嵌入式 Wi-Fi（物联网）基本解决方案

（基于 32 位 MCU）

物联网--MCU 的春天！

修改记录：

修改日期	版本号	作者	描述
2015-01-02 华师图书馆	V1.0	♪ BlueS	内容主要来源： 01）工作期间（4 年）的工作体会 02）研究生期间：学习、个人想法 03）网络收集、整理

目　录

嵌入式 Wi-Fi（物联网）

基本解决方案

（基于 32 位 MCU）

1、引言

1.1 物联网的发展趋势（互联网的 自然 延伸）

从 1969 年互联网的诞生起，我们的生活也就开始悄无声息地变化着。物联网（IOT），即物物相联的网络，被称为是继计算机、互联网之后世界信息产业发展的第三次浪潮。

随着无线连接突破智能手机和 PC 的限制，延伸到更多的嵌入式设备中，数百亿 的设备也将接入到互联网。

1.1.1 物联网发展的基本规律

- 物联网发展的基本规律：一层一层地、从内到外地、越来越快地 加速扩散。（从 PC 台式机 → 笔记本→ 智能手机→ 平台电脑 → … →物物相联）
- 目前，物联网相关的重点扩展领域：安防、智能家居、智能交通（汽车）、智能医疗、传感网等。
- 物联网生态链将越来越重要，也越来越完善。物联网的相关标准和规范，将不断完善和统一。
- 无线网络技术与嵌入式技术的结合，已经是物联网技术的发展重点，有巨大的市场与发展潜力。
- 嵌入式 Wi-Fi（基于 32 位 MCU）与物联网结合，将带来更多创新机会。

1.1.2 物联网与云计算的结合

两者结合基本框架：联网设备⇔云服务器⇔用户移动终端（智能手机），通过移动互联网串联起来。

- 越来越多的事情，让云服务器来处理。 手机上的一些功能、嵌入式设备的一些功能，都将转换到云服务器进行处理（例：数据海量存储、复杂运算等）。因此，智能终端设备的功能将趋于简单化。（在这个方面，与 Chromebook 网络笔记本 类似）
- 更多的数据处理将会放到云端上（例语音、图像识别），而嵌入式设备更多地只负责输入（数据采集）、输出（控制相应的设备）、以及联网功能（与云通信）。
- 大数据处理与数据挖掘。把一大堆嵌入式设备（传感器）数据上传到云服务器，转换成易懂的曲线图，并找出数据背后的规律等（挖出价值）。（例：气候变化、疾病的预测等）

1.2 嵌入式 Wi-Fi 的发展

随著物联网、云计算、大数据、传感网络、智能硬件的兴起，以及在环保低功耗、超小体积的市场需求的驱使之下，集成无线通讯的微控制器（Wireless MCU SoC） 纷纷被开发出来（特别是集成 ARM 内核

和无线处理器 SoC 芯片）。例：

- TI 公司 2014 年 11 月量产的 CC3200（Cortex-M4 + Wi-Fi 射频前端）SoC 单芯片物联网解决方案；
- 联发科 MTK（台湾）在 2014 年 6 月推出的 MT7681（SoC Wi-Fi 单芯片解决方案）；
- 博通公司于 2011 年 3 月推出的 BCM943362 解决方案；
- 乐鑫（国内）2014 年 7 月推出的 ESP8266 解决方案；
- ……

1.2.1 嵌入式 Wi-Fi 应具备哪些功能

嵌入式 Wi-Fi 设备：一个可以在 Wi-Fi 热点区域接入互联网、自动链接至云服务器、环保低功耗的嵌入式无线通信终端。（例：Wi-Fi 智能插座，就是一个完整的嵌入式 Wi-Fi 设备）

- 支持 802.11b/g/n 无线标准；（以及 802.11 ac）
- 支持 TCP / IP / UDP 网络协议栈，提供标准 BSD 套接字应用编程接口，以及 WebSocket；
- 支持无线工作在 Station、AP，以及 Wi-Fi Direct 模式（P2P）；
- 配备常用基本外设模块：UART / SPI / I²C 等通讯接口、GPIO 输入输出、ADC 数据采样等；
- 能可靠、稳定、长时间地工作，不掉线或卡死；
- 提供相应的安全、保密措施。（WPA2、SSL、TSL、HttpS 等）；
- 环保，低功耗等；
- ……

1.2.2 嵌入式 Wi-Fi 与普通 Wi-Fi 的比较

	嵌入式 Wi-Fi 模块（中、低平台）	普通 Wi-Fi 模块（高平台）
适用范围	无线家电、仪表、智能灯泡等智能家居设	笔记本、手机、平板电脑等
主控芯片	模块上集成的 MCU	x86 CPU、ARM 等高速微处理器
接口	UART、SPI、SDIO	SDIO、USB、PCI / PCIe
功耗	低（锂电池供电，充一次电，可工作数月）	高（功耗和功率都很强，需要经常充电）
产品	TI 的 CC3200/ CC3100、联发科的 MT7681 博通的 BCM943362、Marvell 的 88w8686	瑞昱 RTL81xx 系列、Marvell、雷凌 RT 系列、博通 BCM、、高通 Atheros 等
开发设计	内置嵌入式 TCP/IP 协议栈、Wi-Fi 驱动、MAC、无线安全协议等，所有的网络软件封装成一个 UART 或 SPI 接口的设备，使用简单，只需要往 UART 或者 SPI 收发数据即可。从整体软件层面看，不属于网络设备。	需要在主机添加 Wi-Fi 驱动、同时需要依赖主机的网络协议栈等软件平台资源，从整体软件层面上看，属于网络设备，使用时需要遵循网络相关的协议。

1.2.3 无线通信技术的比较（ZigBee、蓝牙、Wi-Fi）

目前流行的三种短距离无线通信技术为 Wi-Fi、ZigBee 和蓝牙，它们各自的协议标准都有着不一样的商业用途及应用领域。

相关比较	ZigBee	蓝牙	Wi-Fi
协议标准	IEEE802.15.4 无线个人局域网 WPAN	IEEE802.15.1 无线个人局域网 WPAN	IEEE802.11 a/b/g/n/ac/ad 无线局域网技术标准 WLAN
工作频率	2.4GHz	2.4GHz	2.4G 或 5GHz

主要用途	无线传感器网络、部分智能家居等相关方面的应用	简化移动通信终端设备之间的连接与通信，如手机、笔记本、音频播放器等之间进行无线通信和数据交互。	Wi-Fi 技术的研究旨在无线局域网，取代布线麻烦的有线网络，实现无线上网功能。而嵌入式 Wi-Fi 有更广泛的用途，并与 ZigBee、蓝牙产生明显竞争。
缺点	数据通信速率较低、 无法直接接入有线网络、 系统扩展困难、维护性差	传播距离短、 网络容量小	成本较高、 最大的缺点是功耗较高、 需要外接电源、锂电池供电
优点	功耗低、成本低、网络容量大、具有自组网功能	功耗低、成本低、 可以用纽扣电池供电	数据通信速度高、系统扩展性好、结构简单可以实现快速组网、 能与现有家庭网络无缝的连接

1.3 物联网--- 32 位 MCU（微控制器）的春天（机遇）

1.3.1 低端 8、16 位 MCU 与 32 位 MCU 的比较

8 位的 MCU 在大多数情况下，没有足够的 RAM 和 Flash 空间来跑 TCP/IP 协议栈（故较难实现联网功能）。但由于 8 位 MCU 的价格低廉，故低端 8 位 MCU 仍存在一部分市场。

但随着 8 位 MCU 与 32 位 MCU 的价格逐渐缩短、以及单片机联网需求的攀升，部分传统的 8 位和 16 位 MCU 产品解决方案将逐渐被 32 的 MCU（ARM cortex 内核）取代。

1.3.2 高性能 MPU 与 32 位 MCU 的比较

一、控制中心的转移

原本由嵌入式设备自己的微处理作为主控制中心，转移到移动智能终端（手机）、或云服务器。

由于智能手机的普及，通过 APP（应用软件），手机（联同其“背后”的云服务器）将一同作为嵌入式设备的控制中心。

某些嵌入式设备在与用户交互过程中，如配置嵌入式设备的参数，读取设备状态值，将不需要按键，也不需要显示屏。而是让移动终端（智能手机）充当按键输入和显示屏输出。手机通过无线的方式向嵌入式设备输入相关的配置参数；嵌入式设备的状态数据、采集的数据，在智能手机上显示。

二、功能的转移

嵌入式设备中越来越多的功能（特别是一些复杂的运算、数据处理），将转移到智能手机或云服务器中进行处理。嵌入式设备的功能将在某种程度上会简化。（功能的转移的同时，也是功耗的转移。）

嵌入式设备只负责数据采集、数据上传、控制输出（功能会越来越简单）；云服务器，负责数据运算，例语音识别、图像处理（功能会越来越强大复杂）；手机或平台作为显示端，与用户交互界面。用户体验：操作会越来越简单自然。

因此，很多在高性能平台（MPU）实现的功能，将不断地往云服务器（更高性能、更快处理速度、更高服务质量）那边转移。特别是那些运算量大（语音识别、图像识别），将由云服务器来实现。导致的结果：硬件的成本将会进一步被压缩，甚至免费提供硬件，转过收取增值服务费。（例百度免费提供的智能手环解决方案）

三、操作系统的转移

目前，大多数 Wi-Fi 技术的实现都是基于 MPU 高性能平台的操作系统（如 Linux、Windows CE、Android、VxWorks），然而随着嵌入式 Wi-Fi 的发展，以及上述的控制中心、功能的转移，基于 32 位 MCU 处理器在更多的应用场合，将变成更为合适的选择。同时，轻量级嵌入式实时操作系统将会得到更广泛地

应用（例： uCOS II/ III、完全开源免费的 FreeRTOS）。

当前情况：一些半导体芯片提供商（如 TI、ST、Atmel），在他们 32 位 MCU 的开发包 SDK 中，均提供了 FreeRTOS、以及自家的 RTOS 的操作例程 Demo。因此，我们可以看到，未来在 32 位 MCU 跑小型嵌入式操作系统的应用将越来越多，并且会越来越规范（这与 PC 机的发展过程有点类似）。

四、功能、控制中心、操作系统的转移所带来的好处

- 提高用户体验：提供更便捷的信息输入方式，操作简单、自然；
- 嵌入式设备工作也会更加稳定（越简单的东西，越可靠）；
- 复杂运算、数据处理移到智能手机、云服务器处理，使嵌入式设备更加省电，延长电池使用时间；
- 转移到云服务器处理，不仅提高运算速度和功能质量等，还可以进行数据收集、大数据分析。
- 嵌入设备由云服务器统一、规范地管理，有利于整个生态链的发展和维护；
- 硬件成本的大幅度降低。

综上分析，32 位 MCU（特别是 SoC 单芯片 Wireless MCU）、轻量级嵌入式操作系统、嵌入式 Wi-Fi 芯片，将会有更大的需求量。后面章节，将重点介绍：如何在 32 位微控制器（MCU）上运行嵌入式操作系统（FreeRTOS），实现嵌入式 Wi-Fi 功能（运行 TCP/IP 协议栈、Wi-Fi 驱动），并结合云服务器、移动智能终端，实现基本的物联网解决方案。

2、嵌入式 Wi-Fi（物联网）系统总体方案

2.1 物联网系统基本组成部分（模块）

在这里，我们将先从宏观上分析物联网的整体系统方案。在下一章节，将从微观上分析嵌入式 Wi-Fi 的具体解决方案。

如图 1 所示，系统主要由若干个嵌入式 Wi-Fi 设备（传感节点）、网关、云服务器、用户智能终端四大部分共同组成。

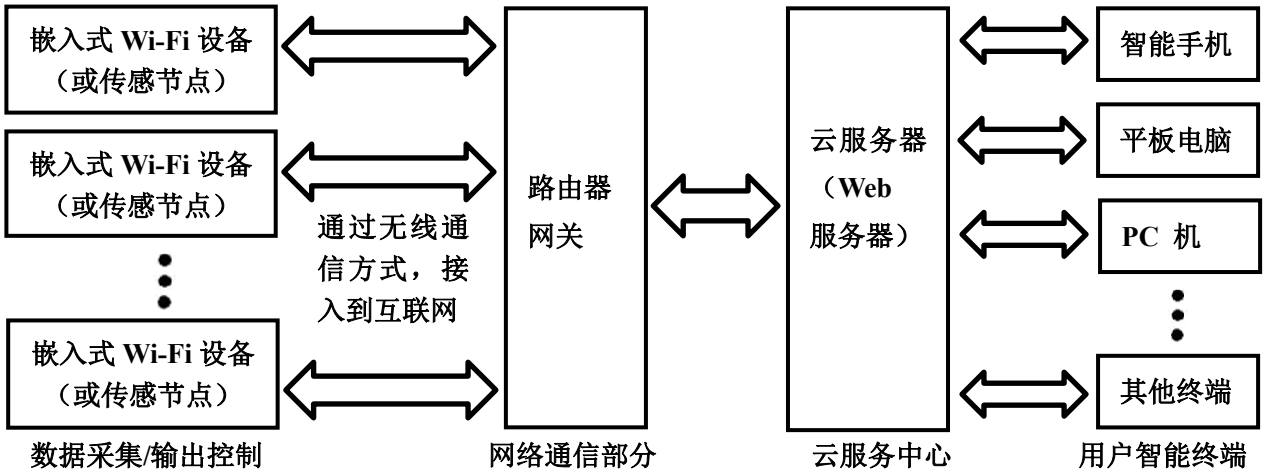

图 1 嵌入式 Wi-Fi（物联网）系统总体方案框架图

（1）嵌入式 Wi-Fi 设备

实现对物体数据采集，并将数据上传到云服务器；同时，也要能接收云服务器发出的控制命令和配置参数等。

（2）路由器网关

利用家庭已有的网络资源——家庭常用的无线路由器。

（3）云服务器（Web 服务器）

实现一个专为嵌入式设备提供指定服务的云服务器。该服务器要能够与嵌入式设备进行双向的数据通信，并且还可以与目前流行的智能终端（智能手机、平板等）进行双向的数据通信。

（4）用户智能终端

实现智能终端与云服务器之间的数据交互，然后通过云服务器，实现智能终端在 Wi-Fi 和 3G/4G 两种无线网络环境下，可以随时随地读取嵌入式设备的信息、或对嵌入式设备进行控制。

2.2 物联网系统案例分析（以本人硕士毕业设计为例）

（1）毕设题目： 一种基于 Wi-Fi 的低功耗报警系统的设计与实现

（2）毕设内容

以 TI 公司的 CC3200 微控制器为核心， 实现了一种基于低功耗 Wi-Fi 的物联网报警系统。系统主要由基于 CC3200 的 Wi-Fi 报警装置、无线路由器、云服务平台、智能手机客户端组成。Wi-Fi 报警装置的加速度传感器，通过相关算法实时对物体的加速度信息进行监控，判断出冲击、移动等状态，并将报警信息通过无线路由器转发至云服务平台，最终由云平台将报警信息发送到用户手机进行相关的报警操作。

（将嵌入式 Wi-Fi 设备 ⇔云服务器 ⇔ 用户端 A PP 通过移动互联网串联起来）如图 2 所示。

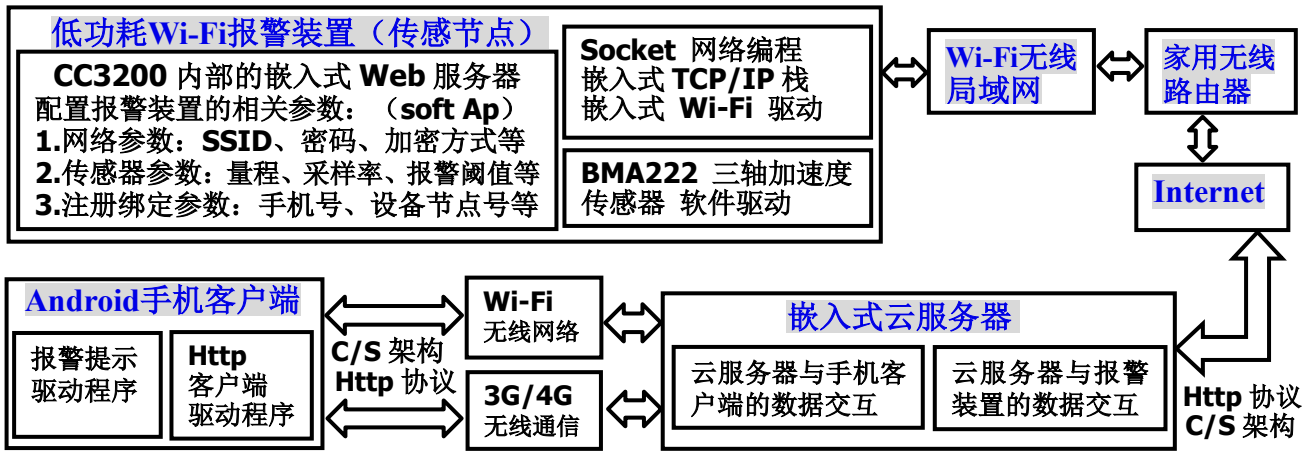

图 2 基于 Wi-Fi 的低功耗报警系统的总体框图

下面将重点分析嵌入式 Wi-Fi 的基本解决方案。

3、嵌入式 Wi-Fi 具体的基本解决方案(★★★)

3.1 嵌入式 Wi-Fi 芯片的几种解决方案分析

3.1.1 单芯片 SoC 解决方案：MCU 和 Wi-Fi IP 核 集成到单个芯片

（1）TI 公司的 CC3200 单芯片 SoC Wi-Fi 解决方案（2014 年 7 月推出）

CC3200 样片于 2014 年 7 月初推出，2014 年 11 月刚量产。TI 的智能插座中，采用 CC3200 解决方案。

CC3200 微控制器，是一款专门针对物联网 (IOT) 应用的 32 位低功耗微控制器。该芯片集成了最新 ARM Cortex-M4 内核（运行频率 80MHz）、Wi-Fi 网络处理器（802.11 b/g/n 射频、基带）、电源管理子系统（DC-DC 转换器），具有高速度、低工作电压、超低功耗、模块化控制灵活等技术特点。

CC3200 由于其本身的特性，具有较好性价比。在量大的时候，已经可以到 2 美元/每颗。

（2）联发科 MTK（台湾）的 MT7681 单芯片 SoC Wi-Fi 解决方案(2014 年 6 月推出)

MT7681 是一款高度集成的 Wi-Fi 的 SoC（片上系统）的单芯片，支持 IEEE802.11b/g/n 单数据流，提供 GPIO 和 PWM 智能控制，以及 UART，SPI，和设备通信的 I2C 接口。MT7681 主要针对如灯泡、门锁、插座等小型设备。采用 uIP　TCP/IP 协议栈。

（3）Qualcomm (高通)的 QCA4002/4004 单芯片 Wi-Fi 解决方案(2013 年 3 月推出)

Qualcomm (高通)推出的高阶 QCA4002/4004 低功耗 Wi-Fi 单芯片，采用内建 CPU 与 Wi-Fi 802.11n (2.4GHz 和 5GHz 频段)收发器的设计，提供 40Mbps 的传输速度。海尔部分洗衣机和空调设备，采用高通 QCA4004 的解决方案。

3.1.2 双芯片分离式：MCU 通过 SPI 或 SDIO 外接 Wi-Fi 处理芯片

（1）TI 公司的 CC3100　Wi-Fi 射频前端芯片

CC3100 由于内部没有集成 MCU 核，故需要与其他 MCU 配合使用，即驱动 CC3100 的 TCP/IP 协议栈要在 MCU 上运行，CC3100 提供驱动 Wi-Fi 射频相关的接口 API。因此，CC3100 虽然使用比较灵活自由，但使用起来，没有 CC3200 方便。

（2）博通公司的 BCM43362（第三方公司，选择此方案较多）(2011 年 3 月推出)

用 ST 公司的 32 位单片机 STM32 外接博通的 Wi-Fi 芯片 BCM43362，是目前双芯片分离式在性价比上，较好的选择。因此，市场上部分第三方公司，比较流行使用此方案。

目前，还有一些公司将 STM32F 与 BCM43362 集成在同一颗芯片上（实际为 2 颗），例：INVENTEK SYSTEMS 公司的 MX1081、MURATA ELECTRONICS 公司的 sn8200。

（3）Marvell (美满)公司的 88w8686、88W8801

88MC200 (MCU 200MHz) + 88W8686 (仅支持 802.11 a/g/b，但不支持 Wi-Fi 802.11n)

88MC200 (MCU 200MHz) + 88W8801 (Wi-Fi 802.11n)，用于小米 Wi-Fi 插座。

（4）国内乐鑫公司的 ESP8266 (2014 年 7 月推出)

国内乐鑫 ESP8266 采用 SDIO 2.0、SPI、UART 接口；集成 RISC 处理器、片上存储器和外部存储器接口；音频 I2S 接口；网络方面，支持 802.11 b/g/n、Wi-Fi Direct (P2P)、soft-AP 等。

ESP8266 是目前为止，最便宜的 Wi-Fi 芯片，已有部分公司开始使用。但该芯片性能，特别是工作的稳定性，不是很清楚。

3.1.3 采用第三方提供的 嵌入式 Wi-Fi 解决方案

第三方公司为了让用户更加“傻瓜式”的方法使用 Wi-Fi 模块，他们一般都会为 Wi-Fi 模块的 SDK 上再封装一套 AT 命令的 API 接口。即，用户通过通过 UART，配合 AT 命令就可以配置嵌入式 Wi-Fi 设备。例：进行 UART 和 TCP、/UDP 的透传、配置网络参数到设备等。例：

- 上海庆科：EMW3162（最新的嵌入式 Wi-Fi 模块）、EMW3280（成熟的 Wi-Fi 模块）
- 济南有人物联网技术：USR-WIFI232-T Wi-Fi 模块（高速 UART 转 Wi-Fi）
- ……

3.2 微控制器 MCU 的几种选择（采用双芯片分离式时）

3.2.1 ST 公司提供 STM32 系列微控制器（32 位 MCU）

如 STM32F10x 系列（Cortex-M3）、STM32F20x、STM32F40x 系列（Cortex-M4），性价比较高，在目前市场占有较高的市场。

3.2.2 TI 公司提供 Tiva 系列微控制器（32 位 MCU）

TI 最近推出的 Tiva 系列的 TM4C12x 微控制器，基于 ARM　Cortex-M4。

3.2.3 Atmel 公司提供 Sam 系列微控制器（32 位 MCU）

例：Sam D20 和 Sam D21 系列，基于最新的 ARM Cortex-M0+，功耗最低的 32 位 MCU（采用两级流水线）。目前，市场上，也有些产品，使用 Cortex-M0+的 MCU 专门来管理各个传感器模块，使系统功耗进一步降低（延长电池使用时间）。

3.3 嵌入式实时操作系统的几种选择

3.3.1 FreeRTOS

FreeRTOS 是一个迷你操作系统内核的小型嵌入式系统。作为一个轻量级的操作系统，具有源码公开、可移植、可裁减 、调度策略灵活的特点，可方便地移植到各种单片机上运行，基本满足较小系统的需要。相对于 uC/OS-II、embOS 等商业操作系统， FreeRTOS 操作系统是完全免费的操作系统。不仅可以用在开源的软件，还可以用在商业软件，而无需付费，也无需公布自己的代码。

FreeRTOS 作为一个轻量级的操作系统，功能包括：任务管理、时间管理、信号量、消息队列、内存管理等，可基本满足较小系统的需要。同 μC/OS-Ⅱ相比而言，FreeRTOS 虽然提供的任务间通讯手段较少，但是占用的空间和消耗的内存更小，运行效率更高。

FreeRTOS 在不同处理器上的移植类似于 uC/OS-II。可移植到多达 30 多种处理器平台上，如 ARM 公司的 ARM7、ARM9、ARM Cortex-M3/4，TI（德州仪器公司）的 MSP430 系列单片机、ST（意法半导体公司）的 ST32 系列单片机、Atmel （爱特梅尔半导体） 公司的 AVR32 单片机等。

3.3.2 uCOS II / III

uC/OS II 是一个可裁减的、抢占式、实时多任务内核，具有高度可移植性，特别适合于微处理器和控制器，是和很多商业操作系统性能相当的实时操作系统(RTOS)。为了提供最好的移植性能，uC/OS II 最大程度上使用 ANSI C 语言进行开发，并且已经移植到近 40 多种处理器体系上，涵盖了从 8 位到 64 位各种 CPU(包括 DSP)。uC/OS III 是一个第 3 代的系统内核，支持现代的实时内核所期待的大部分功能。例如资源管理，同步，任务间的通信等。uCOS II / III 为开源系统，但对于商业应用则是要收费的。

3.3.3 半导体公司提供的自家的 RTOS

一般情况下，半导体公司，也会根据自家的芯片，开发一个用于自家芯片的 RTOS（例：TI 的 ti_rtos）。不仅方便开发人员使用，同时也可以绑定用户。一般也是开源的，但如果从用户的角度，更倾向于使用第三方公开、免费、开源的 RTOS（例 FreeRTOS），因为比较自由。

3.4 嵌入式 TCP/IP 协议栈（状态机实现）的几种选择

对于嵌入式设备，实现嵌入式网络系统的一个重要技术就是网络协议栈。嵌入式 TCP/IP 协议栈，不仅要满足 TCP/IP 规范，还要适合嵌入式终端设备资源紧凑的特点，只保留通信所需要的基本协议，例如 TCP、UDP、ICMP、ARP、IP 等。目前，已有很多公司或组织机构提供了许多商业或开源的精简 TCP/IP 协议栈，可移植到嵌入式微控制器上。

嵌入式 TCP / IP 协议栈各层的功能介绍

TCP/IP 协议栈的层		协议	功能描述
应用层		TFTP、HTTP、SNMP、FTP、SMTP、DNS、Telnet 等	提供常用的应用程序，比如文件传输、电子邮件、远程登录等。
传输层		TCP，UDP	提供应用程序间的通信。其功能包括： 1．格式化信息流；2．提供可靠传输。
网络层		IP、ICMP	负责相邻计算机之间的通信，包括三方面功能： 1．处理来自传输层的分组发送请求； 2．处理路径、流控、拥塞等问题。 3．处理输入数据报；
网络接口层	数据链路层	MAC、SLIP	负责接收 IP 数据包并通过网络发送，或者从网络上接收物理帧，抽出 IP 数据包，交给 IP 层。
	物理层	IEEE802.2 、 IEEE802.3 、IEEE802.11 等	定义物理介质的特性，为设备之间的数据通信提供传输媒体及互连设备。

3.4.1 LwIP（博通的 BCM943362 采用 LwIP TCP/IP 协议栈）

LwIP 是一款专门用于嵌入式系统的开源 TCP/IP 协议栈。目前，使用较为广泛的一个协议栈。

LwIP 最大的优势在于可以移植到操作系统上，也可以在无操作系统的情况下独立运行，且代码量小。LwIP 实现的重点是在保持 TCP 协议主要功能的基础上减少对 RAM 的占用，它只需十几 KB 的 RAM 和 40K 左右的 ROM 就可以运行，这使 LwIP 协议栈适合在低端的嵌入式系统中使用。

3.4.2 uIP（联发科 MTK 的 MT7681 采用 uIP TCP/IP 协议栈）

uIP 去掉了 TCP/IP 协议中不常用的功能。代码容量小巧，实现功能精简。可用于 8 位、16 位的微控制器。另一方面，uIP 不完备的 TCP/IP，限制了其在一些较高要求场合的应用，如可靠性与大容量数据传输。

3.4.3 uC/TCP-IP

uC/TCP-IP 是 Micrium 公司针对嵌入式产品设计的一款 TCP/IP 协议栈。其功能较齐全，但代码量较大，所以主要用在 32 位或 64 位的处理器上。另外，uC/TCP-IP 是一款收费软件。

3.4.4 半导体公司提供的 TCP/IP 协议栈

例：Microchip（微芯）TCP/IP 协议栈，用在 ENC28J60 芯片，支持 8 /16/32 位 PIC 和 dsPIC 器件。

例：TI（德州仪器）的 SimpleLink TCP/IP 协议栈，用在 CC3200、CC3100、CC3000。

3.4.5 全硬件 TCP/IP 协议栈 (WIZnet 公司的 W5500 芯片)

3.5 开发环境的几种选择（以 ARM 内核的 MCU 为例）

3.5.1 开源的开发环境（在 Eclipse 下，搭建自己的环境）(替换 IAR)

开源环境：Elipse+CDT +GCC+OpenOCD，自己搭建开发环境，相对比较麻烦，但没有版权问题。

（1）Eclipse

Eclipse 是一个开放源代码的、基于 Java 的可扩展开发平台。就其本身而言，它只是一个框架和一组服务，用于通过插件组件构建开发环境，Eclipse 拥有较好的灵活性。

（2）CDT

CDT 是 Eclipse 插件，它将把 Eclipse 转换为功能强大的 C/C++ IDE。

（3）安装 GNU Tools for ARM Embedded Processors

由 ARM 维护的一套工具，包括 GCC 编译器，GDB 调试工具等。

- GCC 负责：预处理（Preprocessing）、编译（Compilation）、汇编（Assembly）和连接（Linking）。
- GDB 负责：基于命令行的、功能强大的程序调试工具。

（4）OpenOCD

OpenOCD 是一个开源的 JTAG 上位机程序，目前支持多种芯片，可以增加自己定义的工具的驱动。目的在于：使用 GDB 连接 OpenOCD，作为 GDBServer，从而方便了我们使用 GDB 进行调试。

（5）安装 J-Link 驱动 （FIDI（FT2232）或 ICDI）

在 J-Link 官网 下载驱动并安装，选择具体的操作系统（Windows，OS X，Linux）。

3.5.2 第三方开发的集成开发环境（例 IAR、Keil MDK）

常用的第三方集成开发环境：IAR（Embedded Workbench）、Keil（μVision）。

IAR 和 Keil 虽然可以方便地进行 ARM 处理器（例 STM32）的开发，但这些软件是商业软件（收费），而且仅支持 Windows。另外，两者的代码编辑功能似乎都不太强大。故，最好与其他代码编辑器、阅读器（例 Source Insight）配合使用。

另外 IDE 的版本也是一大问题。例：以前在旧版的 IAR 编译通过的代码，用新版的 IAR 编译不通过。因此，在使用第三方的集成开发环境时，还要注意版本号的统一。

3.5.3 半导体公司提供的开发环境

使用芯片供应商提供的开发环境：例 TI 的 CCS、Microchip 的 MPLAB、Atmel 的 Atmel Studio。

此类开发环境，一般功能较为齐全，但繁琐。并且，在配置较差的电脑上运行时，会比 IAR 卡。例 Atmel Studio，在启动运行时，就要花费较长的启动时间。

3.6 微控制器（MCU）与 Wi-Fi 芯片的几种通信方式

- 对于嵌入式 Wi-Fi，微控制器与 Wi-Fi 芯片的通信，一般采用：SPI 或 SDIO。
- 对于较高平台的普通 Wi-Fi，CPU 与 Wi-Fi 模块的通信，一般采用：USB、PCI/PCIe。

3.7 环保、低功耗的几种实现方式

3.7.1 硬件上实现低功耗

- 充分利用微控制器（MCU）的 Cortex-M4/3/M0+ 内核低功耗特性，让 Wi-Fi 报警装置能在正常状态、低功耗状态和休眠状态之间进行正常切换，使系统在保障通信畅通的同时节省了电量，有效解决了功耗和通信的实时性之间的矛盾。

- 电源模块采用 DC-DC 设计。

3.7.2 软件上实现低功耗

采用嵌入式 Wi-Fi 低功耗软件框架，使功耗达到最低，结合了以下几种控制方式：

- 微控制器的休眠模式选择功耗最低的深度睡眠模式；
- 进入深度睡眠模式之前，禁用未使用的功能模块及时钟模块，进一步降低功耗；
- 微控制器的工作频率越高、使能的时钟模块越多，则功耗越大。故在系统进入深度睡眠模式后，只剩 RTC 时钟仍在工作，时钟源选用频率最小的 32.768 KHz 振荡器，使功耗得到进一步降低；
- 使能 IO 口外部中断。只有发生相关中断时，才将微控制器从休眠模式中唤醒；
- 设备在不进行数据采集和数据通信时，尽可能地让 MCU 进入深度睡眠模式，使其功耗达到最低。

3.8 嵌入式 Wi-Fi 模块的相关测试（功耗测试）

（1）嵌入式设备的测试

包括性能测试、稳定性测试、功耗测试等。下面重点介绍功耗的测试方法：

（2）功耗的常用测试方法

通过在硬件电路中的电源地端，串联大功率、小阻值、高精度的采样电阻。通过串联分压原理，用示波器测得电阻两端的压降，可较为精确的测量出嵌入式设备，在工作时每个阶段消耗的电流和时间值，从而计算出电池的使用寿命。

另外，所使用的采样电阻阻值较小，故在采样电阻上所消耗的功耗很小，可以忽略采样电阻那部分的功耗。如图 3 所示。

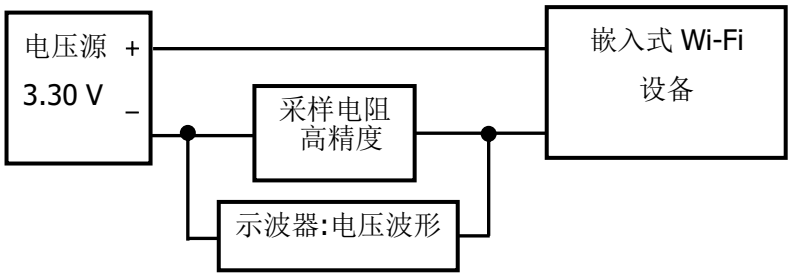

图 3　嵌入式设备的功耗测试框图

4、最终方案的确定(全部开源 All Open) (★★★★★)

4.1 开发环境、调试工具、调试接口 的确定

(1) 开发环境

- 方案 01：**Elipse + CDT 插件 + GCC 编译器（自己搭建开源 IDE 环境）**
- 方案 02：**IAR（需支付一定费用、或破解版）（简单、实用）**

(2) 代码编辑环境、阅读器

Source Insight 3.5 （面向项目开发的程序编辑器和代码浏览器）（与 IDE 配合使用）

(3) 调试工具及方式

调试工具：Jlink （分为两个接口调试方式 ：SWD、JTAG）

- **SWD**（2 线:SWDIO-数据、 SWCLK-时钟）
- **JTAG**（4 线：TMS-模式选择、TCK-时钟、TDI-输入线、TDO-输出线）

主流的 MCU 一般都支持 JTAG 和 SWD，SWD 节省 IO 口，故选用：**SWD 两线调试接口**

4.2 硬件模块的确定：MCU、Wi-Fi 芯片

(1) 电源模块

采用 DC-DC 模块（例：3.3 V 升压降压转换器 TPS63001），还可以采用专门的锂电池电源管理芯片（例：BQ24230）

(2) 微控制器（MCU）（综合考虑：功耗、性能）

- ST 的 32 位单片机（性价比较高），例 STM32F10x（Cortex-M3）、STM32F40x（Cortex-M4）
- TI 公司的 CC3200 单芯片 SoC Wi-Fi 解决方案

(3) Wi-Fi 芯片（网络处理器）

- TI 公司的 CC3200 单芯片 SoC Wi-Fi 解决方案
- TI 公司的 CC3100 Wi-Fi 射频前端芯片
- 博通公司的 BCM43362 单芯片 SoC Wi-Fi 解决方案
- 国内乐鑫公司的 ESP8266 Wi-Fi 射频前端芯片

4.3 软件模块的确定：RTOS、TCP/IP、文件系统

经过第 3 章节的介绍和分析：

(1) 嵌入式实时操作系统

选择 FreeRTOS（开源、免费，用于 MCU 的嵌入式实时操作系统）

(2) 嵌入式 TCP/IP 协议栈

- 若使用 TI 公司的 CC3200：采用 SimpleLink（TCP/IP 协议栈）
- 若使用博通公司的 BCM43362：采用 LwIP （TCP/IP 协议栈）
- 若使用乐鑫公司的 ESP8266：采用 uIP （TCP/IP 协议栈）

(3) 文件系统

FatFS （开源、免费，用于在小型嵌入式系统中实现 FAT 文件系统。）

4.4 最终方案的确定：2 种具体解决方案（开源、免费）(★★★)

主要从芯片的性能、功能、价格、技术支持，以及方案的成熟度等方面综合考虑。
另一个重点考虑的问题：**开源**。使用开源的最大好处：比较自由。

4.4.1 单芯片 SoC、Wi-Fi、物联网解决方案（采用 TI 公司）

CC3200 + FreeRTOS 操作系统 + SimpleLink TCP/IP 协议栈 + FatFS 文件系统

其中，CC3200（TI 公司）集成了 MCU（Cortex-M4）+ Wi-Fi 网络处理器

4.4.2 双芯片 Wi-Fi、物联网解决方案（采用 ST 公司+博通公司 组合）

STM32F 单片机（M3/M4 核） + 博通的 BCM943362（Wi-Fi 网络处理器）
+ FreeRTOS 操作系统 + LwIP TCP/IP 协议栈 + FatFS 文件系统

其中，STM32F 单片机，可以选择 32 位的 STM32F205（Cortex-M3）、或 STM32F407（Cortex-M4）。
另外，这种方案的最大好处：Wi-Fi 网络处理器，不一定采用博通，也可采用其他公司，比较自由。

5、Wi-Fi Connectivity 连接的几种解决方案(★★★★)

如何以最简单、快捷、自然的方法，把网络参数（SSID、密码等）配置到嵌入式设备，并让设备自动链接到云服务器。即：好的用户体验（傻瓜式的配置方法，降低使用门槛），是我们要重点考虑的问题。

- 简单方法：直接把相关的网络参数（例 SSID、加密方式、密码），保存到 Flash 或 EEPROM 中。
- 规范方法：通过 Profile 或 XML 文件，在 Flash 或 EEPROM 中存储网络参数。

5.1 方案 01： 通过 PC 机专门软件，通过 UART 进行配置。

即通过在 PC 机上专门的 PC 软件，把路由器的网络参数，通过 UART 串口发送到嵌入式 Wi-Fi 设备。此种配置方法：可靠、成熟。但需要微控制器提供 UART 接口（虽然嵌入式设备一般都提供 UART 接口，作为 Wi-Fi 的调试信息的输出显示）。另外，用户体验不好。

5.2 方案 02：通过 NFC 无线模块进行配置（但一般不使用此方案）

即在嵌入式设备，通过 IIC 外接 NFC 模块。利用手机上的 NFC 功能和专门的 APP，把路由器的网络参数，一键配置到嵌入式设备。

此方法，虽然与“方案 01”相比，有较好的用户体验，但增加硬件成本、也使硬件复杂化。（越简单的东西，越可靠）

5.3 方案 03：软 AP（SoftAP）+手机 APP (目前流行的 “一键配置”)

基本方法：（通过 LED 不同显示方法，提示配置过程）

- 首先，让嵌入式 Wi-Fi 设备工作在 AP 模式（接入点），并开启一个用于接收网络参数的 TCP 或 UDP 端号，及端口号。（一般采用 UDP 方式,因为 Wi-Fi 在局域网的 UDP 传输，很稳定、不丢包）
- 然后，手机（客户端）接到此嵌入式设备发出的 AP（服务器端），使用专门的手机 APP 通过 UDP 把相关的网络参数（SSID、加密方式、密码）发给嵌入设备。
- 最后，嵌入式 Wi-Fi 根据接收到的网络参数，工作在 Station 模式，连接到路由器（网关），并链接到云服务器。

另外，针对于 iOS 和 Android 两个主流手机操作系统，故需要提供两款 APP。在嵌入式设备联网前，专门用于配置嵌入式设备的相关网络参数。

优点：配置过程简单、快捷。

缺点：需要专门开发多个版本的 APP（针对不同的系统：iOS 版、Android 版等），而且同一手机系统，还有不同版本的支持、维护问题。（例 Android 就要分很多的版本）

最大的缺点在于：用于配置的手机，要先断开与路由器 AP（热点）的连接，之后，才连接到嵌入式 Wi-Fi 设备发出的 AP。（然后，手机再重新连接到路由器的 AP）

5.4 方案 04：直接操作 Wi-Fi PHY 层 +手机 APP(更好的一键配置)

基本方法（通过 LED 不同显示方法，提示配置过程）

- 通过在无线网络 PHY 层上的专有协议，绕过(bypass)网络协议栈、Wi-Fi 驱动和 MAC 层，直接能够与 CC3200 的 PHY 层进行通信。
- 然后再通过定义和管理 RX 过滤功能（添加 MAC 地址过滤、IP 地址过滤），减少传输到主机的数据流量。（即嵌入式 Wi-Fi 设备监听特定的数据流）
- 手机 APP 通过相关 Wi-Fi 驱动，发出特定的数据流（数据帧）
- 嵌入式 Wi-Fi 设备只对过滤后的数据帧进行相应的分析和解码（得到 SSID、密码等网络参数）
- 最后，嵌入式 Wi-Fi 设备通过网络参数，连接到指定网关（路由器）的 AP（扫描、认证、关联），自动链接到云服务平台。

虽然，此方案也需要专门开发多个版本的 APP。

但是，比方案的最大好处：手机在通过 APP 将网络参数配置到嵌入 Wi-Fi 设备时，手机本身不需要断开与路由器 AP 的连接。（故有较好的用户体验）

另外，此方案，需要芯片供应商，提供相应的 Wi-Fi 驱动。

5.5 方案 05：WPS 现在改名为 WSC（可再细分为两种方法）

WPS（Wi-Fi Protected Setup）能够为接入点及 WPS 客户端（嵌入式设备）自动配置 SSID 及 WPA 案例密钥。

具体实现方法：WPS 可再细分为以下 2 种实现方法

（1）输入 PIN 码（个人识别码）：即 8 位十进制数（在路由器的背面标有）

再根据 PIN 码的不同提供者，再分为以下 2 种方法：

- 方法一：在手机（客户端）中，输入路由器（AP）的 PIN 码，来实现 WPS 安全连接。
- 方法二：在路由器（AP）中，输入手机产生的 PIN 码，来实现 WPS 安全连接。

（2）按钮配置 PBC：

- 首先，在路由器（AP）的配置网页中，启用路由器的 WPS 功能。
- 按下客户端（无线网卡、手机）上的 WPS 按钮（可以为模拟按钮），搜索 WPS 网络。
- 在指定时间内（一般为 2 分钟），按下路由器背后的 WPS 按键。
- 片刻后，WPS 安全连接成功建立。（一般 10 秒内可完成）

总结：这种方案，最大的好处：降低用户的使用门槛。即用户无需了解 SSID、WPA 是什么东西，简化 Wi-Fi 无线的安全设置和网络管理。

另外，目前开始流行的 Wi-Fi Direct（直连），也是采用 WPS 简化并建立简单而安全的连接。（Wi-Fi Direct，允许无线网络中的设备之间可以相互通信连接，而不需要路由器的参与，与蓝牙类似）

5.6 方案 06：SoftAP + 嵌入式 Web 服务器+手机浏览器 (不需要 APP)

这种解决方案，与路由器的参数配置极为类似。

基本方法（通过 LED 不同显示方法，提示配置过程）

- 首先，让嵌入式 Wi-Fi 设备工作在 AP 模式（接入点）,并使用其内部的嵌入式 Web 服务器。
- 然后，手机、平板或其他无线终端（客户端）接到此嵌入式设备发出的 AP（服务器端）。
- 在手机或平板的浏览器地址栏输入 http://192.168.0.1 或 10.10.10.1(嵌入式 Web 服务器的网页地址), 可以把网址的二维码印在嵌入设备上，通过扫描二维码，进一步提高用户体验。
- 填写用户名和密码（与访问路由器一样，一般都默认为 admin）,进入后，选择设备的工作模式（AP、Station、Wi-Fi Direct），并在相关菜单栏内，填入指定路由器的 SSID、加密方式、密码等参数。
- 配置成功后，嵌入式 Wi-Fi 设备，通过 Profile 或 XML 文件，存储接收到的网络参数。然后，一般会自动重启。
- 最后，工作在 Station 模式，连接到路由器（网关），并链接到云服务器。

总结：这种方案，最大的好处：开发人员，不需要另外开发 APP，只要有浏览器就可以实现配置。故不需要开发专门的 APP，就可以同时支持主流的、不同手机操作系统。（这种方法，个人认为后面将会有更多种的应用）

另外，利用嵌入式 Web 服务器，还可以通过它查看嵌入式 Wi-Fi 设备的状态（例 Wi-Fi 信号强度、设备上传感器的采样数据和工作状态等）

5.7 方案 07：方案 6 的扩展方案 (不需要手机 APP、更好体验) (★★★)

基本方法（通过 LED 不同显示方法，提示配置过程）

- 首先，让嵌入式 Wi-Fi 设备工作在 Station 模式。
- 在 Station 模式下，让嵌入式 Wi-Fi 设备，扫描附近环境中的 Wi-Fi 热点，并保存到 Profile 配置文件中。
- 然后，再让嵌入式 Wi-Fi 设备工作在 AP 模式（接入点）,但是此 AP 不加密（即不使用 WEP、WPA）。
- 智能终端，连接到此 AP 后，直接在浏览器输入嵌入式 Web 服务器网址或扫描二维码。（不需要 Admin 密码登录）
- 通过网页，查看刚才的“扫描结果”，选择自己特定的 AP 对应的 SSIP，并在网页上连接自己 AP 的密码。嵌入式设备保存网络参数到 Profile 配置文件。
- 最后，让设备工作在 Station 模式，连接到路由器（网关），并链接到云服务器。

注意：只有在需要配置网络参数时，才使用此方法（例：长按某个按键 10 秒后，进行此配置状态）此种方案，有更好的用户体验（过程与手机接到路由器的 AP 类似），为“方案 6”的进一步扩展。

5.8 方案 08：利用 Wi-Fi Direct、mDNS（Bonjour）、UpnP、（有待学习）

6、联网的嵌入式设备，需要重点解决的几个问题

6.1 问题 01：构建“完整”生态系统（待续）

构建“完整”生态系统（集硬件和软件为一体）：以小米为例：小米智能插座、小米**路由器**、小米云服务、小米手机 APP 把嵌入式 Wi-Fi 设备与云服务注册的个人帐号绑定。（资源的重复利用）

在没办法像华为手机、小米手机、苹果手机那样，利用智能手机构建自己的生物圈（跟手机绑定的云服务器）时，我们可以利用第三方提供的平台（例：微信的公众平台）

6.2 问题 02：如何更充分地利用现有“用户的”资源（待续）

“用户的”资源：（充分利用用户资源，降低用户的使用成本）

01）用户现有的联网网络　02）用户现有的家用路由器　03）用户现有的智能终端（手机、平板）

6.3 问题 03：如何更充分地利用现有“云服务”资源（待续）

云服务平台（亚马逊提供、阿里云、新浪云）、微信公众平台的物联网接口、提供通用的、特殊功能的云服务网站等

6.4 问题 04：如何拥有更“自然”的用户体验（待续）

01）最终目的：最大程度地降低最终端的用户（消费者）的使用门槛，拥有更自然的用户体验。

例：让用户在使用嵌入式 Wi-Fi 产品时，甚至没有感觉到背后“云”的存在。例如，用手机控制 Wi-Fi 插座时，用户不需要知道：手机 APP 发出的命令，要先通过移动互联网发给绑定的“云服务器”，再由云服务器通过网络，把控制命令发给绑定的设备（Wi-Fi 插座）。

02）嵌入式设备通过哪种方式，以最简单、自然、安全的方式连接到 Internet（云服务器）并让设备自动链接到云服务器。

03）云服务器框架：如何规范地管理这些联网的、各种各样的嵌入式设备。

规范和标准，将随着物联网的发展，来变成熟。故也是考虑重点。

04）对于半导体产商，要考虑的重点之一：如何降低芯片使用者（开发人员）的开发门槛。

让产品开发人员，在只知道基本的网络知识情况下，就可以利用半导体产商提供的解决方案，以最快的速度实现嵌入式 Wi-Fi 的基本功能。例：像用 AT 命令的方式，来简化嵌入式设备接到云服务器的过程。

6.5 问题 05：两大重要战场（关口）：路由器、浏览器（待续）

一、路由器相关

未来的路由器，将会变得更加智能、并且可能为用户开放一些自定义的窗口，以灵活实现一些特殊的应用功能。（即路由器多元化发展！）

二、浏览器相关

浏览器在将来很有可能成为笔记本和智能手机的唯一客户端（例：Chromebook 网络笔记本）。

浏览器变成最后唯一的 APP（即不再区分是 Android 系统，或是 iOS 系统）网页版的 ERP 管理软件、网页版的 QQ、网页版的微信。

6.6 问题 06：通信协议：嵌入式设备、云服务器、用户终端（待续）

嵌入式设备与云服务器间的通信协议；云服务器与用户终端（手机、平台，微信等）之间的通信协议；

HTTP、HTTPS、SSL/TSL

6.7 问题 07：如何将高平台（例 Linux）实现的功能或驱动，移植到 32 位的低平台 MCU 上

7、嵌入式云服务器（Web 服务器）的设计与实现（物联网系统的搭建）（待续）

嵌入式云服务器是专为嵌入式设备提供指定服务的远程服务器。此类服务器通过为用户提供相应的 API 接口，实现与嵌入式终端设备的数据通信。当各种嵌入式设备实现网络化后，设备将自动链接至云服务器。同时，用户可以通过 PC 机、手机、平板电脑等智能终端来访问服务器，随时随地获取嵌入式设备的相关数据以及监控设备工作状态等。目的是为了使设备数据的接入、存储和监控变得轻松简单。

7.1 云服务器的整体结构

嵌入式 Wi-Fi 设备与云服务器、手机客户端与云服务器的数据交互方式都采用 C/S（客户端/服务器）点对点的架构模型。C/S 架构是 IP 网络上一个非常重要的应用模式，它将服务请求功能（客户端）和服务提供功能（服务器）分开。其中，云服务器对数据进行分析管理，客户端向云服务器发送请求，服务器响应请求后将请求内容返回给客户端实现两者的数据交互，两者分别发挥各自优势相互配合紧密合作。如图 4 所示，嵌入式云服务平台的系统架构图；图 5 所示，为客户端与云服务器数据交互的结构图。

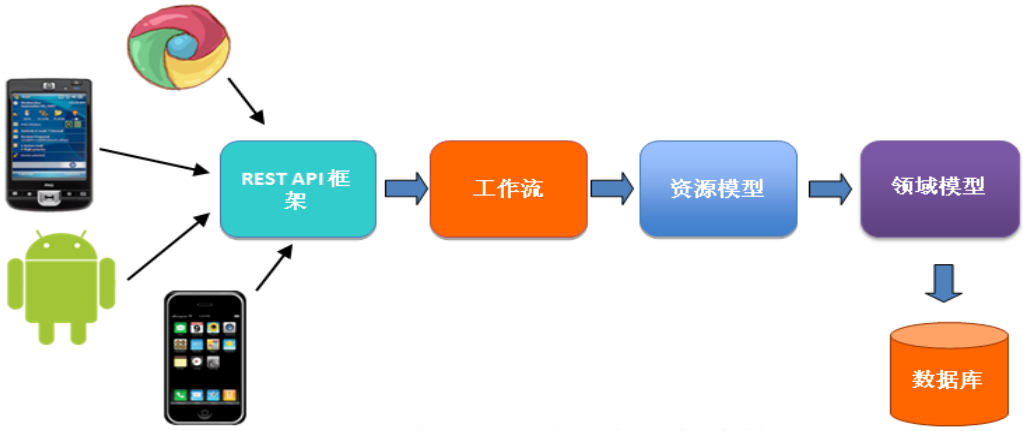

图 4 嵌入式云服务平台的系统架构图

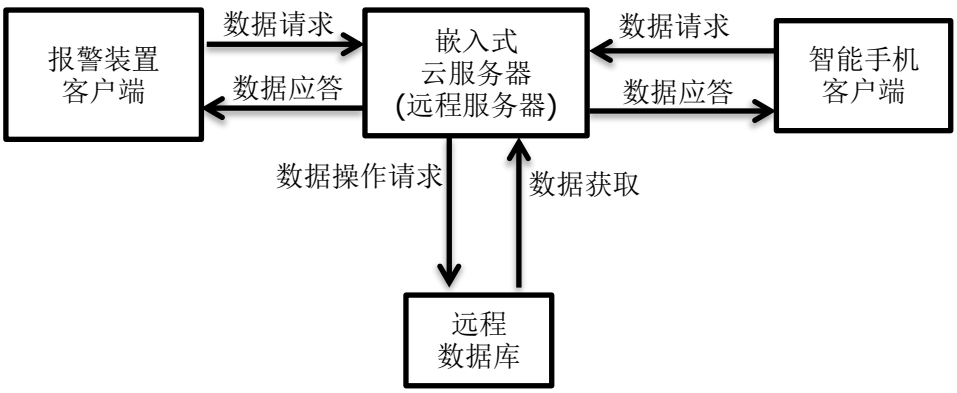

图 5 客户端与云服务器数据交互的结构图

7.2 云服务器基本实现方案

在此云服务平台上，进一步开发专门用于嵌入式 Wi-Fi 设备的云服务器，即通过 HTTP 协议的 GET/POST 请求与客户端进行动态数据交互的 Web 服务器系统。

Web 服务器解析 HTTP 协议，并提供一个可以执行服务器端程序和返回响应的环境。当 Web 服务器接收到一个 HTTP 请求，服务器只是单纯地把请求传递给可以很好处理请求的程序（例：服务器端脚本），再由服务器端程序执行相关的事务处理、数据库连接和消息等功能，最后返回相应的 HTTP 响应，将 HTML 数据返回给请求端。

基本方案：Web 服务器采用 PHP 语言开发，使用 MySQL 作为数据库。软件开发环境选择 Windows 7 + Apache + PHP + MySQL 架构。

7.3 使用第三方提供的云服务平台（待续）

阿里云（有公网固定的 IP）、百度云、腾讯云、新浪云
花生壳：提供二级域名服务

7.4 建立自己的云平台服务（待续）

JSON+REST

搭建了一个简单的 REST 服务， 可以简单的做一个最小的物联网系统， 将嵌入式 Wi-Fi 设备（基于 MCU）连接到互联网。

8、用户交互方式：智能终端（手机/平板）、微信

最终与用户（消费者）打交道的“用户交互方式”：iOS APP、Android APP、第三方（微信公众平台）。

8.1 基于智能手机 APP 的交互方式（Android 版和 iOS 版）

以 Android APP 为例：将此 Android APP 系统的结构分为三个模块：应用配置、功能模块、用户界面布局，如图 6 所示。

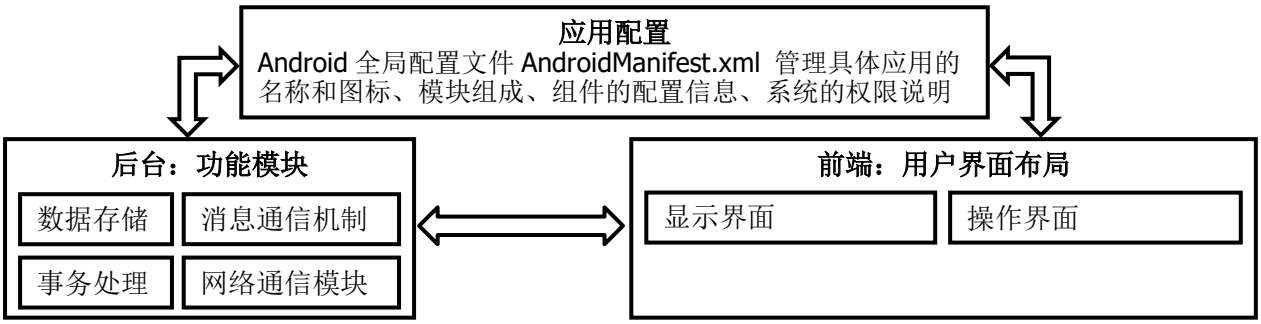

图 6　Android APP 系统的结构的基本框图

8.2 基于微信公众平台的交互方式（利用第三方平台）

利用微信公从平台提供的物联网接口 API，实现与用户的交互。在微信里，通过和设备对话来控制设备。每个嵌入式设备（物体）都有个二维码作为设备 ID 号。

通过云服务器和微信公众帐号一同对嵌入式智能设备进行控制和状态读取。

最大好处：微信公众平台：同时支持 Android 和 iOS。不需要自己开发 APP，故不需要管理和维护 APP 版本、功能等。即将 APP 的版本和开发，转移给第三方（微信公从平台）。这样做，可带来很多便利。

缺点：自己被微信给捆绑住。

8.3 基于 Web 应用程序的交互方式（更通用，支持不同手机系统）

这些程序最大的优点就是不需安装 APP， 也不区分是什么操作系统，只要有浏览器就可以运行。

相当于，浏览器就是一个最终的 APP。原来不同 APP 上实现的功能，尽可能移到浏览器上来实现（即设备输入和输出都由浏览器来控制）

9、嵌入式 Wi-Fi 的实际应用（具体案例）

9.1 在广域网（互联网）中的具体应用（以智能家居为例）

9.1.1 案例 01：百度云摄像头（在无线安防的应用）

百度云摄像头：不仅可通过 PC 机，还可以通过移动终端，进行随时随地采样监控。并且将采样所产生的数据都存储在百度云中，可以为以后的大数据处理分析，提供原材料。另外，从用户角度，不需要考虑存储空间的问题，也不用担心维护（数据、设备丢失）问题等。

9.1.2 案例 02：Wi-Fi 智能插座

例：小米智能插座（Marvell (美满)的 88MC200 (MCU) + 88W8801 (Wi-Fi 802.11n)，通过与小米手机/平板、小米路由器、小米云服务器，建立起来的一个完整的物联网生态链。

9.1.3 案例 03：基于广域网的音乐播放器（Air Play）

随着网络带宽的不断增长，一些只能在局域网内的应用，可以扩展到广域网。例：局域网内 Wi-Fi 音响（例 AirPlay 功能，将手机上音乐推送给 Wi-Fi 音箱），可以扩展为基于互联网（广域网）的 Wi-Fi 音箱（例：把百度音乐或 QQ 音乐（音频流）通过互联网推送到本地的 Wi-Fi 音箱）。

关键：音频流缓冲，不能有间断（不能有一丁点的噪音）

9.2 在局域网中的具体应用

9.2.1 Wi-Fi Direct（Wi-Fi 直连）的应用

一、Wi-Fi Direct 基本功能（改进型的 Ad-Hoc 网络）

- 允许无线网络中的设备，无需通过无线路由器（AP），即可随时随地地建立对等连接。
- 可以支持一对一直连，也可以实现多台设备同时连接，组建小组（与基础设备 BSS 类似）。
- 提供并发机制：设备在加入小组的同时，保持 WLAN 基础设施连接。

二、Wi-Fi Direct 具体功能：

- **创建小组：**由一部 Wi-Fi Direct 设备在创建连接时，自动创建，并负责管理此小组。
- **设备发现：**通过扫描技术，交换设备信息的机制，并建立连接。
- **服务发现：**向其他 Wi-Fi Direct 设务通报自己可提供的功能、服务。（即 Bonjour、UpnP、Web 服务发现等），相当于把小组内的设备，按功能、服务进行分类。
- **能源管理：**采用 P2P-PS 机制，引入机会节能、缺席通知等。

三、Wi-Fi Direct 案例分析：小米的空气净化器

小米的空气净化器（2014 年 12 月初发布）**Wi-Fi Direct = SoftAP + WPS + Station**

智能手机用 Wi-Fi Direct 与嵌入式设备通信（空气净化器），进行控制、通信、数据交互等。即通过手机显示嵌入式设备的状态信息，并能对嵌入式设备进行相关控制或参数配置。

嵌入式 Wi-Fi 在局域网的应用，将会减少 USB 和蓝牙的部分现有应用领域。(例无线硬盘、Wi-Fi 直连)

9.2.2 Wi-Fi Beacon 技术的应用 (数据收集 + 大数据分析)

Wi-Fi Beacon 覆盖范围(200 米) 比蓝牙的范围要大多。但目前用蓝牙实现的信息推送较多，例苹果的 iBeacon。因为蓝牙更省电（一个纽扣电池可以用数年），更重要的 Wi-Fi 不能使用纽扣电池，而要用类似于锂电池类的电池。

不管是 Wi-Fi Beacon，还是蓝牙 Beacon，都将会有更广泛的应用。

另外，使用 Beacon 技术向用户推送信息（广告）的过程，也是在收集数据的过程。故，可以在收集到的数据之后，再进行数据分析，找出数据的价值。（这个才是重点！）

广告系统将更加智能：更加主动、有针对性向特定的潜在客户发送客户感兴趣的商品广告。

以淘宝为例：当你在淘宝上，挑先一样产品（例打印机）时，淘宝系统将至少通过以下两种方式主动推送广告给您。例：把打印机相关的淘宝链接和图片，发到您的邮箱。更厉害的是，当你在浏览一些网页（例看优酷视频）时，网页（视频）旁边的广告窗口，就会一直循环翻页显示打印机相关的广告。（在亚马逊购物时，也一样）

蓝牙 Beacon 或 Wi-Fi Beacon 可用于 数据收集 +大数据分析 + 商品推荐。

案例 01：餐厅

提供推送服务：例我到餐厅吃饭，点菜的时候，用 Beacon。用户可以用手机扫描得到具体的菜单，并且每个菜都有详细的特色介绍、评分等（重点是可以保存这些菜单信息到手机里，方便顾客下次再来吃）（主动推送顾客感兴趣的广告）完善用户体验的同时，提高服务质量。商场也开始部 Beacon，给顾客推送基于位置的内容和个性化信息。

另外，餐厅基于 iBeacon 的顾客忠诚度系统。该系统可以追踪关于顾客的信息：比如最喜欢的位置、在餐厅中用餐市场以及喜欢的菜品等。

案例 02： 博物馆 或旅游景点

给到博物馆参观的智能手机用户提供展览会指南，当参观者站在一副画前，它就会自动推送与该画相关的详细信息。

案例 03：电影院

电影院使用 iBeacon 来给观影人指路，帮助他们找到自己的位置，同时推动优惠活动信息。还可利用这款信息来收集相关信息，以理解客户的趋势和行为。

9.3 基他的具体应用……（待续）

10、其他类型：无线通讯解决方案（待续）

10.1 蓝牙 Bluetooth（基于 IEEE802.15.1 低功耗局域网协议）(待续)

蓝牙与 Wi-Fi 有很多类似的功能，故两者之间存在很多竞争（功耗竞争、局域网连接 Wi-Fi Direct 等）同时，由于他们各自的优点、及部分互补，故是都是目前主流的无线通信技术。

市场流行的几种蓝牙解决方案：

- TI 的 CC2540、CC2541（CC2540 是 MCU 集成低耗电蓝牙(BLE)无线标准）
- Broadcom (博通)的 BCM20732 WICED Smart 低耗电蓝牙模块，则是集成了 Cortex-M4 MCU、RF 收发器、蓝牙 BLE 的堆叠层于单颗芯片上。
- CSR 的蓝牙解决方案。

蓝牙 Bluetooth 能够应用在耳机、玩具、家庭自动化、健康监测、运动仪器等产品，若另外搭配 MCU，则可应用在穿戴式产品。

10.2 ZigBee（IEEE802.15.4 标准的低功耗局域网协议）（待续）

例：TI 公司的 CC2530（或 CC2531）、CC2533（带 USB2.0）或 CC2538（Cortex M3+ Zigbee SoC）

10.3 低于 1GHz 的无线通信：Sub 1G + 6LoWPAN（Ipv6）(待续)

一、简介：TI 公司的 CC1120、CC1180

低成本、低功耗，低于 1GHz 的 6LoWPAN 网络处理器，此处理器用最少的开发成本实现 6LoWPAN 的功能性。

6lowpan 基于 IP 技术-互联网的标准技术。同时，采用 6lowpan 可是实现与其它网络的链接，持与其它 802.15.4 设备的互通，同时也支持和其它 IP 网络的互通，如以太网和 Wi-Fi。

二、解决方案：6LoWPAN（IPv6）+ CC1120（433MHz）