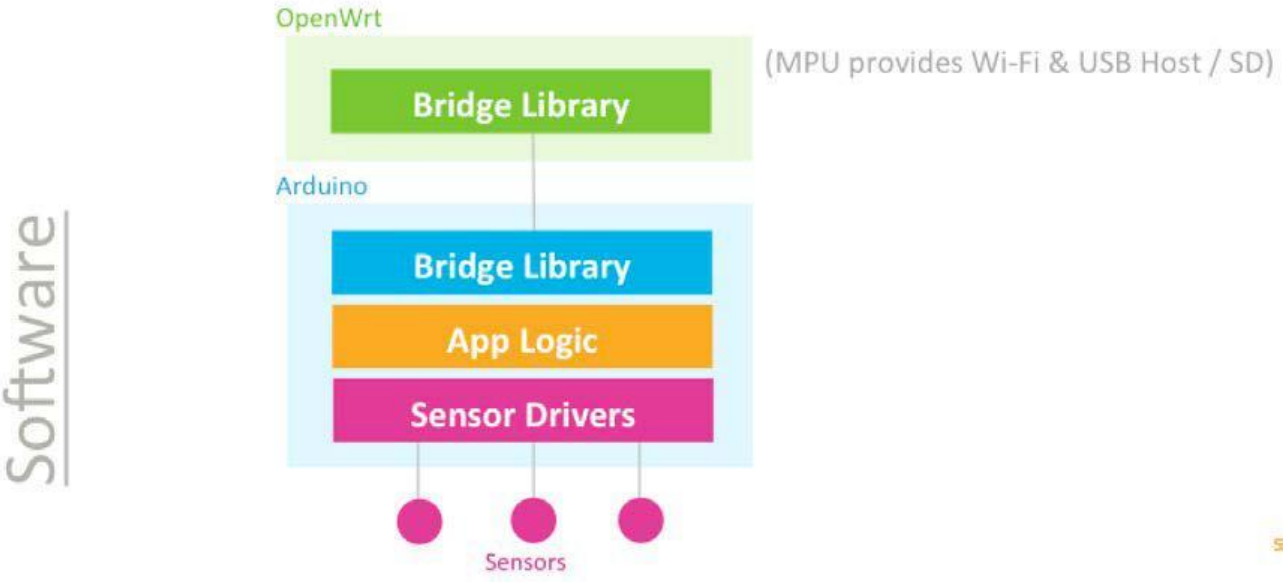


图 55 Yun Bridge 库

6.5 使用简易的 UART 接口编程

在前一节介绍的所有方案中，LinkIt Smart 7688 Duo 的 MPU 和 MCU 使用串口进行通信。MCU 与 OpenWrt Linux 通过 UART 接口进行通信。表 17 说明了单片机与微处理器之间的通信接口。

MCU		MPU	
Arduino	Serial1	Linux	/dev/ttyS0
对应引脚	D0 和 D1	引脚复用	UART0 (GPIO12 和 GPIO13)

表 17 MCU 和 MPU 通讯接口

在下面的小节中，将使用一个示例来演示如何通过简易的 UART 连接实现写入和读取，从而实现 MCU 和 MPU 之间进行通信。

6.5.1 闪烁灯程序——Arduino 侧

MCU 侧编写 Arduino 程序。在本例中，程序简单地监听从 MPU (Linux) 端发送的命令，并根据命令切换板载的 LED 状态。

1) 首先，把 LinkIt Smart 7688 连接到计算机，然后打开 Arduino IDE 并复制粘贴以下程序代码。

```
void setup() {
    Serial.begin(115200);
   // open serial connection to USB Serial port (connected to your computer)
    Serial1.begin(57600);  // open internal serial connection to  MT7688AN
                           // in MT7688AN, this maps to device
    pinMode(13, OUTPUT);
    }

  void loop() {
    int c = Serial1.read();      // read from MT7688AN
    if (c !=  - 1) {
      switch(c) {
        case '0':               // turn off D13 when receiving "0"
          digitalWrite(13, 0);
          break;
        case '1':               // turn off D13  when receiving "1"
          digitalWrite(13, 1);
          break;
      }
    }
  }
```

2) 然后从 IDE 的 Tools>Port 选择正确的端口号（在设备管理器中查看）。

3) 将程序上传到开发板，如图 56 所示。注意，这个时候开发板还没有闪烁——您需要再在 Linux 侧编写一个程序来才能使它闪烁，这将在下一步介绍。

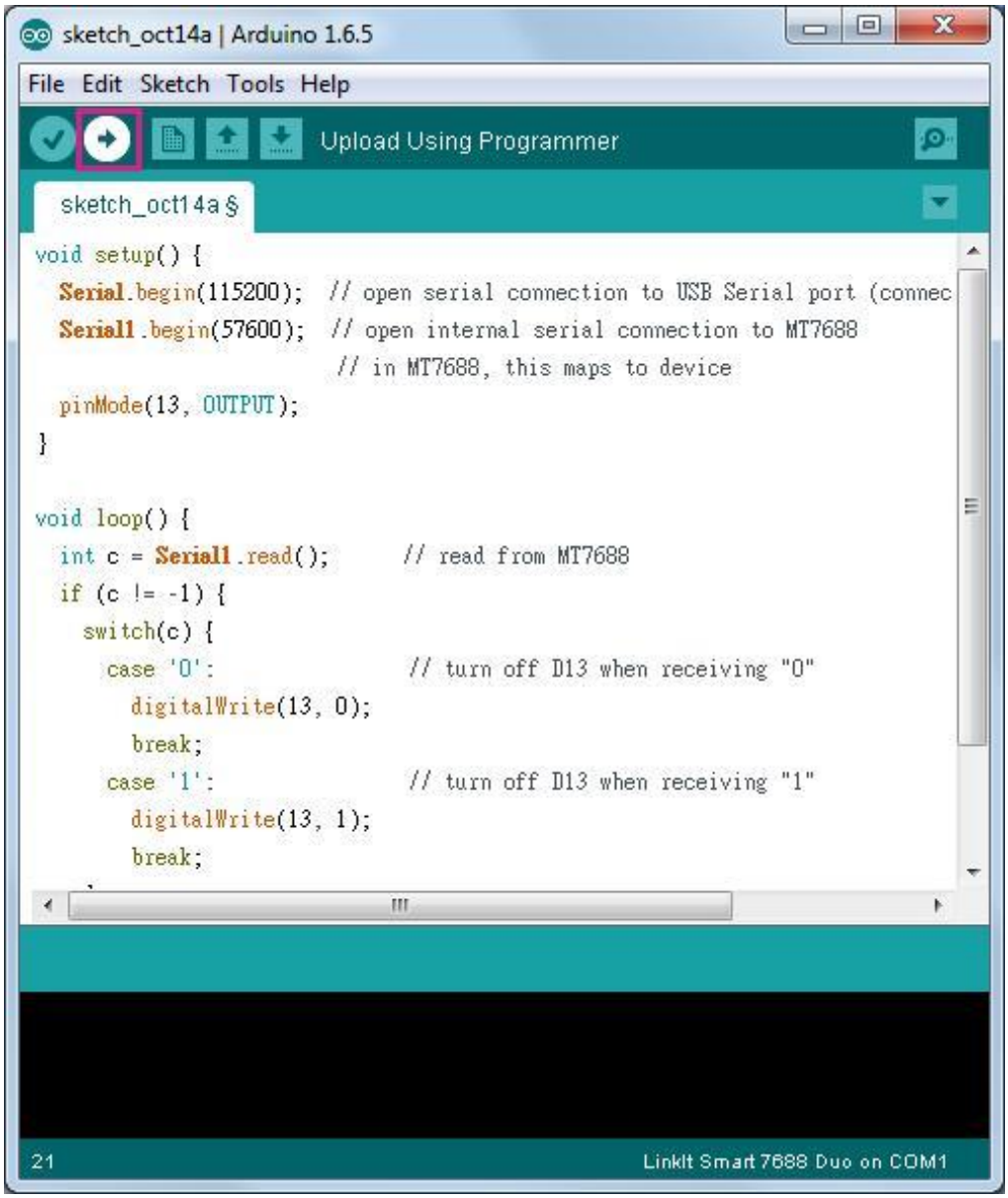


图 56 在 Arduino IDE 中上传程序

6.5.2 闪烁灯程序——Linux 侧

正如在第 3.3 节“各种开发板的程序设计模型”中介绍的那样，您可以使用多种语言在 Linux 侧编程，包括 C/C++、Python 或 Node.js。在本例中，我们需要一个程序将命令发送到 MCU（Arduino）侧使 LED 闪烁，我们使用 Python 语言来说明这个的程序。

1) 使用文本编辑器新建一个文件，复制以下代码并保存。

```
import serial
import time

s = None

def setup():
  global s
  # open serial COM port to /dev/ttyS0, which maps to UART0(D0/D1)
  # the baudrate is set to 57600 and should be the same as the one
  # specified in the Arduino sketch uploaded to ATmega32U4.
  s = serial.Serial("/dev/ttyS0", 57600)

def loop():
  # send "1" to the Arduino sketch on  ATmega32U4.
  # the sketch will turn on the LED attached to D13 on the board
  s.write("1")
  time.sleep(1)
  # send "0" to the sketch to turn off the LED
  s.write("0")
  time.sleep(1)

if __name__ == '__main__':
  setup()
  while True:
    loop()
```

2) 在系统控制台中执行这个 Python 程序——该程序主要是将字符串“1”和“0”写入到/dev/ttyS0 端口，这个端口是映射到 Arduino 中 Serial1 接口上的。上一小节中我们上传的 Arduino 程序将接收该字符串，然后相应地闪烁板载 LED。

现在，您可以进一步扩充 Arduino 程序来驱动其他外设，如 PWM、I2C，或者进一步细化命令消息以实现 Arduino 和 Linux 之间的状态同步。如果开发过程中涉及较多的外设种类，则可以使用一些外部库来实现通信协议。下一节将介绍这些协议的其中一种——Firmata。

6.6 使用 Firmata 协议编程

这一节演示如何使用 Firmata 协议通过串口为 MCU 与 MPU 建立通信。根据使用的接口不同，Firmata 的库也有几种，例如 Pyhont 语言的 pyFirmata 和 PyMata；其他语言如 Node.js 和 cylon.js 例子将在下一节介绍。

6.6.1 Python 例程

以下会使用两个例子来演示如何在 Python 环境中使用 Firmata 协议：pyFirmata 和 PyMata。

MCU 上的串口定义如表 18 所示。

MCU		MPU	
Arduino	Serial1	Linux	/dev/ttyS0
对应引脚	D0 和 D1	引脚复用	UART0 (GPIO12 和 GPIO13)

表 18 MPU 和 MCU 的串口引脚

6.6.2 协议

图 57 说明了 LinkIt Smart 7688 Duo 中的 MPU 和 MCU 如何使用 Firmata 协议和 Python 实现相互通信。

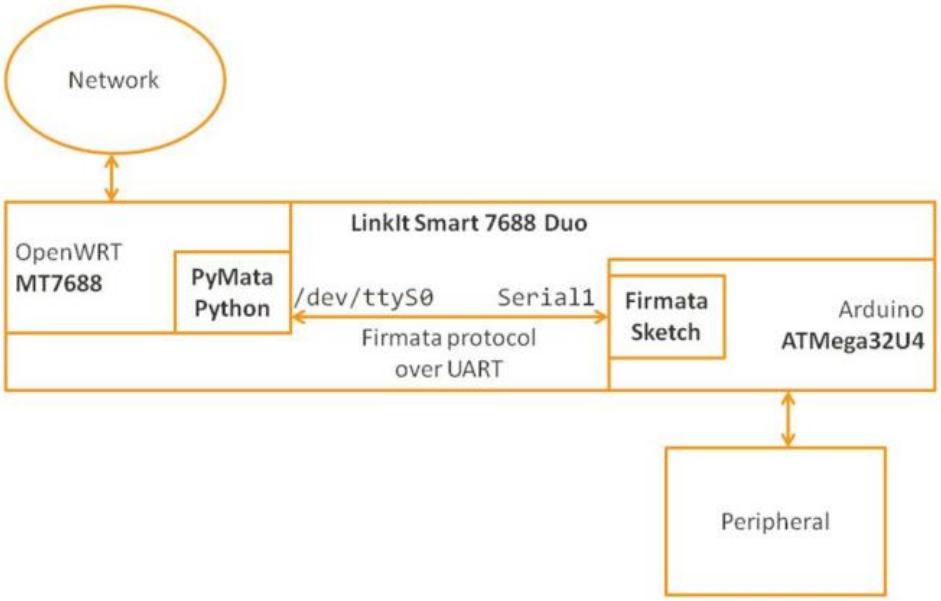


图 57 MPU 和 MCU 通讯原理图

6.6.3 建立开发环境

LinkIt Smart 7688 支持 Arduino1.6.5 和 Python2.7。

Firmata 协议要在 MPU 和 MCU 上运行。首先，您需要安装软件来对 MCU 进行编程。

请确保您已经安装了以下项目：

- 对于 PC
  - Arduino IDE 1.6.5

    用于 Atmega32U4 编程。

- 对于 LinkIt Smart 7688 Duo
  - Python 2.7

    系统映像中已预装载。
  - pyFirmata

    这是一个用于 MCU 的 Firmata 协议的 Python 接口。您可以通过在系统控制台中输入 **pip install pyfirmata** 来安装它。在编写本文档的时候，LinkIt Smart 7688 Duo 支持 pyFirmata-1.0.3-py2.7 版本。
  - PyMata

    这是一个用于 MCU 的 Firmata 协议的 Python 接口。您可以通过在系统控制台中输入 **pip install PyMata** 来安装它。在编写本文档的时候，LinkIt Smart 7688 Duo 支持 pymata-2.1 版本。

注意：LinkIt Smart 7688 在使用 pip 命令安装 pyFirmata 和 PyMata 接口时需处在 STA 模式，详细内容请参阅第 3.4.2 小节“STA 模式”

6.6.4 PyFirmata 方案

- **上传 Arduino Firmata 程序**

  这一小节介绍如何设置 Arduino 以监听串口的 Firmata 命令。

1) 打开 Arduino IDE 1.6.5
2) 在 Arduino IDE 中，打开 File>Examples>Firmata>StandardFirmata。

   这个内置的例程是用于 Arduino Uno 的，它使用的是 Serial 作为串口。然而，LinkIt Smart 7688 Duo 使用的是 Serial1 与 OpenWrt 通讯。因此，您需要将本示例的监听端口从 Serial 改为 Serial1。

   在例程中查找以下行:

```
Firmata.begin( 57600 );
```

3) 改为

```
Serial1.begin( 57600 );
Firmata.begin(Serial1);
```

4) 上传这个程序之后，Arduino 就已经准备好从 Linux 侧接收命令了。

- **使用 pyFirmata 运行闪烁灯程序**

安装 pyFirmata 之后，即可以开始运行闪烁灯例程。

1) 在系统控制台中，创建一个 python 文件并命名为 blink_with_firmata.py
2) 打开该文件并复制以下例程的代码，这个例程的代码已修改为适合 LinkIt Smart 7688 Duo 使用。

```
from  pyfirmata import Arduino, util
from  time  import  sleep

board =  Arduino('/dev/ttyS0')
print   " Start blinking D13 "
while   True:

  board.digital[13].write(1)
  sleep(0.5)

  board.digital[13].write(0)
  sleep(0.5)
```

3) 执行这个 Python 脚本，您应该能看到下以下信息。

```
# python ./blink_with_firmata.py
Start blinking D13
```

如果您能看见 LinkIt Smart 7688 Duo D13 口上的 LED 出现相应的闪烁，那么恭喜您，您已经成功运行这个闪烁灯例程。

要结束这个 Python 程序，请按 **Ctrl-C**。

6.6.5 PyMata 方案

- **上传 Arduino PyMata 程序**

PyMata 是遵循 Firmata 协议的 Python 接口，它使用自己的 Firmata 程序。因此，在把程序上传到 Arduino 之前，先要安装 PyMata 库。

请按以下步骤完成。

1) 把 Pymata 库（PyMata-master.zip）下载并解压到 Arduino 库文件夹中。
2) 在解压的文件夹中，打开 ArduinoSketch 文件夹，找到 library.zip 文件。
3) 解压 library.zip。

4) 把从 library.zip 解压的文件复制到 Arduino 库文件夹 C:\ Program Files (x86)\ Arduino \ libraries\

- **FirmataPlus 例程**

  现在，可以开始设置 Arduino 来监听串口的 Firmata 命令了。

  1) 打开 Arduino IDE 1.6.4 或 1.6.5

  2) 在 Arduino IDE 中，打开 File > Examples > FirmataPlus > FirmataPlus

     这个内置的例程是用于 Arduino Uno 的，它使用的是 Serial 作为串口。然而，LinkIt Smart 7688 Duo 使用的是 Serial1 与 OpenWrt 通讯。因此，您需要将本示例的监听端口从 Serial 改为 Serial1。在例程中查找以下行:

```
Firmata.begin(57600);
```

  3) 改为

```
Serial1.begin(57600);
Firmata.begin(Serial1);
```

  4) 上传这个程序之后，Arduino 就已经准备好从 Linux 侧接收命令了。

- **使用 PyMata 运行闪烁灯程序**

  当您安装了 PyMata 库并在 Arduino 中改了串口后，您就可以开始使用 PyMata 库来运行闪烁灯程序了。

  1) 把闪烁灯程序下载到 LinkIt Smart 7688 Duo。

     在这个示例程序中，您同样需要改变使用的串口。在示例程序中找到一下行：

```
# Create a PyMata instance
board =  PyMata("/dev/ttyACM0" , verbose = True)
```

  2) 改为

```
# Create a PyMata instance
board =  PyMata("/dev/ttyS0", verbose = True)
```

3) 执行这个 Python 脚本，您应该能看到显示以下内容：

```
# python ./pymata_blink.py
Python  Version 2.7.9 (default, Aug 12 2015, 15:09:09)
[GCC 4.8.3]

PyMata version 2.10  Copyright(C) 2013 - 15 Alan Yorinks    All rights
reserved.

Opening Arduino Serial port /dev/ttyS0

Please wait while Arduino is being detected. This can take up to 30  seconds
...
Board initialized in 0 seconds
Total Number of Pins Detected = 30
Total Number of Analog Pins Detected = 12
Blinking LED on pin 13 for 10 times ...
    1
    2
    3
    4
    5
    6
    7
    8
    9
    10
    PyMata close(): Calling sys.exit(0): Hope to see you soon!
```

如果您能看见 LinkIt Smart 7688 Duo D13 口上的 LED 在 Python 程序运行结束之前闪烁了 10 次，那么恭喜您，您已经成功运行这个闪烁灯例程。

您可以在之前下载 PyMata/examples 中查看更多示例，使用的时候只要注意把串口改为/dev/ttyS0 即可，并注意 LinkIt Smart 7688 和 Arduino Uno 之间的硬件差异：

- LinkIt Smart 7688 Duo 工作电压为 3.3V，Arduino Uno 则为 5V。
- 由于驱动马达或伺服电机需要较大电流，因此当您需要驱动这类外设的时候需增加额外的电源。

要查看其他 Firmata 协议的 Python 库，请参阅 Firmata 网站。

6.6.6 Node.js 例程

这一节介绍如何使用 Node.js 实现 Firmata 协议和 MPU 与 MCU 之间的通讯。主要步骤如下：

1) 建立 MCU 程序。在您的计算机上打开 Arduino ide 1.6.5

2) 将 Firmata 例程复制到 Arduino，点击这里查看例程源码。单击“**Raw**”按钮复制代码，如图 58 所示。

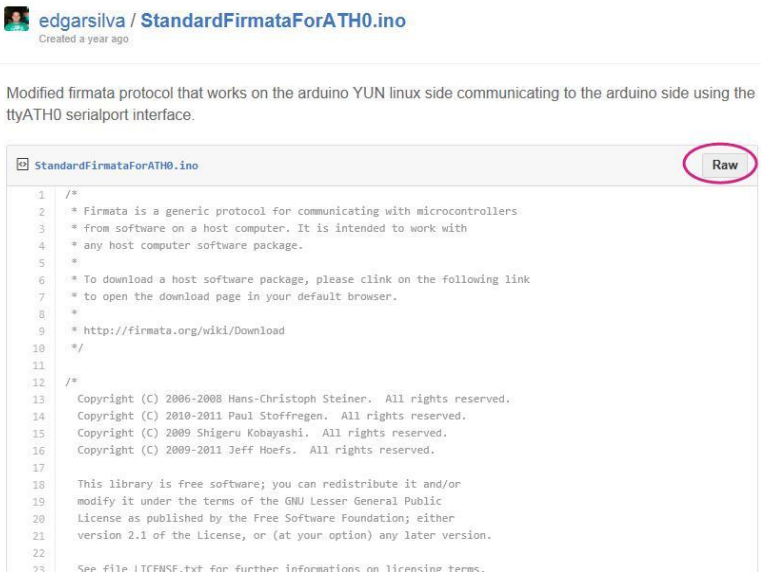


图 58 从 Github 复制例程代码

3) 将源代码复制到 Arduino IDE 并点击 **Upload**，如图 59 所示。

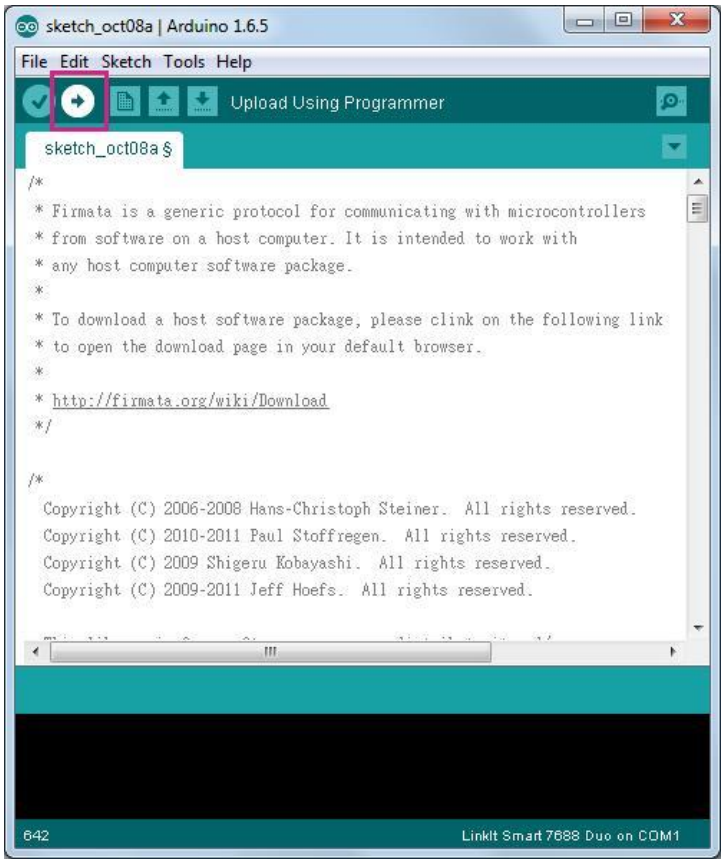


图 59 使用 Arduino IDE 上传示例程序

4) 建立 MPU 程序。您要在计算机上安装 Firmata。虽然您可以使用 NPM 直接在 LinkIt Smart 7688 Duo 上安装 Firmata，但这一过程比较耗时，因此，您应该在计算机上安装 NPM，然后使用 NPM 安装 Firmata。在主机计算机中，创建一个名为 testfirmata 的文件夹，并切换到该目录。下面命令中，$表示命令提示符，而不是命令的一部分。

```
$ mkdir testfirmata && cd testfirmata
```

5) 安装 NPM 之后，使用 npm 初始化文件

```
$ npm init
```

6) 输入以下命令安装 Firmata

```
$ npm install firmata -- save
```

7) 输入以下命令删除 node-serialport 模块

```
$ rm  -rf ./node_modules/firmata/node_modules/serialport
```

Firmata 有一个内置的 node-serial port，当您安装 Firmata 时，它会在您的计算机上创建一个编译文件。但是，由于它已经在 LinkIt Smart 7688 Duo 中建立了，因此您需要删除这个编译文件。

8) 压缩打包 Firmata 文件夹

```
$ tar - cvf ./firmata.tar ./node_modules/firmata
```

9) 用 SCP 把打包的文件传到 LinkIt Smart 7688 Duo

```
$ scp ./firmata.tar root@mylinkit.local:/root/app/node_modules
```

10) 打开 LinkIt Smart 7688 Duo SSH 系统控制台并输入以下命令，创建一个名为 app 的目录并进入该目录。

```
# mkdir app && cd app
```

11) 创建一个名为 app.js 的文件

```
# vim app.js
```

12) 输入 **i**，然后输入以下代码

```
console.log('WWW blink start ...');

var ledPin = 13;
var firmata = require('firmata');

var board = new firmata.Board("/dev/ ttyS0 ",function(err) {
   if (err) {
     console.log(err);
     board.reset();
   return;
   }

   console.log('connected...');
   console.log('board.firmware: ', board.firmware);

   board.pinMode(ledPin, board.MODES.OUTPUT);

   var url = require('url');
   var http = require('http');

        http.createServer(function(request, response) {
          var params = url.parse(request.url, true).query;

          if (params.value.toLowerCase() == 'high') {
            board.digitalWrite(ledPin, board.HIGH);
          } else {
            board.digitalWrite(ledPin, board.LOW);
          }
      response.writeHead(200);
      response.write("The value written was: " + params.value);
        response.end();
        }.bind(this)).listen(8080);

        console.log('Listening on port 8080 ...');
        });
```

13) 按 **Esc** 键并输入**:wq!**，保存并退出编辑器。

14) 在系统控制台中输入以下命令，运行 Node.js 例程。

```
# node app
```

15) 打开浏览器并输入以下命令对 LED 进行控制：

a) 点亮 LED：mylinkit.local:8080?value=high

b) 熄灭 LED：mylinkit.local:8080?value=low

16) 此时，D13 的 LED 应该会根据您使用的命令出现相应的亮灭。

6.6.7 Cylon.js 例程

这一小节介绍如何使用 Cylon.js 实现 Firmata 协议和 MPU 与 MCU 之间的通讯。主要步骤如下：

1) 建立 MCU 程序。在您的计算机上打开 Arduino ide 1.6.5

2) 将 Firmata 例程复制到 Arduino，点击这里查看例程源码。单击“**Raw**”按钮复制代码，如图 58 所示。

3) 建立 MPU 程序。首先您要在计算机上安装 Firmata 和 4 个 Cylon 模块（cylon、cylon-firmata、cylon-gpio 和 cylon-i2c）。Cylon 模块用来与 Arduino 硬件进行通信。虽然您可以直接在 LinkIt Smart 7688 Duo 上使用 NPM 安装 Firmata 和 Cylon，但这一过程比较耗时，因此，您应该在计算机上安装 NPM，然后使用 NPM 安装 Firmata 和 Cylon 模块。安装 Firmata 的步骤已在 6.6.6 节“Node.js 例程”中介绍了，安装 Cylon 也是一样的。因此，下面就只介绍 Cylon 例程。创建一个名为 testcylon 的文件夹，并切换到该目录。下面代码中$表示命令提示符，而不是命令的一部分。

```
$ mkdir testcylon   && cd testcylon
```

4) 安装 NPM 之后，使用 npm 初始化文件

```
$ npm init
```

5) 在计算机上输入以下命令，安装四个 Cylon 模块

```
$ npm install cylon  – -save
$ npm install cylon-firmata – -save
$ npm install cylon-gpio  – - save
$ npm install cylon-i2c – - save
```

6) 输入以下命令删除 node-serialport 模块

```
$ rm  –rf ./node_modules/cylon/node_modules/serialport
$ rm  –rf ./node_modules/cylon-firmata/node_modules/serialport
$ rm  –rf ./node_modules/cylon-gpio/node_modules/serialport
$ rm  –rf ./node_modules/cylon-i2c/node_modules/serialport
```

Cylon 有一个内置的 node-serial port，当您安装 Cylon 时，它会在您的计算机上创建一个编译文件。但是，由于它已经在 LinkIt Smart 7688 Duo 中建立了，因此您需要删除这个编译文件。

7) 压缩打包 Cylon 文件夹

```
$ tar – cvf ./cylon.tar ./node_modules/cylon
```

8) 用 SCP 把打包的文件传到 LinkIt Smart 7688 Duo

```
$ scp ./cylon.tar  root@mylinkit.local:/root/app/node_modules
```

9) 建立 MPU 程序。打开 SSH 系统控制台并输入以下命令，创建一个名为 app 的目录并进入该目录。#是控制台的命令输入提示符，非命令的一部分。

```
# mkdir app && cd app
```

10) 创建一个名为 app.js 的文件并打开文本编辑器

```
# vim app.js
```

11) 输入 **i**，然后输入以下代码

```
var Cylon = require('cylon');

Cylon.robot({
   connections: {
     arduino: { adaptor: 'firmata', port: '/dev/ttyS0' }
   },
   devices: {
      led: { driver: 'led', pin: 13 }
    },
      work: function(my) {
        every((1).second(), my.led.toggle);
      }
    }).start();
```

12) 按 **Esc** 键并输入**:wq!**，保存并退出编辑器。

13) 在系统控制台中输入以下命令，运行 Cylon.js 例程。

```
# node app.js
```

14) 此时，D13 的 LED 应该会每秒闪烁一次。

6.6.8 更新 Atmega32U4 启动引导程序（Bootloader）

本节介绍 ATmega32U4 上对启动引导程序（Bootloader）进行烧写的方法。LinkIt Smart 7688 Duo 已在 MCU (ATmega32U4)中预先烧写了 Arduino 的启动引导程序。您无须再烧写启动引导程序，但万一启动引导程序受损或者需要重新烧写，可以使用下列方法之一来更新启动引导程序。

6.6.9 使用 LinkIt Smart 7688 Duo 对另一块 LinkIt Smart 7688 Duo 烧写启动引导程序

在下面的示例中，使用两块 LinkIt Smart 7688 Duo 开发板，分别称为 A 和 B，A 开发板将作为编程器实现对 B 开发板的烧写。

1) 在 Arduino IDE 中，打开 Tools>Board 选择 Linkit Smart 7688 Duo
2) 打开程序 Example>ArduinoISP
3) 把代码行#define RESET SS 改为#define RESET 10
4) 把这个修改后的程序上传到 A 开发板
5) 把 A 开发板和 B 开发板的 S0、S1、S2、GND 引脚分别相连。
6) 把 A 开发板的 D10 引脚连接到 B 开发板的 RST 引脚
7) 在 Arduino IDE 中，打开 Tools > Programmer，选择 LinkIt Smart 7688 Duo 作为 ISP。
8) 最后，打开 Tools 菜单并选择烧写 Bootloader。

6.6.9.1 使用 OpenWrt 的 AVRDUDE 包烧写 Bootloader

您可以在 Linux 侧（MT7688AN）对 Atmega32U4 进行烧写。系统映像预装载了 AVRDUDE 包，使用这个包，您可以对 Atmega32U4 进行烧写。AVRDUDE 是一个 Linux 上用于 MCU 烧写程序的命令行工具。

1) 将启动引导程序文件复制到 U 盘、SD 卡或计算机上。启动引导程序文件位于以下路径中：
   { ARDUINO_IDE _PREFERENCE_LOCATION} \ Arduino15 \packages\ LinkIt\ hardware\ avr\ 0.1.5 \ bootloaders\caterina\ Caterina- smart7688.hex
   Arduino 参考位置（ARDUINO_IDE _PREFERENCE_LOCATION）请参见图 50。
2) 把启动引导程序传到 LinkIt Smart 7688 Duo 中，您可以使用存储设备或者 SCP 工具完成。在本例中，使用一个名为 USB-A1 的 U 盘。把 U 盘连接到 LinkIt Smart 7688 Duo，在系统控制台中，在根目录下输入以下命令。

```
# cd /Media/USB-A1
```

3) 在系统控制台中，执行以下命令

```
# avrdude -p m32u4 -c linuxgpio -v -e -U flash:w:Caterina-smart7688.hex - U lock:w:0x0f:m
```

4) 启动引导程序烧写进 Atmega32U4 后，屏幕显示应该和图 60 类似。

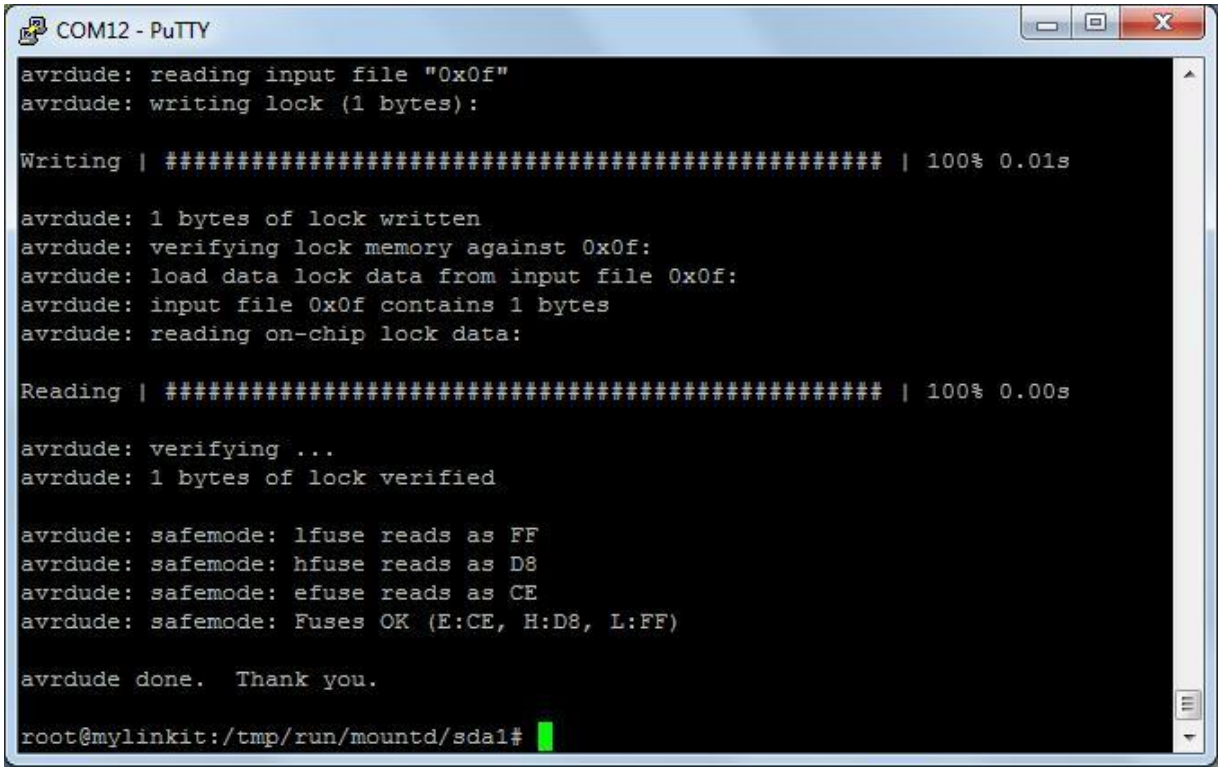


图 60 使用 AVRDUDE 烧写 Bootloader

5) 有关的详细信息，请参阅 AVRDUDE 包。

6.7 使用 Arduino Yun Bridge 库编程

LinkIt Smart 7688 Duo 在一定程度上推广 Yun Bridge 库的使用。为了使 Bridge 库运行，您需要配置系统使各种相关的 Bridge 服务运行。您还需要设置系统控制台映射/dev/ttyS0，使 Arduino 侧可以向 Linux 侧发送命令。幸运的是，LinkIt Smart 7688 Duo 的系统映像默认已经将所有上述配置封装到单个 UCI 配置中。你需要做的就是启用它。您可以通过在系统控制台中输入以下命令并重新启动系统来完成此操作。例如：

```
# uci set yunbridge.config.disabled=’ 0 ’
# uci commit
# reboot
```

重启之后，系统就可以接受来自 Arduino 侧的命令了，把开发板连接到计算机并打开 Arduino IDE，然后选择 **File>Examples>Bridge>TimeCheck**，把 TimeCheck 例程上传到 LinkIt Smart 7688 Duo 并运

行串口监视器，您应该能在串口监视器上看到以下输出：

```
Time Check
00:00:00
03:42:05
03:42:06
```

输出时间戳应与 Linux 系统时间戳相同。

您也可以尝试一下其他 Bridge 库的例程，需要注意的是，某些脚本和服务在默认系统映像上运行可能会有所不同，因此您可能需要相应地修改例程代码。

还请注意，当系统配置使用 Bridge 库时，/dev/ttyS0 将被 Bridge 库占用，此时不建议访问它。例如，当系统配置为使用 Bridge 库时，Firmata 无法运行。

要阻止系统运行与 Bridge 库相关的服务并解除/dev/ttyS0 的占用，请在系统控制台中使用以下命令并重新启动系统：

```
# uci set yunbridge.config.disabled=’ 1 ’
# uci commit
# reboot
```

7.如何编译生成固件和启动引导程序

这一章介绍如何编译生产 LinkIt Smart 7688/7688 Duo 的固件和启动引导程序（bootloader）。

7.1 编译固件

LinkIt Smart 7688 开发平台支持 Makefile 编译，您可以使用 make 命令来编译固件。

本节中的说明适用于 Ubuntu Linux 14.04.3 环境。

1) 第一步，如果您的主机环境不是 Ubuntu，那就要安装虚拟机，大约需预留 50G 的磁盘空间。
   - 从 http://www.ubuntu.com 获取 Ubuntu 14.04.3 LTS 版。
   - 安装虚拟机
2) 编译固件前需要先安装一些必要的软件包。打开 Terminal 并输入以下命令：

```
$ sudo apt-get install git g++ libncurses5-dev subversion libssl-dev gawk
libxml-parser-perl unzip
```

3) 下载 OpenWrt CC 源码，它包含了整个用于编译固件的 OpenWrt 环境（包括 Linux 内核），输入以下命令下载源码：

```
$ git clone git://git.openwrt.org/15.05/openwrt.git
```

4) 创建配置文件 feeds.conf，由默认的模板文件 feeds.conf.default 复制而来：

```
$ cd openwrt
$ cp feeds.conf.default feeds.conf
```

5) 添加 LinkIt Smart 7688 feed 到 feeds.conf

```
$ echo src-git linkit https://github.com/MediaTek-Labs/linkit-smart-7688-feed.git>>
feeds.conf
```

6) 更新所有包的 feed 信息，用来编译 firmware：

```
$ ./scripts/feeds update
```

7) 安装所有的包

```
$ ./scripts/feeds install –a
```

8) 准备内核配置文件，告知 OpenWrt 我们想编译一个 LinkIt Smart 7688 的固件：

```
$ make menuconfig
```

9) 选择以下选项设置目标配置文件：
   - 目标系统（Target System）：Ralink RT288x/RT3xxx
   - 子目标（Subtarget）：MT7688 based boards
   - 目标配置文件（Target Profile）：LinkIt 7688

10) 保存并退出（使用默认的配置文件名，不要改变它）

11) 使用 make 命令开始编译固件。

```
$ make V=99
```

12) 编译完成后，生成的固件文件在以下路径：

    bin/ramips/openwrt-ramips-mt7688-LinkIt7688-squashfs-sysupgrade.bin

13) 现在您可以使用这个文件在 Web UI 上完成固件更新，或者重命名为 lks7688.img，利用 U 盘进行固件升级。

7.2 编译启动引导程序（Bootloader）

LinkIt Smart 7688/7688 Duo 开发平台支持 Makefile 编译，您可以使用 make 命令来编译启动引导程序。

本节中的说明适用于 Ubuntu Linux 14.04.3 环境。

1) 编译启动引导程序前，请参考第 7.1 节“编译固件”准备相应的编译环境。

2) 输入以下命令，下载 Bootloader 源码：

```
$ git clone  https://github.com/MediaTek-Labs/linkit-smart-uboot.git
```

3) 进入 LinkIt Smart 7688 启动引导程序源码文件夹。

```
$ cd linkit-smart-uboot
```

4) 安装编译启动引导程序所需的工具链，这个工具链用于 32 位系统。

```
$ sudo tar xjf buildroot-gcc342.tar.bz2 -C /opt/
```

5) 如果是 64 位机，则需要安装另外的包，用于执行工具链。

```
$ sudo dpkg  --add-architecture i386
$ sudo apt- get update
$ sudo apt- get install libc6:i386 libncurses5:i386 libstdc++6:i386
```

6) 开始编译启动引导程序。

```
$ make
```

7) 编译完成后，最后生成的启动引导程序文件是 uboot.bin

8) 现在，您可以把文件重命名为 lks7688.ldr 并用 U 盘升级启动引导程序。

8.疑难解答

8.1 无法启动固件升级或者升级失败，为什么？

如果您使用 U 盘升级固件，请检查一下内容：

- U 盘和 USB Host 接口已经可靠连接
- USB 电源/Host 线已牢固地连接到电源接口，而不是开发板上的 USB Host 接口

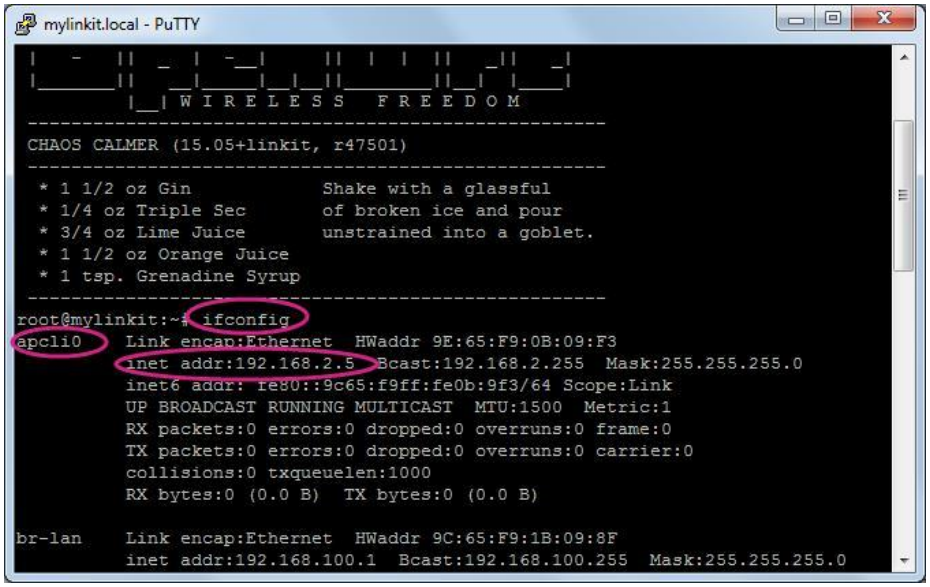
如果您使用 Web UI 升级固件，请检查一下内容：

- 在固件升级过程中需给 LinkIt Smart 7688/7688 Duo 提供稳定的电源。请勿在固件升级过程中拔下 U 盘或断开 USB 线，否则开发板将无法启动。如果发生这种情况，您可以使用启动引导程序启动开发板并完成固件升级过程。有关详细信息，请参阅第 4.6.2 小节“升级启动引导程序”。

在您检查上述项目后，请重试，如果您仍然无法升级固件，请联系 Seed Studio 寻求技术支持或进入 MediaTek Labs 论坛。

8.2 无法用浏览器打开 mylinkit.local，为什么？

如果您使用的是 Windows 7，请安装 Bonjour 打印服务 (32 或 64 位)，重启计算机并再次尝试在浏览器中打开 mylinkit.local。请确保您的 PC 和 LinkIt Smart 7688 在同一个局域网中。



8.2 虚拟机无法用 mylinkit.local 检测到开发板，为什么？

虚拟机的模拟 Linux 环境不支持 mDNS，因此您要有开发板的 IP 地址才能识别它。要查看开发板的 IP 地址，请在系统控制台中输入 ipconfig。IP 地址会显示在 apcli0 下的第一段中，如图 61 所示。当您的 PC 和开发板都在同一个网络下时，你也可以在主机 PC 上 ping 一下 mylinkit.local，获得开发板的 IP 地址。

```
 |    -   ||  _  |  -__|     ||  |  |  ||   _||   _|
 |_______||   __|_____|__|__||________||__|  |____|
          |__| W I R E L E S S   F R E E D O M
 -----------------------------------------------------
 CHAOS CALMER (15.05+linkit, r47501)
 -----------------------------------------------------
  * 1 1/2 oz Gin            Shake with a glassful
  * 1/4 oz Triple Sec       of broken ice and pour
  * 3/4 oz Lime Juice       unstrained into a goblet.
  * 1 1/2 oz Orange Juice
  * 1 tsp. Grenadine Syrup
 -----------------------------------------------------
root@mylinkit:~# ifconfig
apcli0    Link encap:Ethernet  HWaddr 9E:65:F9:0B:09:F3
          inet addr:192.168.2.5  Bcast:192.168.2.255  Mask:255.255.255.0
          inet6 addr: fe80::9c65:f9ff:fe0b:9f3/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

br-lan    Link encap:Ethernet  HWaddr 9C:65:F9:1B:09:8F
          inet addr:192.168.100.1  Bcast:192.168.100.255  Mask:255.255.255.0
```

图 61 STA 模式下查看 LinkIt Smart 7688 的 IP 地址

8.4 无法使用 SSH 方式访问，出现一个错误“Host Identification Has Changed”。

如果您看到一个错误信息类似图 62 所示。

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
5b:34:59:6b:f3:f5:c1:a9:e6:f9:0d:5a:63:2c:4c:21.
Please contact your system administrator.
Add correct host key in /Users/[redacted]/.ssh/known_hosts to get rid of this mess
age.
Offending RSA key in /Users/[redacted]/.ssh/known_hosts:1
RSA host key for mylinkit.local has changed and you have requested strict checki
ng.
Host key verification failed.
```

图 62 Host ID 发生改变的警告

它标明您使用了不同的 LinkIt Smart 7688 开发板或升级了新固件，并尝试使用 SSH 访问系统控制台。 这是因为新的固件或新的开发板创建了一个新的主机 ID，这个 ID 与前一台不同，导致出错。要解决此问题，您需要删除旧的主机 ID。如下所示。

1) 在 Terminal 中，打开 known hosts 文件，本例使用 vi 编辑器，您也可以使用任何您喜欢的编辑器，注意#是终端命令提示符，不是命令的一部分。

```
#   vi ~/.ssh/known_hosts
```

2) 找到 mylinkit.local 所在的位置，如图 63 所示。

图 63 known hosts 文件

3) 输入 **dd**（vi 命令）删除这个入口。

4) 按 **Esc** 键并输入**:wq!**（vi 命令），保存并退出编辑器。

5) 现在再试一下使用 SSH 访问 root@mylinkit.local，应该可以访问系统控制台了。

8.5 附近有多个 LinkIt Smart 7688 热点时，如何确定哪一块是我的？

要查看开发板的硬件地址，在系统控制台中输入 ifconfig，硬件地址会显示在 ra0 后的第一句中，如图 64 所示。最后三组数字应该与您的开发板的热点名称一样。

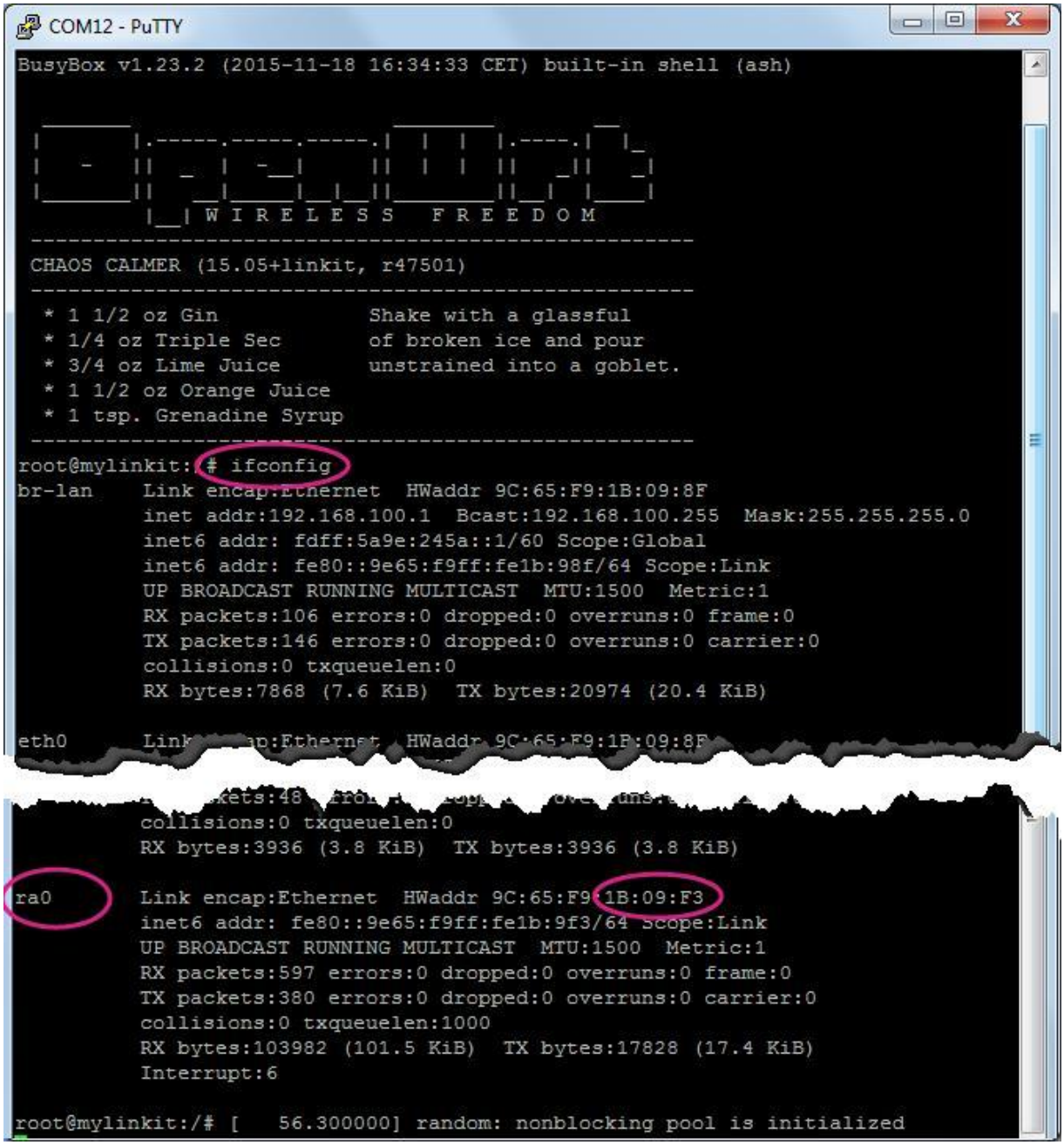


图 64 查看 LinkIt Smart 7688 的硬件地址

8.6 开发板上的 flash 非常慢，似乎损坏了，为什么？

SPI NOR flash 的写入次数有限 (大约 10 万次) ，同时文件系统操作不是原子性的。因此，如果 flash 频繁读写或曾经在写入操作过程中断电，则可能会导致闪存损坏。因此，建议使用临时文件系统 （/tmp）或外部存储器（如 U 盘或 SD 卡）。

8.7 为什么我的开发板由于文件系统损坏而无法启动？

这可能和问题 8.6 一样，是开发板上的 flash 损坏导致。

8.8 如果开发板上的 flash 满了，如何处理？

可以使用外部存储设备进行更多的数据存储，或将根文件系统扩展到外部存储设备上。

8.9 为什么 I2C 设备无法运行？

MT7688AN 不提供 I2C 时钟延展和重复启动功能。

8.10 为什么我的开发板在驱动伺服系统时不断重启？

伺服系统需使用外部电源，不能使用 LinkIt Smart 7688 的 Vout，避免开发板输出功率不足。