在 STM32WB 系列上开始使用 Zigbee®

引言

本应用笔记将指导设计者完成基于 STM32WB 系列微控制器构建特定 Zigbee®应用程序所需的所有步骤。阐明了如何与 STM32WB 系列微控制器连接，同时还整合了与 Zigbee®相关的最重要信息。

为了充分利用本文档中的信息并开发应用，用户须熟悉 STM32 微控制器。

1 概述

本文档适用于 STM32WB 系列基于双核 Arm®的微控制器。

注意： *Arm 是 Arm Limited（或其子公司）在美国和/或其他地区的注册商标。*

1.1 缩略语与定义

表 1. 缩略语与定义

缩略语	定义
API	应用编程接口
APS	应用支持子层
BDB	基本设备特性
HAL	硬件抽象层
IAS	入侵报警系统
IPCC	处理器间通信控制器 IP
MAC	介质访问控制
PAN	个人局域网
SED	休眠终端设备
ZCL	Zigbee 群集库
ZDO	Zigbee 设备对象

1.2 参考文档

- AN5289 使用 STM32WB 系列微控制器构建无线应用
- AN5492《STM32WB 系列中的持久性数据管理 Zigbee®和非易失性存储器》
- AN5491《基于 STM32WB 系列创建制造特定群集》
- AN5498《如何在 STM32WB 系列上使用 ZigBee 群集模板》
- AN5500《ZSDK API 在 STM32WB 系列上实现 Zigbee®》

2 Zigbee 通信协议

2.1 Zigbee 概述

Zigbee 是一种基于 IEEE 802.15.4 的 IOT 协议，用于创建无线个人局域网（WPAN）。它意图在低功耗和低带宽限制条件下，提供简单的网络层以及用于创建可互操作解决方案的标准应用参数文件。

该协议涉及：

- 家庭自动化
- 工业控制系统
- 建筑自动化
- 医疗数据收集和监测
- HVAC 控制
- 无线传感器网络

2.4 GHz 频段的数据吞吐量为每秒写入 250 Kbit，典型距离为 10-20 米。

2.2 Zigbee 网络

2.2.1 设备类型

在 Zigbee 中，有三种逻辑设备类型：

- 协调器（ZC）：这是要启动的第一个节点。协调器负责通过允许其他节点通过网络加入网络来形成网络。协调器负责启动网络并选择某些关键网络参数。建立网络后，协调器将具有路由角色。在集中式网络中，每个 Zigbee 网络必须有且只有一个协调器。
- 路由器（ZR）：路由器是具有路由功能的节点，也能够发送和接收数据。它还能允许其他节点加入网络。Zigbee 网状网络可以具有多个路由器。
- 终端设备（ZED）：该设备是只能发送和接收数据的节点。其本身并不具备路由功能。Zigbee 网状网络可以有多台终端设备。一些终端设备也可以是休眠终端设备，从而实现极低功耗。

2.2.2 网络类型

为了满足广泛应用的需求并确保最佳安全平衡，Zigbee 具有两种类型的网络：分布式和集中式：

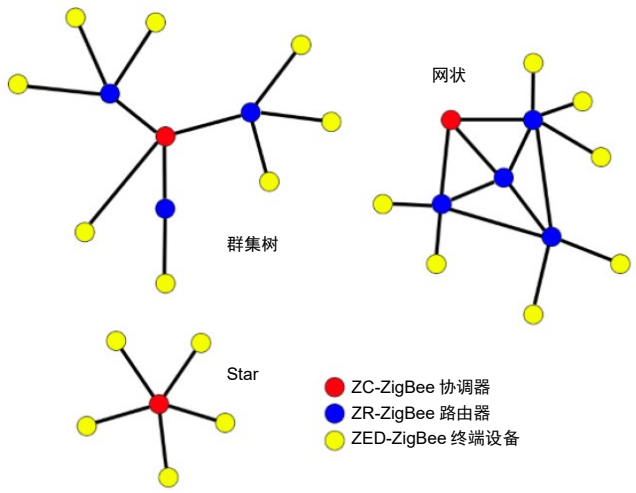
- 分布式网络不具备协调器。在该配置中，任何路由器都可以颁发网络安全密钥。随着越来越多的路由器和终端设备加入网络，已经存在于网络上的路由器可安全地发送网络密钥。网络上的所有设备都使用相同的网络密钥来加密消息。
- 在集中式网络中，有一个名为信任中心（TC）的实体，其通常是协调器。TC 形成了集中式网络，并允许路由器和终端设备加入网络，前提是它们具有适当的凭证。在集中式网络中，只有 TC 可以颁发加密密钥。TC 还在网络上的每台设备加入时为其建立唯一 TC 链路密钥，并根据请求为每对设备建立链路秘钥。

显而易见，集中式网络比分布式网络安全得多。STM32WB 固件包内提供的大多数 Zigbee 样例都使用集中式网络。

2.2.3 Zigbee 网络拓扑

在集中式网络中，Zigbee 支持 3 种类型的网络拓扑，如下图所示。

图 1. Zigbee 网络拓扑（集中式网络）



2.2.4 Touchlink 配网

Touchlink 是一种 Zigbee 功能，可使在物理上彼此靠近的设备即使不处在同一 Zigbee 网络中，也可进行通信。这基于 PAN 间通信机制，其中各设备可以在其本地区域中交换信息，而无需形成或加入相同的 Zigbee 网络。

Touchlink 进程可发现非常接近的两台设备，并将其连接到同一个 PAN 中。Touchlink 涉及两种不同的设备角色：

- 发起设备，即发起 Touchlink 进程的设备。发起设备必须发现可以使其加入到同一 PAN 中其他设备，即目标
- 目标是指正被发现并加入到启动设备 PAN 的设备。

没有参与任何 touchlink 进程的设备（由于离开 Zigbee 网络或形成了新网络）被称为出厂新设备。该设备根据其启动设备/目标角色充当新设备。

对于非出厂新设备，所有 Zigbee 协议栈参数（基本上是网络参数）保持不变。这涉及主要 Touchlink 配网进程步骤，如下：

- 设备发现，包括设备识别
- Zigbee 网络形成和加入用例。

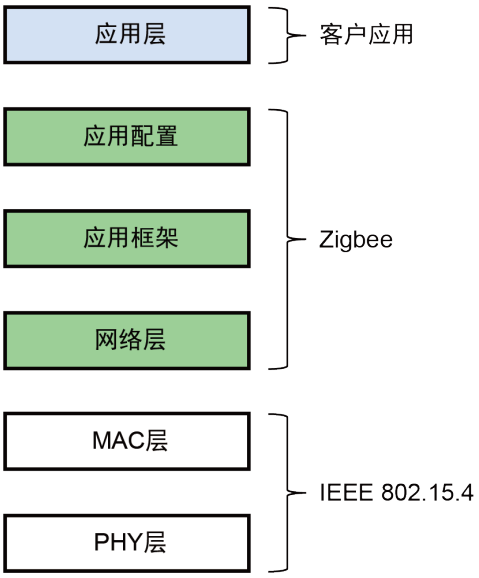
2.3 Zigbee 架构

2.3.1 一般架构

如上文所述，Zigbee 基于 IEEE 802.15.4 标准。Zigbee 为基于数据包的无线电协议提供路由和多跳功能。构建时基于 802.15.4 指定的两层：物理（PHY）层和 MAC 层。

下图描述了 Zigbee 协议栈的主要组件及其与 IEEE 802.15.4 和通用应用层的衔接。

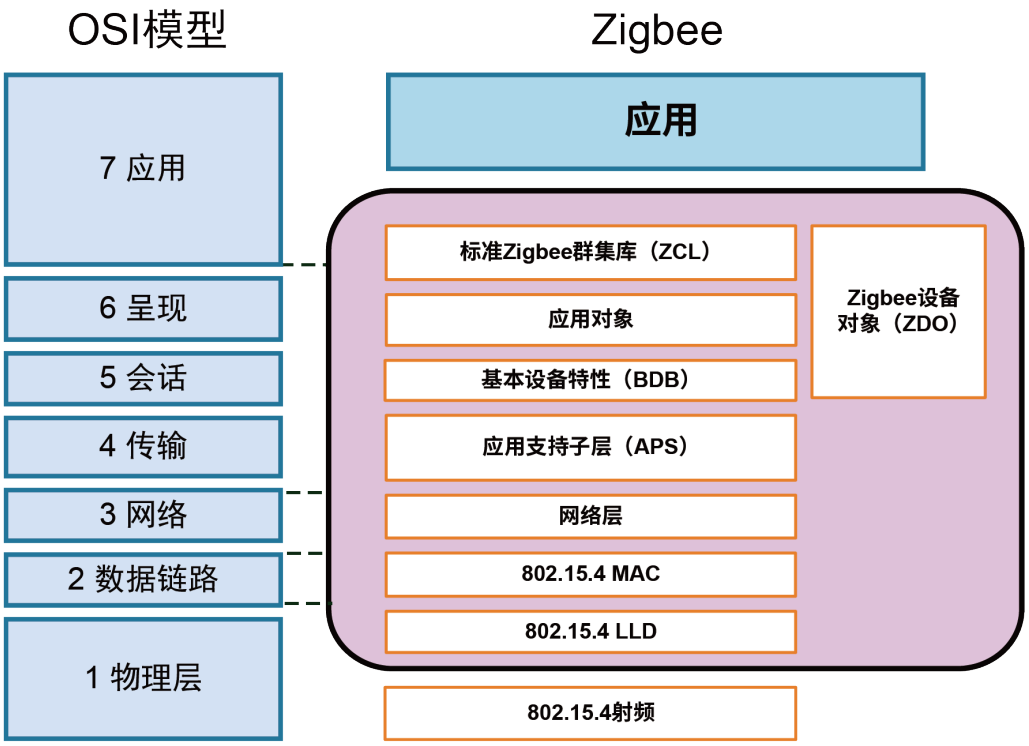
图 2. Zigbee 协议栈概述



2.3.2 Zigbee 协议栈层

ZigBee 规范定义的协议栈层基于 OSI 7 层模型。对于 Zigbee，其涉及网络和应用框架层。Zigbee 协议栈分为多个组件，如下图所示。

图 3. Zigbee 协议栈说明



网络（NWK）层

网络层需要提供确保 IEEE 802.15.4 MAC 子层正确运行的功能，并为应用层提供合适的服务接口。除此之外，这是启动、加入、离开和发现网络的层。

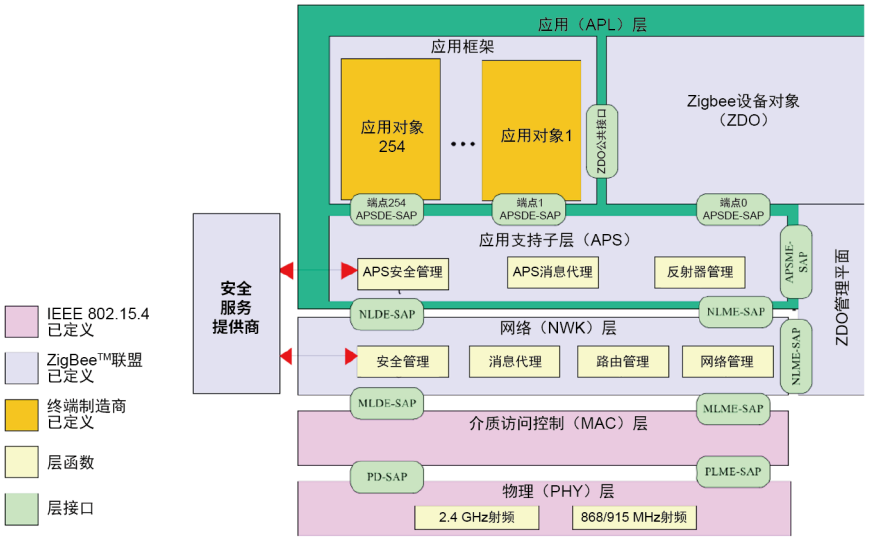
表 2. 网络层功能

Zigbee 网络层函数	协调器	路由器	终端设备
建立 Zigbee 网络	x	-	-
准许其他设备加入或离开网络	x	x	-
分配 16 位网络地址	x	x	-
发现并记录路径以实现高效消息递送	x	x	-
发现并记录单跳邻居列表	x	x	-
路由网络数据包	x	x	-
接收或发送网络数据包	x	x	x
加入或离开网络	x	x	x
进入睡眠模式	-	-	x

应用（APL）层

APL 层由多个子层组成。APL 层的组件如下所示：

图 4. 应用层子层



- **应用支持子层（APS）**

  APS 代表应用支持子层。它通过 ZDO 和制造商定义的应用对象共同使用的一组常规服务，在网络层（NWK）和应用层之间提供接口。

  APS 负责：

  – 绑定管理
  – 绑定设备之间的消息转发
  – 群组地址定义和管理
  – 从 64 位扩展地址到 16 位 NWK 的地址映射
  – 地址（专用表）
  – 数据包的分段和重组
  – 可靠的数据传输

  Zigbee 中的绑定允许连接一个节点上的端点，或“绑定”到另一个节点上的一个或多个端点。

  绑定表将源地址和源端点映射到一个或多个目标地址和端点。该表可用并保存在网络中的所有设备上。

- **Zigbee 设备对象（ZDO）**

  ZDO 组件处理设备管理和通信功能。它包括：

  – 初始化 APS 子层和 NWK 层
  – 设备发现
  – 服务发现
  – 网络管理，包括定义设备的运行模式（ZC、ZR 或 ZED）。
  – 安全管理
  – 发起和/或响应远程绑定请求

- **基本设备特性（BDB）**

  这是一种标准软件组件，用于处理配网、网络安全和持久数据管理等基本操作。该设备不需要端点。

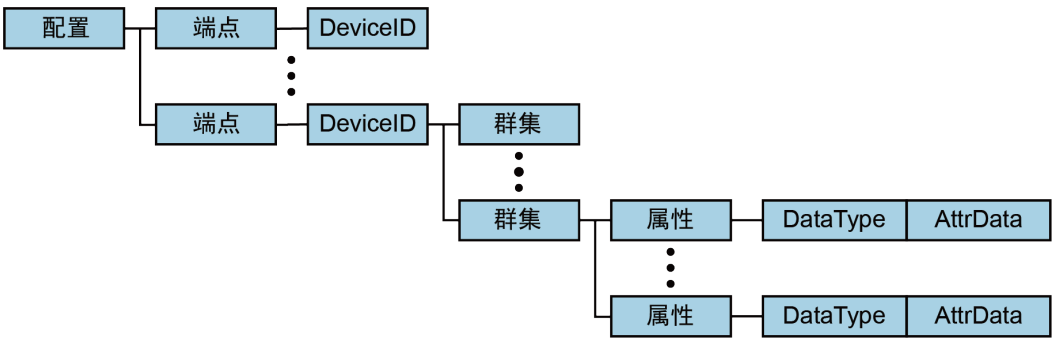
- **应用框架**

  Zigbee 中的应用框架非常丰富，定义了应用对象托管在设备上的环境。

  Zigbee 设备之间的数据交换在客户端服务器模型中执行。这依赖于应用配置文件、群集、属性模型。

  应用配置文件是设备说明的集合，这些设备说明共同构成协作应用。配置文件定义了 Zigbee 物理设备的应用函数的数据交换形式。配置文件由一个或多个端点组成，每个端点具有一个或多个相关联的群集。

图 5. Zigbee 应用配置文件组织



群集是一组命令和属性，用于定义设备可以执行的操作。群集由 ZCL（Zigbee 群集库）管理。

可用于 Zigbee 应用的端点编号介于 1 和 240 之间。

- **Zigbee 群集库（ZCL）**

  ZCL 是管理群集的库。群集可以被视为特定于专有应用（门锁、开关等）种类的一组命令和属性。

  ZCL 由 Zigbee 联盟定义，以便加快公共配置文件的开发和标准化。借助 ZCL，制造商能够快速构建具有一致性和兼容性的 Zigbee 产品。

  群集是命令和属性的相关集合，一起定义了特定功能的接口。命令是群集可以执行的操作。属性是群集中的数据或状态。

2.4 Zigbee 配置文件

2.4.1 Zigbee 应用配置文件

配置文件是不同设备上的应用之间的消息处理协议。它对逻辑组件及其接口进行了描述。

配置文件的目的是实现不同制造商之间的互操作性。

有三种类型的配置文件：

- 公共（标准），由 Zigbee 联盟管理
- 私有，由 Zigbee 供应商定义为限定使用
- 已发布。这涉及以往私有的配置文件，这些配置文件在所有者发布时被发布

配置文件决定所有配置文件都必须具有唯一配置文件标识符。

配置文件使用定义的语言进行数据交换，并且使用一组定义的处理操作。实际上，应用配置文件将指定以下内容：

- 应用区域所需的设备集合
- 每台设备的功能说明
- 用于实现功能的群集集合
- 哪些设备需要哪些群集

可以在设备之间传输的每条信息被称为属性。

属性分组到群集中。所有群集和属性均被给予唯一标识符。有输入群集标识符和输出群集标识符。它链接到客户端/服务器群集架构。

2.4.2 Zigbee 设备配置文件

Zigbee 设备配置文件是由 ZDO 直接执行的设备说明和群集的集合。它适用于所有 Zigbee 设备。

Zigbee 设备配置文件是一种模板，展示了如何编写应用配置文件。它在 Zigbee 应用级规范中定义。

2.5 Zigbee 寻址

在加入 Zigbee 网络之前，采用 IEEE 802.15.4 兼容射频的设备具有全球唯一的 64 位地址。对于 Zigbee，该 MAC 地址被称为扩展地址。

当设备加入 Zigbee 网络时，会收到一个称为 NWK 地址的 16 位地址。这些地址中的任何一个（64 位扩展地址或 NWK 地址）都可以在 PAN 中用于和设备通信。

2.5.1 Zigbee 消息传递

设备一旦加入 Zigbee 网络，就可以向同一网络上的其他设备发送命令。有两种方式对 Zigbee 网络中的设备进行寻址：直接寻址和间接寻址。

直接寻址

发送方必须提供有关目标设备的三条信息。

这属于单播消息，由以下部分组成：

- 修改间接寻址部分
- 设备地址（NWK 或 IEEE 扩展地址）
- 端点编号
- 群集 ID

间接寻址

间接寻址是一种可简化进程的本地功能。因此，即使使用间接寻址，也会发送单播消息。

它要求上述三种类型的信息在绑定表中可用。

发送设备只需要知道自己的地址、端点编号和群集 ID。绑定表条目提供目标信息（设备地址和端点）。

2.5.2 广播寻址

在 Zigbee 网络中，有两个广播级别：

- MAC 层目标地址为 0xFFFF 的广播。任何处于唤醒状态的收发器都将收到数据包。

注意： *数据包由每台设备重新传输三次。该广播类型仅应在必要时使用。*

- 将消息广播到指定设备上的所有端点。对于此类广播，它使用端点编号 0xFF。

2.5.3 群组寻址

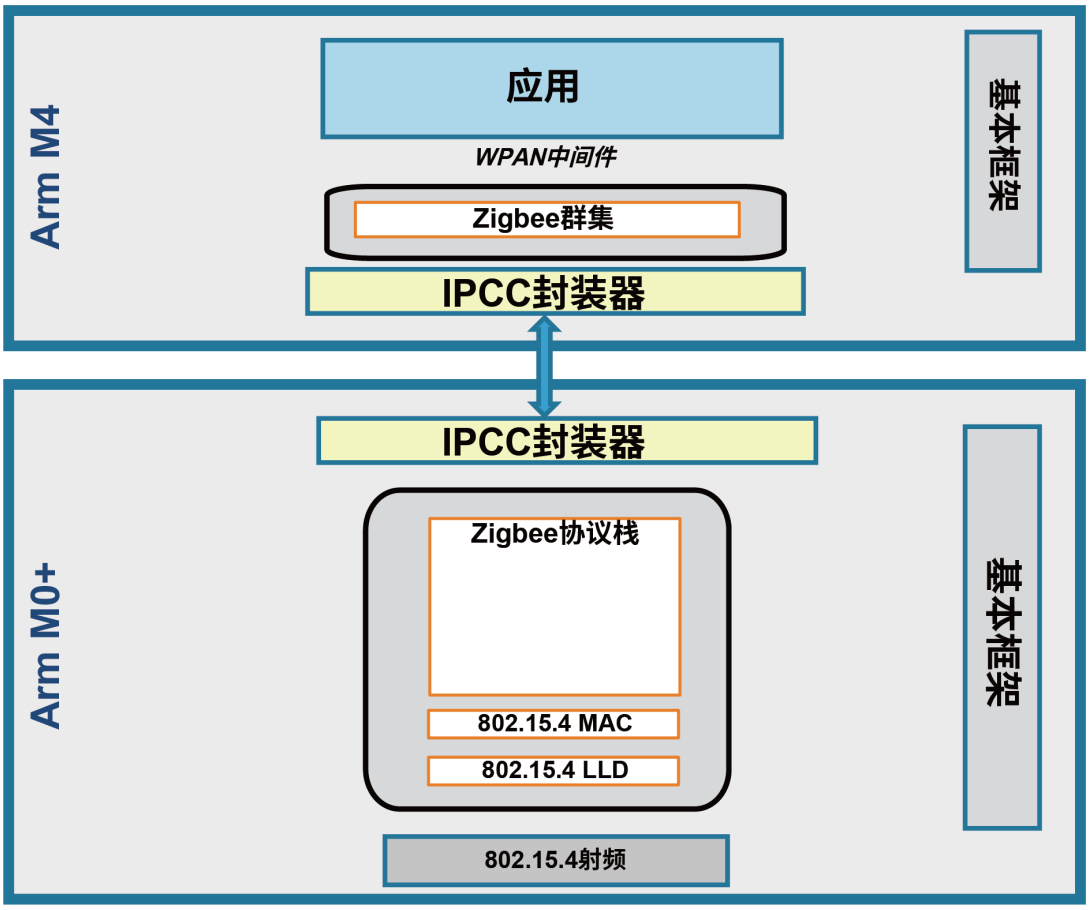
应用可以将多台设备和这些设备上的特定端点分配给单个群组地址。群组分配基于群集 ID、配置文件 ID 和源端点。

3 基于 STM32WB 的 Zigbee

3.1 架构概述

下图概述了总体架构。特别展示了 M4 与 M0 之间的分割。在 M0 上运行的所有代码均以二进制库的形式。客户只能访问 M4 内核，并看到在 M0 上运行的固件（相当于一个黑盒子）。该框架隐藏了 M4 和 M0 之间的所有相互通信。专用 IPCC 通道分配给 Zigbee。

图 6. 基于 STM32WB 的 Zigbee 架构概述

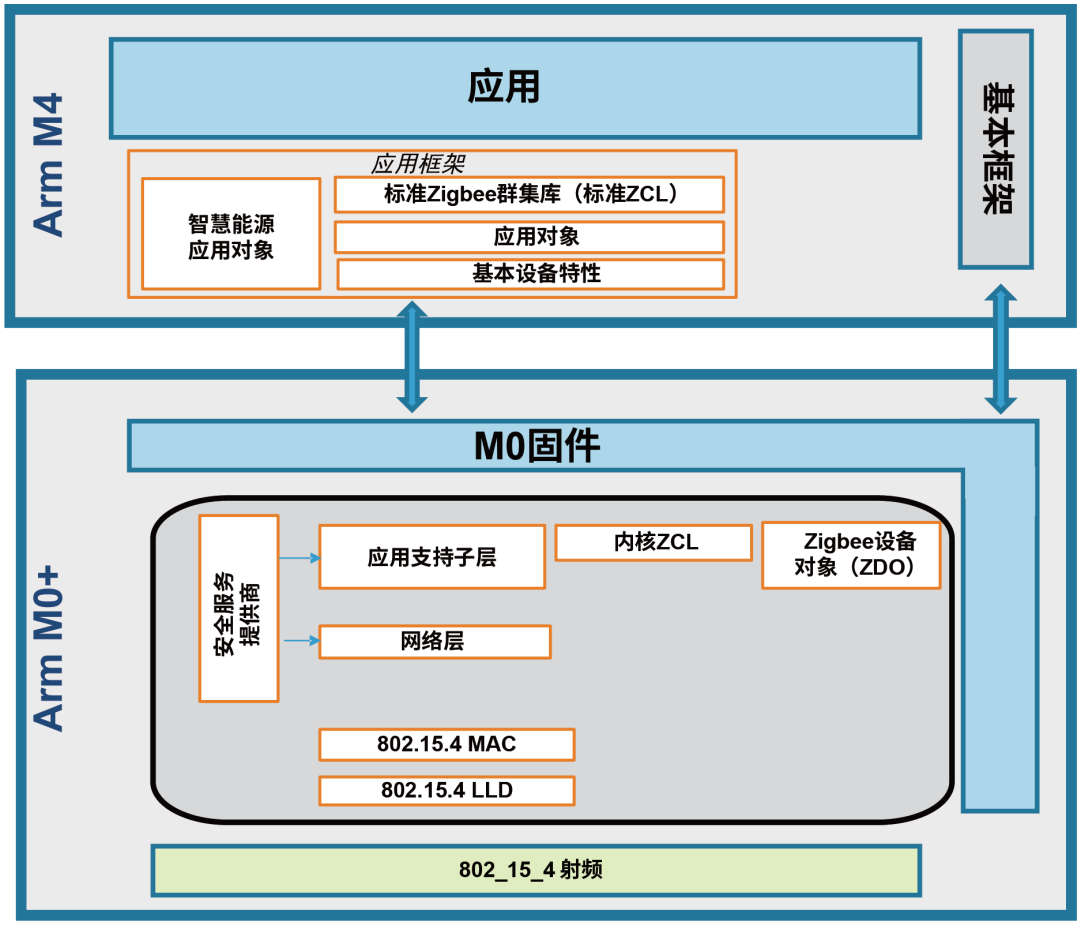


Zigbee 协议栈在 802.15.4 MAC 层上运行，该层本身会使用负责控制无线电的 802.15.4 底层驱动提供的服务。

3.2 STM32WB 上的 Zigbee 协议栈层

图 7. Zigbee 层和模块更详细地展示了不同的 Zigbee 层以及如何在 STM32WB MCU 上实现这些层。

图 7. Zigbee 层和模块



3.3 支持的 Zigbee 固件

STM32WB55 设备支持两种类型的协议栈。这两种协议栈均已获得 Zigbee PRO 2017（第 22 版）认证。

表 3. 支持的固件（独立 Zigbee）

支持的无线协议栈	相关的固件
Zigbee FFD（全功能设备）	stm32wb5x_ZigBee_FFD_Full_fw.bin
Zigbee RFD（精简功能设备）	stm32wb5x_ZigBee_RFD_fw.bin

- FFD 可以适配为网络中的任何角色。它可以是路由器、协调器或终端设备。
- RFD 仅支持终端设备角色。RFD 比 FFD 的占用空间小。当构建作为“休眠终端设备”的应用，以实现最佳低功耗时，必须使用 ZigbeeRFD 协议栈构建此应用。

意法半导体在单个二进制固件中同时支持 BLE 和 Zigbee 协议。

表 4. 支持的固件（Zigbee 并发模式）

支持的无线协议栈	相关的固件
BLE 和 Zigbee（静态模式）	stm32wb5x_BLE_ZigBee_FFD_static_fw.bin

该二进制用于静态并发模式应用。此类应用的例程如下：

Projects\P-NUCLEO-WB55.Nucleo\Applications\BLE_ZigBee 目录。

静态模式

在静态模式下，可以从 BLE 切换到 Zigbee，反之亦然。当 BLE 协议运行时，Zigbee 协议栈则不运行。当 BLE 停止时，系统会切回到 Zigbee。这种情况下，将完全重新初始化 Zigbee 协议栈。

注意： *在 STM32WB 上运行任何 Zigbee 应用之前，用户必须确保在 M0 上下载了正确的固件。如果未下载正确固件，则使用 STM32CubeProgrammer 加载适当的二进制。*

所有可用的 Zigbee 二进制位于：

/Projects/STM32WB_Copro_Wireless_Binaries/STM32WB5x.

请参见

/Projects/STM32WB_Copro_Wireless_Binaries/STM32WB5x/Release_Notes.html

有关如何更改无线协处理器二进制文件的详细过程。

3.4 支持的 Zigbee 群集

STM32WB 系列上可用的 Zigbee 生态系统支持 Zigbee 3.0。Zigbee 3.0 群集符合 ZCL 7 要求。它包括 BDB（基本设备特性），Zigbee 绿色能源和几种特定的 ZCL 群集，如下所示：

表 5. Zigbee 群集列表生态系统

Nb	群集 ID	群集名称
1	0x0000	基本
2	0x0001	功率配置
3	0x0003	识别
4	0x0004	组
5	0x0005	场景
6	0x0006	On/Off
7	0x0008	级别控制
8	0x000a	时间
9	0x0019	OTA 升级
10	0x0020	轮询控制
11	0x0021	节能代理
12	0x0102	窗布
13	0x0202	风扇控制
14	0x0204	恒温器用户接口群集

Nb	群集 ID	群集名称
15	0x0300	颜色控制
16	0x0301	镇流器配置
17	0x0400	亮度测量
18	0x0402	温度测量
19	0x0406	占用感测
20	0x0502	IAS 报警设备（WD）
21	0x0b05	诊断
22	0x1000	Touchlink
23	0x0002	设备温度配置
24	0x0007	开/关开关配置
25	0x0009	报警（闹铃）
26	0x000b	RSSI 定位
27	0x0015	配网
28	0x001a	电源参数文件群集
29	0x0024	最近网关群集
30	0x0101	门锁
31	0x0200	泵配置和控制
32	0x0201	恒温器
33	0x0203	除湿控制
34	0x0401	亮度水平感测
35	0x0403	压力测量
36	0x0405	相对湿度测量
37	0x0500	IAS 区
38	0x0501	IAS 辅助控制设备（ACE）
39	0x0700	价格
40	0x0701	需求响应和负载控制
41	0x0702	计量
42	0x0703	消息
43	0x0704	智能能源隧道（复杂计量）
44	0x0800	密钥建立
45	0x0904	Zigbee 语音
46	0x0b01	仪表识别
47	0x0b04	电气测量

- 通过 STM32_WPAN 中间件可以使用所有这些 47 个群集。此中间件对于 BLE 和 Thread 通用。对于特定需求，客户可以创建其自己的“专有”群集。更多详细信息，请参见[3]。
- 可以在以下目录下找到与这些群集相关的 API：\Middlewares\ST\STM32_WPAN\ZigBee\stack\include
- 默认情况下，所有群集均以单个库的形式提供。尽管如此，也可根据请求访问源代码。

4 STM32WB Zigbee 应用设计

4.1 Zigbee 应用框架

整个 Zigbee 应用框架基于客户端服务器模型。STM32WB 固件包内提供的每一个 Zigbee 应用，都包含两个独立的工程：一个工程处理服务器部分，另一个工程处理客户端部分。要运行这些应用，需要将一个板配置为客户端/协调器模式，并将所有其他板配置为服务器/路由器模式。

注意： *可以独立于其支持的角色（ZC/ZR/ZED），在任何 Zigbee 上映射客户端和/或服务器。此外，单台设备可以同时为客户端和服务器。*

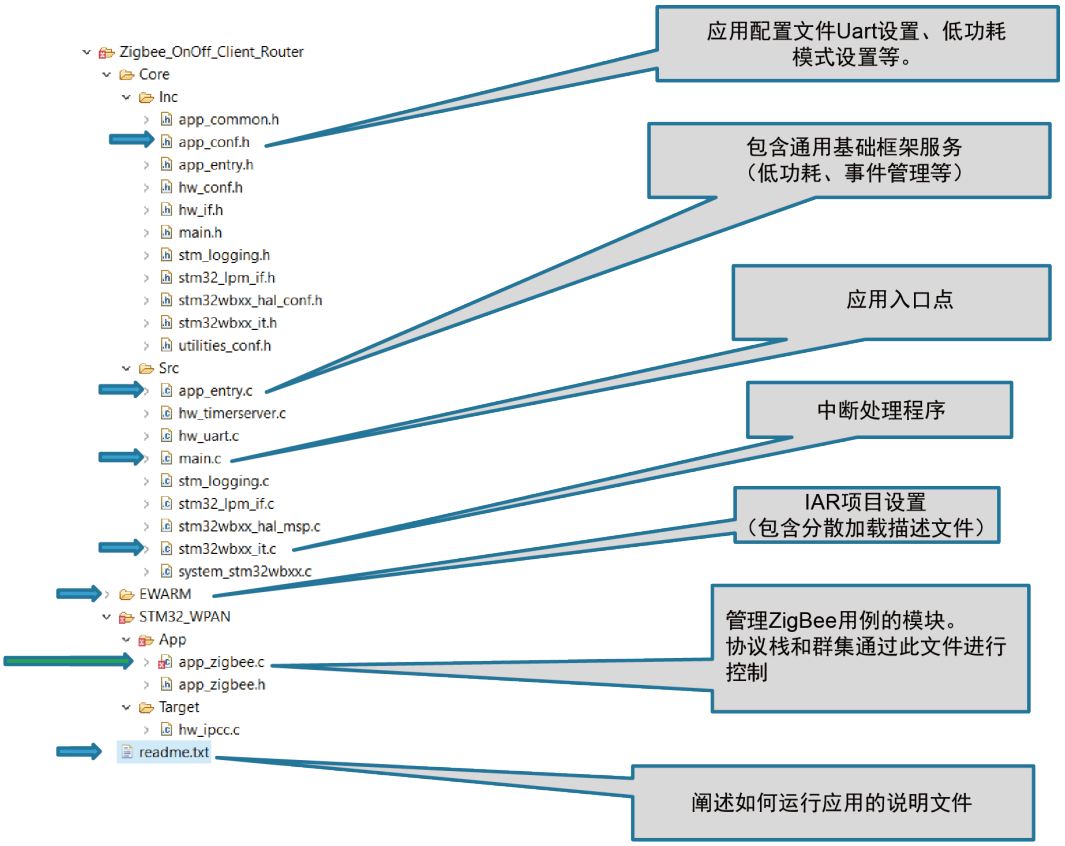
4.1.1 应用框架

所有工程均采用相同的框架构建。在所有工程中，Zigbee 用例均在 app_ZigBee.c 文件中设置、定义和实现。

在以下设置下：Projects\Board_X\Applications\ZigBee\Zigbbe_Y_app\STM32_WPAN\App。

应用工程中的所有其他文件主要用于全局基本框架管理（中断管理、IPCC 封装器、系统启动和配置等）

图 8. Zigbee 开关群集应用



4.1.2 Zigbee 应用架构

本节说明了 Zigbee 用例模块文件 app_zigbee.h 和 app_zigbee.c 中定义的一般 STM32WB Zigbee 应用架构。

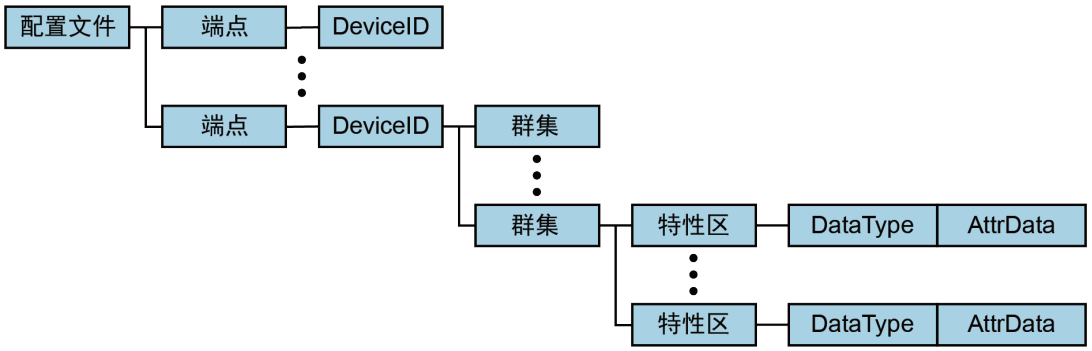
- **应用初始化**

  这是所有 STM32WB 应用常见的步骤，包括 Thread 和 BLE 应用。它包括协议栈层初始化（在这种情况下为 Zigbee 协议栈）和 Zigbee 端点配置步骤。

- **端点管理**

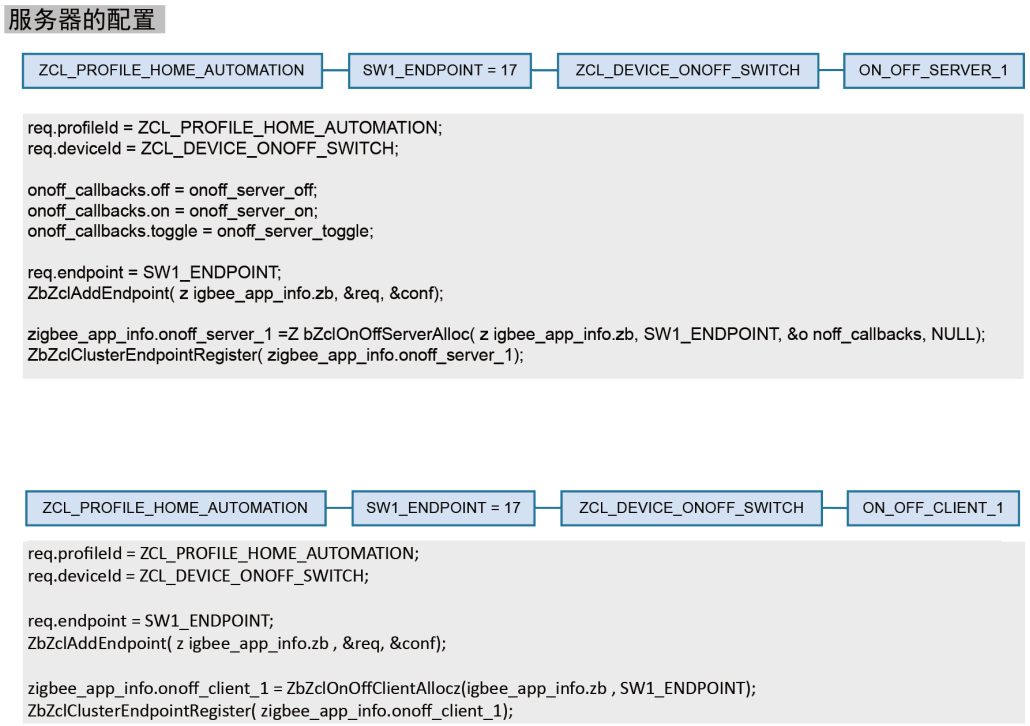
  对于任何 Zigbee 应用，配置文件由一个或多个端点组成，每个端点都有一个或多个群集和垂直的属性结构。所有这些实体之间的通用链路如下：

图 9. Zigbee 端点/群集关系



例如，在“开关”应用中，端点配置管理如下：

图 10. Zigbee 开关应用端点配置



Zigbee 网络管理

该步骤包括形成或加入 Zigbee 网络。它们由特定任务 TASK_ZIGBEE_NETWORK_FORM 进行管理。该任务主要包括 Zigbee 网络配置和相关启动流程。在专门章节提供了更详细信息。

当流程成功时，蓝色 LED 亮起。当由于任何原因加入流程失败时，将重新计划加入任务。

总之，如果应用有需要，就可以使用群组管理对 APS 层进行寻址。组播作为 STM32WB Zigbee 框架，允许访问 APS 层。

群组管理

群组是网络内节点的集合。某些 APS 基元允许较高层请求为特定端点添加特定组的群组成员资格。

例如，对于“开关”应用样例，多个“开关”客户端可以与唯一的“开关”服务器交互，该服务器充当使用群组的协调器。

用户特定代码

在 TASK_ZIGBEE_NETWORK_FORM 任务结束时，如果需要，将启动通用任务 TASK_ZIGBEE_APP_START。在这一点上，从 Zigbee 的角度来看，该应用发挥了作用。

在此，用户可以执行其应用的后续步骤。这包括本地或远程 Zigbee 命令以及相关联的回调。

注意： *在实现任何命令的回调时，用户应等待返回到该回调，然后才请求另一个命令，以避免 IPCC 死锁。实际上，许多 M4 应用命令正通过 IPCC 来控制 M0。出于这一原因，用户应在 Zigbee 命令之后使用可用的调度程序事件 API。M4 CPU 应等待在关联的 Zigbee 命令回调结束时引发的事件。例如，以下是使用 Zigbee 远程写入请求命令的调度程序事件的示例。*

图 11. 使用调度器事件的 Zigbee 命令

```
static void APP_ZIGBEE_RemoteWrite_cb(…){
  …
  /* Unlock the waiting event */
  UTIL_SEQ_SetEvt(EVENT_ZIGBEE_CONTINUE_INIT);
  …
}

…
status = ZbZclWriteReq(zigbee_app_info.commissioning_client, &RemoteWriteReq, APP_ZIGBEE_RemoteWrite_cb, NULL);
UTIL_SEQ_WaitEvt(EVENT_ZIGBEE_CONTINUE_INIT);
…
```

4.1.3 Zigbee 网络启动流程

Zigbee 网络管理中使用的 Zigbee 网络启动流程基于等待事件。实际上，该应用一直等待网络启动结果。

- 对于集中式 Zigbee 网络上的 Zigbee 协调器，它会一直等到网络形成（PAN ID 选择）。
- 对于集中式网络上的 Zigbee 路由器/终端设备，它会一直等到关联结果。

有 3 种主要的启动流程类型。这涉及所有类型的网络：

- 集中式
- 分布式网络
- Touchlink 配网
  - 通常的 Zigbee 启动如 Zigbee_OnOff_Server_Coord 应用例程所示。
  - 具有持久性的 Zigbee 启动，其中恢复了先前在设备断电之前存储的持久性数据。这包括 Zigbee 协议栈和群集参数。特定的 Zigbee 持久数据文档（参考 [1]）给出了详细的说明。请参阅 Zigbee_OnOff_Coord_NVM 和 Zigbee_OnOff_Router_NVM 应用例程。
  - CBKE（基于证书的密钥建立）Zigbee 启动。该启动包括用于加入过程的完整证书交换/验证。

Zigbee 设备网络加入超时

设备已设置或加入 Zigbee 网络后，允许其他设备加入该设备（例如用于 Zigbee 集中式网络的协调器或路由器）的时间是固定的。

因此，在该延迟之后，不准许加入设备的网络（Zigbee 关联许可值）。在该时间之后重置设备时亦是如此。

当这个许可超时时，然后为了一个设备加入 Zigbee 网络，Zigbee 协调器可以让母设备（如 Zigbee 路由器）准许加入。这是通过向母设备发送 ZDO 消息来完成的。

对于协调器，它可以向自身发送 ZDO 消息，使得准许以额外的时间加入。

注意： *Zigbee 持久性数据功能也可用于保持协议栈参数，还能够在重置/关闭后重新连接到网络（或设置）。*

4.1.4 跟踪

来自协议栈本身（在 M0 内核上运行）和来自应用本身（在 M4 端运行）的跟踪均由 M4 管理，并通过 UART（在编译时使用 app_conf.h 文件配置的）进行路由。

要获取跟踪，需要将板连接到超级终端（通过 STLink 虚拟通信端口）。UART 必须按如下方式配置：

- 波特率：115200 波特
- Word Length = 8 Bits
- 停止位=1 位
- 校验位 = 无
- 流控制 = 无

5 STM32WB Zigbee 应用

几个 Zigbee 应用在 STM32WB 固件包中提供。P-NUCLEO-WB55.Nucleo 板和 P-NUCLEO-WB55.USBDongle 上均提供这些应用。

完整的应用列表可在 STM32CubeProjectsList.html 文件（在\Projects 目录下）中找到。

这些应用的主要目的是提供简单的样例，突出显示特定群集的使用。

在 STM32WB 上开始使用 Zigbee 的最简单方法是使用 ZigBee_OnOff_Server_Coord 和 ZigBee_OnOff_Client_Router 应用。

5.1 Zigbee 一般应用

表 6. 可用的 Zigbee 应用

项目名称	说明
Zigbee_APS_Coord 和 Zigbee_APS_Router	该应用旨在展示： • 如何创建 Zigbee 集中式网络 • 如何使用 APSDE 接口直接与 Zigbee 协议栈连接 • 如何在网络上的设备之间发送和接收原始 APS 消息
Zigbee_Commissioning_Client_Coord Zigbee_Commissioning_Server_Router	该应用旨在展示如何创建 Zigbee 集中式网络，以及如何使用配网群集操作配网进程
Zigbee_DevTemp_Server_Coord Zigbee_DevTemp_Client_Router	如何在集中式 Zigbee 网络上使用设备温度群集。
Zigbee_Diagnostic_Server_Coord Zigbee_Diagnostic_Client_Router	如何在集中式 Zigbee 网络上使用诊断。
Zigbee_DoorLock_Server_Coord Zigbee_DoorLock_Client_Router	如何在集中式 Zigbee 网络上使用门锁群集。
Zigbee_IAS_WD_Server_Coord Zigbee_IAS_WD_Client_Router	如何在集中式 Zigbee 网络上使用 IAS WD 群集。
Zigbee_MeterId_Server_Coord Zigbee_MeterId_Client_Router	如何在集中式 Zigbee 网络上使用仪表识别群集。
Zigbee_OnOff_Client_DistribZigbee_OnOff_Server_Distrib	如何在分布式 Zigbee 网络上使用开关群集。
Zigbee_OnOff_Server_Coord Zigbee_OnOff_Client_Router	如何在集中式 Zigbee 网络上使用开关群集。
Zigbee_OnOff_Server_Coord Zigbee_OnOff_Client_SED	如何在集中式 Zigbee 网络上使用开关群集，并将休眠终端设备（SED）作为客户端。 SED 客户端被配置为支持低功耗模式 STOP2，IDLE 时功耗为 3µA。
Zigbee_OnOff_Coord_NVM Zigbee_OnOff_Router_NVM	采用集中式网络使用持久数据说明 Zigbee 开关群集应用。
Zigbee_OTA_Client_Router Zigbee_OTA_Server_Coord Zigbee_OnOff_Client_Router_Ota	Zigbee OTA 群集应用的说明
Zigbee_PollControl_Client_Coord Zigbee_PollControl_Server_SED	如何在集中式 Zigbee 网络上使用轮询控制群集。 轮询控制群集用于在休眠终端设备（SED）上远程操作。

项目名称	说明
Zigbee_PowerProfile_Client_Coord Zigbee_PowerProfile_Server_Router	如何在集中式 Zigbee 网络上使用能源配置文件群集。 该演示展示了如何通过模拟白色食品通用特性，在设备/家庭网关配置中使用能源配置文件群集。
Zigbee_PressMeas_Server_Coord Zigbee_PressMeas_Client_Router	如何在集中式 Zigbee 网络上使用压力测量群集
Zigbee_SE_Msg_Client_Coord Zigbee_SE_Msg_Server_Router	如何在集中式 Zigbee 网络上使用 SE 消息传递群集。

5.2 Zigbee 配网

该应用展示了 Zigbee 配网服务器和配网客户端之间的配网进程。

它展示了设备如何通过调试过程将其 Zigbee 参数（通道、PAN ID）分配给另一个设备。调试基于 PAN 间通信机制，其中设备可以在其本地区域交换信息，而无需形成或加入相同的 Zigbee 网络。

一个设备作为调试设备，另一个作为连接设备。

在该应用中，调试设备接受其 Zigbee 网络中的新连接设备。

该应用需要两块 STM32WBxx_NUCLEO 板。

5.3 休眠终端设备

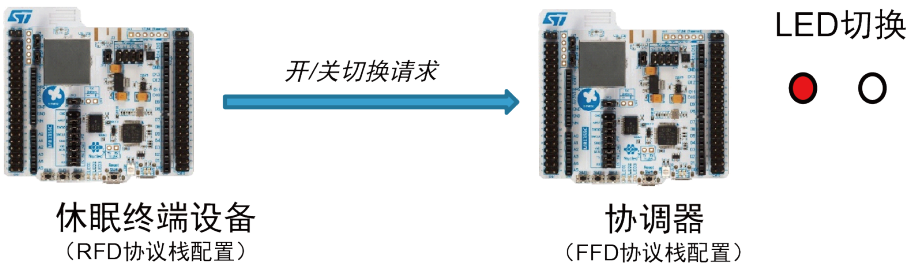
5.3.1 休眠终端设备原理

要运行该应用，需要有两块板

- 板 1：STM32WB55xx Nucleo 板烧录：
  - 无线协同处理器：stm32wb5x_Zigbee_FFD_fw.bin
  - 应用：Zigbee_OnOff_Server_Coord
- 板 2：STM32WB55xx Nucleo 板烧录：
  - 无线协同处理器：stm32wb5x_Zigbee_RFD_fw.bin
  - 应用：Zigbee_OnOff_Client_SED

休眠终端设备（板 2 充当客户端）一旦加入由协调器控制的 Zigbee 网络（板 1 充当服务器），将每秒向协调器发送一个单播开关切换请求。在此阶段，板 1 上的 LED1 在收到来自 SED 的请求时，应每秒切换一次。

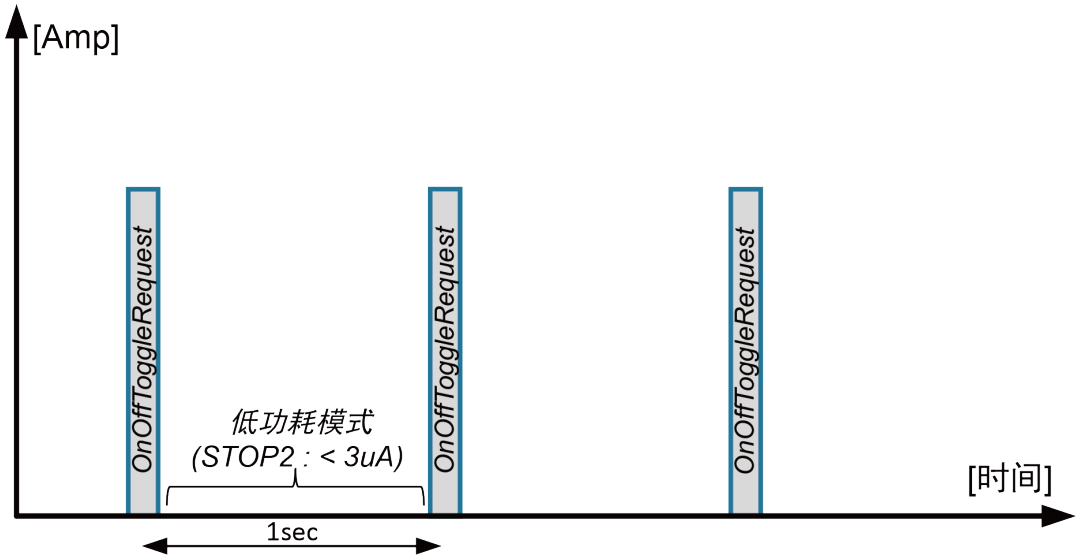
图 12. 休眠终端设备用例



为了在 SED 端实现最低功耗，编译此应用时默认将标志 CFG_FULL_LOW_POWER 设为 0（位于 app_conf.h 文件中）。在此配置下，LED 不再可用，且 M4 内核的调试访问也禁用。

在该配置中，使用 PowerShield 测试工具，可以检查在向协调器发送两个请求之间，SED 是否能够达到低功耗模式（STOP2）。

图 13. 休眠终端设备功耗

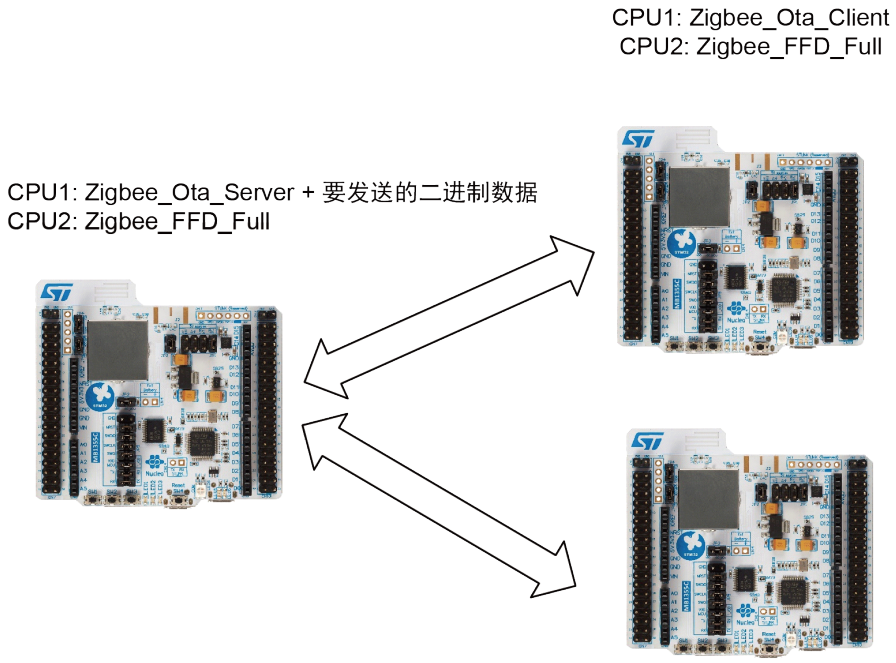


5.4 Zigbee FUOTA

5.4.1 Zigbee FUOTA 原理

目的是使用 Zigbee 协议更新远程设备上的 CPU1 应用二进制文件或 CPU2 无线协处理器二进制文件。

图 14. Zigbee FUOTA 网络拓扑



该 Thread 需要至少两块采用 Zigbee 协议并运行特定应用的 STM32WBxx 板（参见图 13）：

- 一块运行 ZigBee_Ota_Server 应用的板
- 一块或更多块运行 ZigBee_Ota_Client 应用的板

一次只能在一个设备上执行 FUOTA 流程。

服务器发起 FUOTA 配置流程，一个客户端必须响应。

如果多个客户端将逐一更新。

5.4.2 存储器映射

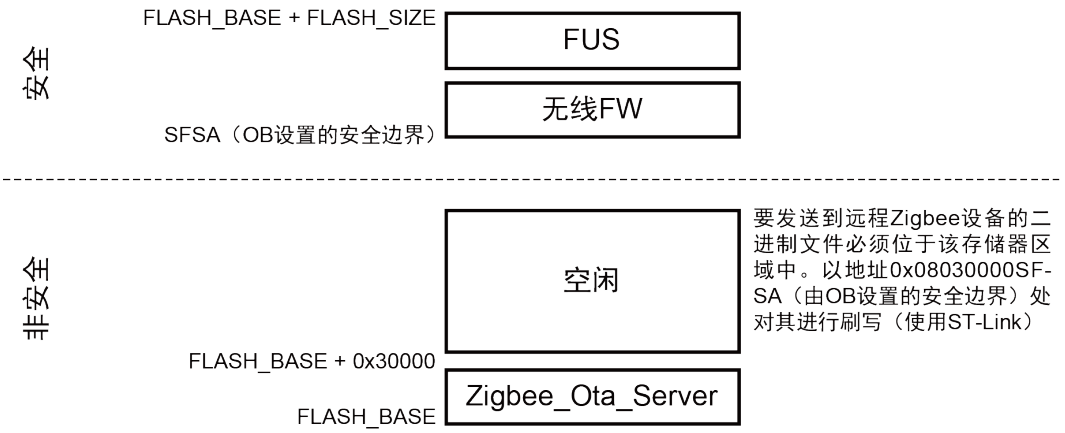
服务器端

必须先将要安装在远程设备上的二进制文件（用于 CPU1 或 CPU2 更新）刷写到服务器端的 “空闲”存储区（参见图 15）。

要传输的二进制文件的大小最大等于：

```
空闲区大小 = SFSA 地址 - （FLASH_BASE - 0x8030000）
```

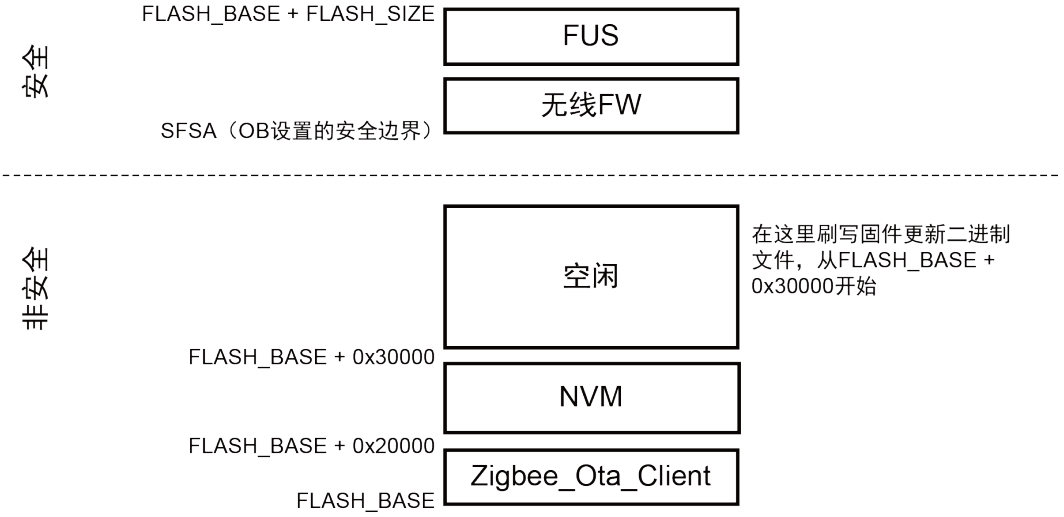
图 15. OTA 服务器（ZigBee_Ota_Server）Flash 存储器映射



客户端

在客户端，收到来自服务器的二进制数据前的 Flash 存储器如图 16 所示。

图 16. FUOTA 客户端 Flash 存储器映射初始状态



在收到 来自服务器端的二进制数据后，Flash 存储器得到更新，如图 17 和图 18（分别对应 CPU1 二进制数据传输和 CPU2 二进制数据传输）所示。

图 17. CPU1 二进制数据传输后的 FUOTA 客户端 Flash 存储器映射

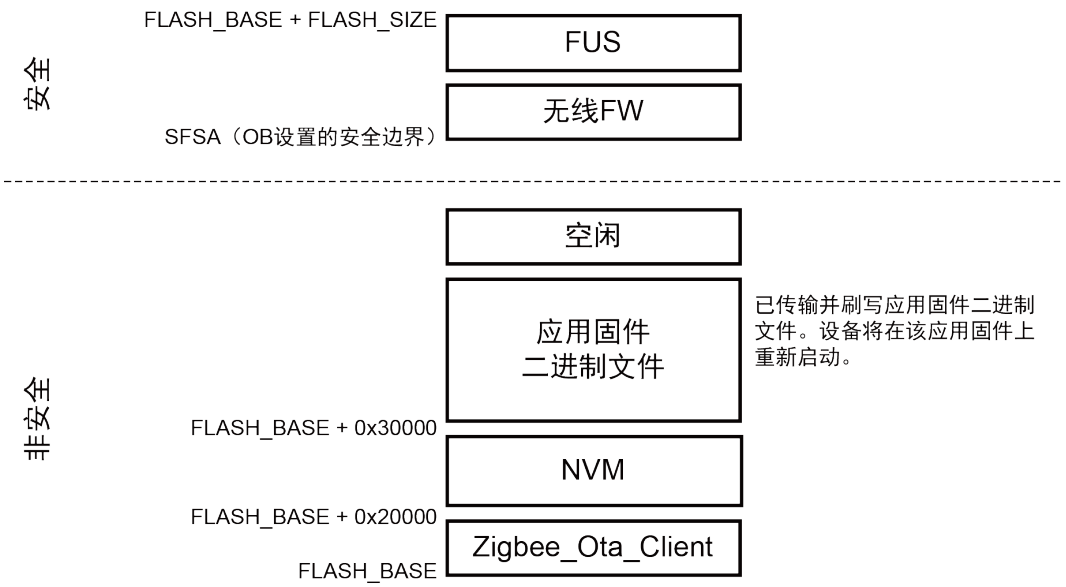
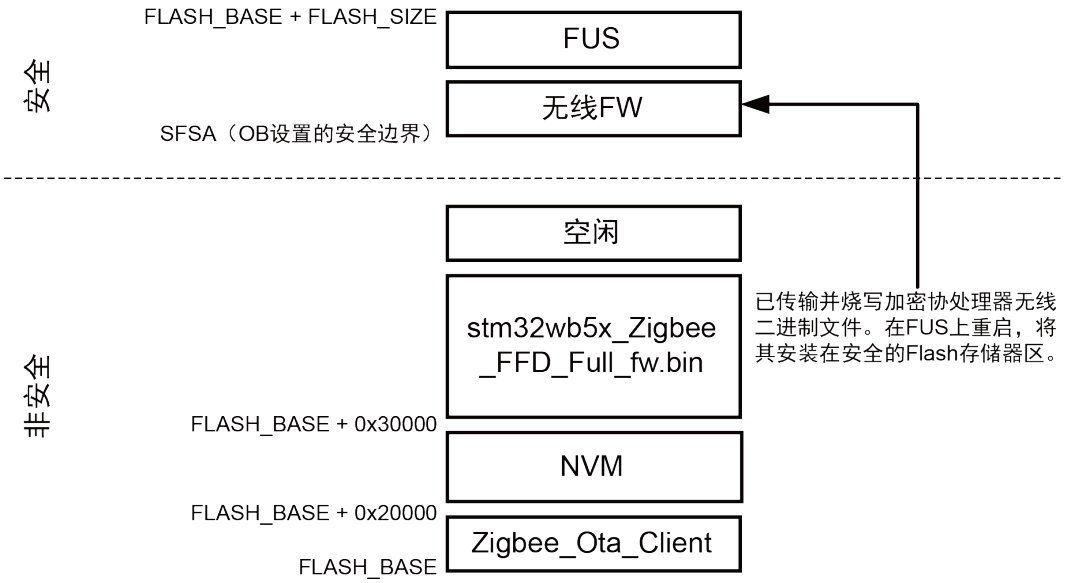


图 18. CPU2 二进制数据传输后的 FUOTA 客户端 Flash 存储器映射



5.4.3 Zigbee FUOTA 协议

这是意法半导体专有协议，可基于 ZCL OTA 群集，使用 Zigbee 更新 CPU2 无线协处理器二进制文件或 CPU1 固件应用。

OTA 文件格式

ZCL OTA 升级基于特定的 OTA 文件格式，该格式由以下部分组成：

- OTA 头文件
- 添加的 OTA 子元素

下图展示了 Zigbee FUOTA 升级进程的 OTA 文件格式

图 19. Zigbee FUOTA 的 OTA 文件格式

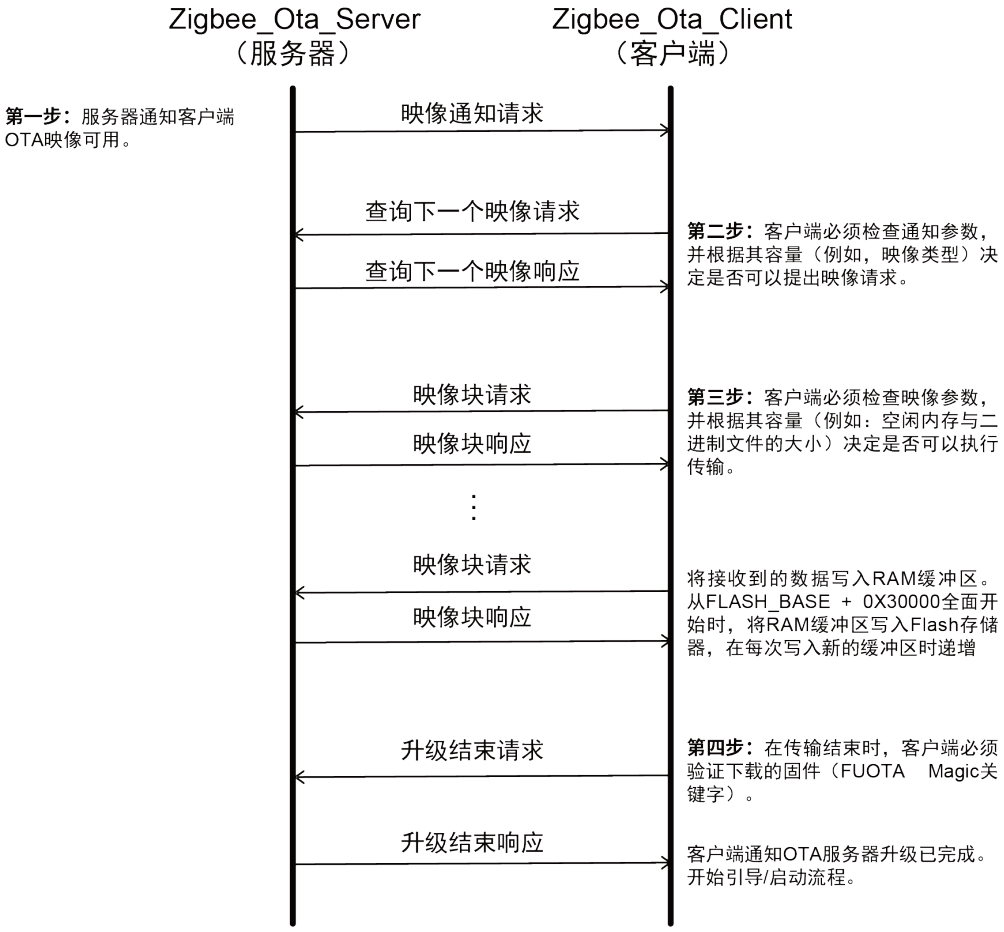
该 OTA 文件格式包含一个子元素（升级映像标签和关联的二进制固件数据）。

对于存储在 OTA 服务器 Flash 存储器中的任何 OTA 固件（有关更多详细信息，请参见第 5.4.2 节“存储器映射”），将动态生成 OTA 文件格式（OTA 头文件、升级映像标签信息）。

Zigbee FUOTA 过程

下图展示了执行固件更新传输的步骤。

图 20. Zigbee FUOTA 协议



1. 服务器发送“映像通知”请求，通知客户端映像可用。它包含：
   - 制造商 ID
   - 映像类型
   - 新文件版本
2. 客户端根据映像参数决定是否请求映像文件（“查询下一个映像”请求）。
3. 向客户端发送“查询下一个映像”响应。它包含：
   - 制造商 ID
   - 映像类型
   - 新文件版本
   - OTA 映像全尺寸

客户端必须检查参数，并根据其容量（如映像的空闲空间）决定是否可以执行传输。

1. 在 OTA 块传输期间，客户端将接收到的固件数据块存储在 RAM 缓存中，并在已满时存储在 Flash 中。
2. 传输完成后（客户端已收到有关 OTA 映像大小的所有必要块），客户端使用 magic 关键字对下载的内容进行验证。
3. 客户端通知服务器。
4. 升级以“升级结束”请求完成，并开始引导/启动流程（关于“升级结束”响应参数）。

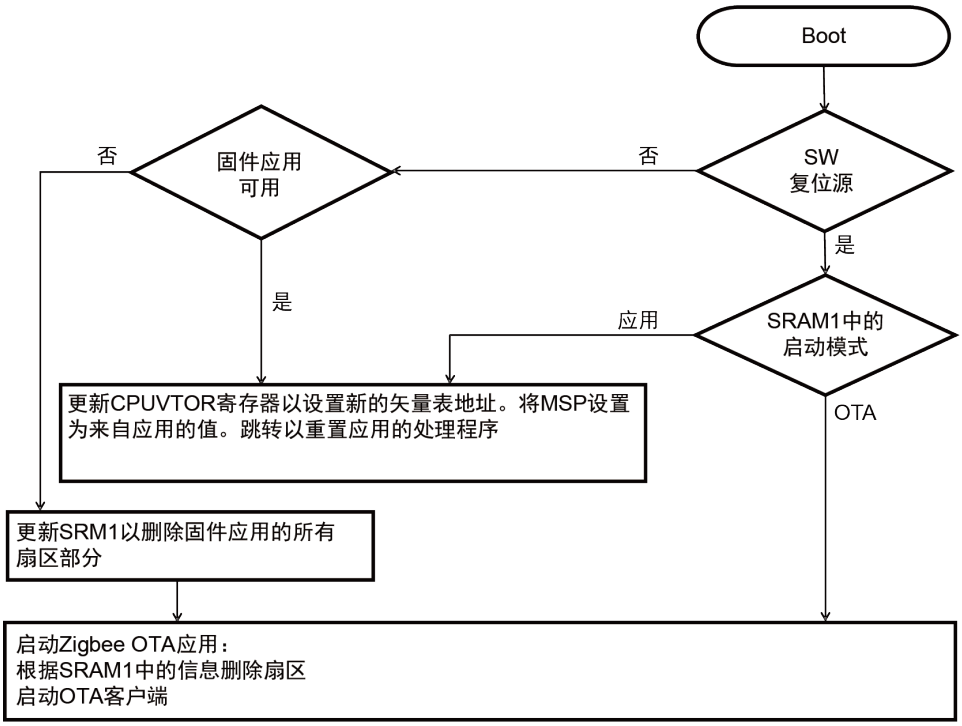
5.4.4 FUOTA 应用启动流程

在将二进制数据传输到远程设备（Zigbee FUOTA 客户端）后，CPU1 应用与 CPU2 协处理器无线二进制数据的更新启动流程并不相同。

CPU1 的 FUOTA

在客户端，在完成二进制数据传输后，将发生如图 21 所示的过程，从而跳转到 OTA 特定的应用（例如：Zigbee_OnOff_Client_Router_Ota）。

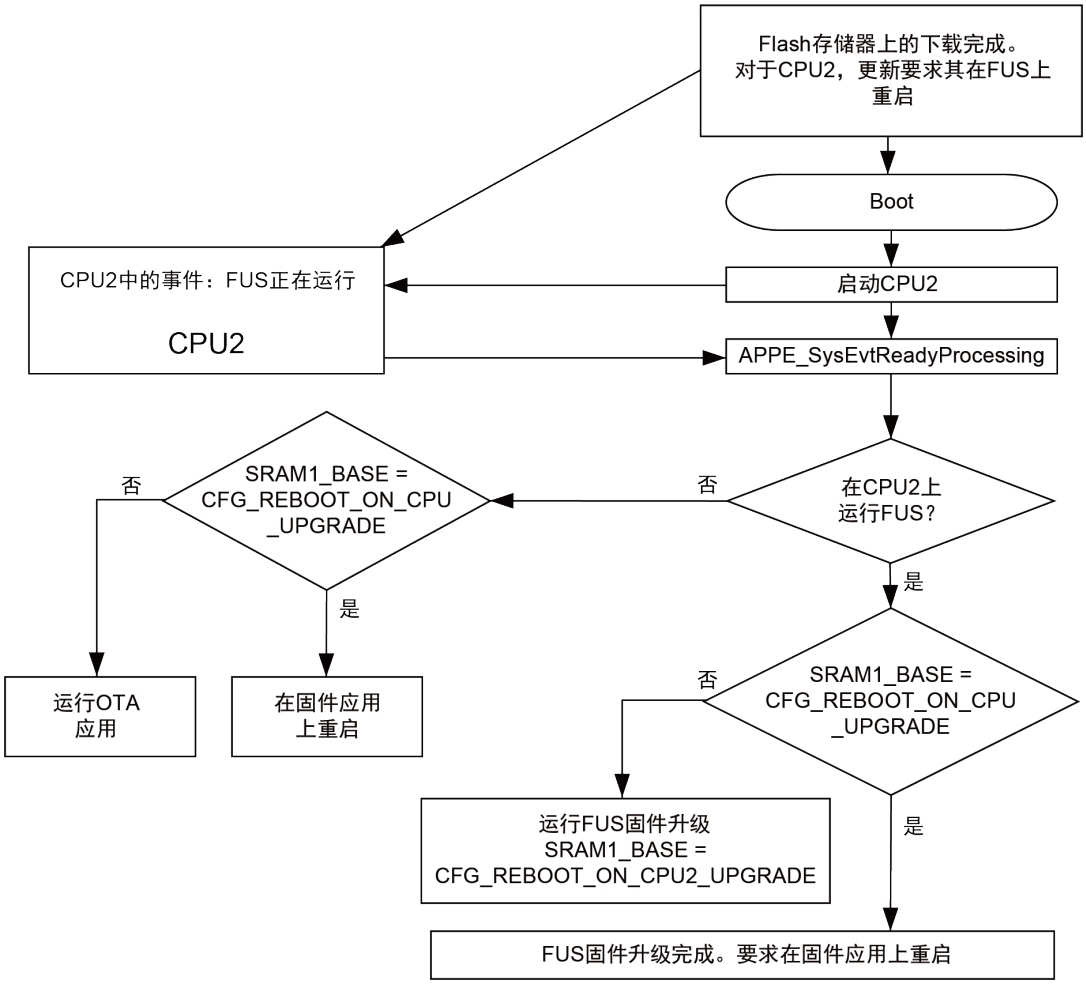
图 21. FUOTA 启动流程



CPU2 的 FUOTA

CPU2 更新涉及 FUS（固件升级服务）软件组件，该组件负责解密和安装安全的二进制文件。图 22 描述了这一进程。

图 22. 更新过程



5.4.5 应用

ZigBee_OTA_Server_Coord

必须将该应用加载到充当 FUOTA 服务器的 STM32WB Nucleo 板上。

ZigBee_OTA_Client_Router

必须将该应用加载到充当 FUOTA 客户端的 STM32WB Nucleo 板上。为了使客户端能够在升级流程后重新连接到 Zigbee 网络，必须保留 Zigbee 协议栈参数。这就是为何将持久性数据与 RAM 缓存和 Flash 存储一起使用的原因。

ZigBee_OnOff_Client_Router_Ota

该应用与 ZigBee_OnOff_Client_Router 几乎相同，区别在于：

- 使用具有 Flash 存储的持久性数据。NVM 配置与 ZigBee_OTA_Client_Router 应用中的配置相同。因此，使用该新应用固件重新启动后，可以恢复以前的 Zigbee 协议栈配置。更新的设备能够重新连接到协调器的网络。

- 使用特殊标签（用于管理数据传输结束和数据一致性）：
  - TAG_OTA_END:在 ZigBee_Ota_Client_Router 应用中检查 Magic 关键字值。
  - TAG_OTA_START:应在二进制映像开头的 0x140 处映射 Magic 关键字地址。

因此，在 0x140 处读取的存储内容等于 Magic 关键字值。

- 必须更新分散加载描述文件以插入上述存储区

IAR 的示例：

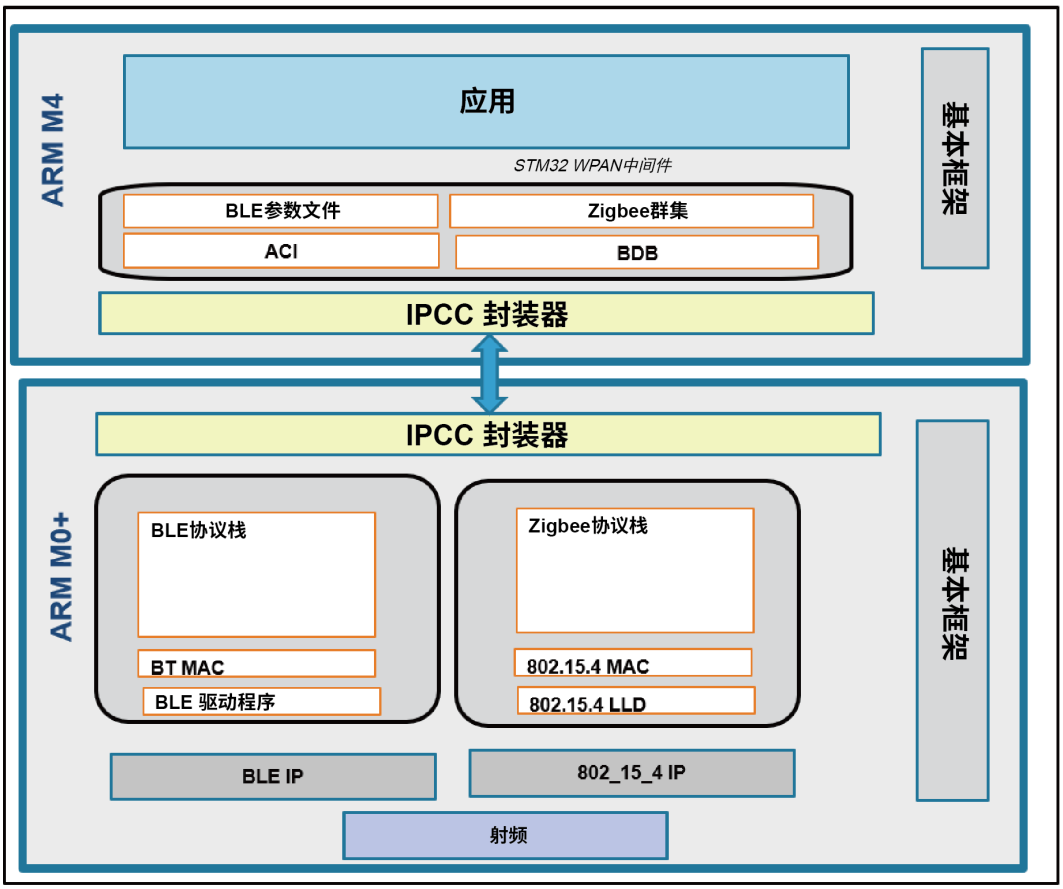
```
Vector table and ROM start @ moved to 0x08030000:
define symbol __ICFEDIT_intvec_start__ = 0x08030000;
define symbol __ICFEDIT_region_ROM_start__ = 0x08030000;
define region OTA_TAG_region = mem:[from
(__ICFEDIT_region_ROM_start__ + 0x140) to
(__ICFEDIT_region_ROM_start__ + 0x140 + 4)];
keep { section TAG_OTA_START};
keep { section TAG_OTA_END };
place in OTA_TAG_region { section TAG_OTA_START };
place in ROM_region { readonly, last section TAG_OTA_END };
```

5.5 静态并发模式

STM32WB 固件包中提供了静态并发模式（BLE/Zigbee）示例。

该应用位于 Projects\P-NUCLEO-WB55.Nucleo\Applications\BLE_Zigbee 目录下。运行该用例时，“静态并发模式”设备可以从 BLE 切换到 Zigbee，反之亦然。该设备通过 BLE 连接到运行“ST BLE Sensor”应用的智能手机，一旦 BLE 活动停止，即可加入 Zigbee 网络。接着，当 Zigbee 应用完全停止后，即可再次连接到 BLE。

图 23. STM32WB 系列上的静态并发模式



版本历史

表 7. 版本历史

日期	版本	变更
2020 年 7 月 23 日	1	初始版本

目录

表格索引

图片目录

重要通知 - 请仔细阅读

意法半导体公司及其子公司（“意法半导体”）保留随时对 ST 产品和/或本文档进行变更、更正、增强、修改和改进的权利，恕不另行通知。买方在订货之前应获取关于意法半导体产品的最新信息。意法半导体产品的销售依照订单确认时的相关意法半导体销售条款。

买方自行负责对意法半导体产品的选择和使用，意法半导体概不承担与应用协助或买方产品设计相关的任何责任。

意法半导体不对任何知识产权进行任何明示或默示的授权或许可。

转售的意法半导体产品如有不同于此处提供的信息的规定，将导致意法半导体针对该产品授予的任何保证失效。

ST 和 ST 标志是意法半导体的商标。关于意法半导体商标的其他信息，请访问 www.st.com/trademarks。其他所有产品或服务名称是其各自所有者的财产。本文档中的信息取代本文档所有早期版本中提供的信息。