STM32W无线射频ZigBee单片机原理与应用

沈建华　郝立平　编著

北京航空航天大学出版社

STM32W 无线射频 ZigBee 单片机原理与应用

沈建华　郝立平　编著

北京航空航天大学出版社

内 容 简 介

STM32W 是基于 IEEE 802.15.4 标准和 ARM Cortex-M3 内核的高性能、低功耗、内嵌网络协议栈的无线射频单片机。全书共 7 章：第 1 章介绍几种短距离无线网络技术及标准、协议；第 2 章介绍 STM32W108 芯片及其电气特性；第 3 章和第 4 章详细描述 STM32W108 的系统模块、射频模块和片内外设的功能、原理和编程结构；第 5 章介绍 STM32W108 的开发环境和工具；第 6 章详细说明 STM32W108 的 MAC、ZigBee(包括安全)和 RF4CE 等网络协议库的结构和使用方法；第 7 章介绍基于 STM32W108 的硬件设计、应用模块和开发套件，并列举了 2 个应用设计实例。

本书适合于从事无线传感网、ZigBee/RF4CE、物联网、无线仪器仪表、无线遥控等应用系统开发的工程技术人员学习参考，也适合作为无线传感网、物联网等实践课程的教材，以及 STM32W 的培训、自学用书。

图书在版编目(CIP)数据

STM32W 无线射频 ZigBee 单片机原理与应用/沈建华，郝立平编著. -- 北京 ：北京航空航天大学出版社，2010.9

ISBN 978-7-5124-0211-9

Ⅰ. ①S… Ⅱ. ①沈…②郝… Ⅲ. ①无线电信号—射频—信号识别②单片微型计算机 Ⅳ. ①TP911.23②TP368.1

中国版本图书馆 CIP 数据核字(2010)第 174597 号

STM32W 无线射频 ZigBee 单片机原理与应用

沈建华 郝立平 编著

责任编辑 刘 晨

*

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号(邮编 100191) http://www.buaapress.com.cn

发行部电话：(010)82317024 传真：(010)82328026

读者信箱：emsbook@gmail.com 邮购电话：(010)82316936

涿州市新华印刷有限公司印装 各地书店经销

*

开本：787 mm×1 092 mm 1/16 印张：23.75 字数：608 千字

2010 年 9 月第 1 版 2010 年 9 月第 1 次印刷 印数：4 000 册

ISBN 978-7-5124-0211-9 定价：45.00 元

前　言

无线通信技术由于其灵活、易用性一直是嵌入式开发应用的热点，从简单的无线遥控到RFID、WSN、WiFi以及GPRS/3G等，每种技术都有其技术特点和适用范围。从实际情况来看，简单的无线应用还是广泛存在的，如遥控门窗、车库等，这类应用从RF到通信协议几乎没有国际标准，有各个厂家自定的产品标准。但随着应用的日趋复杂，对应用系统稳定性、可靠性、兼容性要求越来越高，符合相关国际标准、协议的无线技术得到快速发展。目前得以广泛应用的无线技术，如RFID、WiFi、GPRS/3G等，都对应有较完善的国际标准或行业标准，从物理层到应用层，通信协议的完整性有保证，使得各个厂商的产品可以互联互通，既有利于技术的推广应用，也有利于因为规模效应而降低产品成本。

无线传感网(WSN)技术提出已经有很多年了，其低功耗、低速率、自组网的特点正好弥补了RFID、WiFi等技术的不足，一直被技术、应用等许多领域所看好，但发展却一直比较缓慢。其中的原因，既有WSN技术本身在不断完善，又有除了802.15.4 MAC标准，网络层、应用层协议没有相关标准，使得真正的大规模应用很难展开。前几年的WSN应用，基本上是基于802.15.4 MAC标准，各个厂商或用户自主开发的独立封闭应用，几乎没有可以互联互通的独立产品。

我们从2003年开始从事WSN的研究与应用，先后在MSP430 + CC2420/CC2520、CC2530/2531等硬件平台上，移植了TinyOS 1.0/2.0、ZStack等协议，并自主开发了具有自动路由功能的WSN协议栈、WSN/ZigBee教学、研究平台以及一些独立的应用系统。总体感觉是：WSN/ZigBee的开发应用，由于网络协议的复杂性和实际应用的可靠性要求，对开发人员的技术要求较高，必须熟悉这些协议(而这些协议本身也不完善)，再进行移植、修改和应用开发。无线传感网技术要大规模地普及应用，必须有比较稳定、成熟的网络协议栈，就像从事嵌入式以太网应用开发的人员，一般也是使用稳定、现成的网络协议栈(如TCP/IP)，直接进行应用开发。

ZigBee的目标，就是试图改变上述这种局面，但其发展过程也比较缓慢。ZigBee协议从草案到目前的ZigBee Pro版本，经历了多次大幅度的修改，早期的版本很不完善。几年前，不同厂商的ZigBee模块产品，虽然都声称是ZigBee标准的，但他们之间甚至也不能互联，这种情况一直到ZigBee 2007 Pro的推出，这是一个比较完善的协议标准，多家公司推出的基于ZigBee 2007 Pro标准的协议栈，证明了其良好的性能和兼容性，典型的就是Ember公司的协议栈。由于ZigBee协议的完善及其协议栈的成熟，目前WSN/ZigBee的推广应用已经到了一个转折点，大规模的应用即将开始。美国能源部(U.S. Department of Energy)根据其最初智能电网(Smart Grid)开发标准框架，已经把ZigBee Smart Energy作为家庭局域网络(HAN)能源设备通信标准。

意法半导体(ST)公司于2009年底推出的STM32W系列无线射频WSN/ZigBee单片机，

采用 32 位 ARM Cortex-M3 内核，片上整合 2.4 GHz IEEE 802.15.4 收发器和低功耗 MAC、AES 128 硬件加密引擎，STM32W108 内置 128KB Flash 和 8KB SRAM，具有高性能、低功耗的特点。另外，STM32W108 的发射功率软件可调，最大可达＋7 dbm，是目前 ZigBee 单芯片中最大的，不用外加射频功放（PA），通信距离就可以达到百米左右。最值得注意的是，STM32W 采用硬件固化协议栈的方法（三个版本芯片分别固化了 ZigBee、RF4CE 和 MAC），屏蔽了 RF 部分的寄存器，使用户不必理解、移植有关 WSN/ZigBee 协议栈以及射频部分的技术细节，就可以直接利用协议栈提供的 API 进行自己的应用开发，大大简化了应用系统开发，有利于产品快速上市。如 STM32W108CBU61 芯片固化了由 Ember 公司提供的、经过 ZigBee Alliance 认证的 ZigBee 2007 Pro 协议栈，具有优异的性能和良好的兼容性，可以和其他经过 ZigBee Alliance 认证的第三方产品互联互通。可以相信，STM32W108 的推出，将对无线传感网、物联网应用产生巨大的推动作用。

为了尽快在中国推广 STM32W，ST 公司授权委托华东师范大学计算机系嵌入系统实验室，整理出版一本 STM32W 的中文书。此书的主要内容取自 ST 提供的原版资料（datasheet、user manual 等），应用实例部分取自我们自主开发的一些项目和产品。由于 2.4GHz RF 的特殊性，为方便广大读者快速进行评估、测试，我们开发了与本书配套的开发套件，并由 ST 公司的中国第三方合作伙伴上海庆科信息技术有限公司（www.mxchip.com）负责销售，有需要的读者可以直接和他们联系。

本书共 7 章，其中第 4 章和第 5 章由煤炭科学研究总院上海分院郝立平编写。在本书成稿过程中，得到了 ST（中国）RF 产品经理 Nicon Yang、赵鑫，IAR（中国）总经理 Tony Ye，上海庆科信息技术有限公司王永虹、徐炜，以及北航出版社胡晓柏的大力支持。华东师范大学计算机系周朝丽、洪唯银、许青青、李吉、匡鑫、邢诗宁、镇咸舜等做了很多代码验证、资料整理工作，在此向他（她）们表示衷心的感谢。

由于时间仓促和水平所限，错误之处在所难免，恳请各位读者批评指正，以便我们及时修正。

沈建华

2010 年 7 月于华东师范大学

目　录

第1章 概 述

1.1 标准无线射频技术

目前存在着很多种类的无线数据传输技术，有些技术有着广泛应用，彼此竞争市场，也有些技术是为了一些特殊的应用而设计，在这些特殊应用方面有着突出的优势。

WiFi、蓝牙(Bluetooth)和 WSN/ZigBee 是目前 3 种常见的短距离无线通信技术，表 1.1 列出了它们的主要特征。

WiFi 是目前最成功的无线局域网(WLAN)系统，它基于 IEEE 802.11 技术标准，通过近几年的快速发展，基础架构设施已经比较完善，无线接入热点(AP)覆盖已经很广，价格也很便宜。这样的系统可通过个人权限接入无线互连网，并访问本地网络中的其他系统，比如其他计算机、共享打印机和其他类似设备。通常，WLAN 的带宽和传输延迟比许多其他类型的用户互联网连接(如 ADSL、GPRS 和 3G)要好得多，因为 WLAN 既提供访问互联网服务，也提供本地通信服务。WiFi 在许多场合访问速度更多是受到共享连接和用户人数的限制，而非技术本身。WiFi 的突出优势是传输带宽、技术成熟度、开放的网络协议，以及基础接入产品、设施的完备和广泛覆盖。

蓝牙(Bluetooth)也已经发展了多年，技术也比较成熟，它基于 IEEE 802.15.1 技术标准，主要应用目标是取代短距离的有线电缆。目前蓝牙技术主要应用于无线音频(如蓝牙耳机)，与当初的发展蓝图有很大差距。现在很多蓝牙设备(如手机)都是单一特定功能的，一般不开放其他应用编程接口，这使得用户自己不能通过编程来拓展蓝牙接口的功能，限制了其应用范围，也影响了蓝牙技术的推广。

无线传感网(WSN)是近几年研究、开发的一种新技术，它的应用目标是低数据速率的监测、控制系统，如环境监测、自动抄表系统等，这类应用实时性要求不高、数据传输量较小，不需要高的带宽，但往往要求设备有很低的功耗。另外，这些应用网络节点数较多、形态多变，要求节点能无需配置、自动组网、鲁棒性好。ZigBee 作为 WSN 的一种实现，目前有很高的认知度。WSN、ZigBee 在本章的后续小节将作进一步介绍。

表 1.1　WiFi、蓝牙(Bluetooth)和 WSN/ZigBee 比较

市场名	WiFi	蓝牙(Bluetooth)	WSN/ZigBee
采用标准	802.11	802.15.1	802.15.4
应用目标	Web、Email、多媒体	电缆替代	监测、控制
网络结构	星型	星型	星型、树型、网状
网络规模(节点数)	32	7	上万
带宽/(kb/s)	11000+	720	250
通信距离/m	1～100	10+	1～100
枚举延迟	3 s	10 s	30 ms
电池寿命	几小时	几天	多至几年
优势	速度、覆盖	廉价、方便	低功耗、低成本、可靠、规模

1.2　无线传感网(WSN)技术

无线传感网最初应用于军事领域，近年来越来越多地应用于工业和民用领域，比如工业过程监测和控制、机械健康监测、环境和栖息地监测、医疗应用、家庭自动化、交通管制等。

无线传感网(Wireless Sensor Network，WSN)是由部署在监测区域内大量的廉价微型传感器节点组成，通过无线通信方式形成的一个多跳的自组织的网络系统，如图 1.1 所示，其目的是协作感知、采集和处理网络覆盖区域中被感知对象的信息，并发送给观察者。传感器、感知对象和观察者构成了无线传感网的三个要素。

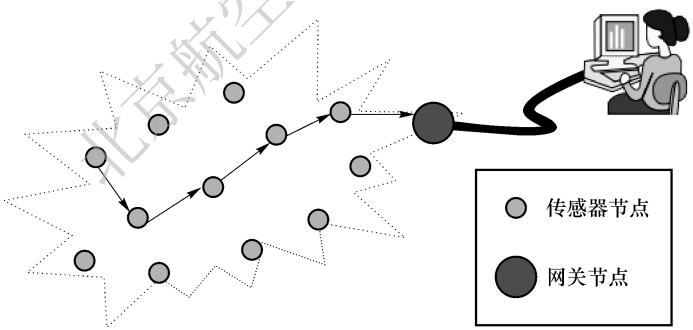

图 1.1　典型的多跳无线传感网络示意图

传感器节点可以看作是一种非常小型的计算机，一般由以下几部分组成：

① 处理器和内存(一般功能都比较有限)。

② 各类传感器(温度、湿度、声音、加速度、全球定位等)。

③ 通信设备(一般是无线电收发器或光学通信设备)。

④ 电源(一般是干电池，也有使用太阳能电池的)。

⑤ 其他设备，包括各种特定用途的芯片，串行并行接口等。

传感器节点的尺寸和成本相差很大，主要受制于电源、内存、运算速度和带宽的限制。除

了传感器节点外，WSN 中一般有一个或几个基站，其作用是从各个传感器节点收集数据，集中处理，然后提交给用户。因此，基站一般拥有更强的数据处理和通信能力以及更持久的电力。

WSN 通常会构成一个无线 ad-hoc 网络，每个传感器都支持多跳路由算法。

1.2.1 特　性

WSN 有一些独特的特性：

- 能够承受恶劣的环境条件。
- 能够应付节点失效。
- 节点的可动性。
- 动态网络拓扑。
- 通信故障容忍。
- 节点的异构性。
- 大规模部署。
- 无人值守操作。
- 节点容量可扩展，仅受网关节点带宽的限制。

1.2.2 标准和规范

目前，有一些标准已经得到批准可以在无线传感网中使用，还有一些正在针对无线传感网的使用而发展，另外还有一些非标准的专有机制和规范也可用于无线传感网中。

其中，6LoWPAN、ISA100、WirelessHART 和 ZigBee/RF4CE 都是基于相同的底层无线电标准 IEEE 802.15.4 的常用于 WSN 的协议。

6LoWPAN 是 IETF 中的一个工作小组，他们已经制定了一个 IPv6 数据包在 IEEE 802.15.4 网络中传输的标准跟踪规范。

EnOcean 是一个实现楼宇自动化无线通信的系统，它不是一个被认可的标准化规范。

国际电工委员会（IEC）在 2010 年 4 月认可 WirelessHART 规范为国际标准（IEC 62591Ed. 1.0）。

IEEE 1451 正在尝试创建智能传感器市场的标准。

ISA100 对工业控制应用提出了更多的协定。

WirelessHART 标准是 HART 协议的扩展，它是专为工业应用设计的。

ZigBee 是 IEEE 标准网络规范，应用范围广泛，它还有一些额外的通信功能，如认证、加密以及上层的应用服务。

1.2.3 软件结构

电源是 WSN 节点最稀缺的资源，它决定了 WSN 网络节点的寿命。WSN 网络要在各种环境中都能应用，包括偏远恶劣的环境，而 ad-hoc 通信是其中关键。基于这个原因，算法和协

议需要解决以下问题：

- 电池寿命最大化。
- 健壮性和容错性。
- 自配置。

WSN 软件研究中的热点问题有：

- 安全。
- 可移动性(传感器节点或基站的移动)。
- 中间层：在软件和硬件之间。

整个 WSN 功能分为三层：最下层是各种敏感单元，负责收集原始信息；中间层是基于传感器智能模块的从节点，负责对原始数据的预处理(包括滤波、补偿、数字化等)和处理后数据的发送；最上层是基于普通 PC 或其他类型上位机(如嵌入式计算机)的主节点，所有传感器的信息在这里进行更高一级处理，如谱分析、模式识别、信息融合、判断决策等。

WSN 多采用五层协议标准：应用层、传输层、网络层、数据链路层、物理层。其中物理层和数据链路层由 IEEE802.15.4 协议模块组成。

1.2.4 操作系统

由于特殊应用要求和硬件平台资源的限制，无线传感网节点的操作系统比通用操作系统简单得多。例如，传感器网络应用通常不需要人机交互，所以操作系统不需要支持用户界面。此外，虚拟内存等内存管理和内存映射功能没有必要使用，或者无法使用。

无线传感网硬件与传统的嵌入式系统并无太多差别，因此可以使用嵌入式操作系统，如 eCos 或 uC/OS。然而这些操作系统大多是实时的，但传感器网络通常不支持实时功能。

TinyOS 是第一个为无线传感器网络设计的操作系统。与其他大多数操作系统不同，TinyOS 基于事件驱动编程模型，而不是多线程。当一个外部事件发生，如收到数据包或传感器读数，TinyOS 调用相应的事件处理程序来处理该事件。之后事件处理程序把任务提交给 TinyOS 内核。TinyOS 系统和程序都是用 nesC 编写的，nesC 是 C 语言的扩展。

也有一些操作系统可以用 C 语言编程，比如 Contiki，MANTIS，BTnut，SOS，Nano-RK。Contiki 支持网络中的模块加载和标准 ELF 文件的运行中加载。Contiki 内核和 TinyOS 一样是事件驱动的，但系统同样支持应用程序多线程驱动。此外，Contiki 还包含了线程模型，它是一种在非常小的内存开销下实现实时多任务系统的方法。与事件驱动的 Contiki 不同，MANTIS 和 Nano-RK 内核基于抢占式多线程。这种情况下，应用程序不需要明确给其他进程让出处理器，而是由内核划分进程的运行时间，并决定现在应该执行哪个进程。Nano-RK 是一种实时内核，很好地控制着任务获得 CPU、网络和传感器的方法。与 TinyOS 和 Contiki 相同，SOS 也是一个事件驱动的操作系统，SOS 最主要的特征就是对装载模块的支持，为了支持模块接口的内在活动性，SOS 还支持动态内存管理。BTnut 基于合作式多线程，使用 C 代码。

LiteOS 是一种为无线传感器网络而新开发的操作系统，它提供类似 UNIX 的抽象，并支持 C 语言。

ERIKA Enterprise 是无线传感网操作系统中的新成员，作为开源的实时内核，ERIKA Enterprise 提供了与用于汽车的 OSEK/VDX API 相似的操作系统 API，以及 uWireless 协议

栈，它还提供了支持保证时隙(GTS)的 802.15.4，这一点对于使用无线传感网的实时流量保障来说至关重要。

1.2.5 算 法

WSN 是由大量的传感器节点组成的，因此，它使用的算法是分布式算法。在 WSN 中最稀缺的资源就是电能，而最耗能的功能是数据传输和闲置监听。因此，WSN 算法研究主要集中于节约能量的研究和设计，比如，使用数据压缩技术来减少传输的数据量，应用拓扑控制算法改变传感器节点的传输功率，或关闭节点但仍保持连接和覆盖。

另一个特征是，由于无线电传输范围的限制以及传输距离的增长导致成本增倍，所以每个节点都可以直接与基站通信是极不可能的，故而数据传输通常是多跳的(节点到节点，最终传输到基站)。

建模、仿真、分析 WSN 得到的算法与事实上使用的协议不同，它更抽象、更一般化、更易于分析。不过，这样的算法与协议设计使用的模型相比不太现实，因为它往往忽略了时间问题、协议开销、路由起始阶段，而且有时还会忽略分布式算法的执行情况。

1.2.6 信息处理

从无线传感器网络收集来的数据通常以数值型数据的形式存储在一个中心基站。此外，开放地理空间联盟(OGC)为互用性接口和元数据编码指定了标准，使异构传感器网络的数据实时整合并连入互联网，允许任何个人通过浏览器监控无线传感网。另外有几种技术可以检索节点的数据，一些协议依赖于淹没机制，另外一些运用 DHT 概念把数据映射到节点。

在无线传感网中，还需要进行信息融合，也称作数据融合，来对数据进行处理，包括筛选、汇总，并对收集的数据进行推理。信息融合对多来源的信息进行整理，使其更便宜、质量更高或关联性更强。在无线传感网域，简单的聚合技术，如最大值、最小值、平均值，都改进为可以降低整体的数据流量，以节约能源。

1.2.7 关键问题

1. 安全问题

无线传感器网络安全方面主要有以下几个问题：

- 信息被非法用户截获。
- 一个节点遭破坏。
- 识别伪节点。
- 如何向已有传感器网络添加合法的节点。

2. 能量效率

无线传感器网络不同于传统的无线网络(如 WLAN 和蜂窝移动电话网络)，除了少数节点需要移动以外，大部分节点都是静止的。因为它们通常运行在人无法接近的，恶劣的，甚至

危险的远程环境中，能源无法替代，设计有效的策略，延长网络的寿命，成为无线传感器网络的核心问题。这些改进涉及物理层、数据链路层和网络层。物理层选择低功耗的调制方式和硬件设计。此外，在 MAC 层和网络层之间加入一个中间层，负责使传感器在不通信时尽可能进入睡眠模式或省电模式，可以大大降低节点的能耗。

3. 时钟同步

无线传感器网络的时钟同步不同于传统的传感器网络。无线传感器与实际的物理环境联系密切，必须采用物理时钟同步，无法使用相对简单的逻辑时钟。无线传感器要求必须采用低能耗工作，时间同步的数据交换受到限制。无线传感器网络覆盖面积大且通常为 Ad-hoc 的结构，不利于采用传统的时间同步方法。无线媒介连接方式不可靠。例如，无线传感器网络与实际的物理环境，监控系统的多传感器信息融合时，上位机需要知道每个原始数据是何时采集的，采样的触发要求每个节点有统一的时钟。无线传感器网络中的通信协议和应用，例如基于 TDMA 的 MAC 协议和敏感时间的监测任务等，也要求节点间的时钟必须保持同步。设计高精度的时钟同步机制是传感器网络设计和应用中的一个技术难点。802.15.4 低速率工作组提供了一种协调件协议 MDP(Mediation Device Protocol)，采用一个伪定义的节点接收网络内所有通信请求，并为通信双方协调会合时间。这个协议不需要额外添加新的硬件，对节点电池寿命的影响也很小，但是，消息的请求对此方案的影响很大。广播时间信标的方法是一种简单实用的同步策略，其基本思想是：节点以自己的时钟记录事情，随后用第三方广播的基准时间加以校正，精度依赖于对这段间隔时间的测量。这种同步机制应用在确定来自不同节点的监测事件的先后关系时有足够的精度。可以考虑精简已有的 NTP(Network Time Protocol)协议的实现复杂度，将其移植到传感器网络中。

4. 定位机制

无线传感器网络中的定位机制与算法包括节点自身定位和外部目标定位两部分，前者是后者的基础。在节点自身定位方面，普通采用了 GPS(Global Postitioning System)技术。对于一些定位精度要求不高的项目，则应用了 LPS(Local Postitioning System)。由于 GSP 不适合中国国情，可以采用一种依赖于自有技术实现传感器网络中节点定位的机制。在北斗一号双星定位系统的支持下，传感器网络中的某些节点就可以找到自己的精确位置，然后参照此基准，利用局部定位算法，其他节点也可以正确定位。此外，在这种模式下，北斗一号的上行数据通路恰好可以作为传感器网络的 sink 链路，将数据回传给控制中心，省去了用飞行器等其他手段收集数据的麻烦。确定了节点的基准位置，利用传统的定位机制和算法，如接收信号的强弱、角度和时间等，以及典型的三角形算法，就可以定位外部目标，这是相对成熟的技术。

1.3 IEEE 802.15.4

IEEE 802.15.4-2006 是低速无线个人局域网(LR-WPANs)的物理层和介质访问控制层的标准，由 IEEE 802.15 工作组维护，它是 WSN/ZigBee、WirelessHART 和 MiWi 标准的基础，这些标准都通过扩展标准中没有定义的上层协议来提供一个完整的网络解决方案。或者，还可以用 6LoWPAN 和标准网络协议建立一个无线嵌入式网络。

IEEE 802.15.4 为一种无线个人局域网提供基础低层，这种网络致力于实现低造价、低速度、无处不在的通信。其主要特征如下：

- 信道访问采用载波监听多路访问/冲突避免(CSMA-CA)。
- 可选时间槽和网络信标。
- 信息确认。
- 多级安全。
- 适合需要超长电池的设备，可以自选延迟来匹配不同的电源需求(传感器，远程监控)。

IEEE 802.15.4 的基本架构是创造一个传输速率为 250 kb/s 的 10 m 通信范围。20 和 40 kb/s 的传输速率在最初的版本中已经定义，100 kb/s 的速率正在加入到当前版本中。低速率往往是影响功耗的一大因素。

与其他 WPANs 标准相比，802.15.4 最主要的特点就是重点实现极低的生产经营成本和简单技术，同时不失灵活性和普遍性。

遵守 802.15.4 的设备可以使用三个可用频段中的一个。设备同样有电源管理功能，比如链路质量和能量检测。

1.3.1 协议架构

设备通过一个概念性的简单无线网络相互交流。网络层的定义基于 OSI 模型，虽然标准中只定义了底层，但它可以使用 IEEE 802.2 逻辑链路控制子层与上层交流，而这个逻辑链路控制子层则是通过一个会聚子层访问介质访问控制子层。这个功能的实现需要依赖外部设备或完全的嵌入式自运作设备。IEEE 802.15.4 协议栈如图 1.2 所示。

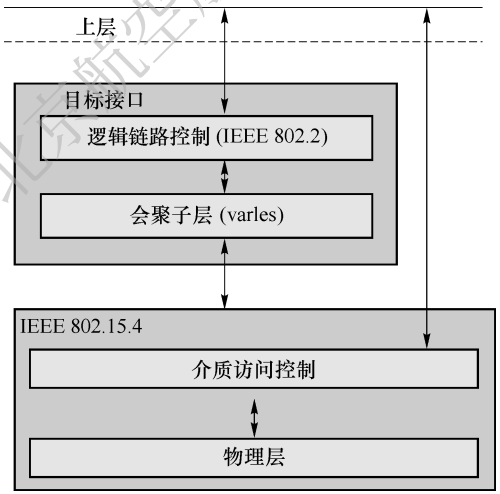

图 1.2 IEEE 802.15.4 协议栈

1. 物理层

物理层(PHY)提供最终的数据传输服务，同时提供物理层管理实体接口，这个实体可以访问每一层的管理功能和维护相关的个人局域网的信息数据库。因此，PHY 管理物理的

RF 收发器和执行通道选择以及能量和信号管理功能。它可以使用三段未授权频带中的一段：

- 868.0-868.6 MHz：欧洲，允许一个通信通道(2003，2006)。
- 902-928 MHz：北美，最多 10 个信道(2003)，延长至 30 个(2006)。
- 2400-2483.5 MHz：世界范围内通用，最多 16 个信道(2003，2006)。

2003 版本中定义了两个基于直接序列扩频(DSSS)技术的物理层：一个工作在 868/915 MHz 频带，传输速率为 20 kb/s 和 40 kb/s；另一个工作在 2 450 MHz 频带，传输速率为 250 kb/s。

2006 版本改善了 868/915 MHz 频带的最大数据速率，使它也支持 100 kb/s 和250 kb/s。此外，还定义了 4 个物理层，使用不同的调制模式。其中三个保留了 DSSS：在 868/915 MHz 频带使用二进制或偏移正交相移键控；在 2 450 MHz 频带使用后者。或者可选的 868/915 MHz 层可以结合使用二进制键控和振幅键控(因此它基于并行，而不是顺序扩频，PSSS)。可以动态切换支持 868/915 MHz 的物理层。

除了上述三个频段，IEEE802.15.4c 研究小组正在考虑在中国新开 314～316 MHz，430～434 MHz 和 779～787 MHz 频段，IEEE 802.15 任务小组在日本修订现有的 802.15.4—2006 标准，以支持 950～956 MHz 频带。第一组标准修定在 2009 年 4 月公布。

2007 年 8 月，IEEE 802.15.4a 发布了新的修正标准，将原有的 4 个物理层扩展为 6 个，新加入的物理层一个使用直接序列超宽带(UWB)；另一个使用调频扩频(CSS)。使用 UWB 的物理层可以使用 3 个频段：低于 1 GHz；3～5 GHz 之间；6～10 GHz 之间。使用 CSS 的物理层可以使用 2 450 MHz ISM 频段。

2009 年 4 月，IEEE 802.15.4c 和 IEEE 802.15.4d 又对现有物理层进行了扩充：一个使用 O-QPSK 或 MPSK，780MHz 频带；另一个使用 GFSK 或 BPSK，950MHz 频带。

物理层除了管理物理 RF 收发器外，还管理以下任务：

- 无线电收发的激活与停用。
- 当前信道上的能量检测。
- 指示接收到的包的链路质量。
- CSMA-CA 的 CCA。
- 选择信道频率。
- 数据发送和接收。

2. MAC 层

介质访问控制(MAC)允许通过使用物理信道传输 MAC 帧。除了数据服务，它还提供了一个管理接口和自身管理对物理信道和网络信标的访问。它同样控制帧校验，保证时间戳，处理节点联合，最终为安全服务提供钩点，如图 1.3 所示。

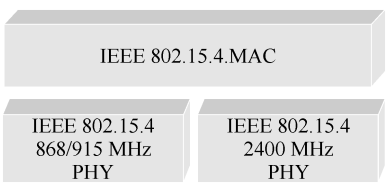

图 1.3　IEEE 802.15.4 协议层

MAC 层管理以下任务：

- 作为 PAN 的协调器要为设备活动产生网络信标。
- 同步信标。

- 支持 PAN 关联和离开。
- 支持设备安全。
- 信道访问使用 CSMA-CA。
- 管理 GTS 机制。
- 在点对点 MAC 实体间提供可靠连接。

3. 更高层

其他更高层和相互操作的子层在标准中没有定义。一些现有的设计，如 ZigBee，就是以 802.15.4 这个标准为基础，提出整体解决方案。TinyOS 栈也采用了 IEEE 802.15.4 的一些硬件定义。

1.3.2 网络模型

1. 节点类型

IEEE 802.15.4 标准定义了两类网络节点。

第一类是全功能设备(FFD)。它可以充当个人局域网的协调器，也可以作为普通节点使用。他实现了通信的一般模式，既可以和其他任何设备交流，也可以传送消息，这时它被称为协调器(如果它负责整个网络，则叫做 PAN 协调器)。FFD 设备适合任何网络拓扑。

第二类是精简功能设备(RFD)。这意味着它是一种非常简单的设备，带有有限的资源和通信能力，因此，它们只能和 FFD 通信，不能充当协调器，RFD 设备只适合星型网络。

2. 拓扑结构

网络可以建为点对点或星型，如图 1.4 所示。每个网络至少需要一个 FFD 作为网络协调器。网络就是由一些分隔适当距离的设备组组成的，每个设备有一个唯一的 64 位标识符，如果条件满足，在一些受限环境中，可以使用 16 位标识符。也就是说，在每个 PAN 域中，通信都有可能使用短标识符。

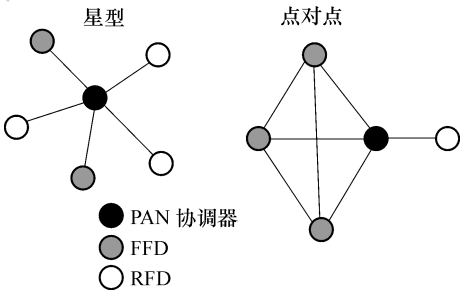

图 1.4 IEEE 802.15.4 星型拓扑和点对点拓扑

点对点网络可以形成任意的连接模式，其扩展仅受限于每对节点间的距离，这是为了可以作为 ad hoc 网络履行自我管理和组织能力的基础。标准中没有定义网络层，所以不直接支持路由，但添加一个附加层就可以增加对多跳通信的支持，不过也增加了进一步拓扑的限制。

IEEE 802.15.4 标准中提到了一种群树结构，如图 1.5 所示，即 RFD 同一时间只与一个 FFD 连接，它们只作为树的叶子，其他节点都是 FFD。这个结构可以作为普通的网状网络进行扩展，这个网状网络的节点是群树网络，每个群都有一个本地协调器，整个网络还有一个总协调器。

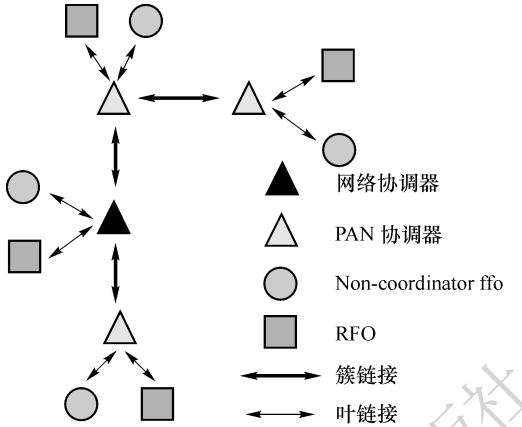

图 1.5　IEEE 802.15.4 群树拓扑

在星型拓扑结构中，网络协调器必须作为中心节点。当一个 FFD 在选择了一个唯一的 PAN 标识符后声明自己是这个 PAN 的协调器时，网络就建立起来了。在此之后，其他设备可以加入该网络，该网络独立于其他所有星型网络。

不过需要注意，IEEE 802.15.4 标准本身并没有实现网络层，它只是定义了这样的一种模型，具体实现是由 ZigBee 等其他高层协议完成的。

1.3.3　数据传输架构

帧是数据传输的基本单位，有 4 种基本类型（数据帧，确认帧，信标帧，命令帧），提供了在简单性和健壮性间的权衡。IEEE 802.15.4 MAC 协议支持两个数据发送模型，如图 1.6 所示，这两个模型能够由 PAN 协调器选择：

- 无信标模型，在这个模型中 MAC 简单地由无槽的 CSMA-CA 管理。
- 信标模型，在这个模型中，信标事先由协调器发送来同步关联的节点并识别 PAN。

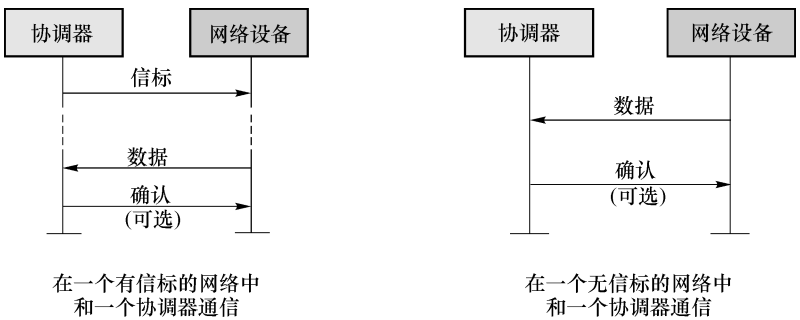

图 1.6　有/无信标网络中设备到协调器传输

访问信道由有槽 CSMA-CA 使用超帧格式结构管理。

超帧结构是由协调器定义的,当两个信标作为自己的限制,并给其他设备提供同步通信和配置信息时会被使用。超帧由 16 个等长槽组成,每个槽都可以进一步分为活动部分和不活动部分,在此期间协调器可以进入省电模式,不需要控制网络。

超帧由于其限制而引发了一些争议,这个问题则由 CSMA/CA 解决。每次传输必须在第二个信标帧到来前结束。正如前面提到的,有定义清晰的带宽需求的应用可以使用来自一个或多个时间槽的域,这样的域最多 7 个,加在超帧的尾部。超帧的第一部分必须给予网络结构和它的设备足够的服务。超帧经常在低延时设备中使用,这种设备即使是在很久不活动的情况下也必须保持连接。

协调器的数据传输需要一个信标同步阶段,即搜索信标来与超帧结构同步。之后是 CSMA/CA 传输,确认帧是可选的。从协调器发出的数据传输通常遵循设备要求:如果使用信标帧,这些帧就用于信号请求,协调器确认请求然后发送数据,再由设备确认。在不使用超帧时,通信过程和上述相同,只有这种情况下没有信标帧追踪挂起信息。

在信标网络中,协调器在信标中指示了数据正在挂起。如果需要的话,这个设备事先使用有槽 CSMA-CA 侦听信标并发送一个数据请求 MAC 命令,如图 1.7 所示。

在无信标网络中,设备使用无槽 CSMA-CA 发送一个数据请求 MAC 命令,如果一个协调器数据发送正在挂起(等待),这个协调器使用无槽 CSMA-CA 发送一个数据帧。否则,协调器发送一个数据帧包含一个 0 长度的负载,如图 1.8 所示。

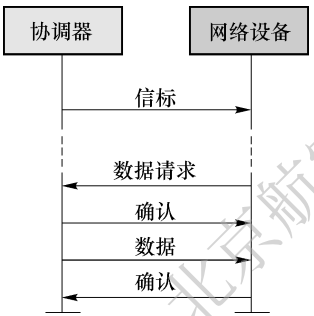

图 1.7　信标网络中协调器到设备通信

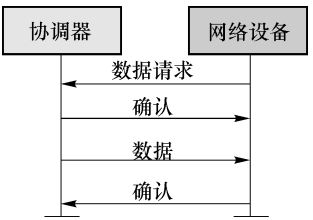

图 1.8　无信标网络中协调器到设备通信

点对点网络可以使用无槽的 CSMA/CA 或同步机制,在这种情况下,任何两个设备间的通信都是可能的,然而在“结构化”模型中,其中一个设备必须是网络协调器。

一般而言,所有执行过程都沿用一个典型的请求－确认/指示－应答分类。

1.3.4　可靠性和安全性

物理介质通过 CSMA/CA 协议访问。当使用信标时,一般的数据传输利用未分配槽,而确认不遵循这个步骤。

确认信息在某些情况下是可选的。无论什么情况下,如果设备不能在一定时间内处理一个帧,它就会简单地认为自己没有收到,超时重发可以进行多次,不过首先要判断是继续尝试还是终止。

这些设备的环境要求电池寿命最大化，协议用定期检查挂起信息的办法实现，至于检查的频率，则依赖于应用需求。

关于安全通信，MAC 子层提供了上层可使用的设备，来达到想要的安全水平。高层进程可以用指定密钥实现对称加密，以保护和限制一组设备或点对点链接的有效载荷，这些设备可在访问控制表中指定。此外，MAC 还会检查成功接收的数据的新鲜度，保证不会把不再有效的旧帧或数据提交给更高层。

除了这个安全模式，还有一个非安全 MAC 模式，在这个模式下允许访问控制表仅仅是为了根据帧的来源决定是否接收它。

1.4 ZigBee

ZigBee 联盟由一些公司组成，它的职能就是维护和发布 ZigBee 规范。ZigBee 规范是一组高层通信协议，使用基于 IEEE 802.15.4—2003 的小型低功耗无线射频，如通过短距离无线与手机连接的无线耳机。ZigBee 是近年来快速发展的 WSN 的规范之一，而且其定义的技术要比其他 WPANs 更简单、实现成本更低，它适合于低数据速率、低功耗并且安全的无线网络。

IEEE 802.15.4 和 ZigBee 的关系与 IEEE 802.11 和 Wi-Fi 联盟的关系很像。ZigBee 1.0 规范在 2004 年 12 月 14 日批准，并且只有 ZigBee 联盟的成员可以使用。ZigBee 2007 规范在 2007 年 10 月 30 日发表。第一个 ZigBee 应用简介——家居自动化，在 2007 年 11 月 2 日发布。根据 NIST 的修正，智能能源 2.0 规范将会去除对 IEEE 802.15.4 的依赖，设备制造商将能够执行任何 MAC/PHY，如 IEEE 802.15.4(x) 和 IEEE P1901。

ZigBee 使用 ISM 频段：欧洲为 868 MHz；美国和澳大利亚为 915 MHz；全球范围内都可以使用 2.4 GHz。无线射频部分既可作为独立元件也可集成在处理器和微控制器中。一般来说，芯片厂商同样提供一个 ZigBee 软件协议栈，这个协议栈也可以单独出售。

ZigBee 能在 15ms 内激活(从睡眠到活动状态)，时延非常低，因此设备反应非常快，尤其是与蓝牙做比较，蓝牙的唤醒延时通常为 3 s。因为 ZigBee 设备大多数时间都在睡眠状态，所以平均功耗非常低，电池寿命可以很长。

ZigBee 联盟致力于实现可靠、低成本、低功耗无线网络监控产品，并且这些产品基于一个开放的全球标准。ZigBee 联盟定义了 ZigBee 规范，并发布了应用简介，允许多个 OEM 厂商设计生产可互操作的产品。对于非商业用途的公众，ZigBee 规范可以免费使用，但如果一个公司想在产品中使用 ZigBee 技术，那么它必须是 ZigBee 联盟的成员。

1.4.1 ZigBee 协议栈

到目前为止，ZigBee 有以下几个版本协议栈：

- ZigBee 2004：在 2004 年发布，同时带有一个家庭控制灯的应用示例。这个栈后来再没有更新过，现在已经很少使用，不能与 ZigBee 2007 通用。
- ZigBee 2006：在 2006 年发布，与 04 版的相比，主要是用“cluster library”替换掉了 MSG/KVP 结构。

- ZigBee 2007：在 2007 Q4 上发布。有两个版本：zigBee 和 zigBee Pro。ZigBee 2007 完全向后兼容 ZigBee 2006 设备，ZigBee 2007 设备可以加入并操作 ZigBee 2006 网络，反之亦然。

ZigBee 和 ZigBee Pro 栈都完全实现了 MAC 层、网络层、安全服务层和应用框架。在其他类型的网络中，实现 ZigBee 和 ZigBee Pro 的设备可以作为端设备进行交互操作。比如，在以一个 ZigBee Pro 协调器为中心形成的网络中，路由器只能是 ZigBee Pro 路由器，但端设备可以是 ZigBee 端设备和 ZigBee Pro 端设备。

ZigBee 栈围绕一个中心协调器形成，使用树型寻址来建立网络，通常使用树型路由（也可以使用基于路由表的路由方式）。这个栈仅支持家居安全方面的应用。

ZigBee Pro 栈围绕一个中心协调器形成，使用一个随机的寻址模式来避免树型的限制，可以使用基于路由表的路由方式，并且可获得如下特征：

- 网络级的组播。
- 多对一和源路由。
- 频率变捷。
- 非对称链路处理。
- PAN ID 冲突。
- 分段。
- 标准和高安全。

如果要实现一个系统，最好使用 ZigBee Pro 栈，而不是 ZigBee 栈，因为 ZigBee Pro 栈有很多对于网络的长期稳定和可靠性来说非常重要的特征，ZigBee 栈则主要用于小型静态网络。

1.4.2 ZigBee 寻址机制

ZigBee 协议栈中包含两种寻址模式，另外，它也支持使用一个测试工具来进行手动分配地址。

ZigBee 栈使用树型寻址。在这种寻址机制下，协调器启动网络并初始化网络空间，路由器和端设备的数量在网络建立的时候就设置好了，它使用一个分布式寻址模式来分配网络地址，这个模式为每个有可能成为父节点的子节点提供一个有限的网络地址子块，这些地址在一个特殊的网络中是唯一的，并且由父节点分给子节点。ZigBee 协调器决定了其他设备的子节点的最大数量，这个数量包括路由器子节点数量和端设备子节点数量。每个子节点地址与它的父节点的地址都是相关的。每个设备都有一个相关深度，这个深度表明了一个传输帧仅使用父节点-子节点链路到达 ZigBee 协调器必须经过的最小跳数。ZigBee 协调器本身的深度为 0，它的子节点的深度为 1。多跳网络的最大深度大于 1。

ZigBee 栈使用的树型寻址机制有一些限制。这些限制包括有可能运行超出本地范围的地址，在这种情况下，新的设备会因为没有合适的父节点而不能加入到网络中。树型路由也要求重新建立网络，包括设备接收新的地址，重新建立父节点和子节点间损坏的链路。由于存在这些限制，就需要为更大的网络开发一种新的寻址机制。

ZigBee Pro 栈使用一个随机地址分配机制。在这种机制下，协调器仍然用来启动网络，每

个设备(路由器和端设备)加入到网络后,从加入到的那个设备那里获得一个随机分配的地址,设备使用网络级的消息来进行地址上的冲突检测,来确保地址是唯一的,这个地址然后由设备保存,即使父节点改变了。

在 ZigBee Pro 下,对于网络的建立和配置有一些测试工具,这些测试工具可以被用来提供手动的地址分配。

在使用 ZigBee 的 EmberZNet 中,我们推荐使用带有任意地址分配机制的 ZigBee Pro 栈。

1.4.3 硬件和软件

ZigBee 适用于开放的和互操作性的设备,标准从物理层开始定义。物理层和 MAC 层采用 IEEE 802.15.4 标准。网络层、安全层和应用层由 ZigBee 联盟开发。它支持多个系统,如网关,还有一些试运行工具可用来简化 ZigBee 网络的开发和部署。标准的扩展和附加部分还在继续开发。ZigBee 协议架构如图 1.9 所示。

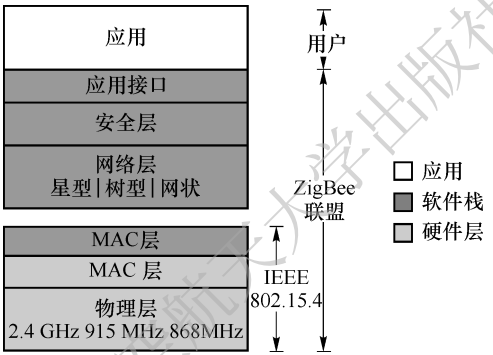

图 1.9 ZigBee 协议架构

ZigBee 的硬件和软件都可由厂商提供,另外还可以提供一些压缩的低成本的模块以及应用程序参考设计,软件开发者能够通过扩展由厂商提供的示例程序开发一个完整的 ZigBee 应用程序。

Ember ZigBee 协议栈包含一些典型的示例程序和一些通用的功能或者工具:

- 使用 ZigBee 串库的家庭自动化和智能能源参考应用。
- 在部署后,通过无线 bootloaders 来允许系统软件的更新。
- 把 ZigBee 网络连接到其他系统的网关接口。
- 微控制器的编程工具。
- 网络抓包器和调式工具允许观看和分析网络操作。

ZigBee 的软件很容易在小型廉价的微处理器上开发。ZigBee 使用的无线电设计需要经过非常仔细的优化,以保证在大规模生产中实现低成本生产。

1.4.4 协 议

ZigBee 网络是基于 IEEE 802.15.4 MAC 层和物理层的。802.15.4 标准在 2.4 GHz 带

宽上的数据速率是 250 kb/s，在 900/868 MHz 带宽上的数据速率是 40/20 kb/s。很多芯片公司在 2.4 GHz 带宽上都提供了解决方案，只有一小部分公司支持 900/868 MHz 带宽。ZigBee 采用 802.15.4—2003 的标准版本。

IEEE 802.15.4 标准在信标网络中的 MAC 层上提供了一些选项，在当前任何版本的 ZigBee协议栈中，均没有使用有保证的时间槽。例如，为了节省代码空间，这些选项通常不包含在 ZigBee 软件栈中。对于 IEEE 802.15.4 标准 MAC 层，ZigBee 有做出一些特定的修改。

ZigBee 协议基本的信道访问模式是“载波侦听，多路访问/碰撞避免”(CSMA/CA)。但是有三种例外：信标的发送基于固定的时间安排，不适用 CSMA；消息确认也没有使用 CSMA；面向有低延时实时要求的信标网络的设备会使用保证时槽(GTS)，不能用 CSMA。

ZigBee 当前版本的协议栈支持信标和无信标网络。在无信标网络中，设备使用无槽 CSMA/CA 信道访问机制，在这种类型的网络中，ZigBee 路由器通常需要自己的接收器一直处于活跃状态，这就需要非常强的电源供应。而这一点在异构网络中也是允许的，在这种网络中一些设备的接收器持续运行，而其他设备仅在检测到外部事件时发送数据。异构网络的一个典型的例子就是无线灯开关：灯节点的接收器从连接到主电源后就持续运行，而电池供电的开关节点可以保持睡眠。如果开关被拨动，则开关节点醒来，发送一个命令给灯节点，然后收到一个应答，之后就返回睡眠状态。在这样的网络中，灯节点至少是一个 ZigBee 路由器，开关节点是一个典型的 ZigBee 终端设备。

在信标网络中，特定的网络节点会让 ZigBee 路由器定时发送信标，来确保它们对于网络的其他节点来说是存在的。在两个信标传输的间隙，节点可以睡眠，这样可以降低它们的工作周期，延长它们的电池寿命。在数据速率为 250 kb/s 下，信标间隔时间可以从 15.36 ms 到 251.658 24 s；在数据速率为 40 kb/s 下，信标间隔时间可以从 24 ms 到 393.216 s；在数据速率为 20 kb/s 下，信标间隔时间可以从 48 ms 到 786.432 s。不过，低工作周期和长信标间隔时间需要精准的定时，这与低生产成本产生矛盾。

一般情况下，ZigBee 协议会让无线电工作的时间降到最小，以减少电力使用。在信标网络中，节点仅需要在信标被发送时活动。在无信标网络中，功耗绝对是不对称的：有些设备始终活跃；而其他设备则大部分时间都在睡眠。

无线设备使用直接序列扩频编码，它由调制器中的数字流管理。BPSK 用于 868/915 MHz 频段；OQPSK 用于 2.4 GHz 频段。2.4 GHz 频段的每个信道的数据速率为 250 kb/s；915 MHz 的为40 kb/s；868 MHz 的为 20 kb/s。对于 ZigBee Pro 来说，传输范围在 10～75 m 之间，最大可达到 1 500 m，尽管它在很大程度上依赖于特定的环境。无线射频的最大输出功率通常为 0 dBm(1 mW)。

1.4.5 设备类型

ZigBee 设备有三种类型：

- ZigBee Coordinator (ZC)：功能最强的设备，负责形成网络，包括在扫描可用信道后选择一个合适信道以及一个扩展 PAN ID。网络形成后，协调器作为网络树的根，以及与其他网络沟通的桥梁。每个网络只有一个 ZigBee 协调器，它是网络中最初开始工

作的设备。它能够存储网络中的信息，包括网络密钥。

- ZigBee Router（ZR）：除了运行应用程序，它还作为中间路由器，把数据传递给其他设备。路由器也可以作为端设备。路由器不能睡眠，网络建立后，它必须一直处于活动状态。
- ZigBee End Device（ZED）：只能与父节点沟通（或者是协调器或者是路由器），不能从其他设备中获得数据。这种设计使得节点大部分时间都可以睡眠，从而获得很长的电池寿命。ZED 需要的内存容量最少，因此造价比 ZR 和 ZC 都便宜。

1.4.6 网络拓扑

ZigBee 支持两种基本拓扑类型：

- 树型网络。
- 网状网络。

在树型网络中，以协调器为“根”，路由器分散在树的分支上，端设备则是树的“叶子”，如图 1.10 所示。

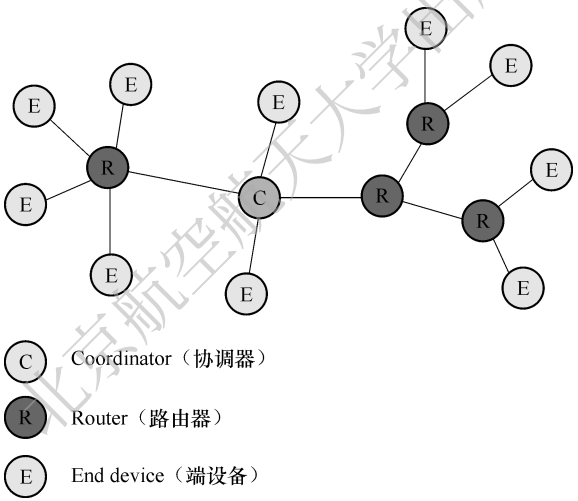

图 1.10　树型拓扑

网状网络因为有中继功能，因此无线通信更为可靠。例如，如果节点 A 不能直接发送一个信息给节点 B，那么它可以通过一个或者多个中间节点把信息发送给 B。

EmberZNet 支持三种网状网络拓扑类型（见图 1.11）：

- 星型网络
- 全网状网络
- 混合网状网络

在一个星型网络中，hub 是所有通信的中心节点，因此 hub 有可能变成网络或处理带宽的瓶颈，传输范围也会受到 hub 通信半径的限制。在 EmberZNet 栈中，这个拓扑由一组端设备和一个作为他们父节点的协调器构成，协调器节点作为网络的 hub。

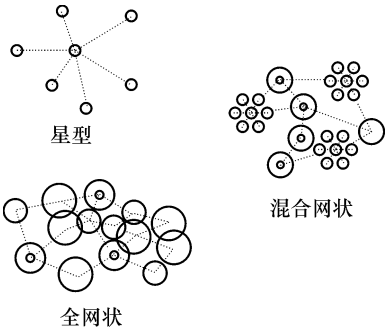

图 1.11　星型、全网状和混合网状网络拓扑

在一个全网状网络中，所有节点都是路由器节点或协调器节点。因为所有节点能够中继信息给其他节点，所以这个拓扑的可靠性非常好。但同时，造价高昂。

混合网状网络拓扑结合了星型和网状拓扑策略。在这种网络中存在有很多星型网络，它们的 hub 能够在网状网络中通讯。混合网络有着比星型网络更长的通信距离，比网状网络更多的分层设计功能。这个拓扑由 EmberZNet 栈形成，它使用路由设备作为星型子网的 hub，每个 hub 都有端设备与它相连。

1.4.7　路由机制

ZigBee 有多种路由机制，在基于网络和期待的流量模型上使用。应用开发者需要选择使用哪种机制，并且这个选择应该作为系统架构和设计的一部分。在实际应用操作中，可能会使用多种路由机制，因为一些设备执行一对一的通信，同时所有设备都可能与一个中心检测设备通信。以下讨论的路由类型有树型路由、路由表路由、组播路由、广播和多对一/源路由。

在 ZigBee 栈中，路由是沿着树型拓扑的链路发起的，即使这个路由可能是间接的，比如，两个节点可能是互为邻居，但是它们在树的同等深度上，也就是说，它们离协调器的跳数是一样的。如果它们通过不同的父节点加入到网络中，信息从一个节点传输到另一个节点的树型路由就是尽可能地一级一级的向上传递信息，直到信息被传递到它们共同的祖先上。这个树型路由机制假定网络树型拓扑一直是稳定的，树路径从来不变，因此不需要路由发现功能。设备之间的路由是确定的，可以用数学方式计算出来。注意 EmberZNet 栈不支持树型拓扑。

路由表(或网状)路由也存在于 ZigBee 栈中。这个路由表路由是从 AODV 下一跳路由中衍生出来的，当一个节点由于发现了一个路由表路由而发送信息给目的节点，树型栈首先尝试路由传送，如果树型路由传送失败了并且信息指定了 APS 重传可选项，树型栈转而求助于路由表路由。

ZigBee Pro 栈从不使用树型路由来进行信息传送，因为它有一个更优的路由策略，而且 ZigBee Pro 网络中不存在树型结构。当一个节点发送一个路由请求，希望发现到另一个节点的路径时，路由就形成了。当在两个节点间的路由被发现后，源节点给路由中的第一个节点发送它的信息，这个节点是在源节点的路由表中指定的。每个节点使用自己的路由表来决定把信息传递给目标节点所要走的下一跳。如果一次路由失败了，路由错误被发送给信息的源端，然后重新进行发现路由。

组播路由提供了一个一对多路由选项。当一个设备想要发送信息给一组设备时，使用一个组播，比如灯开关发送一个命令给 10 个灯，在这种机制下，所有设备加入到一个组播组中，只有组播组中的成员才可以接收信息，即使其他设备将路由这些组播信息。一个组播是一个过滤的有限广播，因此应该在应用中按需使用，因为过度使用广播机制会降低网络性能。组播信息不需要被确认。

广播路由是一种用来把一个信息发送给网络中的所有节点的机制。有一些网络级的广播可选项只向路由器发送信息或者也向睡眠的端设备发送广播。广播信息被网络中的所有供电设备重复三次，以确保传送到所有的设备。广播是一种可靠的信息发送方式，不过它应该被保守地使用，因为过度使用广播会影响网络的性能。重复广播会限制网络中的其他数据传输流量。

多对一路由是一个简单的机制，允许一个完整的网络中有一条到一个中心控制或者检测设备的路径。在一般的路由表路由下，中心设备和其周围紧密环绕的设备需要路由表空间来为网络中的每个设备存储下一跳，考虑到 ZigBee 网络中使用的设备的存储空间是有限的，这些大的表是不合适的。在多对一路由下，中心设备发送单个路由发现，这个路由发现在所有路由器中建立一个路由表项，来提供到中心设备的下一跳，这样，只使用单个表项，网络中的所有设备就都有了到中心设备的一下跳的路径。但是，在网络中的中心设备也经常需要回送发送的信息，这个可能导致路由表的增大。取而代之，到达中心设备的信息首先使用一个路由记录信息来存储使用的下一跳，中心设备在一个路由记录表中存储这些下一跳路由，要发送的信息在其网络头部包含了这些路由记录，然后信息使用自己的网路头部中的下一跳进行路由，而不再使用路由表。这为大规模网络提供了路由机制，且不会增加所有设备的内存需求。不过要注意，如果要存储这些路由记录的话，中心设备确实需要一些额外存储空间。

1.4.8 应　用

ZigBee 协议适用于要求低数据速率和低功耗的嵌入式应用中。ZigBee 网络目前的重点是定义一个通用的、廉价的、自组织的网状网络，可用于工业控制、嵌入式传感、医疗数据收集、烟雾和入侵警报、建筑物自动化、家居自动化等。最终实现的网络将使用非常少的能量，个人设备要通过 ZigBee 认证，必须保证电池寿命在两年以上。

ZigBee 网络的通用特性如下：

- 低功耗：使用合适的工作周期的设备，在一个 AA 类电池上能够运行很多年。
- 低数据率：2.4 GHz 带宽支持 250 kb/s 的无线数据率。通过网络实际可维持的流量低于理论上的无线容量。所以 ZigBee 可以很好地用于采样和检测的应用环境中。
- 大小型网络：ZigBee 网络规模可以从很少的设备变化到成千的设备，而且通讯流畅。网络层是为很多不同的数据发送机制（路由类型）而设计的，在预期使用基础上优化网络操作。
- 范围：典型的设备提供充足的范围来覆盖一个正常的家庭，并且随时可用的功率放大设计很大程度上扩大了范围。用于物理层的分布式扩频技术，在很大程度上增强了网络的抗干扰性。
- 简单的网络安装、启动和操作：ZigBee 标准支持多种网络拓扑结构和简单的协议，对于

形成和加入网络来说允许系统的自我配置，当发生路由问题时，允许自我解决。

ZigBee 网络的典型应用范围包括：

- 家庭娱乐与控制：智能照明系统，先进的温度控制，安全和保安，电影和音乐。
- 家具警报：水传感器，电源传感器，能源监控，烟雾和火灾探测器，智能家电和访问传感器。
- 移动服务：m-付款，m-监控，m-安全和访问控制，m-医疗保健和远程协助。
- 商业大厦：能量监控，HVAC，灯控，进出控制。
- 工业厂房：过程控制，资产管理，环境管理，能源管理，工业控制设备，机器对机器（M2M）的通信。

本书介绍的 STM32W108 有自带 ZigBee Pro 协议栈固件的版本，用户可以借此快速开发支持 ZigBee 的各类传感、控制类电子产品。

1.5 RF4CE

RF4CE 广泛用于无线遥控音频、视频等消费类电子产品，如电视机和机顶盒。2009 年 3 月 3 日，RF4CE 联盟同意与 ZigBee 联盟合作，共同提供一个以无线电频率为基础的远程控制标准化规范。它比现有的遥控解决方案拥有更多优势，包括更好的通信和更高的可靠性，增强的功能和灵活性，互操作性，并且没有视线障碍。

RF4CE 标准是一种基于无线射频的遥控标准化规范，与一般的 RF 遥控不同，它不需要中介基站。RF4CE 可以置于电子产品内部，这使它成为设备之间双向通信的理想技术。

RF4CE 的特性包括：

- 工作在 2.4 GHz 频带，基于 IEEE 802.15.4 标准。
- 3 个以上的信道。
- 遥控器和消费类电子设备都有省电机制。
- 发现机制。
- 配对机制。
- 通过行业标准 AES-128 的安全认证。
- 安全密钥生成机制。
- 支持标准规范以及厂商特定规范。

RF4CE 使用星型拓扑，这种拓扑是专为简单的设备到设备通信而设计的，它不需要协议提供的完整的网状网络功能。它的内存需求很小，所以成本很低。同时，这种简单的设备到设备的拓扑也利于开发和测试，可以缩短上市时间。

RF4CE 可用于娱乐设备、车库开门器和无钥匙进入系统等场合，特别是在家居控制中，RF4CE 不但可以减少家电控制的数目，还可以增强功能性，使家电控制的操作更简单。

对于 RF4CE 网络来说，第一个要解决的问题就是设备的配对。在 RF4CE 网络中有两种设备：控制设备和目标设备。控制设备就是 RF 遥控器；目标设备可以是电视、DVD 机、空调等。配对是两个设备之间的一种逻辑链接，必须在设备交换信息之前建立，通常配对是在控制设备和目标设备之间进行的，不过两个目标设备之间也可以配对，只有配对成功的设备才能进行通信，一对一、一对多、多对一、多对多的配对都是可行的，故而一个目标设备可以被多个控制设备控制，一个控制设备也可以控制多个目标设备，也就是说，RF4CE 网络是支持多 PAN 网络的，如图 1.12 所示。

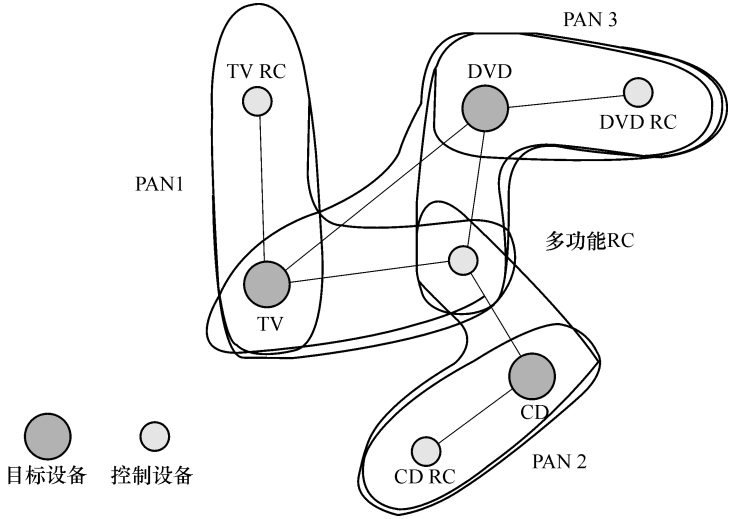

图 1.12　RF4CE 多 PAN 网络

建立一个配对，首先要一个设备在网络覆盖范围内发现另一个 RF4CE 设备，比如，按下遥控器的一个按钮，然后可用设备回应这个发现请求，发送自己的详细信息以供控制设备鉴别，一旦控制设备鉴定成功，两个设备间的配对就完成了。

除了设备配对，RF4CE 还有一点数击的功能。比如，你可以只按下一个按钮，就能够打开电视、机顶盒、音响，甚至还可以调暗灯光、关闭百叶窗等，它帮助你把你的电子产品组合成一个系统，而不再是一个个单一的设备。

RF4CE 协议在网络层有内置的安全保障功能，以保证配对设备间的通信是安全的、可靠的，这个安全保障包括 AES-128 位加密和一个安全密钥生成机制，可以防止对信息内容的修改，以及信息被记录。

RF4CE 最大的好处就是控制设备和目标设备之间的双向通信，这样一来，用户就可以看到网络范围内所有目标设备的详细信息和状态，便于对配对等操作的设定，也更容易对系统进行控制。另外，随着很多设备都开始具有互联网功能，RF 控制器也有了新的功能，比如，你可以使用你的 RF 遥控器浏览互联网，就像使用鼠标一样。你也可以设定当你按下某个目标设备的按钮时，RF 遥控器的蜂鸣器就会响，这样就绝不会再发生找不到遥控器的事情了。

RF4CE 还充分利用了 IEEE 802.15.4 的抗干扰技术以及多通道机制。它可以与运行在同一环境中的其他协议共用 2.4 GHz 频段。RF4CE 支持三个及其以上的通道。网络第一次启动时，每个设备都选择最佳通道，一旦信道质量降低，目标设备可以自动切换到其他可用信道。控制器如果在原信道找不到目标设备，就会搜索其他信道，找到目标设备现在的位置。

另外，现在常用的红外遥控器通常需要瞄准接收器，而 ZigBee RF4CE 控制器就不需要。你可以在网络覆盖范围内的任何地方进行控制，而且这个范围要比红外遥控的范围大得多。

最后，RF4CE 设备还有一个优点，那就是省电机制。RF4CE 遥控器可以在不使用时将自己关闭，这就意味着它的电池寿命要比普通红外遥控器的多很多倍。RF4CE 目标设备在等待时也可以进入睡眠，不会一直耗电。当设备不再使用时，它们会把自己关闭，在用户定义的时间间隔内醒来，查看是否有新的命令给自己。用户可以完全控制睡眠时间长度，从而不会有延时问题。

本书介绍的 STM32W108 也有自带 RF4CE 协议栈固件的版本，用户可以借此快速开发

支持 RF4CE 的消费类电子产品。

1.6 6LoWPAN

一个大型的无线传感网络中分布着极为庞大的传感器节点，每个节点都需要一个 IP 地址，在 IEEE 802.15.4 上使用 IPv4 无法满足要求，而 6LowPan 的出现可以实现在 IEEE 802.15.4 底层上使用 IPv6，解决了 IP 地址不足的问题。

IETF 组织于 2004 年 11 月正式成立了 IPv6 overLR-WPAN(简称 6LowPan)工作组，着手制定基于 IPv6 的低速无线个人区域网标准，即 IPv6over IEEE 802.15.4，旨在将 IPv6 引入以 IEEE 802.15.4 为底层标准的无线个人区域网。其出现推动了短距离、低速率、低功耗的无线个人区域网络的发展。

6LowPan 技术底层采用 IEEE 802.15.4 规定的 PHY 层和 MAC 层，网络层采用 IPv6 协议。重点研究的技术有适配层、路由、报头压缩、分片、IPv6、网络接入和网络管理等技术。

目前在 IEEE 802.15.4 上实现传输 IPv6 数据包的关键技术如下：

① IPv6 和 IEEE 802.15.4 的协调。IEEE802.15.4 标准定义的最大帧长度是 127 字节，MAC 头部最大长度为 25 字节，剩余的 MAC 载荷最大长度为 102 字节。如果使用安全模式，不同的安全算法占用不同的字节数，比如 AES-CCM-128 需要 21 字节；AES-CCM-64 需要 13 字节；而 AES-CCM-32 需要 8 字节。这样留给 MAC 载荷最少只有 81 个字节，而在 IPv6 中，MAC 载荷最大为 1 280 字节，IEEE 802.15.4 帧不能封装完整的 IPv6 数据包，因此，要协调二者之间的关系，就要在网络层与 MAC 层之间引入适配层，用来完成分片和重组的功能。

② 地址配置和地址管理。IPv6 支持无状态地址自动配置，相对于有状态自动配置的来说，配置所需开销比较小，这正适合 WSN 设备特点。同时，由于 WSN 设备可能大量、密集地分布在人员比较难以到达的地方，实现无状态地址自动配置则更加重要。

③ 网络管理。网络管理技术对 WSN 网络很关键。由于网络规模大，而一些设备的分布地点又是人员所不能到达的，因此 WSN 网络应该具有自愈能力，要求 WSN 的网络管理技术能够在很低的开销下管理高度密集分布的设备。由于在 IEEE802.15.4 上转发 IPv6 数据提倡尽量使用已有的协议，而简单网络管理协议(SNMP)又为 IP 网络提供了一套很好的网络管理框架和实现方法，因此，6LowPan 倾向于在 WSN 上使用 SNMPv3 进行网络管理。但是，由于 SNMP 的初衷是管理基于 IP 的互联网，要想将其应用到硬件资源受限的 WSN 网络中，仍需要进一步调研和改进，比如，限制数据类型、简化基本的编码规则等。

④ 安全问题。由于使用安全机制需要额外的处理和带宽资源，并不适合 WSN 设备，而 IEEE802.15.4 在链路层提供的 AES 安全机制又相对宽松，有待进一步加强，因此寻找一种适合 WSN 的安全机制就成为 6LowPan 研究的关键问题之一。

作为短距离、低速率、低功耗的无线个域网领域的新兴技术，6LowPan 以其廉价、便捷、实用等特点，向人们展示了广阔的市场前景。凡是要求设备具有价格低、体积小、省电、可密集分布特征，而不要求设备具有很高传输率的应用，都可以应用 6LowPan 技术来实现。比如，建筑物状态监控、空间探索等方面。目前 6LowPan 技术在国外还处于研发阶段，未见真实应用的报道，如果将来 6LowPan 技术逐步普及，将给人们带来全新的工作、生活体验。

1.7 STM32W108 简介

WSN/ZigBee 技术经过多年的推广，已逐渐进入真实应用阶段，并有多家半导体公司推出了基于 802.15.4 的 RF 芯片。STM32W108 是意法半导体(ST)公司最新推出的一个完全集成的系统级芯片，该芯片集成了符合 IEEE 802.15.4 标准的 2.4GHz 收发器、32 位 ARM Cortex-M3 微处理器、Flash 闪存、RAM 存储器以及基于 ZigBee 系统使用的很多通用外设。

ST 微电子(ST Microelectronics)也是 ZigBee 联盟成员，并参与了联盟中的许多活动，可以帮助、支持客户使用 ZigBee 技术。

STM32W108 与目前其他 2.4 GHz SoC 芯片最大的区别/优势主要有三点：一是在保持低功耗的基础上，采用了 32 位 ARM Cortex-M3 内核，有别于其他 8、16 位处理器，提高了更强大的处理能力，并有广泛的 ARM 开发工具、群体支持；二是芯片内部带有功率放大器(PA)，发射输出功率可达 +7 dBm，无需外部功放就可以获得较大的通信距离；三是 STM32W108 芯片不同版本分别固化了 802.15.4 MAC、ZigBee、RF4CE 等协议栈，用户无需理解、开发网络协议，就可以进行符合相关标准的无线网络产品开发，可大大简化产品开发的技术复杂度，缩短产品上市时间。

STM32W108 的内部结构见图 1.13，本节介绍其主要特征，本书的后续章节将详细介绍其内部模块和使用方法。

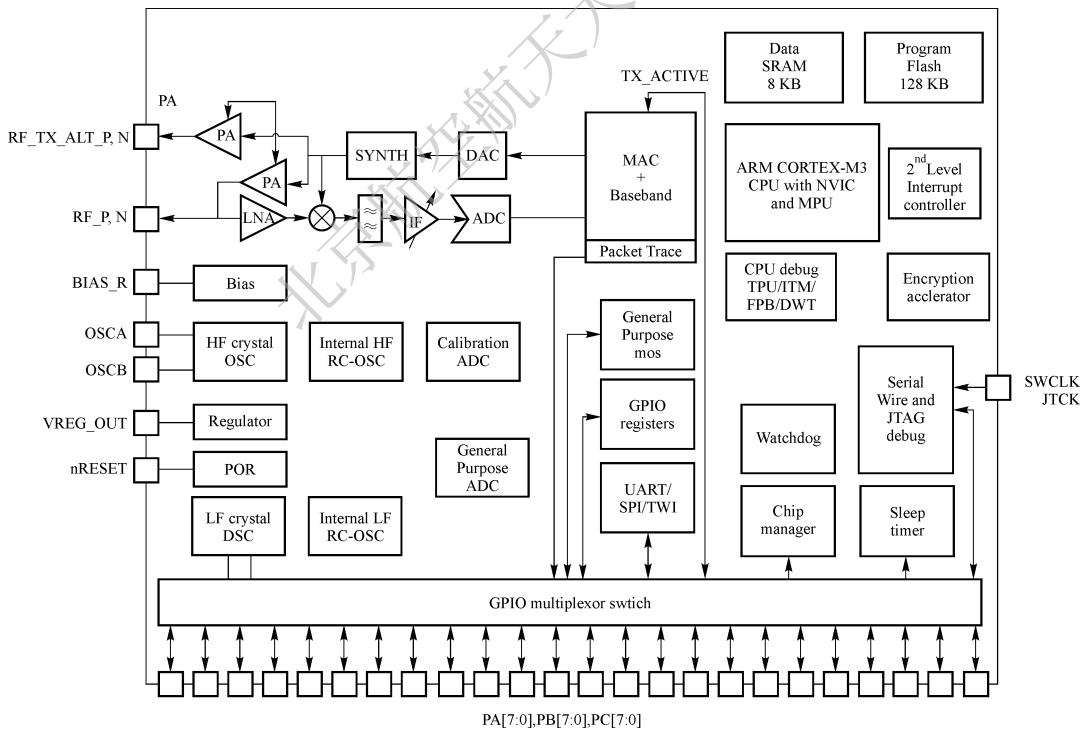

图 1.13 STM32W108 框图

STM32W108 集成了一个经过优化的 ARM Cortex-M3 微处理器，它是业界领先的 32 位

高性能内核，使用 ARM Thumb 2 指令集，具有高性能、低功耗、高内存利用率的特点。它支持两种不同的操作模式：特权模式和非特权模式。网络协议栈软件运行在特权模式，可以访问芯片的所有资源；应用程序运行在非特权模式，对于访问 STM32W108 的资源有一定限制，即允许应用程序开发人员调度事件，但同时防止其对内存和寄存器的某些禁区进行修改。这种架构可以增加系统的稳定性和可靠性。该处理器在使用外部晶振时，运行在 12 MHz 或 24 MHz；在使用内部高频 RC 振荡器时，运行在 6 MHz 或 12 MHz。

STM32W108 的 RF 收发器采用高效架构，超出 IEEE 802.15.4:2003 标准动态范围要求 15 dB。集成的接收信道滤波器允许其他使用 2.4 GHz 频段的通信标准共用这一频段，如 IEEE 802.11 和蓝牙。内部集成有稳压器、VCO、环路滤波器、功率放大器等。可选的高性能无线模式(升压模式)是软件可选的，以提高动态范围。

STM32W108 无线接收器是一种低中频、超外差接收器，它可以和其他设备共用 2.4GHz 频段。接收器使用差分信号通路，以减少对噪声干扰的敏感性。信号经过射频放大、下变频、滤波，最后经过 ADC 量化。

STM32W108 无线发射器使用一种高效的架构，可以将数据流直接调制成 VCO 频率。集成功率放大器提供了较大的输出功率。数字逻辑电路控制发射路径和输出功率校准。如果 STM32W108 使用外部功率放大器，就需要使用 TX_ACTIVE 或 nTX_ACTIVE 信号控制外部转换逻辑电路的时序。

集成的 4.8 GHz 的 VCO 和环路滤波器，最大限度地减少了片外电路。只再需要一个 24 MHz 晶振及负载电容来构成 PLL 振荡器信号。

为了保持 ZigBee 和 IEEE 802.15.4:2003 标准要求的严格的时间机制，STM32W108 在硬件上集成了许多 MAC 功能部件。MAC 硬件处理自动应答发送和接收、自动退避时延、为发送清除信道评估、自动过滤接收包。MAC 中还集成了数据包跟踪接口，它可以捕获所有 STM32W108 发送和接收到的数据包。MAC 的接口将片内 RAM 和收发基带模块连接起来。MAC 提供了基于硬件的 IEEE 802.15.4 数据包级过滤。它提供了一个精确的符号时基，以尽量减少软件栈在同步和时序方面要做的工作。此外，它还提供了对 IEEE 802.15.4 CSMA-CA 算法的定时和同步援助。

STM32W108 有 128 KB 的 Flash 内存和 8 KB 的 RAM 空间，以及 ARM 配置存储保护单元(MPU)。STM32W108 HAL 软件采用有效的磨损均衡算法，可以优化嵌入式 Flash 的寿命。

STM32W108 提供了许多先进的电源管理功能，使电池寿命更长。内部高频 RC 振荡器使处理器执行代码的速度非常快。多种睡眠模式都可达到小于 1 μA 的功耗，同时保证 RAM 中的数据不丢失。为了支持用户自定义应用，片上外设还包括 USART、SPI、TWI、ADC 以及通用定时器，同时还有 24 个 GPIO。此外，还提供了集成稳压器、上电复位电路、睡眠定时器等。

STM32W108 有 24 个 GPIO 引脚与其他外设共用或作为复用引脚，外部设备可以使用各种 GPIOs 的复用功能。集成的串行控制器 SC1 可以被配置为 SPI、TWI 或 USART；串行控制器 SC2 可以被配置为 SPI 或 TWI。

STM32W108 有一个通用 ADC，可以在单端或差分模式下，从 6 个 GPIO 引脚对模拟信号进行采样。它也可以采样 VDD_PADSA、VREF 和 GND。ADC 有两个可选电压范围：0V

到 1.2V(正常)以及在高电源电压供给下的 0.1V 到 电源电压－0.1 V。ADC 有 DMA 模式，在这个模式下，ADC 采样得到的数据被自动发送到 RAM 中。集成的参考电压 VREF 可以供给外部电路使用，外部参考电压也可以供给 ADC 使用。

STM32W108 包含 4 个振荡器：一个高频 24MHz 外部晶体振荡器、一个高频 12 MHz 内部 RC 振荡器、一个可选低频 32.768 kHz 外部晶体振荡器和一个 10 kHz 内部 RC 振荡器。多个时钟源给应用设计带来了很大的灵活性。

STM32W108 具有超低功耗，可选的深睡眠状态时钟。睡眠定时器可使用外部 32.768 kHz 晶体振荡器或内部 10 kHz RC 振荡器产生的 1 kHz 时钟作为计数时钟。另外，所有时钟都可以在低功耗模式下被禁用，在这个模式下，只有 GPIO 引脚上的外部事件才能唤醒芯片。STM32W108 从深睡眠到执行第一条 ARM® Cortex-M3 指令的启动时间非常快(一般为 100 μs)。

STM32W108 有三个电源域：时钟开启的电压源为 GPIO 焊盘和关键芯片功能部件供电；低电压源为芯片的剩下的部分供电，低电压供电在深睡眠时被禁用，以减少耗电；集成稳压器可产生 1.25 V 和 1.8 V 电压。1.8 V 稳压输出为模拟电路、RAM 和 Flash 内存供电；1.25 V 稳压输出为核心逻辑电路供电。

除了两个通用定时器，STM32W108 还有一个看门狗定时器、一个 32 位睡眠定时器以及一个 NVIC 中的 ARM? 标准系统事件定时器。看门狗定时器保证系统可以从软件崩溃和 CPU 锁定中恢复出来；睡眠定时器为系统定时，并在特定时间将系统唤醒。

STM32W108 同时支持 ARM 串行线调试(SWD)和 JTAG 调试接口。这些接口提供了实时的、非侵入性的编程和调试功能。串行线和 JTAG 提供相同的功能，但不能同时使用。串行线接口使用芯片 2 个引脚；JTAG 接口使用 5 个引脚。

后续章节将详细介绍 STM32W108 内部各种外设的特性、结构和编程使用方法。

第 2 章

STM32W108 引脚与电气特性

本章介绍 STM32W108 引脚定义、电气特性，以及型号命名规则和芯片封装，这些内容主要与 STM32W108 应用时的硬件设计相关。

2.1 STM32W108 的引脚

STM32W108 有两种封装引脚定义，分别是 48 脚 VFQFPN 封装（图 2.1）和 40 脚 VFQFPN 封装（图 2.2），后者少一些 GPIO 引脚。它们的封装尺寸在本章最后有描述。

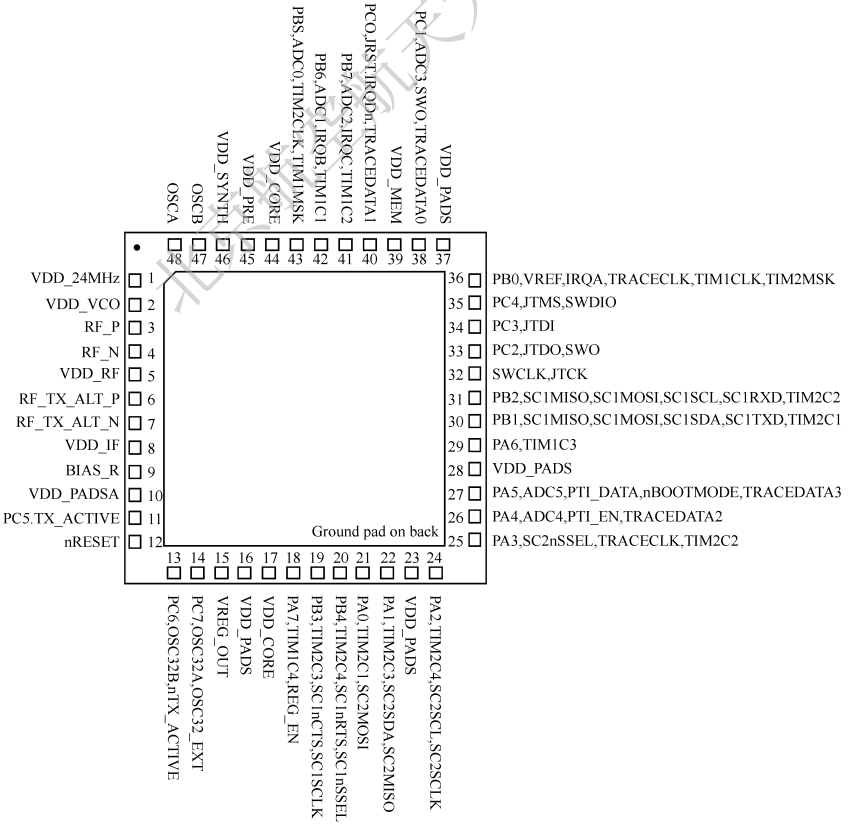

图 2.1　48 脚 VFQFPN 引脚分配

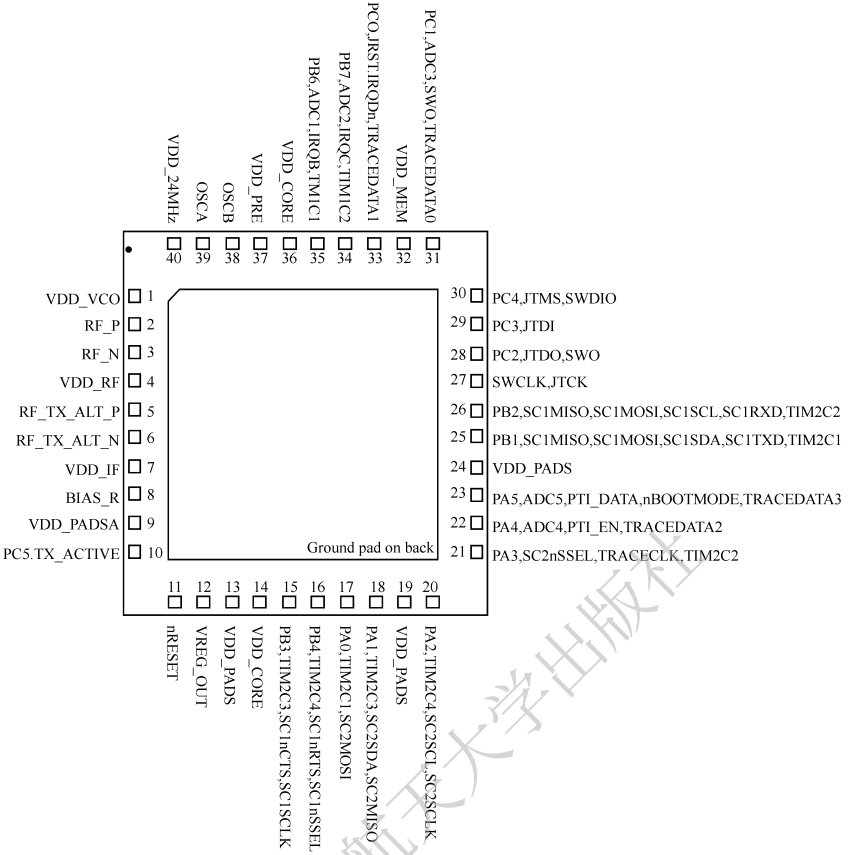

图 2.2　40 脚 VFQFPN 引脚分配

表 2.1 是对这些引脚功能的具体描述，在系统设计时应充分注意每个引脚的功能特性。

表 2.1　引脚描述

48 引脚封装引脚号	40 引脚封装引脚号	信　号	方　向	描　述
1	40	VDD_24MHz	电源	1.8 V 高频晶振供电
2	1	VDD_VCO	电源	1.8 V VCO 供电
3	2	RF_P	输入/输出	差分(RF_N)接收器输入/发送器输出
4	3	RF_N	输入/输出	差分(RF_P)接收器输入/发送器输出
5	4	VDD_RF	电源	1.8 V RF 供电(LNA 和 PA)
6	5	RF_TX_ALT_P	输出	差分(RF_TX_ALT_N)发送器输出(可选)
7	6	RF_TX_ALT_N	输出	差分(RF_TX_ALT_P)发送器输出(可选)
8	7	VDD_IF	电源	1.8 V IF 供电(混频器和滤波器)
9	8	BIAS_R	输入	偏置设置电阻
10	9	VDD_PADSA	电源	模拟部分供电(1.8 V)

续表 2.1

48 引脚封装引脚号	40 引脚封装引脚号	信　号	方　向	描　述
11	10	PC5	输入/输出	数字输入/输出
		TX_ACTIVE	输出	逻辑层控制外部 Rx/Tx 切换开关，在 Tx 模式下，STM32W108 基带控制 TX_ACTIVE 并驱动它为高；用 GPIO_PCCFGH[7:4]选择替换输出功能。
12	11	nRESET	输入	低有效芯片复位(内部上拉)
13		PC6	输入/输出	数字输入/输出
		OSC32B	输入/输出	32.768 kHz 晶振； 用 GPIO_PCCFGH[11:8]选择模拟功能。
		nTX_ACTIVE	输出	反相 TX_ACTIVE 信号(见 PC5)； 用 GPIO_PCCFGH[11:8]选择替换输出功能
14		PC7	输入/输出	数字输入/输出
		OSC32A	输入/输出	32.768 kHz 晶振； 用 GPIO_PCCFGH[15:12]选择模拟功能
		OSC32_EXT	输入	数字 32 kHz 时钟输入源
15	12	VREG_OUT	电源	稳压器输出(唤醒时 1.8V，深睡眠时 0V)
16	13	VDD_PADS	电源	芯片供电(2.1～3.6 V)
17	14	VDD_CORE	电源	1.25 V 数字部分电源退耦
18		PA7	输入/输出 高电流	数字输入/输出； 用 GPIO_DBGCFG[4]禁止 REG_EN。
		TIM1_CH4	输出	定时器 1 通道 4 输出； 用 TIM1_CCER 来使能定时器输出； 用 GPIO_PACFGH[15:12]选择替换输出功能； 用 GPIO_DBGCFG[4]来禁止 REG_EN
			输入	定时器 1 通道 4 输入(不能重映射)
		REG_EN	输出	外部稳压器使能，开漏输出(复位后使能)
19	15	PB3	输入/输出	数字输入/输出
		TIM2_CH3 (见引脚 22)	输出	定时器 2 通道 3 输出； 用 TIM2_OR[6]使能重映射； 在 TIM2_CCER 上使能定时器输出； 用 GPIO_PBCFGL[15:12]选择替换输出功能
			输入	定时器 2 通道 3 输入； 用 TIM2_OR[6]使能重映射

续表 2.1

48 引脚封装引脚号	40 引脚封装引脚号	信　号	方　向	描　述
19	15	UART_CTS	输入	串口控制器 1 的 UART CTS； 用 SC1_UARTCF[5]使能； 使用 SC1_MODE 选择 UART
		SC1SCLK	输出	串口控制器 1 的 SPI 主时钟； 在 TIM2_CCER 上关闭定时器输出或者用 TIM2_OR[6]禁止重映射； 用 SC1_SPICFG[4]使能主 SPI； 用 SC1_MODE 选择 SPI； 用 GPIO_PBCFGL[15:12]选择替换输出功能
			输入	串口控制器 1 的 SPI 从时钟； 用 SC1_SPICFG[4]使能从 SPI； 用 SC1_MODE 选择 SPI
20	16	PB4	输入/输出	数字输入/输出
		TIM2_CH4 (见引脚 24)	输出	定时器 2 通道 4 输出； 用 TIM2_OR[7]使能重映射； 在 TIM2_CCER 上使能定时器输出； 用 GPIO_PBCFGH[3:0]选择替换输出功能
			输入	定时器 2 通道 4 输入； 用 TIM2_OR[7]使能重映射
		UART_RTS	输出	串口控制器 1 的 UART RTS； 在 TIM2_CCER 上关闭定时器输出或者用 TIM2_OR[7]禁止重映射； 用 SC1_UARTCFG[5]使能； 用 SC1_MODE 选择 UART； 用 GPIO_PBCFGH[3:0]选择替换输出功能
		SC1nSSEL	输入	串口控制器 1 的 SPI 从设备选择； 用 SC1_SPICF[4]使能从设备； 用 SC1_MODE 选择 SPI
21	17	PA0	输入/输出	数字输入/输出
		TIM2_CH1 (见引脚 30)	输出	定时器 2 通道 1 输出； 用 TIM2_OR[4]来禁止重映射； 在 TIM2_CCER 上使能定时器输出； 用 GPIO_PACFGL[3:0]选择替换输出功能
			输入	定时器 2 通道 1 输入 用 TIM2_OR[4]禁止重映射

续表 2.1

48 引脚封装引脚号	40 引脚封装引脚号	信　号	方　向	描　述
21	17	SC2MOSI	输出	串口控制器 2 的 SPI 主数据输出;在 TIM2_EECR 上关闭定时器输出或者用 TIM2_OR[4]使能重映射; 用 SC2_SPICFG[4]使能主设备; 用 SC2_MODE 选择 SPI; 用 GPIO_PACFGL[3:0]选择替换输出功能
			输入	串口控制器 2 的 SPI 从数据输入; 用 SC2_ SPICF[4]使能从设备; 用 SC2_MODE 选择 SPI
22	18	PA1	输入/输出	数字输入/输出
		TIM2_CH3 (见引脚 19)	输出	定时器 2 通道 3 输出; 用 TIM2_OR[6]禁止重映射; 在 TIM2_CCER 使能定时器输出; 用 GPIO_PACFGL[7:4]选择任意输出功能
			输入	定时器 2 通道 3 输入; 用 TIM2_OR[6]禁止重映射
		SC2SDA	输入/输出	串口控制器 2 的 TWI 数据输入/输出; 在 TIM2_CCER 上关闭定时器输出或者是用 TIM2_OR[6]使能重映射; 用 SC2_MODE 选择 TWI; 用 GPIO_PACFGL[7:4]选择替换的开漏输出功能
		SC2MISO	输出	串口控制器 2 的 SPI 从数据输出; 在 TIM2_CCER 上关闭定时器输出或者用 TIM2_OR[6]使能重映射; 用 SC2_SPICFG[4]使能从设备; 用 SC2_MODE 来选择 SPI; 用 GPIO_PACFGL[7:4]选择替换输出功能
			输入	串口控制器的 SPI 主数据输入; 用 SC2_SPICFG[4]使能从设备; 用 SC2_MODE 来选择 SPI
23	19	VDD_PADS	电源	芯片供电(2.1～3.6 V)
24	20	PA2	输入/输出	数字输入/输出
		TIM2_CH4 (见引脚 20)	输出	定时器 2 通道 4 输出; 用 TIM2_OR[7]禁止重映射; 在 TIM2_CCER 上使能定时器输出; 用 GPIO_PACFGL[11:8]选择替换输出功能
			输入	定时器 2 通道 4 输入; 用 TIM2_OR[7]禁止重映射

续表 2.1

48 引脚封装引脚号	40 引脚封装引脚号	信　号	方　向	描　述
24	20	SC2SCL	输入/输出	串口控制器 2 的 TWI 时钟； 在 TIM2_CCER 上关闭定时器输出或者用 TIM2_OR[7]来使能重映射； 用 SC2_MODE 来选择 TWI； 用 GPIO_PACFGL [11:8]来选择替换的开漏输出功能
		SC2SCLK	输出	串口控制器 2 的 SPI 主时钟； 在 TIM2_CCER 上关闭定时器输出或者用 TIM2_OR[7]使能重映射； 用 SC2_SPICFG[4]使能主设备； 用 SC2_MODE 来选择 SPI； 用 GPIO_PACFGL[11:8]来选择替换输出功能
			输入	串口控制器 2 的 SPI 从时钟； 用 SC2_SPICFG[4]使能从设备； 用 SC2_MODE 选择 SPI
25	21	PA3	输入/输出	数字输入/输出
		SC2nSSEL	输入	串口控制器 2 的 SPI 从选择； 用 SC2_SPICFG[4]使能从设备； 用 SC2_MODE 选择 SPI
		TRACECLK (见引脚 36)	输出	同步的 CPU 跟踪时钟； 关闭 TIM2_CCER 上的定时器输出或者用 TIM2_OR[5]使能重映射； 在 ARM 核上使能跟踪接口； 用 GPIO_PACFGL[15:12]选择替换输出功能
		TIM2_CH2 (见引脚 31)	输出	定时器 2 通道 2 输出； 用 TIM2_OR[5]禁止重映射； 使能 TIM2_CCER 上的定时器输出； 用 GPIO_PACFGL[15:12]来选择替换输出功能
			输入	定时器 2 通道 2 输入； 用 TIM2_OR[5]禁止重映射
26	22	PA4	输入/输出	数字输入/输出
		ADC4	模拟	ADC 输入 4，用 GPIO_PACFGH[3:0]选择模拟功能
		PTI_EN	输出	包跟踪接口(PTI)的帧信号； 在 ARM 核上禁止跟踪接口； 用 GPIO_PACFGH[3:0]选择替换输出功能

续表 2.1

48 引脚封装引脚号	40 引脚封装引脚号	信　号	方　向	描　述
26	22	TRACEDATA2	输出	同步的 CPU 跟踪数据位 2； 在 ARM 核上选择 4 线同步跟踪接口； 在 ARM 核上使能跟踪接口； 用 GPIO_PACFGH[3:0]来选择替换输出功能
27	23	PA5	输入/输出	数字输入/输出
		ADC5	模拟	ADC 输入 5，用 GPIO_PACFGH[7:4]选择模拟功能
		PTI_DATA	输出	包跟踪接口(PTI)的数据信号； 在 ARM 核上禁止跟踪接口； 用 GPIO_PACFGH[7:4]选择替换输出功能
		nBOOTMODE	输入	复位后内置引导激活； 在 NRST 复位期间或复位后立即有效。见系统模块的复位章节。
		TRACEDATA3	输出	同步的 CPU 跟踪数据位 3； 在 ARM 上选择 4 线同步跟踪接口； 在 ARM 核上使能跟踪接口； 用 GPIO_PACFGH[7:4]选择替换输出功能
28	24	VDD_PADS	电源	芯片供电(2.1～3.6 V)
29	25	PA6	输入/输出高电流	数字输入/输出
		TIM1_CH3	输出	定时器 1 通道 3 输出； 使能 TIM1_CCER 上的定时器输出； 用 GPIO_PACFGH[11:8]来选择替换输出功能
			输入	定时器 1 通道 3 输入(不能被重映射)
30		PB1	输入/输出	数字输入/输出
		SC1MISO	输出	串口控制器 1 的 SPI 从数据输出； 关闭 TIM2_CCER 上的定时器输出，或者用 TIM2_OR[4]禁止重映射； 用 SC1_MODE 选择 SPI； 用 SC1_SPICR 选择从设备； 用 GPIO_PBCFGL[7:4]来选择替换输出功能
		SC1MOSI	输出	串口控制器 1 的 SPI 主数据输出； 关闭 TIM2_CCER 上的定时器输出，或者用 TIM2_OR[4]禁止重映射； 用 SC1_MODE 来选择 SPI； 用 SC1_SPICR 来选择主设备； 用 GPIO_PBCFGL[7:4]来选择替换输出功能

续表 2.1

48 引脚封装引脚号	40 引脚封装引脚号	信　号	方　向	描　述
30	25	SC1SDA	输入/输出	串口控制器 1 的 TWI 数据； 关闭 TIM2_CCER 上的定时器输出，或者用 TIM2_OR[4]禁止重映射； 用 SC1_MODE 来选择 TWI； 用 GPIO_PBCFGL[7:4]来选择替换的开漏输出功能
		SC1TXD	输出	串口控制器 1 的 UART 发送数据； 关闭 TIM2_CCER 上的定时器输出，或者用 TIM2_OR[4]禁止重映射； 用 SC1_MODE 来选择 UART； 用 GPIO_PBCFGL[7:4]来选择替换输出功能
		TIM2_CH1 （见引脚 21）	输出	定时器 2 通道 1 输出； 用 TIM2_OR[4]使能重映射； 在 TIM2_CCER 上使用定时器输出； 用 GPIO_PACFGL[7:4]选择替换输出功能
			输入	定时器 2 通道 1 输入； 用 TIM2_OR[4]禁止重映射
31	26	PB2	输入/输出	数字输入/输出
		SC1MISO	输入	串口控制器 1 的 SPI 主数据输入； 用 SC1_MODE 选择 SPI； 用 SC1_SPICR 选择主设备
		SC1MOSI	输入	串口控制器 1 的 SPI 从数据输入； 用 SC1_MODE 选择 SPI； 用 SC1_SPICR 选择从设备
		SC1SCL	输入/输出	串口控制器 1 的 TWI 时钟； 关闭 TIM2_CCER 上的定时器输出，或者用 TIM2_OR[5]禁止闭重映射； 用 SC1_MODE 来选择 TWI； 用 GPIO_PBCFGL[11:8]来选择替换的开漏输出功能
		SC1RXD	输入	串口控制器 1 的 UART 接收数据； 用 SC1_MODE 选择 UART
		TIM2_CH2 （见引脚 25）	输出	定时器 2 通道 2 输出； 用 TIM2_OR[5]使能重映射； 在 TIM2_CCER 上使能定时器输出； 用 GPIO_PBCFGL[11:8]来选择替换输出功能
			输入	定时器 2 通道 2 输入； 用 TIM2_OR[5]使能重映射

续表 2.1

48 引脚封装引脚号	40 引脚封装引脚号	信　号	方　向	描　述
32	27	SWCLK	输入/输出	串行线调试器时钟输入/输出； 仅在串行线模式选择时(见 JTMS 描述,引脚 35)
		JTCK	输入	来自调试器的 JTAG 时钟输入； 当在 JTAG 模式下时选择(默认模式,见 JTMS 描述,引脚 35) 使能内部下拉
33	28	PC2	输入/输出	数字输入/输出； 用 GPIO_DBGCFG[5]使能
		JTDO	输出	JTAG 数据输出； 当在 JTAG 模式下选择(默认模式,见 JTMS 描述,引脚 35)
		SWO	输出	串行线异步跟踪输出； 在 ARM 核上选择异步跟踪接口； 在 ARM 核上使能跟踪接口； 用 GPIO_PCCFGL[11:8]使能串行线模式(见 JTMS 描述,引脚 35)； 使能内部上拉
34	29	PC3	输入/输出	数字输入/输出 用 GPIO_DBGCFG[5]使能数字 IO 或串行线模式(见 JTMS 描述)
		JTDI	输入	来自调试器的 JTAG 数据输入； 在 JTAG 模式下选择(默认模式,见 JTMS 描述,引脚 35) 使能内部上拉
35	30	PC4	输入/输出	用 GPIO_DBGCFG[5]使能数字输入/输出
		JTMS	输入	来自调试器的 JTAG 模式选择； 在 JTAG 模式下选择(默认模式)； 在上电或 NRST 拉低后 JTAG 模式被使能； 通过调试器,使用 ARM 定义的协议,可以选择串行线模式； 使能内部上拉
		SWDIO	输入/输出	串行线调试(SWD)双向数据； 使能串行线模式(见 JTMS 描述)； 通过调试器,使用 ARM 定义的协议来选择串行线调试模式； 使能内部上拉

续表 2.1

48 引脚封装引脚号	40 引脚封装引脚号	信　号	方　向	描　述
36		PB0	输入/输出	数字输入/输出
		VREF	模拟输出	ADC 参考输出； 用 GPIO_PBCFGL[3:0]使能模拟功能
		VREF	模拟输入	ADC 参考输入； 用 GPIO_PBCFGL[3:0]使能模拟功能； 用 ST 一个系统功能使能参考输出
		IRQA	输入	外部中断源 A
		TRACECLK (见引脚 25)	输出	同步的 CPU 跟踪时钟； 在 ARM 核上使能跟踪接口； 用 GPIO_PBCFGL[3:0]来选择替换输出功能
		TIM1CLK	输入	定时器 1 外部时钟输入
		TIM2MSK	输入	定时器 2 外部时钟屏蔽输入
37		VDD_PADS	电源	芯片供电(2.1～3.6 V)
38	31	PC1	输入/输出	数字输入/输出
		ADC3	模拟	ADC 输入 3； 用 GPIO_PCCFGL[7:4]使能模拟功能
		SWO(见引脚 33)	输出	串行线异步跟踪输出； 在 ARM 核上选择异步跟踪接口； 在 ARM 核上使能跟踪接口； 用 GPIO_PCCFGL[7:4]选择替换输出功能
		TRACEDATA0	输出	异步 CPU 跟踪数据位 0； 在 ARM 核上的选择 1、2 或 4 线异步跟踪接口； 在 ARM 核上使能跟踪接口； 用 GPIO_PCCFGL[7:4]选择替换输出功能
39	32	VDD_MEM	电源	1.8V 供电(flash 和 RAM)
40	33	PC0	输入/输出高电流	数字输入/输出； 用 GPIO_DBGCFG[5]使能数字 IO 或串行线模式(见 JTMS 描述,引脚 35)并禁止 TRACEDATA1
		JRST	输入	来自调试器的 JTAG 复位输入； 在 JTAG 模式时选择(默认模式,见 JTMS 描述),并禁止 TRACEDATA1； 使能内部上拉
		IRQDn①	输入	默认外部中断源 D
		TRACEDATA1	输出	异步 CPU 跟踪数据位 1； 在 ARM 核上选择 2 或 4 线异步跟踪接口； 在 ARM 核上使能跟踪接口； 用 GPIO_PCCFGL[3:0]选择替换输出功能

续表 2.1

48 引脚封装引脚号	40 引脚封装引脚号	信　号	方　向	描　述
41	34	PB7	输入/输出高电流	数字输入/输出
		ADC2	模拟	ADC 输入 2 用 GPIO_PBCFGH[15:2]使能模拟功能
		IRQC①	输入	默认外部中断源 C
		TIM1_CH2	输出	定时器 1 通道 2 输出； 使能 TIM1_CCER 定时器输出； 用 GPIO_PBCFGH[15:12]选择替换输出功能
			输入	定时器 1 通道 2 输入(不能重映射)
42	35	PB6	输入/输出高电流	数字输入/输出
		ADC1	模拟	ACD 输入 1； 用 GPIO_PBCFGH[11:8]使能模拟功能
		IRQB	输入	外部中断源 B
		TIM1_CH1	输出	定时器 1 通道 1 输出； 在 TIM1_CCER 上使能定时器输出； 用 GPIO_PBCFGH[11:8]选择替换输出功能
			输入	定时器通道 1 输入(不能重映射)
43		PB5	输入/输出	数字输入/输出
		ADC0	模拟	ADC 输入 0； 用 GPIO_PBCFGH[7:4]使能模拟功能
		TIM2CLK	输入	定时器 2 外部时钟输入
		TIM1MSK	输入	定时器 2 外部时钟屏蔽输入
44	36	VDD_CORE	电源	1.25 V 数字核电源退耦
45	37	VDD_PRE	电源	1.8 V 分频器供电
46		VDD_SYNTH	电源	1.8 V 合成器供电
47	38	OSCB	输入/输出	24 MHz 晶振，使用外部时钟输入到 OSCA 时浮空
48	39	OSCA	输入/输出	24 MHz 晶振或外部时钟输入
49	41	GND	地	在封装的底部中央的接地焊盘

① 用 GPIO_IRQSEL 和 GPIO_IRQDSEL 寄存器，IRSC 和 IRQD 外部中断能够被重映射到任何数字 I/O 引脚。

2.2　操作条件

2.2.1　绝对最大额定值

超过绝对最大额定值可能会造成器件的永久性损坏，长时间在最大额定值条件下使用，也可能会影响器件的可靠性。

表 2.2　电压特性

参　数	最小值	最大值	单　位
输入电源电压(VDD_PADS)	−0.3	+3.6	V
模拟、存储器、内核电压(VDD_24MHz、VDD_VCO、VDD_RF、VDD_IF、VDD_PADSA、VDD_MEM、VDD_PRE、VDD_SYNTH、VDD_CORE)	−0.3	+2.0	V
RF_P,N、RF_TX_ALT_P,N 电压	−0.3	+3.6	V
RF 输入功率(正确接收包的最高电平见 RF 接收特性表) RX 信号经过一个无损耗的匹配变压器(balun)	−	+15	dBm
任何 GPIO (PA[7:0]、PB[7:0]、PC[7:0])、SWCLK、NRST、VREG_OUT 电压	−0.3	VDD_PADS+0.3	V
BIAS_R、OSCA、OSCB 电压	−0.3	VDD_PADSA+0.3	V

表 2.3　电流特性

符　号	参　数	最大值	单　位
IVDD	流入 VDD/VDDA 电源线的总电流	150	mA
IVSS	流出 VSS 地线的总电流	150	
IIO	任何 I/O 和控制引脚的输出拉电流	25	
	任何 I/O 和控制引脚的输出灌电流	−25	
IINJ(PIN)	NRST 引脚的注入电流	±5	
	HSE OSC_IN 和 LSE OSC_IN 引脚的注入电流	±5	
	任何其他引脚的注入电流	±5	
ΣIINJ(PIN)	总注入电流(所有 I/O 和控制引脚的总和)	± 25	

表 2.4　热特性

符　号	参　数	值	单　位
TSTG	储存温度范围	−40～+140	℃
TJ	最高结温	150	℃

2.2.2　正常操作条件

STM32W108 在正常工作条件见表 2.5。

表 2.5　正常操作条件

符　号	参　数	最小值	典型值	最大值	单　位
−	输入电源电压(VDD_PADS)	2.1	—	3.6	V
−	模拟和存储器输入电压 (VDD_24MHz、VDD_VCO、VDD_RF、VDD_IF、VDD_PADSA、VDD_MEM、VDD_PRE 和 VDD_SYNTH)	1.7	1.8	1.9	V
−	内核输入电压(VDD_CORE)	1.18	1.25	1.32	V
TOPER	工作温度范围	−40	—	+85	℃

2.2.3 上电操作条件

1. 上电复位（POR HV 和 POR LV）

STM32W108 对 3 个供电域的电压进行监测，如果电源电压低于低电压阈值，就会发出复位信号，如果电源电压高于高电压阈值，复位解除。电源复位有以下三种复位电压检测电路：

① POR HV 检测始终供电域(always on domain)，电压阈值如表 2.6 所列。

② POR LVcore 检测核供电域(core domain)，电压阈值如表 2.7 所列。

③ POR LVmem 检测存储供电域(memory domain)，电压阈值如表 2.8 所列。

表 2.6 POR HV 电压阈值

参 数	测试条件	最小值	典型值	最大值	单 位
始终上电域复位解除		1.0	1.2	1.4	V
始终上电域复位发出		0.5	0.6	0.7	V
电源电压上升时间	0.5～1.7 V	—	—	250	μs

表 2.7 POR LVcore 电压阈值

参 数	测试条件	最小值	典型值	最大值	单位
1.25 V 域复位解除		0.9	1.0	1.1	V
1.25 V 域复位发出		0.8	0.9	1.0	V

表 2.8 POR LVmen 电压阈值

参 数	测试条件	最小值	典型值	最大值	单位
1.8 V 域复位解除		1.35	1.5	1.65	V
1.8 V 域复位发出		1.26	1.4	1.54	V

POR LVcore 和 POR LVmem 复位源合并为一个信号复位源 POR LV 给复位模块，因为这两个事件的检测都需要复位相同的系统模块。

2. NRST 复位引脚(表 2.9)

低电平有效的 NRST 引脚用来复位系统，这个引脚是一个施密特触发输入。为了保证良好的抗干扰和开关信号振铃，这个引脚经过一个专门的滤波器，最后产生复位源 RSTB 给复位模块。

表 2.9 RSTB 的复位滤波器特性

参 数	最小值	典型值	最大值	单 位
复位滤波器时间常数	2.1	12.0	16.0	ms
保证复位的复位脉宽	26.0	—	—	ms
保证不会复位的复位脉宽	0	—	1.0	ms

2.3 时钟频率

2.3.1 高频内部时钟特性(表 2.10)

表 2.10 高频 RC 振荡器特性

参 数	测试环境	最小值	典型值	最大值	单 位
复位频率		6	12	20	MHz
频率步进			0.5		MHz
占空比		40		60	%
电源变化测试:1.8～1.7 V			5	%	

2.3.2 高频外部时钟特性(表 2.11)

表 2.11 高频晶体振荡器的特性

参 数	测试环境	最小值	典型值	最大值	单 位
频率	—	—	24	—	MHz
精度	—	−40	—	+40	ppm
占空比	—	40	—	60	%
相位噪声(at 100 kHz offset)	—	—	—	−120	dBc/Hz
最大偏置时的启动时间	—	—	—	1	ms
最佳偏置时的启动时间	—	—	—	2	ms
电流消耗	—	—	200	300	μA
最大偏置时的电流消耗	—	—	—	1	mA
高 ESR 的晶体	—	—	—	100	Ω
−负载电容	—	—	—	10	pF
−晶体电容	—	—	—	7	pF
−晶体功耗	—	—	—	200	μW
低 ESR 的晶体	—	—	—	60	Ω
−负载电容	—	—	—	18	pF
−晶体电容	—	—	—	7	pF
−晶体功耗	—	—	—	1	mW

2.3.3 低频内部时钟特性(表 2.12)

表 2.12 低频 RC 振荡器特性

参 数	测试条件	最小值	典型值	最大值	单 位
标称频率	经过整修	9	10	11	kHz
模拟修正步长	—	—	1	—	kHz
电源相关	电压变化从 3.6 V 到 3.1 V 或 2.6V 到 2.1 V(无重新校正)	—	—	1	%
频率温漂	温度变化从－40℃到＋85℃时频率变化(无重新校正)	—	2	—	%

2.3.4 低频外部时钟特性(表 2.13)

表 2.13 低频晶体振荡器特性

参 数	测试条件	最小值	典型值	最大值	单 位
频率	—	—	32.768	—	kHz
精度	初始、温度和老化	－100	—	＋100	ppm
负载电容 xin	—	—	27	—	pF
负载电容 xout	—	—	18	—	pF
晶振 ESR	—	—	—	100	kΩ
启动时间	—	—	—	2	s
消耗电流	At 25℃，VDD_PADS=3.0 V	—	—	0.5	μA

2.3.5 ADC 特性

表 2.14 描述了在 25℃、VDD_PADS 为 3.0V、采样时钟频率为 1MHz 时，测得的 ADC 主要参数。ADC_HVSELP 和 ADC_HVSELN 编程为 0，禁用输入缓冲。单端测试在 $f_{input}=7.7\%\ f_{Nyquist}$，0 dBFS 电平(满量程是 1.2Vp-p)的条件下进行。差分测试在 $f_{input}=7.7\%\ f_{Nyquist}$，－6 dBFS level(满量程是 2.4Vp-p)的条件下进行。

表 2.15 描述了 ADC 的其他特性。

表 2.14 1 MHz 采样率下 ADC 的主要参数

参 数	性 能							
ADC_PERIOD	0	1	2	3	4	5	6	7
转换时间/μs	32	64	128	256	512	1024	2048	4096

续表 2.14

参　数	性　能							
奈奎斯特采样频率/kHz	15.6	7.81	3.91	1.95	0.977	0.488	0.244	0.122
3 dB 截止/kHz	9.42	4.71	2.36	1.18	0.589	0.294	0.147	0.0736
INL(peak)	0.04	0.04	0.05	0.09	0.2	0.3	0.6	1.2
INL(RMS)	0.02	0.02	0.02	0.03	0.05	0.1	0.2	0.4
DNL(peak)	0.04	0.03	0.04	0.04	0.04	0.09	0.12	0.1
DNL(RMS)	0.02	0.01	0.01	0.01	0.01	0.03	0.03	0.03
ENOB(单周期测试)	5.5	7.0	7.5	10.0	11.5	12.0	12.5	12.5
SNR/dB								
单端	35	44	53	62	69	74	75	74
差分	36	45	54	62	68	71	73	73
SINAD/dB								
单端	35	44	53	61	66	67	68	67
差分	35	44	53	62	68	71	73	72
SDFR/dB								
单端	63	69	70	70	70	69	70	70
差分	61	70	75	74	78	80	84	89
THD/dB								
单端	−55	−63	−67	−69	−69	−69	−69	−69
差分	−57	−64	−74	−82	−87	−93	−94	−93
ENOB(从 SNR)								
单端	5.8	7.3	8.8	10.3	11.5	12.3	12.5	12.3
差分	7.0	8.5	10.0	11.3	12.3	12.8	13.1	13.1
ENOB(从 SINAD)								
单端	5.8	7.3	8.8	10.1	11.0	11.1	11.3	11.1
差分	7.0	8.3	9.8	11.3	12.3	12.8	13.1	13.0
等效的 ADC 位数	5 [15:11]	6 [15:10]	7 [15:9]	8 [15:8]	9 [15:7]	10 [15:6]	11 [15:5]	12 [15:4]

表 2.15　ADC 其他特性

参　数	最小值	典型值	最大值	单　位
VREF	1.17	1.2	1.23	V
VREF 输出电流	—	—	1	mA
VREF 负载电容	—	—	10	nF
外部 VREF 电压范围	1.1	1.2	1.3	V

续表 2.15

参　数	最小值	典型值	最大值	单　位
外部 VREF 输入阻抗	1	—	—	MΩ
最小输入电压				V
输入缓冲禁用	0	—	—	
输入缓冲使能	0.1	—	—	
最大输入电压				V
输入缓冲禁用	—	—	VREF	
输入缓冲使能	—	—	VDD_PADS − 0.1	
单端信号范围				V
输入缓冲禁用	0	—	VREF	
输入缓冲使能	0.1	—	VDD_PADS − 0.1	
差分信号范围				V
输入缓冲禁用	−VREF	—	+VREF	
输入缓冲使能	−VDD_PADS + 0.1	—	+VDD_PADS − 0.1	
共模范围				V
输入缓冲禁用				
输入缓冲使能	0	VDD_PADS/2	VREF	
输入偏移	−10	—	10	mV
输入阻抗				MΩ
1 MHz 采样时钟	1	—	—	
6 MHz 采样时钟	0.5	—	—	
不采样	10	—	—	

2.4　直流电气特性

表 2.16 列出了直流电气特性。

表 2.16　DC 电气特性

参　数	条　件	最小值	典型值	最大值	单位
稳压器输入电压（VDD_PADS）		2.1	—	3.6	V
电源电压范围（VDD_MEM）	稳压器输出或外部输入	1.7	1.8	1.9	V
电源电压范围（VDD_CORE）	稳压器输出	1.18	1.25	1.32	V

续表 2.16

参　数	条　件	最小值	典型值	最大值	单位
深睡眠电流					
静态电流，内部 RC 振荡器禁用	−40 ℃，VDD_PADS = 3.6 V	—	0.4	—	μA
	+25 ℃，VDD_PADS = 3.6 V	—	0.4	—	μA
	+85 ℃，VDD_PADS = 3.6 V	—	0.7	—	μA
静态电流，包括内部 RC 振荡器	−40 ℃，VDD_PADS = 3.6 V	—	0.7	—	μA
	+25 ℃，VDD_PADS = 3.6 V	—	0.7	—	μA
	+85 ℃，VDD_PADS = 3.6 V	—	1.1	—	μA
静态电流，包括 32.768 kHz 振荡器	−40 ℃，VDD_PADS = 3.6 V	—	0.8	—	μA
	+25 ℃，VDD_PADS = 3.6 V	—	1.0	—	μA
	+85 ℃，VDD_PADS = 3.6 V	—	1.5	—	μA
静态电流，包括内部 RC 振荡器和 32.768 kHz 振荡器	−40 ℃，VDD_PADS = 3.6 V	—	1.1	—	μA
	+25 ℃，VDD_PADS = 3.6 V	—	1.3	—	μA
	+85 ℃，VDD_PADS = 3.6 V	—	1.8	—	μA
模拟深睡眠电流（调试模式）	调试器不工作	—	300	—	μA
复位电流					
静态电流，NRST 有效	Typ at 25 ℃/3 V Max at 85 ℃/3.6 V	—	1.2	2.0	mA
处理器和外围电流					
ARM Cortex-M3、RAM 和 Flash	25 ℃、1.8 V 存储器和 1.25 V 内核电压，ARM Cortex-M3 在 12 MHz 晶振下运行，无线和所有外设均关闭	—	6.0	—	mA
ARM Cortex-M3、RAM 和 Flash	25 ℃、1.8 V 存储器和 1.25 V 内核电压，ARM Cortex-M3 在 24 MHz 晶振下运行，无线和所有外设均关闭	—	7.5	—	mA
ARM Cortex-M3、RAM 和 Flash 休眠电流	25 ℃、1.8 V 存储器和 1.25 V 内核电压，ARM Cortex-M3 在 12 MHz 晶振下运行，无线和所有外设均关闭	—	3.0	—	mA
ARM Cortex-M3、RAM 和 Flash 休眠电流	25 ℃、1.8 V 存储器和 1.25 V 内核电压，ARM Cortex-M3 在 6 MHz 高频 RC 振荡器下运行，无线和所有外设均关闭	—	2.0	—	mA
串行控制器电流	每个控制器都工作在最大数据速率	—	0.2	—	mA

续表 2.16

参　数	条　件	最小值	典型值	最大值	单位
通用定时器电流	每个定时器都工作在最大时钟速率	—	0.25	—	mA
通用 ADC 电流	工作在最大采样频率,DMA 使能	—	1.1	—	mA
Rx 接收电流					
无线接收器、MAC 和基带	ARM® Cortex-M3 睡眠	—	22.0	—	mA
总接收电流(无线接收器、MAC、基带、CPU、RAM、Flash)	VDD_PADS = 3.0 V, 25 ℃, ARM Cortex-M3 工作在 12 MHz	—	25.0	—	mA
	VDD_PADS = 3.0 V, 25 ℃, ARM Cortex-M3 工作在 24 MHz	—	26.5	—	mA
升压模式 Rx 总电流(无线接收器、MAC、基带、CPU、RAM、Flash)	VDD_PADS = 3.0 V, 25 ℃, ARM Cortex-M3 工作在 12 MHz	—	27.0	—	mA
	VDD_PADS = 3.0 V, 2 5℃, ARM Cortex-M3 工作在 24 MHz	—	28.5	—	mA
Tx 发射电流					
无线电发射器、MAC 和基带	25 ℃、1.8 V core; 功率输出+3 dBm; ARM Cortex-M3 睡眠	—	26.0	—	mA
总发射电流(无线发射器、MAC、基带、CPU、RAM、Flash)	VDD_PADS = 3.0 V,25 ℃; 最高功率设置(+7 dBm); ARM Cortex-M3 工作在 12 MHz	—	42.0	—	mA
	VDD_PADS = 3.0 V,25 ℃; +3 dBm 功率设置; ARM Cortex-M3 工作在 12 MHz	—	29.5	—	mA
	VDD_PADS = 3.0 V,25 ℃; 0dBm 功率设置; ARM Cortex-M3 工作在 12 MHz	—	27.0	—	mA
	VDD_PADS = 3.0 V,25 ℃; 最低功率设置; ARM Cortex-M3 工作在 12 MHz	—	21.0	—	mA

图 2.3 显示了不同发射功率下电流的变化(ARM Cortex-M3 运行在 24 MHz)。

图 2.4 显示了在 ST 参考设计板测试的设置发射功率与典型输出功率。

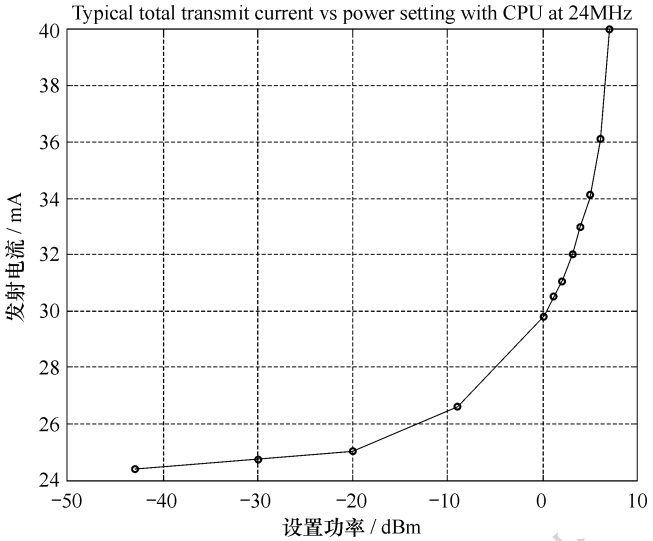

图 2.3　发射功率消耗

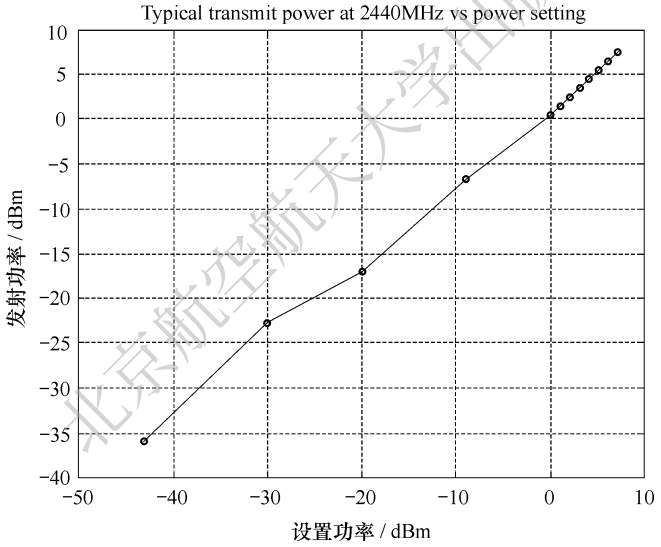

图 2.4　发射输出功率

2.5　数字 I/O 特性

表 2.17 列出了 STM32W 的数字 I/O 特性。数字 I/O 电源由三个 VDD_PADS 引脚提供，这些引脚上的电压决定了 I/O 电平。

表 2.17　数字 I/O 特性

参　数	条　件	最小值	典型值	最大值	单　位
电源电压(稳压器输入)	VDD_PADS	2.1	—	3.6	V

续表 2.17

参　数	条　件	最小值	典型值	最大值	单位
低施密特开关门限	VSWIL 施密特输入门限从高到低	0.42 × VDD_PADS	—	0.50 × VDD_PADS	V
高施密特开关门限	VSWIH 施密特输入门限从低到高	0.62 × VDD_PADS	—	0.80 × VDD_PADS	V
逻辑 0 的输入电流	IIL	—	—	−0.5	μA
逻辑 1 的输入电流	IIH	—	—	+0.5	μA
输入上拉电阻值	RIPU	24	29	34	kΩ
输入下拉电阻值	RIPD	24	29	34	kΩ
逻辑 0 的输出电压	VOL(标准输出 IOL = 4 mA,大电流为 8 mA)	0	—	0.18 × VDD_PADS	V
逻辑 1 的输出电压	VOH(标准输出 IOH = 4mA,大电流为 8mA)	0.82 × VDD_PADS	—	VDD_PADS	V
输出高电流(标准电流引脚)	IOHS	—	—	4	mA
输出低电流(标准电流焊点)	IOLS	—	—	4	mA
输出高电流大电流引脚:PA6、PA7、PB6、PB7、PC0	IOHH		—	8	mA
输出低电流大电流引脚:PA6、PA7、PB6、PB7、PC0	IOLH	—	—	8	mA
总输出电流(所有 I/O 引脚)	IOH+IOL	—	—	40	mA
OSC32A 的输入电压门限		0.2 × VDD_PADS	—	0.8 × VDD_PADS	V
OSCA 的输入电压门限		0.2 × VDD_PADSA	—	0.8 × VDD_PADSA	V

2.6　非 RF 系统电气特性

表 2.18 列出了 STM32W108 的非 RF 系统级特性,即进入和唤醒深睡眠的时间。

表 2.18　非 RF 系统电气特性

参数	条件	最小值	典型值	最大值	单位
系统从深睡眠中唤醒时间	在 6 MHz 内部 RC 时钟下,从唤醒事件到 CPU 执行第一条指令,包括电源上升时间和振荡器启动时间	—	100	—	μs
进入深睡眠的关机时间	从最后一条 ARM® Cortex-M3 指令到深睡眠模式	—	5	—	μs

2.7 RF 电气特性

2.7.1 Rx 接收

表 2.19 列出了 STM32W108 上集成的 IEEE 802.15.4 接收器的关键参数。

表 2.19 Rx 接收特性

参 数	条件	最小值	典型值	最大值	单 位
频率范围		2400	—	2500	MHz
灵敏度(升压模式)	1%PER,有 IEEE 802.15.4—2003 定义的 20 字节数据包	—	−102	−96	dBm
灵敏度	1% PER,有 IEEE 802.15.4 −2003 定义的 20 字节数据包	—	−100	−94	dBm
高边邻道抑制	−82 dBm IEEE 802.15.4 信号	—	35	—	dB
低边邻道抑制	−82 dBm IEEE 802.15.4 信号	—	35	—	dB
2nd高边邻道抑制	−82 dBm IEEE 802.15.4 信号	—	46	—	dB
2nd低边邻道抑制	−82 dBm IEEE 802.15.4 信号	—	46	—	dB
其他信道抑制	−82 dBm IEEE 802.15.4 信号	—	40	—	dB
802.11g 的抑制,+12MHz 或 −13 MHz	−82 dBm IEEE 802.15.4 信号	—	36	—	dB
正常最大输入信号电平		0	—	—	dBm
同信道抑制	−82 dBm IEEE 802.15.4 信号	—	−6	—	dBc
相对频率误差 (IEEE 802.15.4 要求 2x40ppm)		−120	—	+120	ppm
相对定时误差 (IEEE 802.15.4 要求 2x40ppm)		−120	—	+120	ppm
线形 RSSI 范围	如 IEEE 802.15.4 定义	40	—	—	dB
RSSI 范围		−90	—	−40	dBm

2.7.2 Tx 发射

表 2.20 列出了 STM32W108 上集成的 IEEE 802.15.4 发射器的关键参数。

表 2.20 Tx 发射特性

参 数	条 件	最小值	典型值	最大值	单 位
最大输出功率(升压模式)	设置为最高功率	—	8	—	dBm
最大输出功率	设置为最高功率	1	5	—	dBm

续表 2.20

参　数	条　件	最小值	典型值	最大值	单　位
最小输出功率	设置为最低功率	—	−55	—	dBm
误差矢量幅度	IEEE 802.15.4 定义最大为 35%	—	5	15	%
载波频率误差		−40	—	+40	ppm
最佳传输功率的负载阻抗		—	200+j90TBC	—	Ω
相对 PSD 屏蔽	3.5 MHz 外	−20	—	—	dB
绝对 PSD 屏蔽	3.5 MHz 外	−30	—	—	dBm

2.8　型号命名与封装

2.8.1　STM32W108 型号命名

STM32W108 是一个单芯片的无线单片机，从内置不同固件(Firmware)的角度看，可分为 5 个细分型号(见图 2.5 STM32W108 型号命名)，分别是空片(无后缀)、Ember ZigBee 协议栈(后缀为 1)、RF4CE 协议栈(后缀为 3)和 IEEE 802.15.4 MAC(后缀为 4)，这几种芯片适合不同的应用和软件开发，目前 ST 推荐使用的是 STM32W108CBU61/4，具体协议使用方法参见第 6 章。

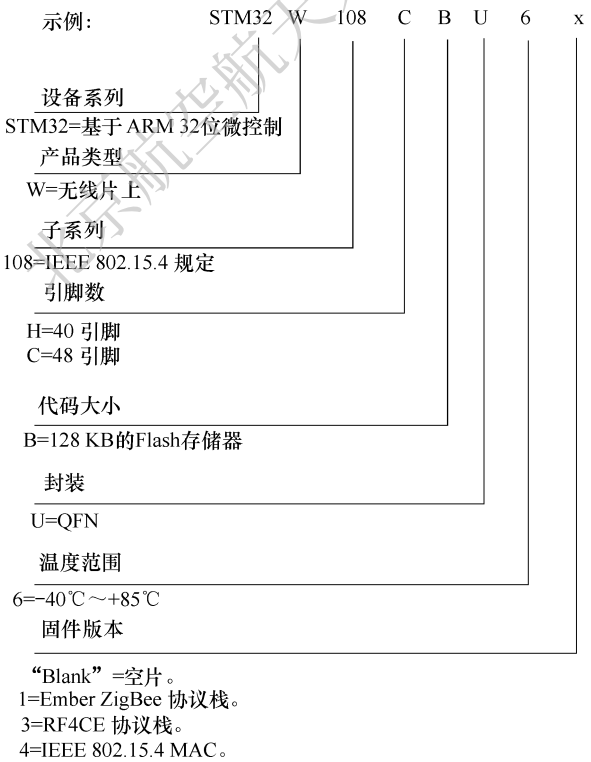

图 2.5　STM32W108 型号命名

2.8.2 STM32W108 封装尺寸

STM32W108 有两种封装，分别是 VFQFPN48 和 VFQFPN40，它们的封装形式和尺寸详见图 2.6、图 2.7 和表 2.21、表 2.22。

为了满足环保要求，ST 根据设备的环境要求提供不同等级的 ECOPACK 封装，ECOPACK 规格、等级的定义和产品状态请访问：www.st.com。

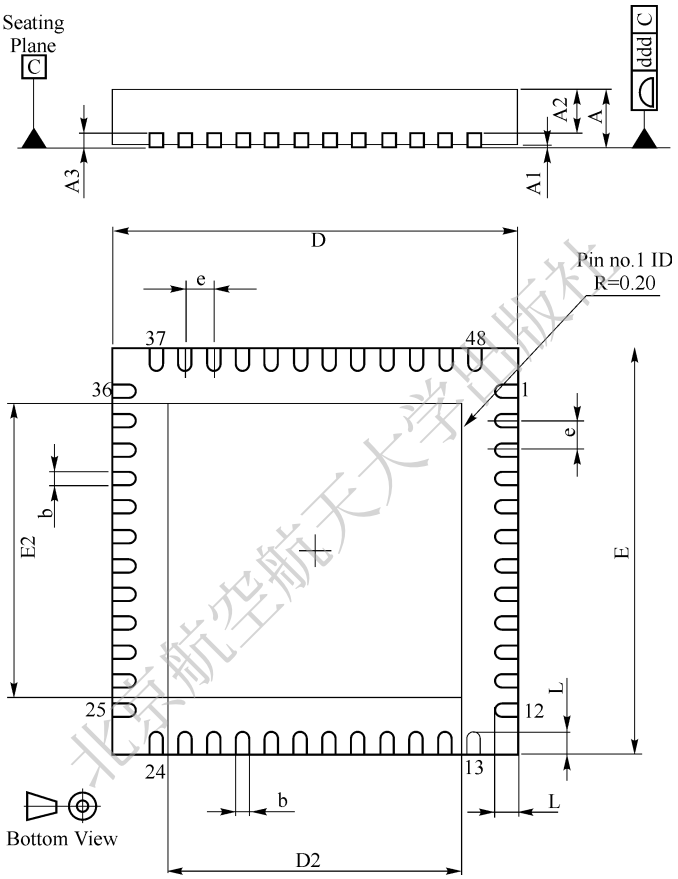

图 2.6 VFQFPN48 7 mm×7 mm 封装

表 2.21 VFQFPN48 7 mm×7 mm 封装尺寸

Symbol	Millimeters			Inches[①]		
	Min.	Typ.	Max.	Min.	Typ.	Max.
A	0.800	0.900	1.000	0.0315	0.0354	0.0394
A1		0.20	0.050		0.0008	0.0020
A2		0.650	1.000		0.0256	0.0394
A3		0.250			0.0098	
b	0.180	0.230	0.300	0.0071	0.0091	0.0118

Symbol	Millimeters			Inches①		
	Min.	Typ.	Max.	Min.	Typ.	Max.
D	6.850	7.000	7.150	0.2697	0.2756	0.2815
D2	2.250	4.700	5.250	0.0886	0.1850	0.2067
E	6.850	7.000	7.150	0.2697	0.2756	0.2815
E2	2.250	4.700	5.250	0.0886	0.1850	0.2067
e	0.450	0.500	0.550	0.0177	0.0197	0.0217
L	0.300	0.400	0.500	0.0118	0.0157	0.0197
ddd			0.080			0.0031

① 英寸值是毫米值四舍五入到小数点后 4 位转换而来。

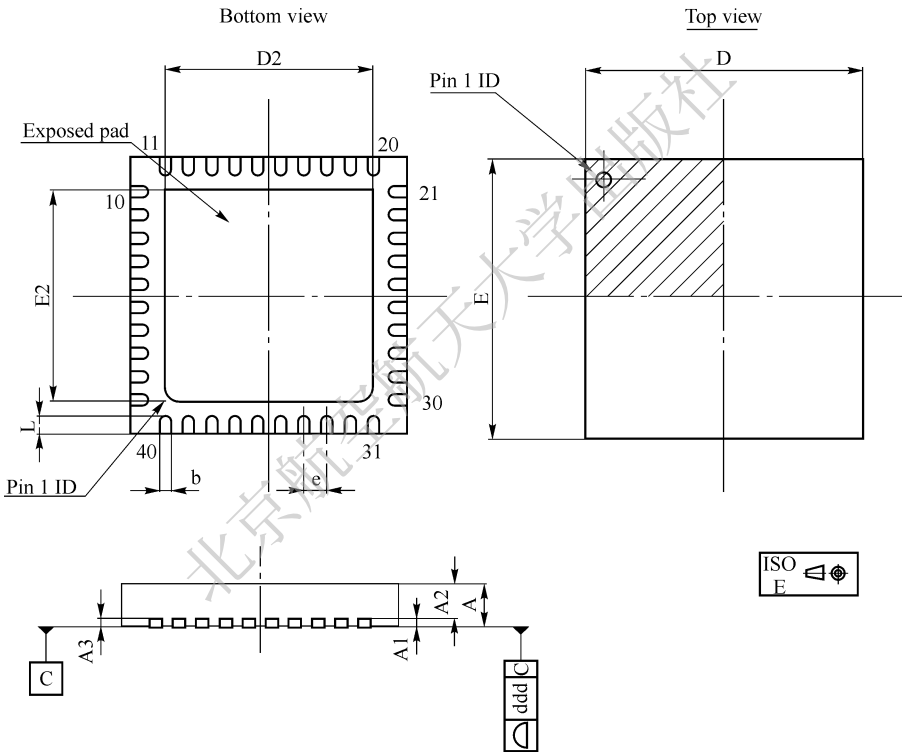

图 2.7　VFQFPN40 6mm×6mm 封装

表 2.22　VFQFPN40 6mm×6mm 封装尺寸

Symbol	millimeters			inches①		
	Min.	Typ.	Max.	Min.	Typ.	Max.
A	0.800	0.900	1.000	0.0315	0.0354	0.0394
A1		0.020	0.050		0.0008	0.0020
A2		0.720	1.070		0.0283	0.0421

续表 2.22

Symbol	millimeters			inches①		
	Min.	Typ.	Max.	Min.	Typ.	Max.
A3		0.200			0.0079	
b	0.180	0.250	0.300	0.0071	0.0098	0.0118
D	5.900	6.000	6.100	0.2323	0.2362	0.2402
D2	4.500	4.550	4.700	0.1772	0.1791	0.1850
E		6.000			0.2362	
E2	4.500	4.550	4.700	0.1772	0.1791	0.1850
e		0.500			0.0197	
L	0.350	0.400	0.450	0.0138	0.0157	0.0177
ddd			0.080			0.0031

① 英寸值是毫米值四舍五入到小数点后 4 位转换而来。

第 3 章

STM32W108 系统模块

STM32W108 是一个基于 ARM Cortex-M3 内核的 MCU 与无线射频(RF)结合的 SoC，内部既有一般 MCU 的通用资源和外设，也有特殊的射频模块。本章介绍 STM32W108 中的系统共用模块和射频模块，其他一些通用外设，如 GPIO、通用定时器、串行接口、中断等，将在第 4 章进行详细说明。

STM32W108 的系统模块包括电源、复位、时钟、系统定时器、电源管理、加密引擎和调试接口，图 3.1 显示这些模块以及它们之间的交互。

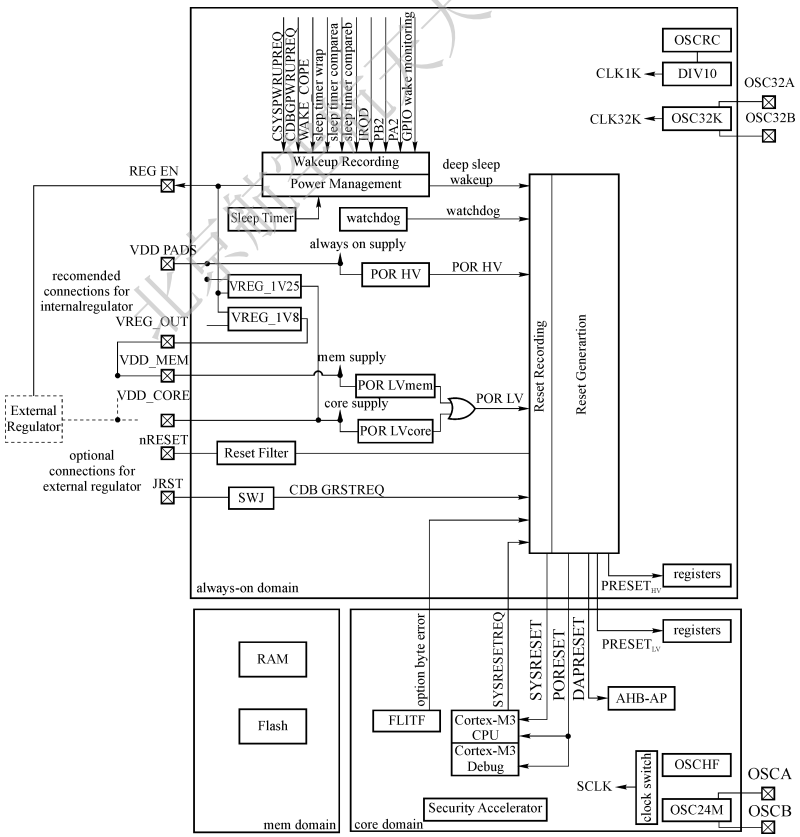

图 3.1　STM32W108 系统模块图

3.1 内部供电域

STM32W108 内部电路分为 3 个供电域：

- 始终上电域(always on domain)：包括 GPIO 控制器和睡眠定时器，这些模块必须始终保持供电。
- 核心域(core domain)：包括 CPU、嵌套中断向量控制器(NVIC)和通用外设，为了节省功耗，这个域可以进入深睡眠模式。
- 存储域(memory domain)：包括 RAM 和 FLASH 存储器，这个域由电源控制器来管理。当进入深睡眠模式时，这个域的 RAM 部分将由始终上电域(always on domain)供电，以保持 RAM 中的内容，而其他稳压器将被关闭。在深睡眠模式，FLASH 部分将完全断电。

3.1.1 内部稳压电源

推荐的电源配置是使用内部稳压器来给核心域和存储域供电，内部稳压器(VREG_1V25 和 VREG_1V8)产生 1.25 V 和 1.8 V 的额定电压。1.25 V 是由内部稳压器供给核心域和外部引脚的；1.8 V 的供电由内部稳压器供给外部引脚，同时可以回送芯片给存储域供电。在深睡眠模式下电压稳压器失效。

当使用内部稳压器时，芯片的 4 个 VDD 引脚必须连接 2.1～3.6 V 范围的电源，稳压器的 VREG_1V8 输出引脚(VREG_OUT)必须连接到 VDD_MEM、VDD_PADSA、VDD_VCO、VDD_RF、VDD_IF、VDD_PRE 和 VDD_SYNTH 引脚，VREG_1V25 稳压输出必须与 VDD_CORE 引脚相连。

当内部稳压器使能时，需在外部输出引脚(VREG_OUT 和 VDD_CORE)加一个适当的电容以避免振荡。1.8V 稳压器能提供的最大输出电流为 50 mA；1.25 V 稳压器能提供最大的输出电流为 10 mA。稳压器由芯片的数字部分控制，如表 3.1 所示。

表 3.1 集成稳压器特性

规格参数	最小	典型	最大	单位	说明
输入电压范围	2.1		3.6	V	VDD_PADS
1V8 稳压器输出	−5%	1.8	+5%	V	初始化后稳压器输出
1V8 稳压器复位后输出	−5%	1.75	+5%		复位后稳压器输出
1V25 稳压器输出	−5%	1.25	+5%	V	初始化后稳压器输出
1V25 复位后稳压器输出	−5%	1.45	+5%	V	复位后稳压器输出
1V8 稳压器电容		2.2		μF	低 ESR 的钽电容 ESR 大于 2 Ω 小于 10 Ω 退耦小于 100 nF 陶瓷电容
1V25 稳压器电容		1.0		μF	陶瓷电容器(0603)
1V8 稳压器输出电流	0		50	mA	稳压器输出电流
1V25 稳压器输出电流	0		10	mA	稳压器输出电流
空载电流		600		μA	没有负荷时电流

规格参数	最小	典型	最大	单位	说明
1V8 稳压器电流限制		200		mA	短路电流限制值
1V25 稳压器电流限制		25		mA	短路电流限制值
1V8 稳压器启动时间		50		μs	0 V 到上电复位,2.2 μF 电容
1V25 稳压器启动时间		50		μs	0 V 到上电复位,1.0 μF 电容

STM32W108 在深睡眠模式下可以用开漏(open-drain)GPIO PA7 来控制外部稳压器(参见通用输入输出所述),PA7 输出低电平禁止外部稳压器工作。当用一个外部稳压器时,电流消耗增加大约 2mA,内部稳压器需要通过软件禁用。

3.1.2 外接稳压电源

片上稳压器可以选择不用,核心域和存储域可以由外部稳压器来供电,为简化外部电源设计,核心域可以用一个外部 1.8 V 稳压器来供电。注意:如果核心域供电用 1.8 V 代替 1.25 V,则功耗也会增加。稳压器使能信号 REG_EN 用来控制外接稳压器,这是一个开漏输出信号,需要外接上拉电阻。如果不需要用 REG_EN 来控制外接稳压器,则它可以被禁用。

使用外接稳压器时,4 个 VDD_PAD 引脚必须有 1.8 V 到 3.6 V 的电源供电,VREG_1V8 稳压器输出引脚(VREG_OUT)必须浮空,外部额定 1.8 V 电源需要连接到 VDD_CORE、VDD_MEM、VDD_PADSA、VDD_VCO、VDD_RF、VDD_IF、VDD_PRE 和 VDD_SYNTH 引脚。

3.2 复位与时钟

复位与时钟是系统模块中两个与系统启动运行相关的部件。

3.2.1 复 位

STM32W108 的复位来自多个源,每个复位源注入到复位检测控制逻辑,根据系统的状态和复位事件来复位系统的不同部分。

1. 复位源

上电复位(高电压 POR HV 和低电压 POR LV),参见第 2 章中的上电操作条件。

(1) 树型狗复位

STM32W108 有一个由内部 1 kHz 参考时钟驱动的树型狗定时器(参见树型狗定时器章节),当树型狗定时器超时溢出时,它将产生一个复位源 WATCHDOG_RESET。

(2) 软件复位

ARM Cortex-M3 CPU 可以在软件控制下引发一个复位,即产生复位源 SYSRESETREQ 给复位模块。

注意:当使用某些外部仿真/调试器时,如果调试器发出 SYSRESETREQ,芯片有可能被锁住而需要一个外部引脚复位,或者需要重新上电。建议最好不要在应用程序里直接写 SCS_AIRCR 寄

存器，可以使用 EMBERZNET HAL 复位 API，这样可以保证芯片在安全的时钟周期触发复位。

(3) 可选字节错误(option byte error)

FLASH 存储控制器包括一个状态机，在系统启动时从 FLASH 里的信息块读取配置信息，在从 FLASH 读取的可选字节上将执行错误检测，如果检测失败，将发出一个错误信号，由复位源 OPT_BYTE_ERROR 传递信号到复位模块。

如果一个可选字节错误被检测出，系统将再次启动并重新检测，如果这个错误被再次检测出，这个过程将再被重复直到第三次失败时停止。系统将进入一个可以进行恢复的仿真深睡眠状态，在这个状态中，FLASH 存储器读出保护被强制开启，以防止应用安全受到威胁(如试图通过调试接口读取 FLASH 中的代码)。

(4) 调试复位

串行线/JTAG 接口(SWJ)用于访问 SWJ 调式端口(SWJ-DP)的寄存器。通过设置 SWJ-DP 寄存器中的 CDBGRSTREQ 位，将产生复位源 CDBGRSTREQ 给复位模块。

(5) JRST

STM32W108 的一个引脚可以遵循 JTAG 标准的需求配置为 JTAG 复位。这个 JRST 输入与所有其他的复位源独立，当它有效时只复位 JTAG TAP，不复位其他任何片上硬件。如果 STM32W108 处于串行线模式(非 JTAG 模式)或 SWJ 被禁用，这个输入无效。

(6) 深睡眠复位

电源管理模块通知复位模块进入和退出深睡眠状态。深睡眠复位用于以下情形：在进入深睡眠之前；存储域和核心域断电时；在深睡眠状态时；从深睡眠中唤醒时；在重新上电时，直到由 POR LV 检测到可靠的电源电压。

在深睡眠时，电源管理模块允许一个特殊的仿真深睡眠状态，以维持存储域和核心域的供电。

2. 复位记录

STM32W108 记录最后一个使系统重启的复位条件，记录的复位条件包括：

- POWER_HV　　始终上电域电源失效。
- POWER_LV　　核心域或存储域电源失效。
- RSTB　　NRST 引脚复位。
- W_DOG　　树型狗定时器超时。
- SW_RST　　ARM Cortex-M3 内核的 SYSERSETREQ 软件复位。
- WAKE_UP_DSLEEP　　从深睡眠唤醒。
- OPT_BYTE_FAIL　　从 FLASH 存储器中读取可选字节时检测出错误。

复位事件源寄存器(RESET_EVENT)用于读取上一个复位事件，除了保持原始复位事件的 OPT_BYTE_FAIL 位以外，其他位都互斥。

注意：当 CPU LOCKUP 在软件上标志为复位条件时，CPU LOCKUP 就不再专门作为一个复位事件。设置 CPU LOCKUP 来表明 CPU 进入了一个无法恢复的异常，执行停止，但是不产生复位，这样是为了让调试器来说明这个错误的原因。建议在真实应用(一般无调试器)中，使能树型狗，从而使 STM32W108 可以被自动重启。

3. 复位产生

复位模块响应复位源，并且产生以下的复位信号：

- PORESET　复位 ARM Cortex-M3 CPU 和 ARM CORTEX-M3 系统调试部件(断

点、数据监视和跟踪、指令跟踪宏单元、嵌套中断向量控制器)。ARM 定义了 PORESET 作为上电复位。

- SYSRESET　复位 ARM Cortex-M3 CPU,但不复位核心调试和系统调试部件。因此一个运行的系统可以在不影响调试配置的情况下被复位。
- DRPRESET　复位 SWJ 的 AHB 访问端口(AHB-AP)。
- PRESETHV　始终上电域的外设复位,用于在深睡眠期间需要保留配置的外设。
- PRESETLV　核心域的外设复位,用于在深睡眠期间不需要保留配置的外设。

表 3.2 说明了不同复位源产生的复位信号。

表 3.2　复位源产生的复位信号

复位源	产生的复位信号				
	PORESET	SYSRESET	DAPRESET	PRESETHV	PRESETLV
POR HV	X	X	X	X	X
POR LV(在深度睡眠状态)	X	X	X		X
POR LV(不在深度睡眠状态)	X	X	X	X	X
RSTB	X	X		X	X
树型狗复位		X		X	X
软件复位		X		X	X
可选字节错误 R	X	X			X
正常深度睡眠模式	X	X	X		X
仿真深度睡眠		X			X
调试复位		X			

4. 复位寄存器

复位源寄存器(RESET_EVENT)。

地址偏移:0X4000002C。

复位值:0X00000001。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CPU_LOCKUP	OPT_BYTE_FAIL	WAKE_UP_DSLEEP	SW_RST	W_DOG	RSTB_PIN	POWER_LV	POWER_HV
								r	r	r	r	r	r	r	r

BIT 7　CPU_LOCKUP:为‘1’时,内核锁定复位。

BIT 6　OPT_BYTE_FAIL:为‘1’时,可选字节加载失败(可能与其他位同时有效)复位。

BIT 5　WAKE_UP_DSLEEP:为‘1’时,从深睡眠中唤醒复位。

BIT 4　SW_RST:为‘1’时,软件复位。

BIT 3　W_DOG:为‘1’时,树型狗超时复位。

BIT 2　RSTB_PIN:为‘1’时,外部引脚复位。

BIT 1　POWER_LV：为‘1’时，核心域电源失效(或者之前失效)复位。

BIT 0　POWER_HV：一直为‘1’，表示正常供电。

3.2.2　时　钟

STM32W108 片上集成了 4 个时钟振荡器：

- 高频 RC 振荡器。
- 24 MHz 晶体振荡器。
- 10 kHz RC 振荡器。
- 32.768 kHz 晶体振荡器。

图 3.2 是 STM32W108 时钟结构图，显示了所有时钟源以及它们的路径供给区域。

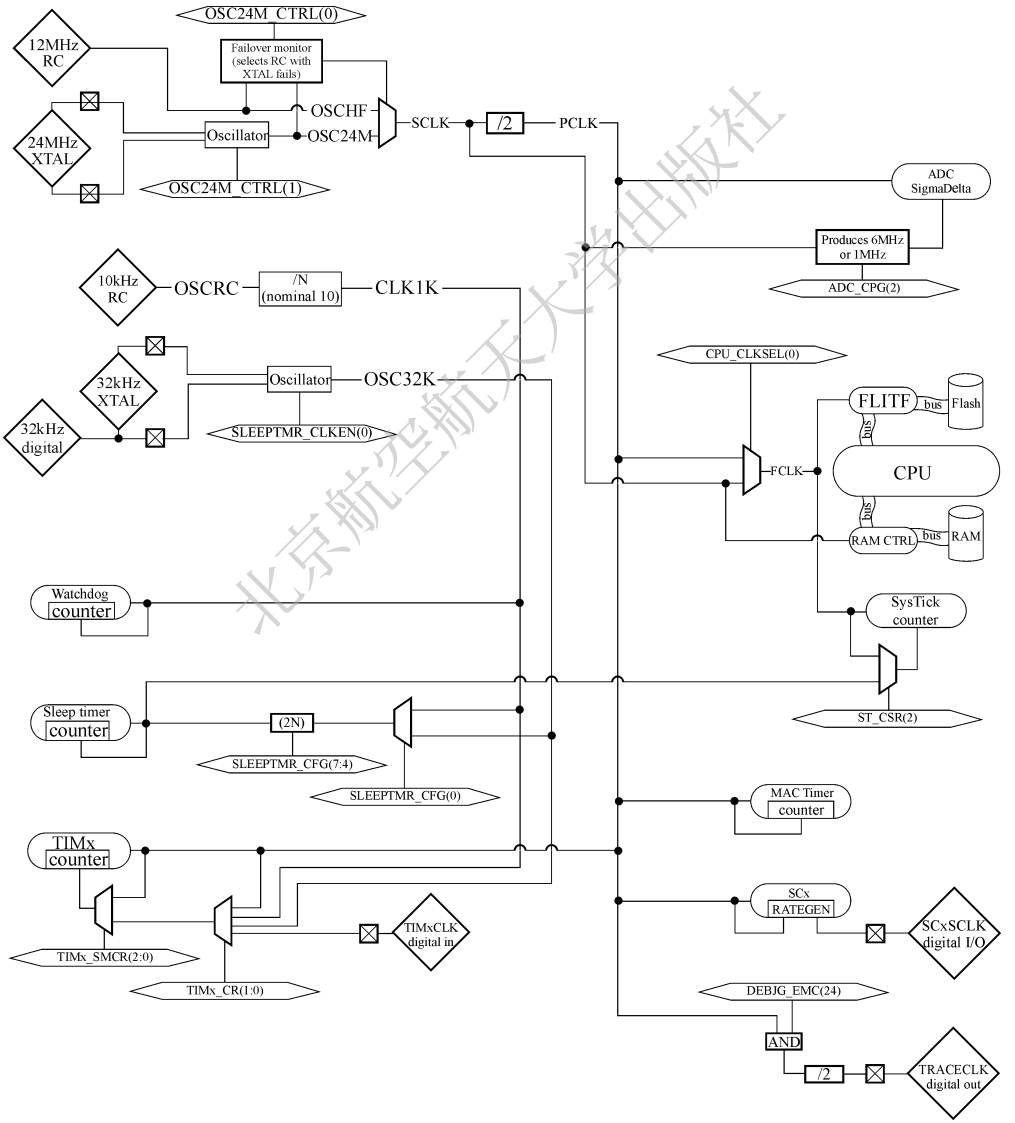

图 3.2　STM32W108 时钟

1. 高频内部 RC 振荡器(OSCHF)

当核心域上电时,高频 RC 振荡器(OSCHF)作为系统时钟源,复位时的正常频率为 12 MHz。除了射频外设,其他大多数外设运作都可使用 OSCHF 时钟源。应用软件必须明确,外设可能受不同的时钟速度驱动,依赖于使用 OSCHF 或者 OSC24M。OSCHF 的频率误差是 0.5 MHz,可用高频晶体振荡器来校准,OSCHF 校正后的精度可达±250 kHz ±40 ppm。由于 OSCHF 频率精度较低,UART 和 ADC 等外设可能无法使用。

2. 高频晶体振荡器(OSC24M)

高频晶体振荡器(OSC24M)需要外接一个 24MHz、精度为±40 ppm 的晶振,既要考虑到元件成本和电流消耗,外接晶振也要满足 ESR 要求范围,这个晶体振荡器有一个软件可编程的偏置电路,可以减少电流消耗。当使用 OSC 24 MHz 时钟源时,包括射频部分的所有外设都可以正常运作。

如果这个 24 MHZ 的晶振失效,一个硬件故障机制将会强制系统切换到使用高频 RC 振荡器作为主时钟源,并产生一个不可屏蔽中断(NMI)信号发送到 ARM Cortex-M3 嵌套向量中断控制器(NVIC)。

3. 低频内部 RC 振荡器(OSCRC)

低频内部 RC 振荡器(OSCRC)作为一个内部定时参考,复位后正常频率是 10 kHz,由 ST 软件校准这个时钟到 10 kHz。从这个 10 kHz 振荡器(OSCRC),ST 软件校准 N 分频,产生一个 1kHz 的参考时钟 CLK1K。

4. 低频晶体振荡器(OSC32K)

低频 32.768 kHz 晶体振荡器(OSC32K)作为片上定时器的一个可选的定时参考,这个振荡器设计使用一个外接的手表晶振。

5. 时钟切换

STM32W108 的主系统时钟有 2 个切换机制,提供 4 种时钟模式,如表 3.3 所列。

OSC24M_CTRL 寄存器的 OSC24M_SEL 位用来选择使用高频 RC 振荡器(OSCHF)或者高频晶体振荡器(OSC24M)作为主系统时钟(SCLK)。外围时钟(PCLK)的频率始终是 SCLK 频率的一半。

CPU_CLKSEL 寄存器的 CPU_CLK_SEL 位用来选择使用 PCLK 或者 SCLK 作为 ARM Cortex-M3 CPU 时钟(FCLK)。默认和推荐的操作模式是以较低的 PCLK 频率(12MHz)来运行 CPU。如果使用较高的 SCLK(24 MHz),可以提供更好的处理性能,但会增加系统的功耗。

除了这些模式,当 FLASH 编程使能时,硬件将调用进一步的自动控制,为了保证 FLASH 控制器的定时精度,在进行 FLASH 编程和擦除操作时,FCLK 频率将被强制设置为 12MHz。

表 3.3　系统时钟模式

OSC24M_SEL	CPU_CLK_SEL	SCLK	PCLK	FCLK	
				FLASH 编程未激活	FLASH 编程激活
0（OSCHF）	0（NORMAL CPU）	12 MHz	6 MHz	6 MHz	12 MHz
0（OSCHF）	1（FAST CPU）	12 MHz	6 MHz	12 MHz	12 MHz
1（OSC24M）	0（NORMAL CPU）	24 MHz	12 MHz	12 MHz	12 MHz
1（OSC24M）	1（FAST CPU）	24 MHz	12 MHz	24 MHz	12 MHz

6. 时钟切换寄存器

（1）XTAL 或 OSCHF 主时钟选择寄存器（OSC24M_CTRL）

地址偏移量：0X4000401C。

复位值：0X00000000。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														OSC24M_EN	OSC24M_SEL
														rws	rws

BIT 1　OSC24M_EN：当设置为“1”时，24 MHz 晶体振荡器作为主时钟。

BIT 0　OSC24M_SEL：当设置为‘0’时，选择 OSCHF；当设置为“1”时，选择 XTAL。

（2）CPU 时钟源选择寄存器（CPU_CLK_SEL）

地址偏移：0X4000 4020。

复位值：0X0000 0000。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CPU_CLK_SEL
															rws

BIT 0　CPU_CLK_SEL：当设置为“0”时，选择 12 MHz CPU 时钟。当设置为“1”时，选择 24 MHz CPU 时钟。

注意：这个时钟选择也同时决定了 RAM 控制器的运行速率是与 HCLK 一样（CPU_CLK_SEL =“1”），还是 HCLK 速率的两倍（CPU_CLK_SEL = ‘0’）。

3.3　系统定时器

3.3.1　树型狗定时器

STM32W108 集成了一个树型狗定时器，使能后可以为软件故障或 ARM Cortex-M3

CPU 锁定提供保护。在上电后，树型狗定时器默认是被禁止的。看门狗定时器使用校准的 1 kHz时钟(CLK1K)作为工作参考时钟，溢出时间为 2.048 s。在 1.792 s 时，会产生一个低水印(LOW WATER MARK)中断并触发一个 NMI 给 ARM CORTEX-M3 NVIC 作为初期警告。树型狗定时器使能以后，必须在它溢出之前通过写 WDOG_RESTART 寄存器来周期性地复位看门狗定时器。

在仿真器/调试器停止 ARM Cortex-M3 时，看门狗定时器可以被暂停。可以通过设置 SLEEP_CONFIG 寄存器的 DBG_PAUSE 位来使能这个功能。

如果在深睡眠状态时低频内部 RC 振荡器(OSCRC)被关闭，CLK1K 将停止，随之看门狗定时器也将停止计数，并在深睡眠状态下暂停。

为了防止意外改变看门狗定时器的使能/禁止位，需要用两步来处理：如果要使能看门狗定时器，应用程序必须首先写使能码 0XEABE 到 WDOG_CTRL 寄存器，然后设置 WDOG_EN 寄存器位。如果要禁止看门狗定时器，应用程序必须首先写禁止码 0XDEAD 到 WDOG_CTRL 寄存器，然后设置 WDOG_DIS 寄存器位。

3.3.2 睡眠定时器

STM32W108 集成了一个 32 位的定时器专门用于系统计时，或在特定情况下从睡眠状态中唤醒，这个睡眠定时器可以使用校准的 1 kHz 时钟参考(CLK1K)，或者 32 kHz 晶振(CLK32K)，默认的时钟源是内部 1 kHz 时钟，睡眠定时器的时钟源由 SLEEPTMR_CLKSEL 寄存器来选择。

睡眠定时器有一个预分频器，分频系数以 2^N 的形式，其中 N 可以编程设置为 1～2^15，这个分频器考虑到要设置很长的睡眠周期。定时器提供两个比较输出和回卷(WRAP)检测，它们都可以用来产生中断或者唤醒事件。

当仿真器/调试器终止 ARM Cortex-M3 时，睡眠定时器被暂停，不需要设置任何寄存器位。

为了节省电流，在深睡眠状态下，低频内部 RC 振荡器(OSCRC)可以被关闭。如果 OSCRC 在深睡眠状态下被关闭，而且低频 32.768 kHz 晶体振荡器也没有被使用，那么这个睡眠定时器在深睡眠状态下也将暂停，睡眠定时器将不会发出唤醒事件来唤醒 STM32W108。

3.3.3 事件定时器

SYSTICK 定时器是在嵌套向量中断控制器(NVIC)中的一个 ARM 标准系统定时器。SYSTICK 定时器可以由 FCLK(进入 CPU 的时钟)或者睡眠定时器的时钟提供时钟驱动。FCLK 可以通过设置 CPU_CLK_SEL 位来选择 SCLK 或者 PCLK(参见时钟切换)。

3.4 电源管理

STM32W108 电源管理系统能实现最低的深睡眠电流消耗，并且仍然能保证灵活的唤醒源、定时器和调试器操作。STM32W108 有 4 个睡眠模式：

- 空闲睡眠(Idle Sleep):使 CPU 进入空闲状态,指令执行被暂停直到中断发生。所有的供电域都保持有效供电,并且没有部件被复位。
- 深睡眠 1:这是主要的一种深睡状态,在这种状态下,核心供电域关闭,睡眠定时器激活。
- 深睡眠 2:睡眠定时器不工作,以进一步降低节省功耗,其他和深睡眠 1 模式相同,在这种模式下,睡眠定时器不能唤醒 STM32W108。
- 深睡眠 0(也就是仿真深睡眠):芯片在不关闭核心域供电的情况下,模仿一个真的深睡眠状态。核心域保持供电,并且除了系统调试部件(ITM, DWT, FPB, NVIC)外,所有的外围设备保持在复位状态。这种睡眠模式的目的,是为了让 STM32W108 的软件能在维持调试配置(如断点)的情况下实现深睡眠周期。

3.4.1 唤醒源

在深睡眠状态下,STM32W108 可以经过一定的步骤回到运行状态,唤醒源根据深睡眠 1 和深睡眠 2 分成两种情况。

以下的唤醒源在深睡眠 1 和 2 情况下都有效:

- GPIO 激活唤醒:由于 GPIO 状态的改变而唤醒。
- 串行控制器 1 唤醒:由于 GPIO 引脚 PB2 状态的改变而唤醒。
- 串行控制器 2 唤醒:由于 GPIO 引脚 PA2 状态的改变而唤醒。
- IRQD 唤醒:由于 IRQD 状态的改变而唤醒,因为 IRQD 可以配置指向任何 GPIO,所以这个唤醒源也就是 GPIO 唤醒的另一种形式。
- 设置 CDBGPWRUPREQ 唤醒:由于设置了 SWJ 调试端口中的 CDBGPWRUPREQ 位而唤醒。
- 设置 CSYSPWRUPREQ 唤醒:由于设置了 SWJ 调试端口中的 CSYSPWRUPREQ 位而唤醒。

以下的唤醒源仅在深睡眠 1 模式下有效,因为在深睡眠 2 模式下,睡眠定时器没有激活:

- 睡眠定时器比较 A 匹配唤醒。
- 睡眠定时器比较 B 匹配唤醒。
- 睡眠定时器计数回卷(wrap)唤醒。

由于 SWJ 需要写存储器来设置中断源,而且 SWJ 在深睡眠 0 模式下只能访问部分寄存器,以下的唤醒源只在深睡眠 0 状态下有效:

- 写 WAKE_CORE 寄存器位唤醒。

唤醒记录模块监测所有可能的唤醒源。因为事件不断地被记录(不仅是在深睡眠状态),由于另外一个事件可能在第一个唤醒事件时,或者是在 STM32W108 唤醒过程中发生,所以可能会有多种唤醒源被记录。

3.4.2 基本睡眠模式

电源管理状态图(图 3.3)显示了电源管理控制器的基本操作。

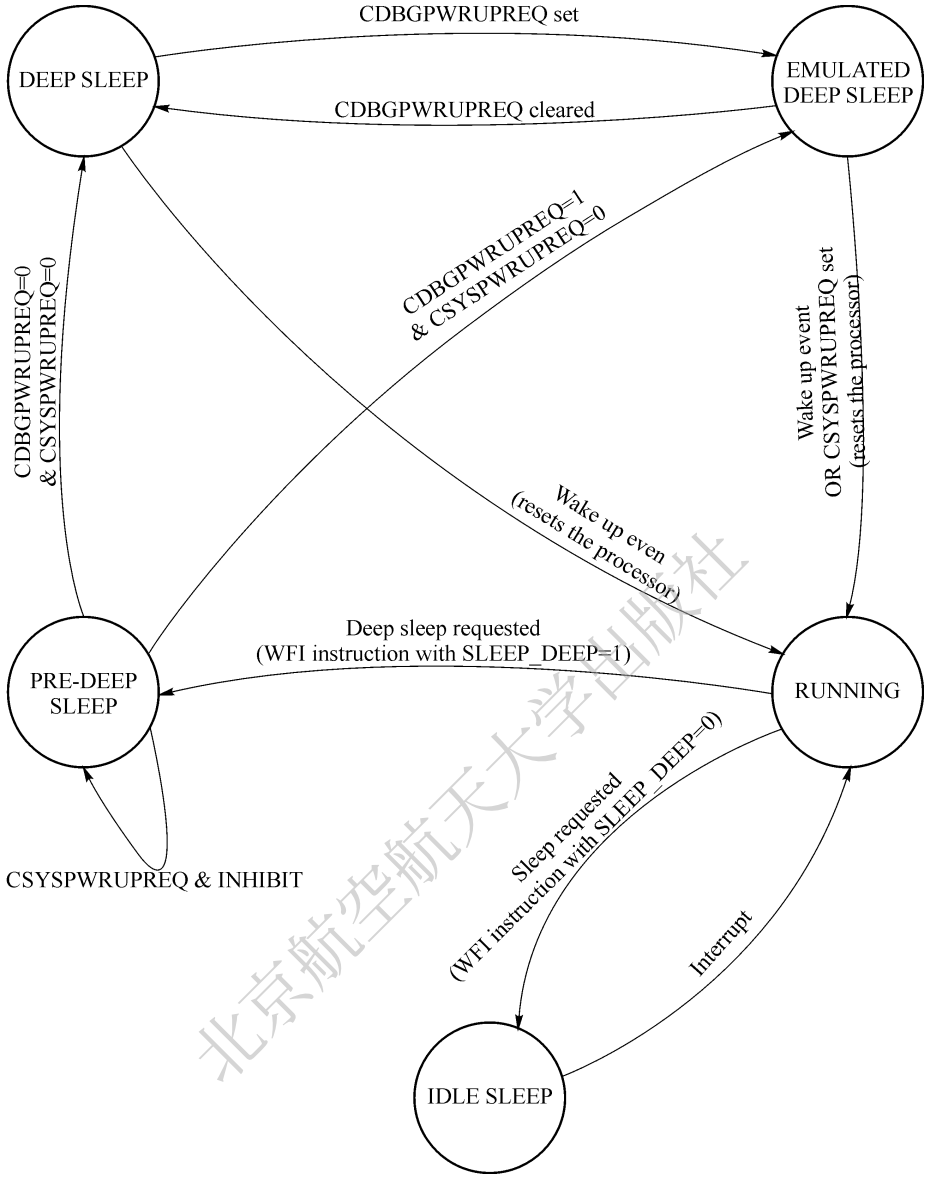

图 3.3　电源管理状态图

应用程序在整个运行过程中，一般会要求下面两种低功耗模式：

- 空闲睡眠模式(Idle Sleep)。通过执行一条 WFI 指令，清除 Cortex 系统控制寄存器(SCS_SCR)的 SLEEPDEEP 位，系统进入空闲睡眠状态，这种状态使得 CPU 进入暂停执行的空闲状态，直到中断发生，图 3.3 中的下部表示了这一过程。在 STM32W108 处于空闲睡眠状态时，核心逻辑保持供电。
- 深睡眠模式(Deep Sleep)。通过执行一条 WFI 指令，设置 Cortex 系统控制寄存器(SCS_SCR)的 SLEEPDEEP 位，系统进入深睡状态。在图 3.3 中，这个触发了其中主循环的状态转换，导致 STM32W108 核心逻辑电源关闭，只保留始终供电域(always-

on)的电源。当预定事件发生时,触发唤醒。

如果需要进入深睡眠模式,STM32W108 首先会进入一个预深睡眠(pre-deep sleep)状态,这种状态防止芯片的任何一个部分掉电或者复位,直到 SWJ 进入空闲状态(通过清除 CSYSPWRUPREQ 位),这种预深睡眠状态保证了调试操作不会被打断。

在深睡眠状态下,STM32W108 等待一个使它能回到运行状态的唤醒事件。通过完成一个复位周期给 ARM Cortex-M3 核心逻辑上电,并由 ST 软件恢复进入深睡眠时的堆栈和应用程序状态。

3.4.3 可选的深睡眠

在深睡眠状态(深睡眠 1)下,内部低频 RC 振荡器(OSCRC) 默认保持运行。为了节省功耗,在深睡眠状态下,OSCRC 可以被关闭,这种模式就是深睡眠 2。因为 OSCRC 被禁用,睡眠定时器和看门狗定时器不起作用,它们都不能唤醒芯片,除非低频 32.768 kHz 晶体振荡器被启用。非定时器的唤醒源仍然可以产生作用,一旦唤醒事件发生,OSCRC 重启并且被使能。

3.4.4 睡眠模式下使用调试器

仿真器/调试器使用 SWJ 和 STM32W108 进行连接、通信。当连接调试器后,SWJ 调试端口的 CDBGPWRUPREQ 位被设置,STM32W108 只能进入深睡眠 0(仿真深睡眠状态)。CDBGPWRUPREQ 位用来指示调试工具是否连接到芯片上,因此在系统调试组件中可以显示调试状态。因为深睡眠 0 不会引起一个上电周期或核心域复位,只能使用深睡眠 0 来维持系统调试组件的状态。SWJ 调试端口的 CSYSPWRUPREQ 位指示调试器需要访问 STM32W108 上的存储器。在 STM32W108 唤醒状态下,CSYSPWRUPREQ 是被设置的,STM32W108 不能进入深睡眠直到这个位被清除,这保证了 STM32W108 不会打断存储器的调试通信。

同时清除 CSYSPWRUPREQ 和 CDBGPWRUPREQ 位使得 STM32W108 进入一个真睡眠状态(深睡眠 1 或 2)。这两个信号也都可以作为唤醒源,因此当调试器连接到 STM32W108 并开始访问芯片,STM32W108 可以自动脱离深睡眠状态。当调试器在 STM32W108 处于深睡眠状态下发起一个访问,SWJ 自动拖延调试器一小段时间,直到 STM32W108 正常供电并准备就绪。

3.5 内部存储器

本节介绍 STM32W108 的片上存储器结构,图 3.4 是 STM32W108 的存储器映射图。

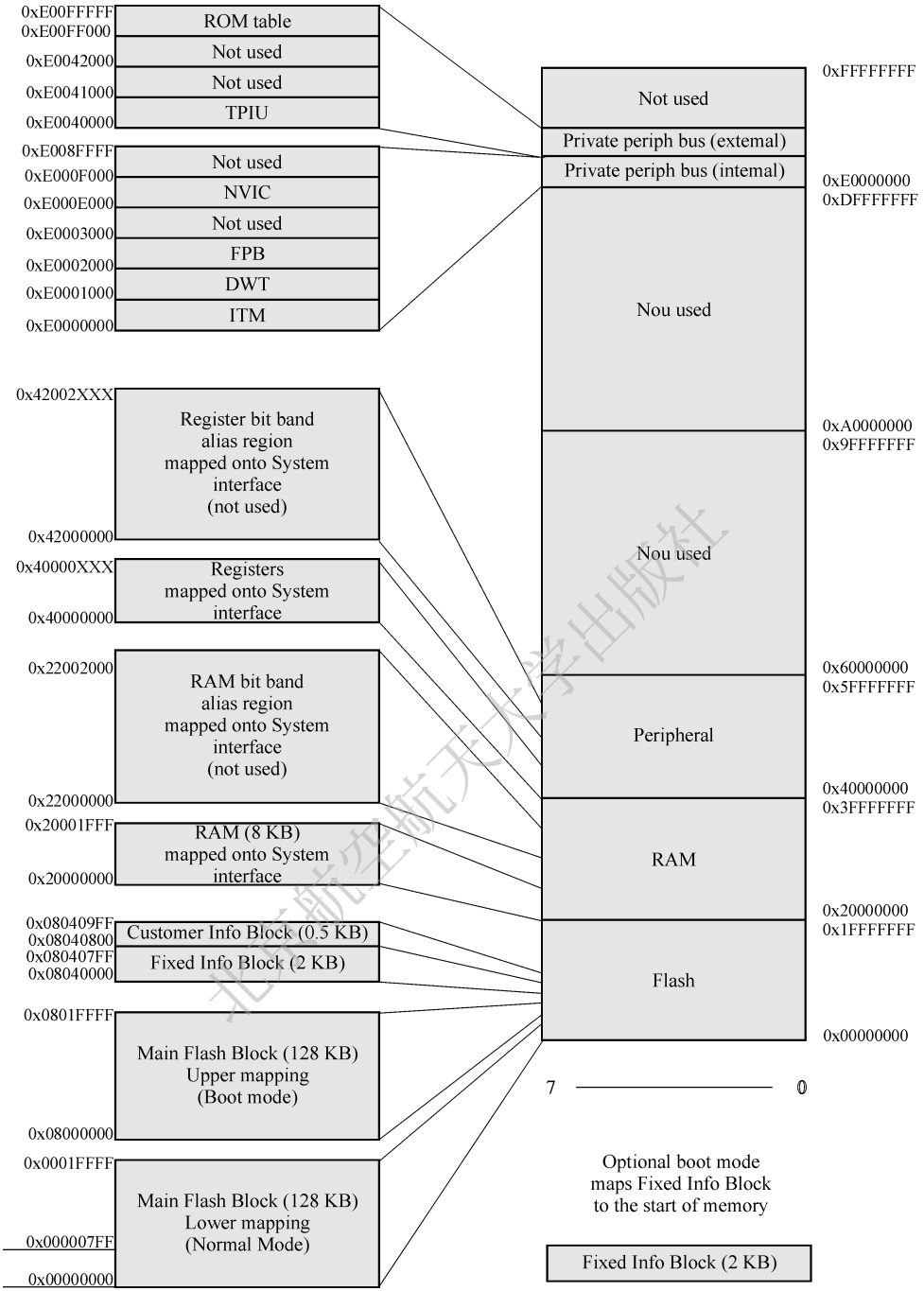

图 3.4　STM32W108 存储器映射

3.5.1　Flash 存储器

STM32W108 提供了总共 130.5 KB 的 Flash 存储器，分为三块：

- 主 Flash 块(MFB)。
- 固定信息块(FIB)。
- 用户信息块(CIB)。

MFB 被分成 128 个 1 KB 的页,CIB 是单个 512 字节页,FIB 是单个 2 KB 页。最小擦除单元是一个页,最小可写单元是一个对齐的 16 位半字。Flash 能保证 1000 次擦写周期,在室温下 Flash 单元能够保留数据大于 100 年。

Flash 编程可以通过串行线/JTAG 接口、或引导程序软件来实现。通过引导程序编程需要指定的软件来进行无线或串口下载。一个简化的、仅串口连接的引导程序也可以通过预编程到 FIB 中来实现。

3.5.2 随机访问存储器 SRAM

STM32W108 有 8 KB 的静态 RAM,起始地址是 0X20000000。尽管 ARM Cortex-M3 允许在这个地址范围进行位绑定访问,但标准 MPU 配置不允许使用位绑定这个特征。

RAM 在物理上连接到 AHB 系统总线,因此 ARM Cortex-M3 微处理器和调试器都可以访问 RAM。指令和数据以字节、半字或字为单位访问 RAM。标准 MPU 配置不允许从 RAM 开始执行,除非是为了特殊目的,例如编程主 Flash 块。对于这个总线,RAM 看似一个 32 位存储器,并且在大部分情况下,读或写访问是 0 等待的。在更高的 CPU 时钟频率下,RAM 要求两个等待周期。这是由硬件决定的,对用户应用程序来说是透明的、无需配置。

1. 直接存储器访问(DMA)

STM32W108 有几个外设配置了 DMA 控制器,允许与 RAM 的自动数据传输。DMA 可应用于无线(802.15.4 MAC)、通用 ADC 和 2 个串口控制器。在串口控制器中,DMA 是全双工的,对 RAM 的读写可以同时被请求。所以总共支持 6 个 DMA 通道。

STM32W108 集成了一个 DMA 仲裁器,通过一个固定的优先级模式确保外设公平地使用微处理器资源,这个固定的优先级与每个主设备要求的存储器带宽相适应。这个优先级模式是(从高到低):通用 ADC、串口控制器 2 接收、串口控制器 2 发送、无线 MAC、串口控制器 1 接收、串口控制器 1 发送。

2. RAM 存储保护

STM32W108 集成了两个存储保护机制。第一个存储器保护机制是按照 ARM Cortex-M3 存储保护单元(MPU)实现的,MPU 可用于保护任何存储器区域,MPU 的配置通常是由软件完成的。第二个存储器保护机制是通过一个细粒度的 RAM 保护模块来实现的,它允许 RAM 分段成 32 字节的块,任何一个块可以被标记为写保护,如果在用户模式下尝试写入一个保护的 RAM 块,会产生一个 AHB 系统总线错误信号。在系统模式下写任何 RAM 块都是允许的,在用户模式和系统模式下对 RAM 读是不受限制的。这种细粒度的 RAM 保护模式的主要目的,是通报堆栈有错误的对存储器系统区域的写入。使用一组寄存器来配置 RAM 保护单元,这些寄存器中的位与 RAM 地址有映射关系,每一位代表 RAM 中一个 32 字节的块,当某一位被置位时,一个 RAM 块就被写保护了。

细粒度 RAM 存储保护机制对 DMA 控制器也是有效的,如果一个寄存器位使能了存储

器保护，也包括对被保护存储器的 DMA 写，如果一个 DMA 写试图写入被保护的 RAM 块，就会产生一个管理中断。同时，这个错误的地址和外设标识将被捕获，用于后续的调试。注意：只有能写数据到 RAM 的外设，才能产生这个中断，例如 RF 接收到的数据包，或者串口接收到的字符。

3.5.3 存储保护单元

STM32W108 包含 ARM Cortex-M3 的存储保护单元——MPU。MPU 能够控制访问权限，并拥有多达 8 个地址区域的特性，每个地址区域被分划成 8 个相同的子区域。可参考 ARM Cortex-M3 技术参考手册(DDI O337A)获取有关 MPU 更详细的描述。

ST 软件(固化在芯片内)按照一个标准配置来配置 MPU，应用程序软件不能修改。这个配置是为有效检测非法指令或者数据访问而设计的。如果有一个非法访问，MPU 可捕获关于访问类型、被访问地址以及进攻软件位置等信息，这简化了软件的调试，并增加了外设的可靠性。STM32W108 的这个 MPU 配置，不允许访问 RAM 和寄存器的位绑定地址别名区域，如果尝试那样操作，会产生了一个总线错误。

3.6 硬件 AES 加速器

STM32W108 包含一个可以从 ARM Cortex-M3 访问的硬件 AES 加密引擎。基于 NIST 的 CCM、CCM* 、CBC-MAC 和 CTR 模式都由硬件实现，应用软件无需参与加密算法、模式的具体实现，可以大大降低 CPU 的工作负荷，这些模式在 IEEE 802.15.4—2003 规范中描述，除了 CCM* 是在 ZigBee 安全服务规范 1.0 中描述，具体可以参考相关资料。

3.7 无线射频模块

STM32W108 无线射频模块包含了一个模拟前后端和一个数字基带，如图 3.5 所示。

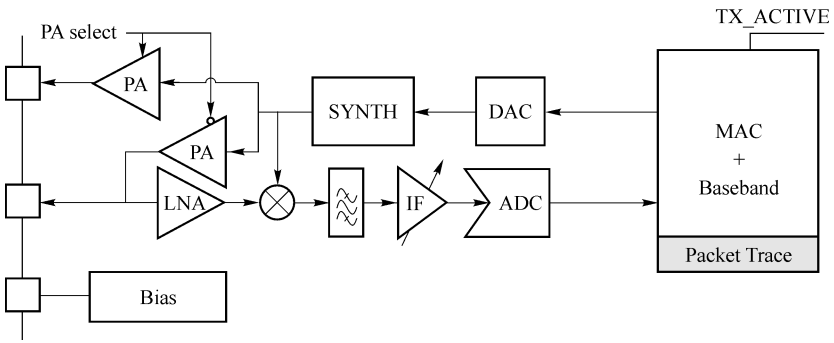

图 3.5 STM32W108 无线射频结构图

3.7.1 接收(Rx)通道

Rx 通道采用一个低 IF、超外差接收器,这个接收器使用复杂的混合、多相滤波来抑制像频。在模拟部分,天线输入的 RF 信号先被放大、混频合成一个 4 MHz 的中频(IF),混频器的输出经滤波、组合、放大,然后被一个 12 Mb/s 的 ADC 采样,这个数字化的信号在数字基带上被解调。Rx 通道内的滤波改善了 STM32W108 与环境内其他 2.4 GHz 收发器的共存性,如 IEEE 802.15.4、IEEE802.11g 和蓝牙等无线设备。数字基带也提供了 Rx 路径的增益控制,既可以有能力接收大范围的信号,也可以容忍较大的干扰。

1. Rx 基带

STM32W108 的 Rx 数字基带实现了一个相干解调器以获得最佳性能。基带在芯片级解调 O-QPSK 信号并同步 IEEE 802.15.4 定义的前导序列。自动增益控制(AGC)模块每 1/4 信标连续地调节模拟增益,直到前导序列被检测到,一旦检测到,对这个包的剩余接收增益将被固定。基带把解调数据变成 4 位的数字信号,缓冲后传送给硬件 MAC 模块来进行包组装和过滤。

另外,Rx 基带提供校正并控制 Rx 模拟部分,包括 LNA、Rx 基带过滤器和调制模块。ST 的 RF 驱动软件包含校正算法,这个算法可以减少硅片制造和温度变化带来的影响。

2. RSSI 和 CCA

STM32W108 每 8 个信号周期以及在一个接收包的末尾计算一次 RSSI。超出温度范围时,RSSI 的线性范围被指定为至少 40 dB。在室温下,线性范围大概为 60 dB(−90 dB～−30 dB 输入信号)。

STM32W108 的 Rx 基带支持 IEEE 802.15.4−2003 RSSI CCA 方式,如果 RSSI 超过了阈值,信道就会报告介质繁忙。

3.7.2 发送(Tx)通道

STM32W108 的 Tx 通道采用一个模拟前后端和数字基带来产生 O-QPSK 调制信号,这是一个面积小、能效高的两点调制架构,来调制由合成器产生的 RF 信号,调制过的 RF 信号被送到一个集成功放(PA),最后从 STM32W108 输出。

1. Tx 基带

在数字域的 STM32W108Tx 基带,把 4 位的信号扩展成 IEEE 802.15.4 定义的 32-chip 序列,它也给软件提供接口来校正 Tx 模块,以减少硅片制造、温度和电压变化的影响。

2. TX_ACTIVE 和 nTX_ACTIVE 信号

对于需要外部功放(PA)的应用,Tx 通道提供了 TX_ACTIVE 和 nTX_ACTIVE 二个信号,这两个信号是反相的,他们可用于外部 PA 的电源管理和 RF 开关逻辑。在发送模式下,Tx 基带驱动 TX_ACTIVE 信号为高;在接收模式下,TX_ACTIVE 信号为低。TX_ACTIVE 是 PC5 的替换功能;nTX_ACTIVE 是 PC6 的替换功能。详细可参考 GPIO 替换功能、信号分配章节的相关描述。