armsky™
斯凯科技
http://www.armsky.net

ZigBee 快速入门

目 录

1. ZigBee 是什么？

ZigBee 和 IEEE 802.15.4 是基于标准的协议，它们为无线传感器网络应用提供所需要的网络基础设施。802.15.4 定义了物理层（PHY）和媒体访问控制（MAC）层，ZigBee 定义了网络（NWK）层和应用层（APL）。

对于传感器网络应用，关键的设计要求围绕着电池寿命长，成本低，占地面积小和网状网络等问题，以支持在一个互操作多应用环境中大量设备之间的通信。

1.1 典型应用

ZigBee 无线网状网络的冗余、自配置和自愈能力对许多应用来说是非常理想的，主要包括：

- 能源管理和提高效率 — 提供更多的信息和控制能源使用，为用户提供更好的服务和更多的选择机会，更好地管理资源，帮助减少对环境的冲击。
- 家居自动化 — 提供对照明、采暖、制冷、安全和家庭娱乐系统更灵活的管理。
- 楼宇自动化 — 整合并集中管理照明、采暖、制冷和安全。
- 工业自动化 — 扩大现有的生产和过程控制系统可靠性。

ZigBee 的互用性意味着这些应用可以一起工作，提供更大的好处。

1.2 ZigBee 目标

ZigBee 标准被开发以解决以下需求：

- 低成本
- 安全
- 可靠和自愈
- 灵活可扩展
- 低功耗
- 容易且不昂贵的部署
- 使用全球无限制无线电频段
- 智能化的网络建立和信息路由

1.3 关于 ZigBee 联盟

ZigBee 联盟是一个由 285 家公司一起工作的联合体，以实现基于一个开放的全球标准的、可靠的、具有成本效益、低功耗、无线网络的、检测和控制产品。其重点是以下方面：

- 定义网络、安全及应用软件层
- 提供互操作性和一致性测试规范
- 全球性地促进 ZigBee 品牌以建立市场意识
- 管理该技术的发展

1.4 产品认证

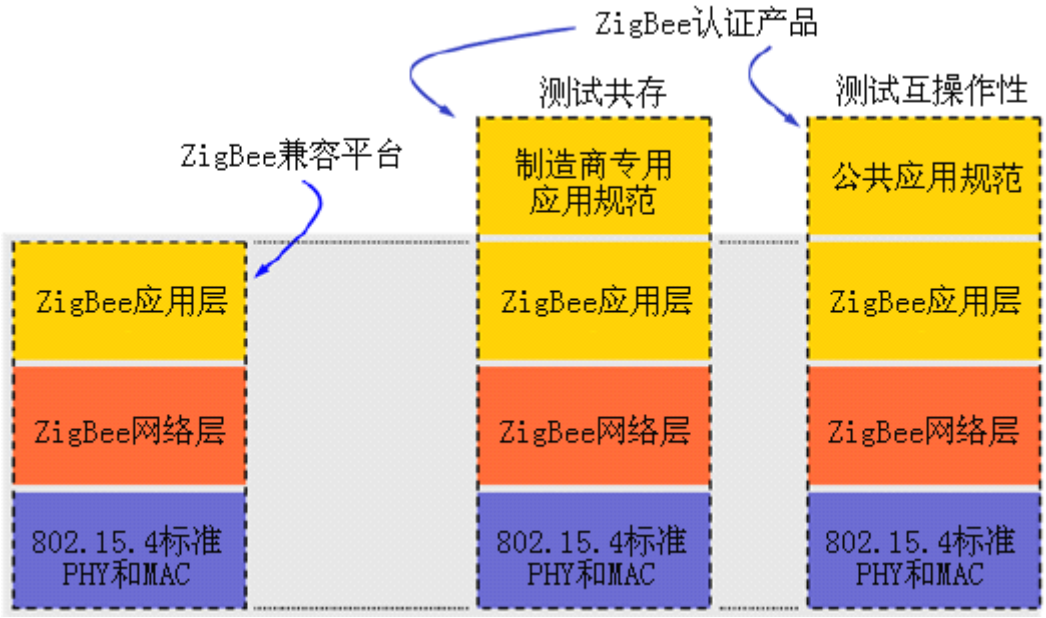
要使用 ZigBee Alliance 标识的产品，它必须首先成功完成 ZigBee 认证项目。这就确保了该产品符合在 ZigBee 规范里的标准描述。

只有那些通过了 ZigBee 认证的产品才可以使用 ZigBee 标识。

这里有两个 ZigBee 认证测试项目：

- ZigBee 兼容平台（ZCP）
  ZCP 适用于组件或平台，它们被用来创建终端产品的模块。

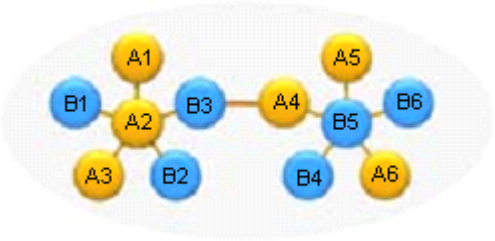
- ZigBee 认证产品
  该项目适用于那些建立在 ZigBee 兼容平台上的终端产品。顺利完成之后，这些产品可以使用 ZigBee 标识。



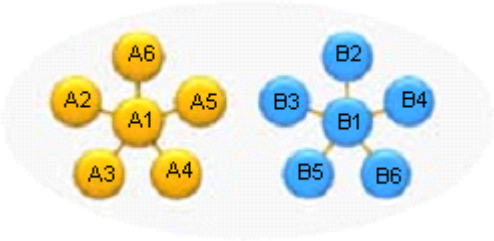
使用公共应用规范的产品被测试以保证与其他 ZigBee 终端产品的互操作性。

将运行为“封闭系统”的使用制造商专用应用规范的产品，被测试以保证它们能和其他 ZigBee 系统共存：也就是说，它们不影响到其他 ZigBee 认证产品和网络的运行。

应用规范将在本文档的后面被描述。



互操作性

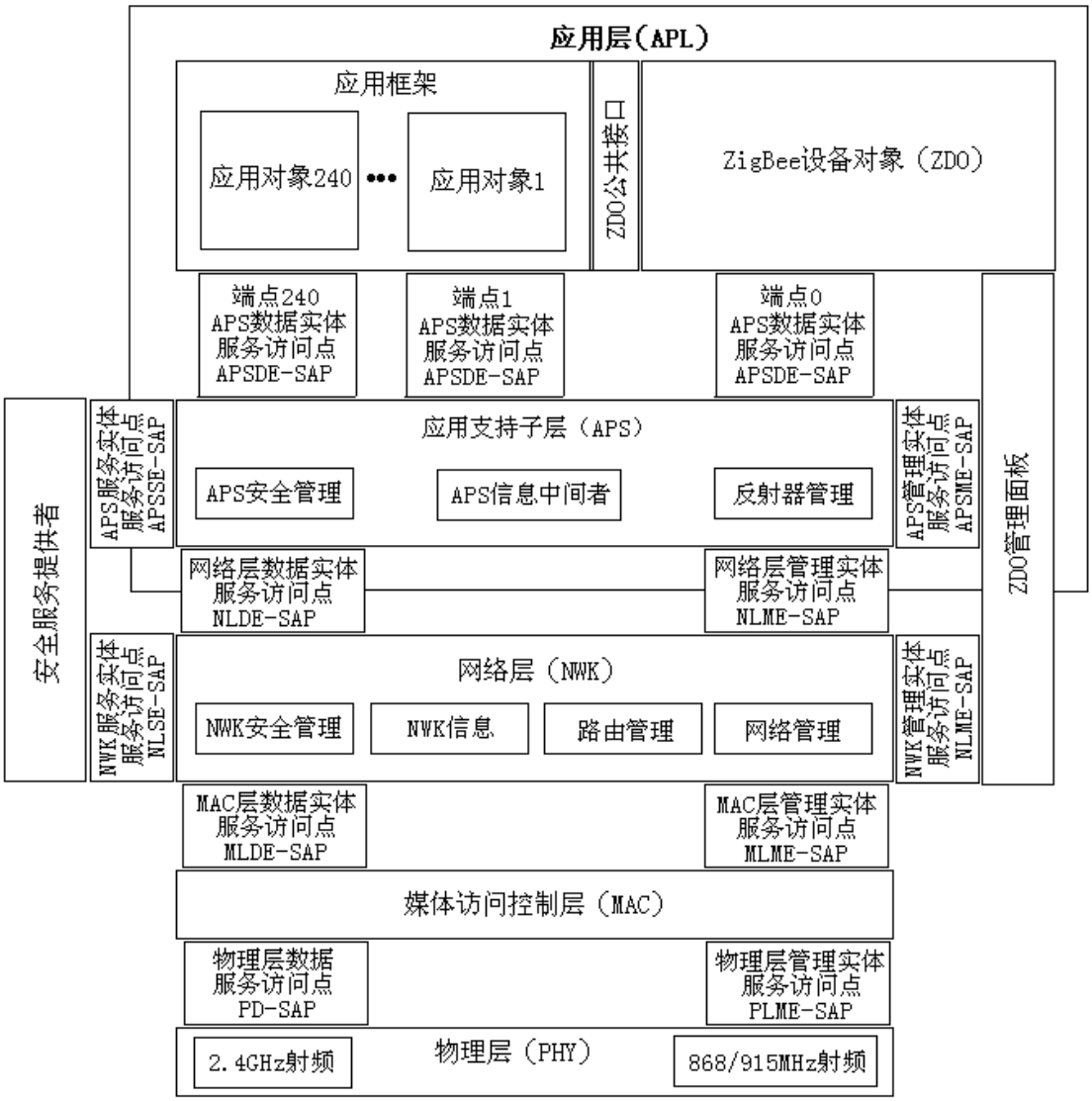


共存性

2. ZigBee 协议栈

ZigBee 位于 IEEE802.15.4 物理层（PHY）和媒体访问控制（MAC）层的上面：

ZigBee 协议栈



每一层为它的上层提供一套特定的服务。每一个服务实体通过一个服务访问点（SAP）为上层提供服务。

2.1 ZigBee

2.1.1 应用（APL）层

ZigBee 协议栈的顶层由应用框架、ZigBee 设备对象（ZDO）和应用支持（APS）子层组成。

2.1.2 应用框架

提供了一个如何在 ZigBee 协议栈上建立一个规范（以帮助确保该规范可以产生一种前后一致的方式）的描述。它也规定了规范的一系列的标准数据类型，协助服务发现的描述符，传输数据的帧格式，和一个键值对结构以快速开发基于属性的简单规范。。

2.1.3 应用对象

在一个端点上的软件，它控制 ZigBee 设备。一个单一的 ZigBee 节点可支持多达 240 个应用对象。每一个应用对象支持的端点编号为 1～240（端点 0 保留给 ZigBee 设备对象（ZDO））。

2.1.4 ZigBee 设备对象（ZDO）

定义一个设备在网络中的角色（协调器、路由器或终端设备），发起和/或回应绑定和发现请求，并在网络设备间建立一个安全关系。它也提供定义在 ZigBee 设备规范（用于 ZigBee 试运转）里的一套丰富的管理指令。ZigBee 设备对象总是为端点 0。

2.1.5 ZigBee 设备对象管理面板

使用 ZigBee 设备对象促进应用支持子层和网络层之间的联系。允许 ZigBee 设备对象处理为网络访问和安全的来自应用的请求，使用 ZDP（ZigBee 设备规范）信息。

2.1.6 应用支持（APS）子层

负责提供一个数据服务给应用和 ZigBee 设备规范。它也提供一个管理服务以维护绑定链接和它自己绑定表的存储。

2.1.7 安全服务提供者（SSP）

为使用加密的层（网络层和应用支持子层）提供安全机制。通过 ZDO 来初始化和配置 。

2.1.8 网络层（NWK）

通过在 MAC 层的调用行动来处理网络地址和路由。它的任务包括启动网络（协调器）、分配网络地址、添加和删除网络设备、路由信息、请求安全和执行路由发现。

2.2 IEEE 802.15.4

2.2.1 媒体访问控制层（MAC）

负责为一个节点和它的直接近邻之间提供可靠通讯，帮助避免碰撞和提高效率。MAC 层也负责组装和分解数据包和帧。

2.2.2 物理层（PHY）

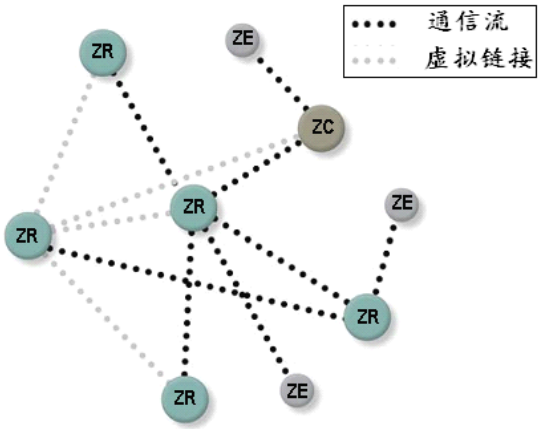
提供接口给物理传输介质（例如无线电）。物理层由两个层组成，它们运行在连个不同的频率范围。较低频率的物理层涵盖了 868MHz 欧洲频段和 915MHz 频段的使用国家，比如美国和澳大利亚。较高频率的物理层（2.4GHz）用于几乎世界各地。

3. ZigBee 网络

3.1 设备类型

ZigBee 网络包含以下设备类型：

- 协调器（ZC）
- 路由器（ZR）
- 终端设备（ZE）



3.1.1 协调器

该设备启动和控制网络。协调器存储关于网络的信息，包括作为认证中心和作为安全密钥的贮藏所。

3.1.2 路由器

这些设备扩展网络覆盖面，在障碍周围动态路由，并且提供备份路由以防网络拥挤和设备失败。它们可以联系到协调器和其它路由器，并且支持子设备。

3.1.3 终端设备

这些设备可以发送或接收一个信息，但是不能执行任何路由操作。它们必须被联系到协调器或者一个路由器，并且不支持子设备。

3.2 网状网络拓扑结构

网状拓扑结构，也被称为点到点，是由互联的路由器和终端设备组成的一个网状结构。每个路由器通常至少通过两个路径来连接，并且可以为它的邻居转发信息。

如上图所示，一个网状网络包含一个单一的协调器，以及多个路由器和终端设备。

网状拓扑结构支持“多跳”通讯，这些数据通过跳跃从一个设备到另一个设备，使用最可靠的通讯联系和最符合成本效益的路径，直到到达它的目的地。

这种多跳能力也帮助提供容错功能，如果一台设备失败或经历冲突，该网络可以使用剩下的设备重新路由它自己。

3.2.1 优点

- 该拓扑结构具有高可靠性和稳定性。任何独立路由器可能变得不可用，替换路由可以被发现和使用。
- 利用中间设备来转发数据，即网络范围可以被明显增加，使网状网络可高度扩展。
- 仅仅是通过添加更多的路由器到网络中，可以消除弱信号和死区。

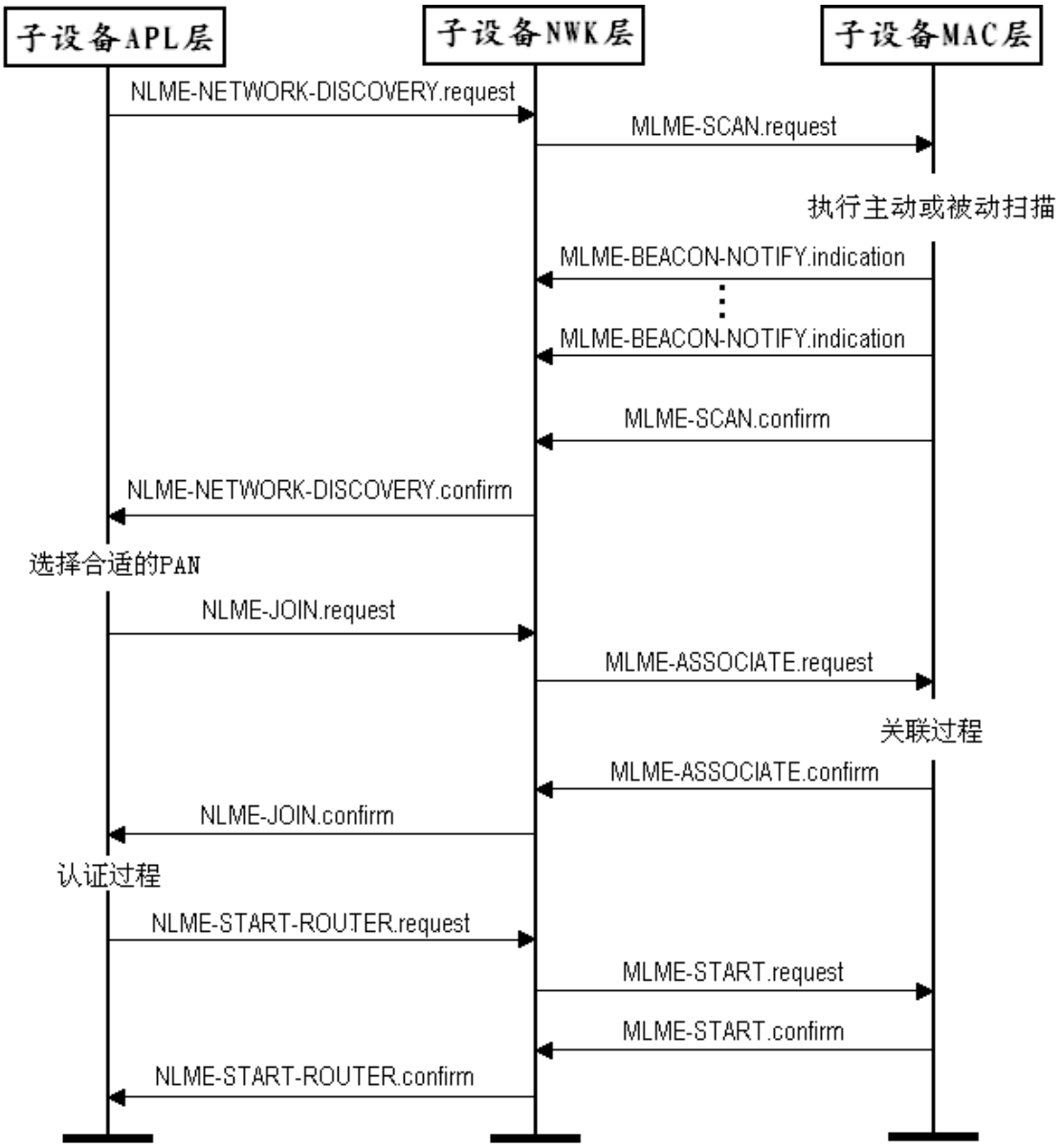
3.3 加入一个 ZigBee 网络

当网络中的设备允许一个新设备加入网络时，这两个设备就构成了父子关系。新加入的设备是子设备，而第一个设备是父设备。一个子设备可以通过下面两种方式加入网络：通过 MAC 层关联过程加入网络或由先前指定的父设备直接加入网络。只有 ZigBee 协调器或 ZigBee 路由器能够允许设备加入网络，而 ZigBee 终端设备则不能。

3.3.1 MAC 层关联

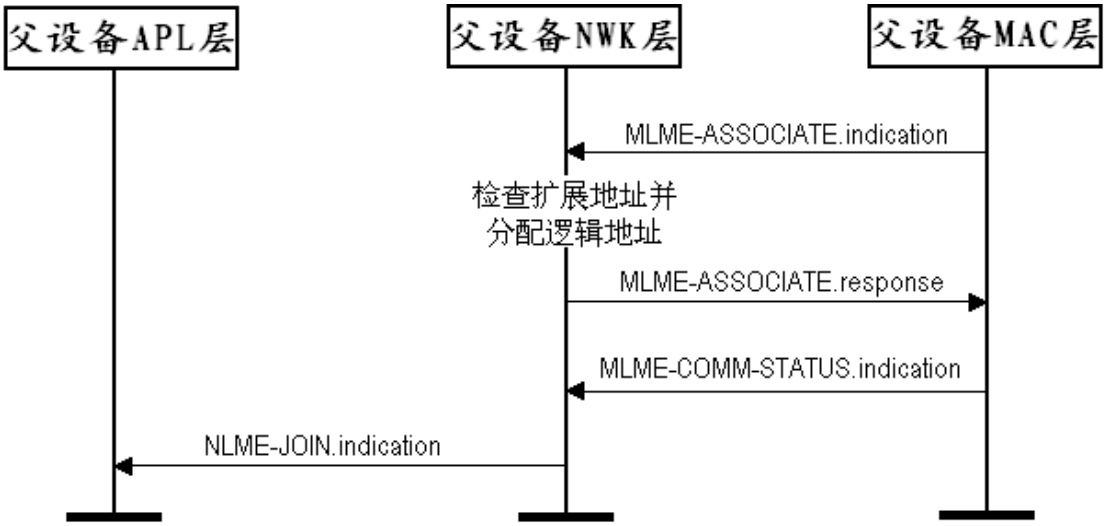
子设备通过 MAC 层关联加入网络的过程是通过向 NWK 层发送 NLME-NETWORK-DISCOVERY.request 原语来初始化的。该请求原语中 ScanChannels 参数指定了网络发现过程中要扫描的信道列表，Scanduration 参数指定了扫描每个信道花费的时间。NWK 层收到网络发现请求原语后，就向 MAC 子层发送 MLME-SCAN.request 原语，请求 MAC 层执行被动扫描或主动扫描。在扫描过程中，每接收到一个有效负载长度非零的信标帧，扫描设备的 MAC 层就向 NLME 发送一个 MLME-BEACON-NOTIFY.indication 指示原语。该指示原语中包含的信息由信标设备地址信息、是否允许关联以及信标有效负载等。扫描设备的 NLME 将检查信标有效负载中的协议 ID 字段，看它是否与自身的 ZigBee 协议标识匹配。如果不匹配，该信标就被忽略；如果匹配，扫描设备就把接收信标中的相关信息拷贝到近邻表中。当 MAC 层完成扫描想 NLME 发送 MLME-SCAN.confirm 原语后，NWK 层就向其上层发送 NLME-NETWORK-DISCOVERY.confirm 原语，把侦听到的每个网络的描述信息传递给应用层。每个网络描述信息包括 ZigBee 版本、协议栈配置文件、PAN ID、逻辑信道以及是否允许加入网络等。接收到 NLME-NETWORK-DISCOVERY.confirm 原语后，应用层就获知了设备邻近区域内存在网络的信息。如果要从中选择一个网络加入，设备应用层就向 NLME 发送 NLME-JOIN.request 原语，原语中 PANId 参数设置为选定网络的 PAN 标识，RejoinNetwork 参数设置为 FALSE，JoinAsRouter 参数则根据加入网络的设备是否是路由设备来设置。只有尚未加入网络的设备才能初始化通过 MAC 关联加入网络的过程，任何其他设备初始化该过程时，NLME 将中止该过程并告知其上层。这是通过向上层发送 Status 参数为 INVALID_REQUEST 的证实原语 NLME-JOIN.confirm 来实现的。一个尚未加入网络的设备 NWK 层收到 NLME-JOIN.request 原语后，将从近邻表中搜索合适的父设备。一个合适的父设备应有期望的 PAN ID，应允许关联并且链路成本至多为 3.如果近邻表记录中有潜在父设备字段，则该字段的值也应为 1.如果近邻表中没有合适的父设备，NLME 就向上层发送状态参数为 NOT_PERMITTED 的 NLME-JOIN.confirm 原语。如果近邻表中有多个设备可以作为父设备，则选择其中与 ZigBee 协调器深度最小的设备为父是设备。一旦确定了合适的父设备，NLME 就向 MAC 层发送 MLME-ASSOCIATE.request 原语。关联状态通过证实原

语 MLME-ASSOCIATE.confirm 反馈给 NLME。如果通过关联加入网络不成功，NWK 层收到的来自 MAC 层的 MLME-ASSOCIATE.confirm 原语中状态参数将指示出失败原因。如果状态参数指示近邻设备拒绝新设备加入，那么意图加入网络的设备应把近邻表中该近邻设备对应记录中的潜在父设备字段设为 0，以防止 NWK 层再次发送关联请求给这个拒绝关联的近邻设备。每次发送 MLME-SCAN.request 原语的时候，近邻表中每个记录的潜在父设备字段都设为 1.如果要加入网络的设备把 JoinAsRouter 参数设置为 TRUE，但潜在的父设备不允许新的路由器关联（如它关联的路由器已经达到最大值 nwkMaxRouters），加入请求也会失败。这种情况下，NLME-JOIN.confirm 原语中的状态为 NOT_PERMITTED。此时，子设备的应用层可以尝试以终端设备加入网络，吧 NLME-JOIN.request 原语中的 JoinAsRouter 参数设置为 FALSE。如果设备尝试加入网络失败，则 NWK 层将针对第二个设备重新初始化 MAC 层关联过程。NWK 层将重复这个过程知道设备成功加入到 PAN 或尝试了所有合适的父设备。如果设备不能成功加入到应用层指定的 PAN 中，NLME 将通过发送 NLME-JOIN.confirm 原语来中止加入网络的过程，此时，设备得不到有效的逻辑地址，不能在网络中发送数据。如果子设备成功加入到网络中，NWK 层收到 MLME-ASSOCIATE.confirm 原语中包含一个 16 位的逻辑地址，子设备以后就可以使用该逻辑地址来通信。子设备的 NWK 层还要设置相应近邻表记录中的 Relationship 字段，指示该近邻设备是它的父设备。如果设备试图加入一个安全网络中成为路由器，那么它在发送信标之前需要等待父设备的认证。如果设备成功加入网络，并且收到上层发送 NLME-START-ROUTER.request 原语，设备 NWK 层就向 MAC 层发送 MLME-START.request 原语，设置超帧配置并在要求的时候开始发送信标帧。只有 BeaconOrder 参数不等于 15 时，路由器才发送信标帧。PANId、LogicalChannel、BeaconOrder 和 SuperframeOrder 参数都应设置为其近邻表中父设备对应记录的值。PANCoordinator 和 CoorRealignment 参数都应设为 FALSE。接收到 MLME-START.confirm 原语后，NWK 也向上层发送一个同样状态的 NLME-START-ROUTER.confirm 证实原语。子设备通过关联加入网络的信息流程如下图所示。



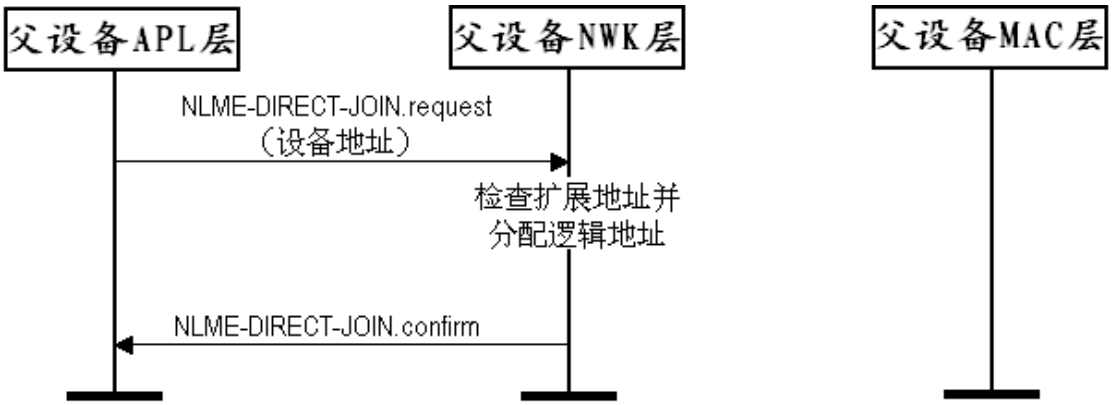
ZigBee 协调器或路由器通过 MAC 层关联把一个设备加入到网络的过程是由 MAC 层指示原语 MLME-ASSOCIATE.indication 来初始化的。收到该指示原语后，潜在父设备的 NLME 首先判断想加入的设备是否已经存在于网络中，即 NLME 搜索近邻表看是否有有匹配的 64 位扩展地址。如果找到匹配的扩展地址，NLME 将获得对应的 16 位网络地址并向 MAC 层发送关联响应；如果找不到匹配的扩展地址，NLME 在可能的情况下将为新设备分配一个唯一的 16 位网络地址。ZigBee 协调器为每个潜在的父设备分配了有限的地址空间，一旦地址空间占满，父设备就不接收入网请求了。如果潜在的父设备用尽了分配的地址空间，NLME 将中止设备加入网络的过程，并在随后的 MLME-ASSOCIATE.response 响应原语中反映这一事实。如果潜在的父设备接受了新设备的入网请求，NLME 将在近邻表中为新加入的子设备增加一条记录，记录设备信息，并向 MAC 层发送 MLME-ASSOCIATE.response 响应原语，指示关联成功。响应传输到子设备的状态通过 MLME-COMM-STATUS.indication 指示原语反馈给网络层。如果响应命令传输到子设备不成功，即 MLME-COMM-STATUS.indication 原语的状态不是 SUCCESS，则 NLME 将中止设备加入网络的过程；如果响应命令传输成功，

NLME 将向上层发送 NLME-JOIN.indication 原语，告知一个新设备已经加入到网络中。父设备接收入网请求的信息流程如下图所示。

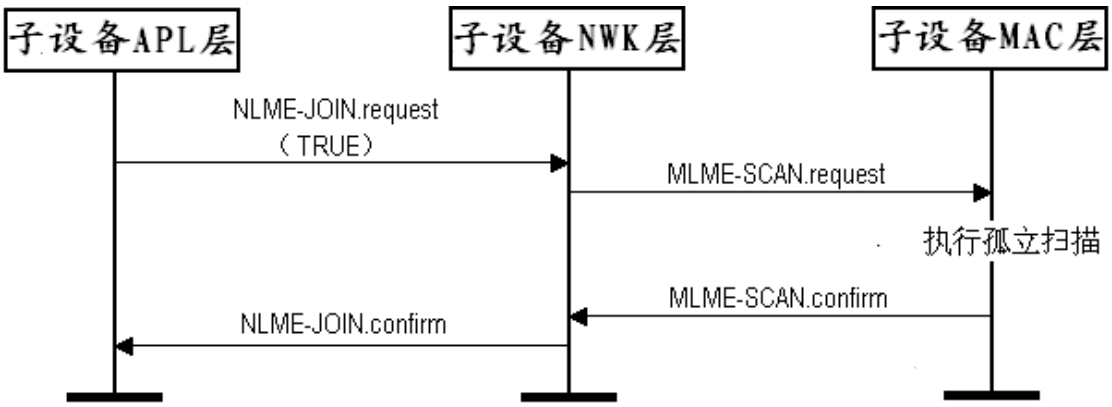


3.3.2 直接加入

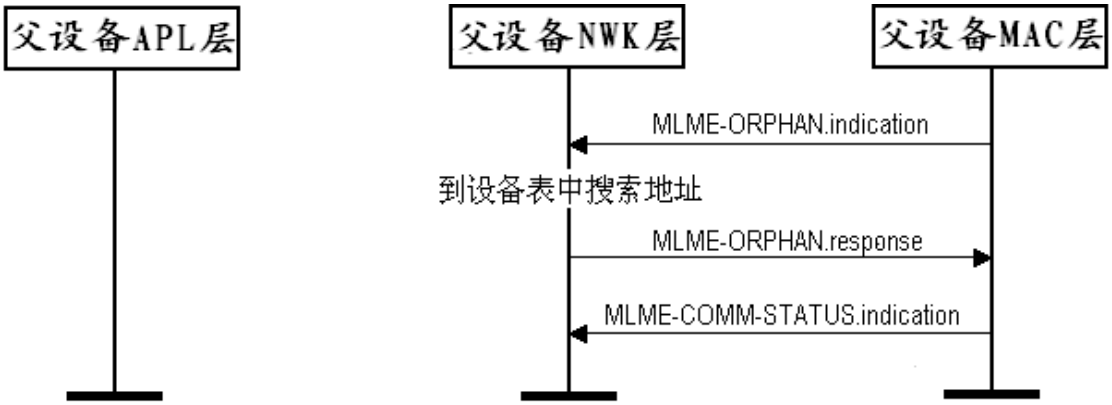
ZigBee 协调器或路由器把设备直接加入到网络的过程通过 NLME-DIRECT-JOIN.request 原语来初始化，原语中 DeviceAddress 参数设为将被加入到网络的设备地址。收到 NLME-DIRECT-JOIN.request 原语后，NLME 首先判断指定的设备是否已经存在于网络中，这是通过搜索近邻表判断是否有匹配的 64 位扩展地址来实现的。如果找到匹配的地址，NLME 将中止该过程，并向上层发送 Status 参数为 ALREADY_PRESENT 的证实原语 NLME-DIRECT-JOIN.confirm；如果找不到匹配的地址，NLME 在可能的情况下将为新设备分配一个 16 位网络地址。每个潜在父设备分到的地址空间是有限的，如果它有足够的空间接受一个新设备，则它还要在近邻表中增加一条新纪录，记录新设备的信息。如果地址空间容量不够，NLME 将中止该过程，向上层发送 Status 参数为 TABLE_FULL 的 NLME-DIRECT-JOIN.confirm 原语。如果有足够的空间，NLME 就向上层发送 Status 参数为 SUCCESS 的证实原语 NLME-DIRECT-JOIN.confirm。ZigBee 协调器或 ZigBee 路由器吧设备直接成功加入网络的过程如下图所示。



一个被直接加入到网络中的子设备为了完成与父设备的关系建立，将启动孤立申明过程，即子设备通过孤立申明加入网络；一个加入到网络中的子设备又与父设备失去联系时，要重新加入网络也要启动孤立申明过程，即子设备通过孤立申明重新加入网络。子设备通过孤立申明加入网络的过程是通过 NLME-JOIN.request 原语来启动的，原语中 RejoinNewwork 参数设为 TRUE。收到来自上层的 NLME-JOIN.request 原语后，NLME 首先请求 MAC 子层执行在所有可用信道上的孤立扫描。NLME 向 MAC 层发送 MLME-SCAN.request 原语启动孤立扫描,扫描结果通过 MLME-SCAN.confirm 原语反馈给 NLME。如果孤立扫描成功（即设备找到其父设备），NLME 将向上层发送 Status 参数为 SUCCESS 的证实原语 NLME-JOIN.request，把设备加入或重新加入网络请求成功的消息通知给上层。如果孤立扫描失败，NLME 将中止请求入网过程，并向上层发送 Status 参数为 NO_NETWORKS 的证实原语 NLME-JOIN.request，把设备没有找到网络的消息通知给上层。下图是一个子设备通过孤立扫描加入或重新加入网络的信息流程。



设备的 MAC 层向上层发送 MLME-ORPHAN.indication 原语告知一个孤立设备的存在。只有 ZigBee 协调器或 ZigBee 路由器才可以接受 MLME-ORPHAN.indication 原语，其他设备收到 MLME-ORPHAN.indication 原语时 NLME 将中止该过程。ZigBee 协调器或 ZigBee 路由器收到 MLME-ORPHAN.indication 原语后，首先判断孤立设备是否是它的子设备。这个判断过程是通过比较孤立设备与近邻表中子设备的扩展地址来实现的。如果 ZigBee 协调器或 ZigBee 路由器发现孤立设备是它的子设备，NLME 将获取该子设备的 16 位网络地址并通过孤立响应发送给 MAC 子层。孤立响应时通过向 MAC 层发送 MLME-ORPHAN.response 原语来实现的，孤立响应命令向子设备传送的结果状态通过 MLME-COMM-STATUS.indication 原语反馈给 NLME。如果 ZigBee 协调器或 ZigBee 路由器发现孤立设备不是它的子设备，NLME 就通过孤立响应原语把这一情况反映给 MAC 层。下图是父设备把孤立的子设备加入或重新加入到网络过程的信息流程。



设备的近邻表包含了传输范围内每个设备的信息。近邻表中存储的信息有多种用途，但并不是每个 ZigBee 设备运行时都要求下面提到的全部字段。近邻表中每个记录应包含近邻设备的下列信息：PAN 标识、扩展地址（如果近邻设备是父设备或子设备）、网络地址、设备类型、关系。另外，有些信息不是近邻表记录中必须的字段但具体实现时可以包含这些信息：空闲时接收机开启指示、任何近邻设备的扩展地址、深度、信标阶数、允许入网指示、发送失败指示、潜在父设备指示、平均 LQI、逻辑信道、接收信标帧时间戳、信标发送时间偏移；近邻表中甚至还可以包含近邻设备的其他信息。设备每次接收到近邻设备的任何帧时，都要对近邻表中相应的记录进行更新。近邻表中各信息字段的定义如下：

PAN 标识 近邻设备的 16 位 PAN 标识，其取值范围是 0x0000～0x3FFF。这是每一个近邻表记录都必须包含的信息。

扩展地址 每个设备唯一的 64 位扩展地址。如果近邻设备是该设备的父设备或子设备，就必须包含该字段。

网络地址 近邻设备的 16 位网络地址，其取值范围是 0x0000～0xFFFF。这是每一个近邻表记录都必须包含的信息。

设备类型 近邻设备的类型，0x00 表示 ZigBee 协调器，0x01 表示 ZigBee 路由器，0x02 表示 ZigBee 终端设备。这是每一个近邻表记录都必须包含的信息。

关系 近邻设备与当前设备之间的关系，0x00 表示近邻设备是父设备，0x01 表示近邻设备是子设备，0x02 表示近邻设备是兄弟设备，0x03 表示其他关系。这是每一个近邻表记录都必须包含的信息。

空闲时接收机开启指示 指示近邻设备的接收机在 CAP 的空闲期间是否开启，TRUE 表示开启，FALSE 表示关闭。如果近邻设备是父设备或是 ZigBee 路由器/ZigBee 协调器的子设备，近邻表记录必须包含该字段。

近邻设备的扩展地址 近邻设备的 64 位扩展地址。这是近邻表记录中的可选字段。

深度 近邻设备在网络拓扑中的深度，即到 ZigBee 协调器的最小跳数，深度值 0x00

表示近邻设备是 ZigBee 协调器。这是近邻表记录中的可选字段。

信标阶数 它指定了发送信标的频率。这是近邻表记录中的可选字段。

允许入网指示 指示近邻设备是否接受其他设备的入网请求，TRUE 表示接受入网请求，FALSE 表示不接受入网请求。这是近邻表记录中的可选字段。

发送失败指示 其值指示此前向及您设备发送是否失败，其取值为 0x00～0xFF，值越大表示失败次数越多。这是近邻表记录中的可选字段。

潜在父设备指示 指示近邻设备是否被指定为一个潜在的父设备，0x00 表示近邻设备不是潜在的父设备，0x01 表示近邻设备是潜在的父设备。这是近邻表记录中的可选字段。

平均 LQI 当前设备到近邻设备的链路质量估计。这是近邻表记录中的可选字段。

逻辑信道 近邻设备工作使用的逻辑信道。这是近邻表记录中的可选字段。

接收信标帧时间戳 设备接收到近邻设备最近一个信标帧的时间（用符号数表示），这个值等于设备接收信标时打的时间戳，其取值范围是 0x000000～0xFFFFFF。

信标发送时间偏移 近邻设备发送信标和它的父设备发送信标之间的时间差（用符号表示），设备用接收到近邻设备信标时间戳减去这个时间差就可以计算出近邻设备的父设备发送信标的时间。信标发送时间偏移的取值范围是 0x000000～0xFFFFFF。

3.4 路由功能

ZigBee 协调器和路由器应提供以下路由功能：代表上层转发数据帧；代表其他 ZigBee 路由器转发数据帧；为后面的数据帧建立路由而参与路由发现；代表终端设备参与路由发现；参与端到端路由修复；参与本地路由修复；使用路由发现和路由修复中指定的 ZigBee 路径成本度量。此外，ZigBee 协调器和路由器还可能提供下列路由功能：为记住最好的可用路由而维护路由表；代表上层启动路由发现；代表其他 ZigBee 路由器启动路由发现；启动端到端路由修复；代表其他 ZigBee 路由器启动本地路由修复。

3.4.1 路由成本

在路由发现和维护中，ZigBee 路由算法使用路径成本度量来进行路由比较。为了比较这个度量，这里赋予路径中每段链路度量—链路成本，构成路径的所有链路的成本之和就是整条路径的度量—路径成本。更正式地，如果把一组有序的设备[D_1，D_2，…，D_L]定义长度为 L 的路径 P，而其中的一段链路[D_i，D_{i+1}]定义为长度为 2 的子路径，那么路径成本可以表示为：

$$C\{P\}=\sum_{i=1}^{L-1}C\{[D_i,D_{i+1}]\}$$

这里 $C\{[D_i,D_{i+1}]\}$ 是链路成本，链路 1 的成本 C{1}是取值在[0…7]范围内的函数。它定义为：

$$C\{1\}=\begin{cases}7\\ \min\left(7,round\left(\frac{1}{p_1^4}\right)\right)\end{cases}$$

其中 p_1 表示包在链路 1 上传递的概率。这样，具体实现时可以选择常数 7 作为链路成本，也可以选择反应概率 p_1 的函数作为链路成本。如果一个设备提供了链路成本的这两种选项，那么通过设置 NIB 属性 nwkReportConstantCost 的值为 TRUE 可以强制使用常数 7 作为链路成本。现在剩下的问题就是如何得到概率 p_1。p_1 的估计是一个实现问题，完全由设计者按照自己的方式来实现。一种是可以通过在一段时间内统计丢帧来计算 p_1，人们普遍认为这种方法得到的概率 p_1 是最精确的。然而，最直接的方法是基于 MAC 和 PHY 层提供的每帧平均 LQI 来估计 p_1，一个函数可以把平均 LQI 映射到 C{1}上，根据 LQI 通过查表就可以得到链路成本。

3.4.2 路由表

ZigBee 路由器或 ZigBee 协调器可能维护了一个路由表，路由表中存放的信息如表 1 所列。路由表记录中路由状态信息的取值如表 2 所列。ZigBee 路由器或 ZigBee 协调器还可能预留一些路由表记录专用于路由修复和在其他路由能力都耗尽的时候才使用。在后面的路由算法中会用到“路由表能力”这个术语，所谓路由表能力是指设备使用路由表能够建立起一条到达特定目的设备的路由。如果一个设备是 ZigBee 协调器或 ZigBee 路由器，它维护的路由表中有空闲的路由表记录或已经有一个与目的设备对应的路由表记录，并且正在尝试路由修复的设备预留了专用于路由修复的路由表记录，那么就说它具有“路由表能力”。如果 ZigBee 路由器或 ZigBee 协调器维护了一个路由表，那么它还应该维护一个路由发现表。路由发现表包含的信息如表 3 所列。路由表记录在设备中是长期存在的，而路由发现表记录仅维持一次路由发现操作的时间并且可以重复使用。如果一个设备维护了一个路由发现表，并且路由发现表中有空闲的记录，那么就说这个设备具有“路由发现表能力”。如果一个设备既有路由表能力，又有路由发现表能力，那么就或设备具有“路由能力”。

表 1

字 段 名	字段长度	描 述
目的地址	2 字节	本路由最终目的设备的 16 位网络地址
状态	3 位	路由状态
下一个跳点地址	2 字节	去往目的地址的路由上下一跳的 16 位网络地址

表 2

数 值	状 态
0x0	ACTIVE（活动）
0x1	DISCOVERY_UNDERWAY（正在执行路由发现）
0x2	DISCOVERY_FAILED（路由发现失败）
0x3	INACTIVE（不活动）
0x4～0x7	预留

表 3

字 段 名	字段长度/字节	描述
路由请求标识	1	路由请求命令帧的序号，该序号在每次设备发起路由请求时递增
源地址	2	路由请求发起设备的 16 位网络地址
发送地址	2	最新发送最低成本路由请求命令帧的设备网络地址。 该信息用于决定最后路由应答命令帧应遵循的路径
前期成本	1	从路由请求源地址设备到当前设备的累加路径成本
剩余成本	1	从当前设备到目的设备的累加路径成本
到期时间	2	用以设定路由发现到期时间的递减定时器上的时间（单位为 ms）， 其初始值是 nwkcRouteDiscoveryTime 的属性值

3.4.3 基本路由算法

设备 NWK 接收到数据帧时按照下面的程序为帧安排路由。如果 NWK 接收到的数据帧来自其上层并且目的地址为广播地址，NWK 层就按照后续部分将要介绍的程序来广播该数据帧。如果接收设备是 ZigBee 路由器或 ZigBee 协调器，帧的目的设备是 ZigBee 终端设备并且还是接收设备的子设备，那么接收设备将使用 MCPS-DATA．request 原语把帧直接发送给目的设备，即下一跳的目的地址就等于最终目的地址。具有路由能力的设备应检查 NWK 帧头部分帧控制字段中的发现路由子域，如果发现路由子域的值等于 0x02，那么设备将立即启动路由发现过程。如果发现路由子域的值不等于 0x02，设备将在路由表中查找与帧目的地址对应的记录。如果找到帧目的地址对应的路由表记录，并且其中路由状态的值为 ACTIVE，那么设备就用 MCPS-DATA．request 原语转发收到的帧。转发帧时，MCPS-DATA．request 原语中 SrcAddrMode 和 DstAddrMode 参数值应都为 0x02，即使用 16 位短地址；原语中 SrcPANId 和 DstPANId 参数值都应为转发设备的 MAC PIB 属性 macPANId 的值；SrcAddr 参数应设为转发设备的 MAC PIB 属性 macShortAddress 的值，DstAddr 参数应设为目的地址对应的路由表记录中下一跳地址字段的值；TxOptions 参数与 0x01 逐位相与操作后结果总是非零值，即转发的帧要求确认。如果找到帧目的地址对应的路由表记录，但其中路由状态的值为 DISCOVERY_UNDERWAY，则表示设备正在为该帧执行路由发现操作。此时设备可以选择缓存该帧以等待路由发现完成，也可以选择在 NIB 属性 nwkUseTreeRouting 等于 TRUE 时用分级路由把该帧沿着树传递。如果沿树传递帧，NWK 头的帧控制字段中的发现路由子域应设为 0x00。如果找到帧目的地址对应的路由表记录，但其中路由状态的值为 DISCOVRY_FAILED 或 INACTIVE，则在 NIB 属性 nwkUseTreeRouting 等于 TRUE 时刻采用分级路由把该帧沿着树传递。如果设备路由表中找不到帧目的地址对应的路由记录，设备将检测 NWK 头部分帧控制字段的发现路由子域。如果发现路由子域的值等于 0x01，设

备将启动路由发现过程;如果发现路由子域的值等于 0x00 并且 NIB 属性 nwkUseTreeRouting 等于 TRUE，设备将用分级路由把该帧沿着树传递。如果发现路由子域的值等于 0x00 并且 NIB 属性 nwkUseTreeRouting 等于 FALSE，设备路由表中又找不到帧目的地址对应的路由记录，则 NLDE 将丢弃该帧并向上层发送 Status 参数等于 INVALID_REQUEST 的 NLDE-DATA．request 原语。

对于一个没有路由能力的设备，如果 NIB 属性 nwkUseTreeRouting 等于 TRUE，设备将采用分级路由沿着树转发帧。在分级路由算法中，如果帧的目的设备是当前设备的后代设备，那么设备就把帧转发给适当的子设备。如果帧的目的设备就是当前设备的一个子设备，并且该子设备是终端设备，那么该终端设备的 macRxOnWhenIdle 状态可能会导致不能立即递交帧。当子设备的 macRxOnWhenIdle 值等于 FALSE 时，设备只能采用间接传输来递交帧。如果帧的目的设备不是当前设备的后代设备，那么设备将把帧提交给父设备。ZigBee 网络中除 ZigBee 协调器之外的每一个设备都是 ZigBee 协调器的后代设备，而任何一个 ZigBee 终端设备都没有后代设备。对一个地址为 A、深度为 d 的 ZigBee 路由器，如果下面的逻辑表达式成立，那么地址为 D 的目的设备就是该路由器的后代设备：

A＜D＜A+Cskip（d—1）

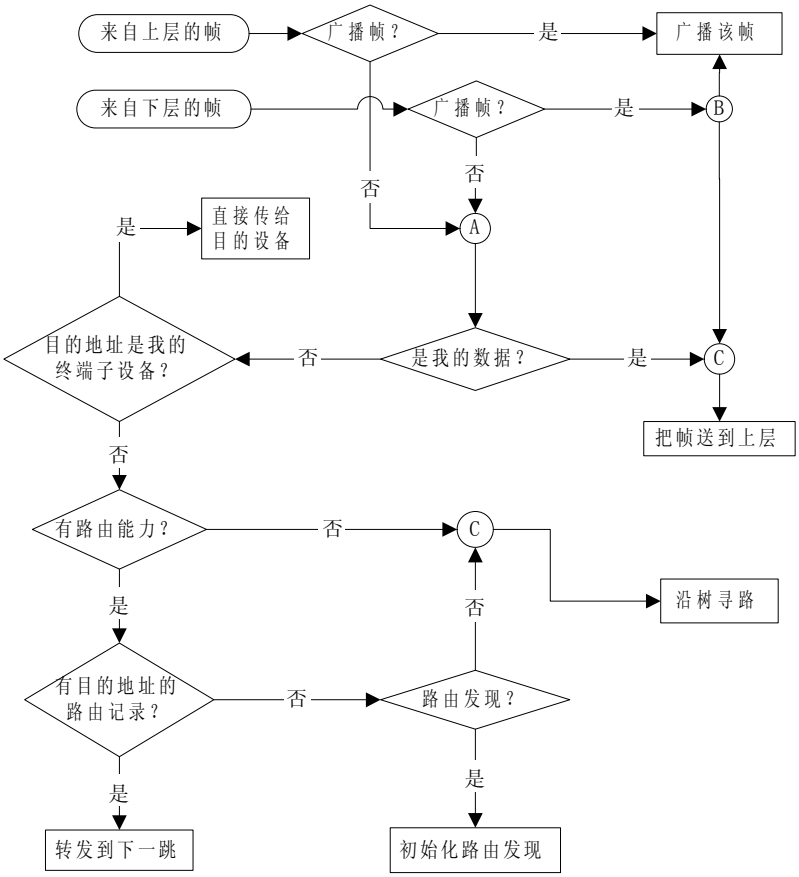
如果确定了帧的目的设备是当前接收设备的后代，那么当目的设备是当前接收设备的子设备且为终端设备时，下一跳地址 N 为：

N=D

这里 D＞A+Rm×Cskip（d）。当目的设备是当前接收设备的其他后代设备时，下一跳的地址 N 为：

$$N = A + 1 + \left[\frac{D - (A+1)}{Cskip(d)} \right] \times Cskip(d)$$

如果 NWK 层接收到的数据帧来自 MAC 子层并且目的地址为广播地址，NWK 层将首先重新广播该帧再把该帧送给上层处理。如果 NWK 层接收到的来自 MAC 子层的数据帧不是广播帧，那么 NWK 层就将判断该帧的目的地址是否等于当前接收设备的逻辑地址。如果帧的目的地址等于当前设备的逻辑地址，NWK 层就把帧传递给上层处理；否则，当前接收设备就只是该帧的一个中间设备，此时 NWK 层对该帧的处理过程与上文介绍的 NWK 层处理来自上层的单播数据帧的过程一样。下图是 ZigBee 的基本路由算法。



3.4.4 路由发现

路由发现是网络中的设备相互配合，发现并建立路由的过程。路由发现总是针对特定的源设备和目的设备执行的。在下面三种情况下，NWK 层将启动路由发现过程：第一种情况是 NWK 层收到来自上层的 NLDE-DATA.request 原语中 DiscoverRoute 参数值为 0x02；第二种情况是 NWK 层收到来自上层的 NLDE-DATA.request 原语中 DiscoverRoute 参数值为 0x01 并且没有 DstAddr 参数对应的路由表记录；第三种情况是 NWK 层收到来自 MAC 层的数据帧，NWK 头的目的地址不是当前设备的地址或是广播地址，并且其帧控制字段中发现路由子域的值等于 0x02 或 0x01，当前设备路由表中没有 NWK 头目的地址对应的记录了。在上述任何一种情况下，如果当前设备没有路由能力并且 NIB 属性 nwkUseTreeRouting 的值等于 TRUE，NWK 层将用分级路由算法沿树递交该数据帧；如果设备没有路由能力并且 NIB 属性 nwkUseTreeRouting 的值等于 FALSE，NWK 层将丢弃该帧。

对具有路由能力的设备，如果其路由表中没有帧目的地址对应的记录，它将创建一条状态为 DISCOVERY_UNDERWAY 的路由表记录；如果路由表中有一条对应于帧目的地址的记录，并且状态为 ACTIVE，那么设备将使用该路由转发数据帧并保持该路由表记录的状态不变；如果路由表中存在帧的目的地址对应的记录，但状态不等于 ACTIVE，那么设备将使用该路由表记录，并把其状态设为 DISCOVERY_UNDERWAY；同时，设备还要建立相应的路由发现表记录。

每个发送路由请求命令帧的设备都要维护一个计数器，用来产生路由请求标识。每当设备产生一个新的路由请求命令帧时，路由请求计数器加 1，并把路由请求计数器的值存放在设备路由请求发现表的路由请求标识字段中。路由发现表中路由请求定时器的计时周期应设为 nwkcRouteDiscoveryTime 毫秒，计时期满后，设备将把对应的路由请求记录从路由发现表中删除。如果此时目的地址对应的路由表记录中 Status 字段的值仍然是 DISCOVERY_UNDERWAY，并且路由发现表没有该目的地址对应的其他记录，则设备同时还要删除该路由表记录。NWK 层可选择缓存接收的帧等待路由发现或在 NIB 属性 nwkUseTreeRouting 等于 TRUE 时，把 NWK 头帧控制字段中的发现路由子域设为 0 并沿着树递交帧。

设备创建了路由发现表和路由表记录后，就要按照前面介绍过的帧格式构造路由请求命令帧的有效负载部分。命令帧标识字段应设为 0x01 表示路由请求；路由请求标识字段应设为路由发现表记录中存放的值；目的地址字段应设为该路由发现过程所指向目的设备的 16 位网络地址；路径成本字段应设为 0。准备好路由请求命令广播帧后，NWK 层就调用 MCPS-DATA.request 原语把它递交给 MAC 子层。路由发现过程的启动设备广播路由请求命令帧时，NWK 在初次广播之后还应重复广播 nwkcInitialRREQRetries 次，即总计广播 nwkcInitialRREQRetries+1 次路由请求命令帧。每两次广播时间的时间间隔是 nwkcRREQRetryInterval 毫秒。

接收到路由请求命令帧后，设备将判断自己是否具有路由能力。如果设备没有路由能力，它将检测接收的帧是否来自有效路径。如果接收帧来自设备的一个子设备并且源设备是该子设备的一个后代或者接收帧来自设备的父设备并且源设备不是当前的后代，那么这样的路径就是有效的。如果接收到的路由请求命令帧不是来自有效路径，设备将丢弃该帧。如果路由请求命令帧是来自有效路径，设备将检测路由请求命令帧的目的设备是否是当前设备或当前设备的终端子设备。如果当前设备或其一个终端子设备是该路由请求命令帧的目的设备，它将回应一个路由应答命令帧。设备用路由应答命令帧回应路由请求时，设备将构造一个帧类型为 0x01（命令帧）的帧。路由应答的源地址应设为路由应答产生设备的 16 位网络地址，目的地址应设为计算出的下一跳的地址，而对应的路由请求发起设备的地址则是应答帧的最终目的地址，它是放在有效负载部分的发起地址字段内的。计算出当前设备到下一跳设备的链路成本并插入到路由应答命令帧的路径成本字段。路由应答命令发起设备将调用 MCPS-DATA.request 原语把路由应答单播发送到下一跳设备。如果当前设备不是路由请求命令帧的目的地址，设备将计算从前一跳设备到其自身的这段链路的成本，并累加到路由请求命令帧的路径成本值中。然后，设备将调用 MCPS-DATA.request 服务原语吧路由请求命令帧向目的地址单播转发。此时单播发送的下一跳地址的决定方法与数据帧一样，即把当前的路由请求命令帧看作一个数据帧，而该数据帧的目的设备则是路由请求命令帧有效负载部分目的地址字段指定的设备。

如果接收路由请求命令帧的设备具有路由功能，它将检测路由请求命令帧的目的设备是否是当前设备或当前设备的终端子设备。如果当前设备或其一个终端子设备是路由请求命令帧的目的设备，设备将判断是否存在对应于路由请求标识和源地址字段的路由发现表记录。如果路由发现表中没有这样的记录，设备将产生该路由请求的路由发现表记录。路由发现表记录中的各字段根据路由请求命令帧中对应的字段设置，唯一的例外就是路由发现表中前期成本字段的值要在路由请求命令帧路径成本的基础上累加前一跳设备到当前设备的链路成

本。如果 nwkSymLink 属性值为 TRUE，即对称链路路由，则设备还要创建一个路由表记录 。路由表记录中目的地址字段设置为路由请求命令帧的源地址，下一跳字段设置为路由请求命令前一个发送设备的地址，状态字段设置为 ACTIVE。然后设备应向路由请求命令帧的发起设备发送一个路由应答命令。如果路由发现表中存在一个对应于路由请求标识和源地址字段的路由发现表记录，设备将判断当前路由发现过程中得到的路径成本是否小于现存的路由发现表记录中对应的前期成本字段的值。如果当前路由发现得到的路径成本大于现存的路由发现表记录中的前期成本，设备将丢弃该路由请求命令帧，不需再作进一步处理；否则，设备就把路由发现表记录中前期成本和发送设备地址字段的值更新为当前路由发现得到的路径成本值和路由请求命令帧的前一个发送设备的地址。同样，如果 nwkSymLink 属性值为 TRUE，则设备还要创建一个路由表记录。路由表记录中目的地址字段设置为路由请求命令帧的源地址，下一跳字段设置为路由请求命令前一个发送设备的地址，状态字段设置为 ACTIVE。然后设备应向路由请求命令帧的发起设备发送一个路由应答命令。在上述任何一种情况下，如果设备是代表其终端子设备发送路由应答命令，则路由应答命令帧有效负载中应答地址字段应设为终端子设备的地址而不是应答设备的地址。

如果具有路由能力的设备不是接收到的路由请求命令帧的目的设备，它将判断是否存在该路由请求命令的路由请求标识和源地址对应路由发现表记录。如果不存在这样的路由发现表记录，设备将产生一条记录，并把路由请求定时器的定时周期设为 nwkcRouteDiscoveryTime 毫秒。如果存在路由请求目的地址对应的路由表记录但状态值不是 ACTIVE，则把状态值设备 DISCOVERY_UNDERWAY；如果不存在路由请求目的地址对应的路由表记录，则创建一条记录。如果 nwkSymLink 属性值为 TRUE，则设备还要创建一个路由表记录。路由表记录中目的地址字段设置为路由请求命令帧的源地址，下一跳字段设置为路由请求命令前一个发送设备的地址，状态字段设置为 ACTIVE。当路由请求定时器计时期满后，设备将把对应的路由请求记录从路由发现表中删除。如果此时目的地址对应的路由表记录中 Status 字段的值仍然是 DISVOVERY_UNDERAY，并且路由发现表没有该目的地址对应的其他记录，则设备还要同时删除该路由表记录。如果路由发现表中存在一个对应于路由请求标识和源地址字段的路由发现表记录，设备将判断当前路由发现过程中得到的路径成本是否小于现存的路由发现表记录中对应的前期成本字段的值。如果当前路由发现得到的路径成本大于现存的路由发现表记录中的前期成本，设备将丢弃该路由请求命令帧，不需再作进一步处理；否则，设备就把路由发现表记录中前期成本和发送设备地址字段的值更新为当前路由发现得到的路径成本值和路由请求命令帧的前一个发送的地址。如果 nwkSymLink 属性值为 TRUE，设备还要把目的地址字段等于路由请求命令帧源地址的所有路由表记录中的下一跳字段更新为路由请求命令帧的前一个发送设备的地址；状态值设为 ACTIVE。最后，设备将调用 MCPS-DATA.request 原语转播路由请求命令帧。

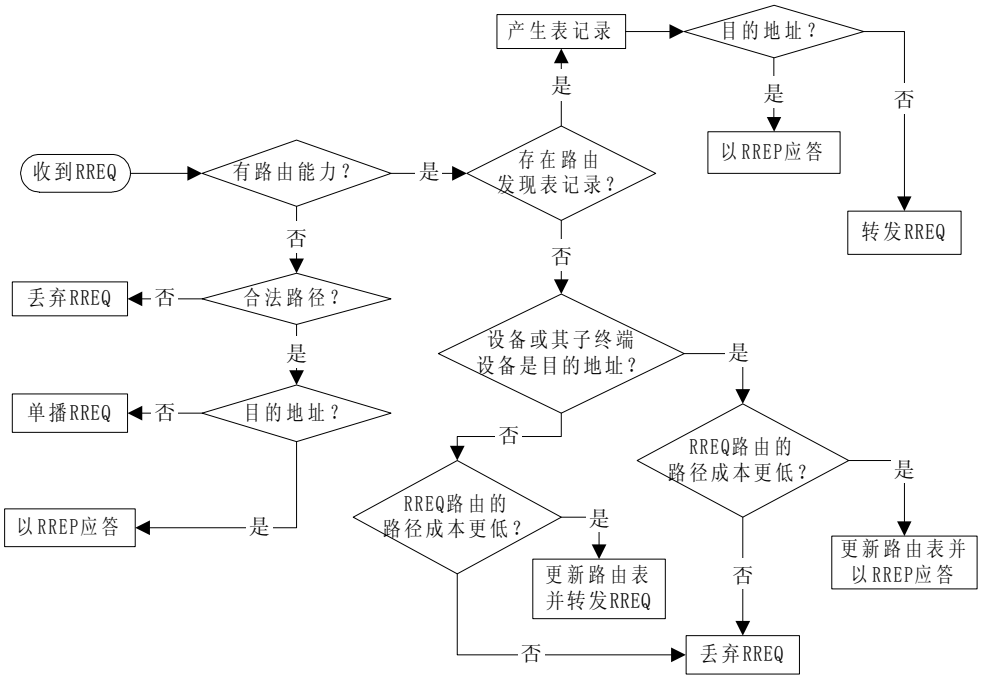
NWK 层重复转播路由请求命令帧时，两次广播之间的随机延时按照下面的公式来计算：

2×R[nwkcMinRREQJitter，nwkcMaxRREQJitter]

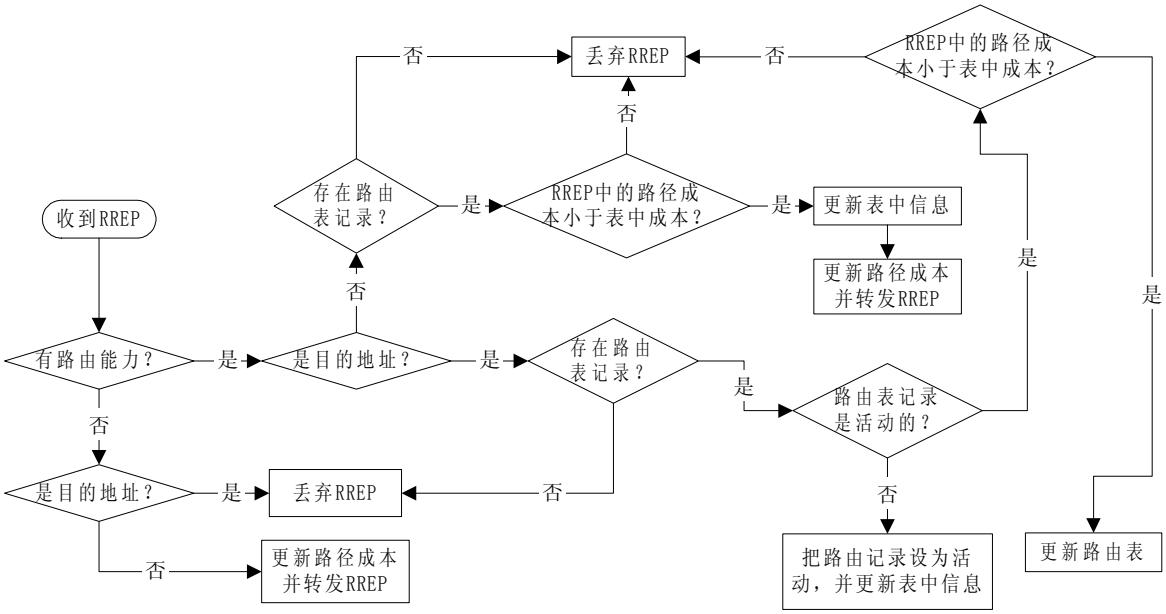
这里 R[a，b]表示在区间[a，b]上的随机函数，时延抖动的单位的 ms。具体实现时，实现者可以调整抖动量时的转播时路由成本大的路由请求命令比路由成本小的路由请求命令延时更大。NWK 层爱第一次转发路由请求命令帧后还要再重复广播 nwkcRREQRetries 次，即最多可以转播 nwkcRREQRetries+1 次受到的路由请求命令帧。NWK 层在重复广播路由请求命

令帧的过程中，如果收到源地址和路由请求标识相同、路径成本更低的路由请求命令帧，则可以选择丢弃正在等待重发的路由请求命令帧。设备同样要把路由请求命令帧有效负载部分目的地址字段的值对应的路由表记录中的状态字段设置为 DISCOVERY_UNDAYWAY；如果这样的路由表记录不存在，则设备要创建一条这样的路由表记录。

当用路由应答命令帧来响应路由请求时，设备将构造一个帧类型字段值为 0x01 的 NWK 命令帧，设备中有一条对应于路由请求源地址和路由请求标识的路由发现表记录。路由应答命令帧头的源地址字段设为当前设备的 16 位网络地址；目的地址字段设为相应的路由发现表记录中发送设备地址字段的值，即路由请求命令帧前一跳设备的地址。路由应答命令帧有效负载部分的 NWK 命令标识字段应设为 0x02，表示路由应答命令；路由请求标识字段的设置与路由请求命令帧中该字段的值相同；发起地址字段的值设为路由请求命令帧头部分的源地址字段的值；路径成本字段的值应设为当前设备到对应的路由发现表记录中发送设备地址字段指定设备的链路成本。构造好路由应答命令帧后，设备就调用 MCPS-DATA.request 原语把它单播发送到最终目的设备（即路由请求的发起设备），而当前发送的下一跳就是路由发现表记录中发送设备地址字段指定的设备。下图就是设备接收到路由请求命令帧时的处理过程。



设备接收到路由应答命令帧时，将按照下图的流程进行处理。



如果接收设备没有路由能力但设备 NIB 属性 nwkUseTreeRouting 的值为 TRUE，则设备将沿树向前转发路由应答。如果接收设备没有路由能力且设备 NIB 属性 nwkUseTreeRouting 的值为 FALSE，设备将丢弃路由应答命令帧。设备在向前转发路由应答命令帧之前要更新有效负载中路径成本字段的值：计算当前设备与下一跳设备之间的链路成本并累加到路由应答命令的路径成本字段中。

如果接收设备具有路由能力，设备将把自身地址与路由应答命令帧中发起地址字段的值进行比较，判断其自身是否是路由应答命令帧的目的设备。如果接收设备是路由应答命令帧的目的设备，它就到路由发现表中查找路由应答命令帧有效负载中路由请求标识对应的记录。如果没有对应的路由发现表记录，设备将丢弃该路由应答命令帧并终止路由应答处理；如果存在这样的路由发现表记录，则设备还要到路由表中查找路由应答命令帧有效负载中应答地址对应的记录。如果没有对应的路由表记录，设备将丢弃该路由应答命令帧并删除路由请求标识对应的路由发现表记录，终止路由应答处理。如果对应的路由表记录和路由发现表记录都存在，但路由表记录中状态字段的值为 DISCOVERY_UNDERWAY，则设备将把路由表记录中的状态修改为 ACTIVE 并把下一跳字段的值设置为路由应答命令帧前一个发送设备的地址；同时，设备还要把路由发现表记录中剩余成本字段设置为路由应答命令帧中路径成本字段的值。如果对应的路由表记录中状态字段的值已经是 ACTIVE，则设备将比较路由应答命令帧中的路径成本和路由发现表记录中的剩余成本。如果路由应答命令帧中的路径成本小于剩余成本，设备将更新路由发现表中的剩余成本字段和路由表中的下一跳字段的值；如果路由应答中的路径成本不小于剩余成本，设备将丢弃该路由应答，不作进一步处理。如果接收路由应答的设备不是目的设备，设备就到路由发现表中查找路由应答命令帧有效负载中发起地址和路由请求标识对应的记录。如果没有这样的路由发现表记录，设备将丢弃该路由应答命令帧；如果存在这样的路由发现表记录，设备将比较路由应答命令帧中的路径成本值和路由发现表记录中剩余成本值。如果路由发现表记录中的剩余成本值小于路由应答命令中的路径成本值，设备就丢弃该路由应答命令帧；否则，设备将查找路由应答中应答地址对应的路由表记录。如果存在路由发现表记录而找不到相应的路由表记录，设备将丢弃路由应

答命令帧。如果找到相应的路由表记录，设备就把路由表记录中下一跳字段更新为路由应答命令帧前一个发送设备的地址，并把路由发现表记录中剩余成本字段的值更新为路由应答命令帧的路径成本值。完成这些路由记录更新后，设备将向目的地址转发路由应答。在转发路由应答之前，设备还要更新路由应答命令帧中的路径成本值。设备找到路由发现表中对应于路由请求标识和源地址的记录，该记录中发送设备地址字段的值即为路由应答的下一跳地址。计算当前设备到下一跳的链路成本并累加到路由应答命令帧的路径成本字段中，NWK 命令帧头部分的目的地址字段设为下一跳地址，然后设备就调用 MCPS-DATA.request 原语把路由请求命令单播发到下一跳设备。MCPS-DATA.request 原语中 DstAddr 参数应设为从路由发现表中得到的下一跳地址。

3.4.5 路由维护

每个设备的 NWK 层针对它需要发送数据帧的每个近邻设备都有一个失败计数器。任何一个发送链路失败计数器的值超过 nwkcRepairThreshold 时，设备就要启动路由修复程序。实现者可以选择一种简单的计算失败的方法来产生失败计数器的值，也可以选择更精确的时间窗方法。需要注意的是，网络中不要过于频繁地启动路由修复，否则就会拥塞网络，影响正常的业务支持。

Mesh 网络中的路由修复。当 mesh 网络中一条链路或一个设备失败时，上行设备将启动路由修复程序。如果由于没有路由能力或其它限制使得上行设备不能启动路由修复，设备将向源设备发送路由错误命令，NWK 命令中的错误代码将指示出链路失败的原因。如果上行设备能够启动路由修复，它将广播路由请求命令来修复路由，路由请求命令中源地址设为失败链路上行设备的地址，目的地址设为传输失败的帧的目的地址。该路由请求命令帧有效负载中名列选项字段的路由修复子域应设为 1，表示这是路由修复的路由请求命令。当一个设备正在修复一个特定目的设备的路由时，它不应向该目的设备发送帧。对路由修复启动时要传送到目的设备的帧和在路由修复完成前又到达的帧，修复设备要么把它们缓存起来，直到路由修复完成，要么丢弃这些帧，具体采取哪种措施根据设备的能力来决定。当一个路由节点接收到路由请求命令帧时，它将根据前面介绍过的路由发现过程来处理。如果该路由节点或它的一个终端子设备是路由请求命令帧的目的地址，那么该路由节点将回应看一个路由应答命令帧。路由应答命令帧有效负载中路由修复子域的值设为 1，表示路由修复应答。如果在 nwkcRouteDiscoveryTime 毫秒内失败链路的上行设备没有收到路由应答命令帧，它将向失败帧的源设备发送一个路由错误命令帧。如果上行设备在规定的时间内收到了路由修复应答命令帧，它将按照新的路由转发那些缓存的数据。接收到路由错误命令帧的源设备如果没有路由能力但 NIB 属性 nwkUseTreeRouting 值等于 TRUE，它将采用分级路由算法沿着树向目的设备单播发送路由请求命令帧；如果源设备具有路由能力，它将启动正常的路由发现过程。如果一个 RFD 类型的终端设备不能向其父设备发送信息，它将启动孤立申明过程。如果终端设备孤立扫描成功并与父设备重新建立通信，那么该终端设备将恢复此前在网络中的操作。如果该终端设备的孤立扫描失败，它将尝试通过新的父设备重新加入网络，此时新的父设备要为该终端设备分配一个新的 16 位网络地址。如果由于临近区域内没有能够继续接受子设备的设备使得终端设备找不到父设备，那么该终端设备将不能重新加入到网络中。此时可能就需要人为干预，使得其重新加入网络。

树状网络中的路由修复。当树状网络中的一个设备与父设备的信标失去同步或不能向父设备发送消息时，它可能启动孤立扫描过程搜索其关联的父设备或启动关联过程寻址一个新的父设备。如果孤立扫描失败或设备重新与一个新的父设备关联，它将从新的父设备收到一个新的 16 位网络地址并恢复此前在网络中的操作。这样，网络同样还是以树状拓扑运行。设备在尝试重新加入网络、得到新的网络地址之前，应使用 MAC 层解关联程序与它所有的子设备解除关联。如果该设备不能访问其子设备，那么它就认为该子设备已经与网络解关联并把该子设备的 16 位网络地址从近邻表中删除，然后设备才重新加入网络，开始以新的网络地址运行。如果一个设备不能向它的子设备发送消息，它将丢弃该消息并向帧的发起设备发送路由错误命令帧，表示消息没有送达目的地址。

4. 应用规范，簇和端点

一个**应用规范**描述了为某一特定应用所使用的一个设备集合，并且隐含地描述了这些设备之间的信息模式。例如，被定义为家居自动化（HA）和智能能源的应用规范。每一个应用被分配一个规范 ID 用以唯一地标识这个应用。

应用规范有两种类型：

- **公共应用规范** — 由 ZigBee 联盟开发的可互操作的应用软件，用以完成某一特定任务。
- **制造商专用规范** — 由某一公司开发的运行在一个 ZigBee 设备上的私有应用规范。

一个应用规范内的所有设备，通过**簇**的方式彼此进行通信。簇可被输入给一个设备，也可从一个设备被输出。例如，在 HA 规范中有一个簇专门负责照明子系统的控制。一个簇 ID 唯一地标识了一个特定规范范畴内的所有簇。

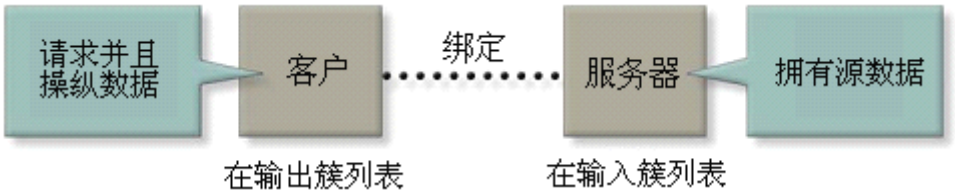
一个**端点**定义了一个设备内的一个通信实体，一个特定应用通过它被执行。例如，一个远程控制可为主卧室的灯光控制分配端点 6，为供暖和空调系统分配端点 8，为控制安全系统分配端点 12。这将允许该远程控制与这些设备单独通信，该远程控制可以识别出数据包是来自哪个应用和设备的。

在任何一个 ZigBee 设备内可使用 240 个端点，端点编号为 1～240。端点 0 被保留给 ZigBee 设备对象（ZDO）使用，它提供控制和管理命令。

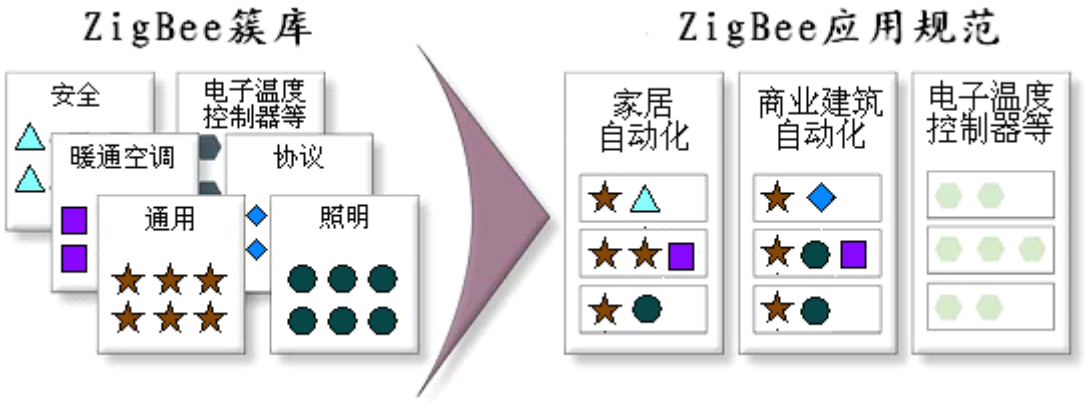
4.1 ZigBee 簇库（ZCL）

ZCL 是一个可被任何应用使用的簇库。这允许公共簇可以在若干不同的功能领域被重复使用。举例来说，同样的照明簇可被需要照明控制的任何应用所使用，例如家居自动化和商业建筑自动化。

每一个簇有两个“端” — 客户和服务器。



ZCL 内的簇被组织到一些不同功能领域，包括照明、HVAC（暖气、通风、空调）、测量和遥感、安全和通用。



每一个簇具体定义

- 强制的和可选的属性
- 簇专用命令
- 功能的描述

每一个设备具体定义

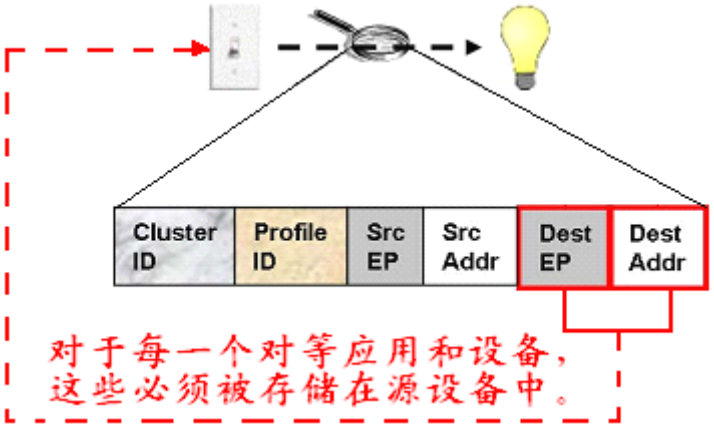
- 强制的和可使用的簇
- ZLC 中“空闲参数”的值
- 任何额外的功能的描述

4.2 绑定

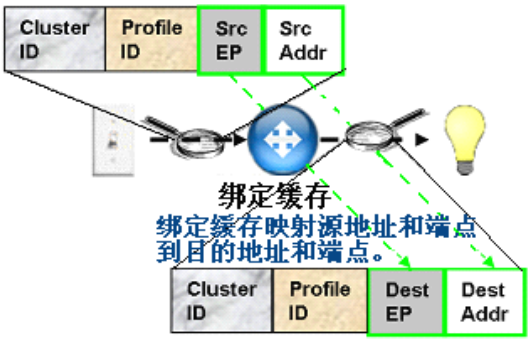
绑定是两个端点之间的连接，每一个绑定支持一个特定应用规范，每一个信息类型由该规范中的一个簇提供。

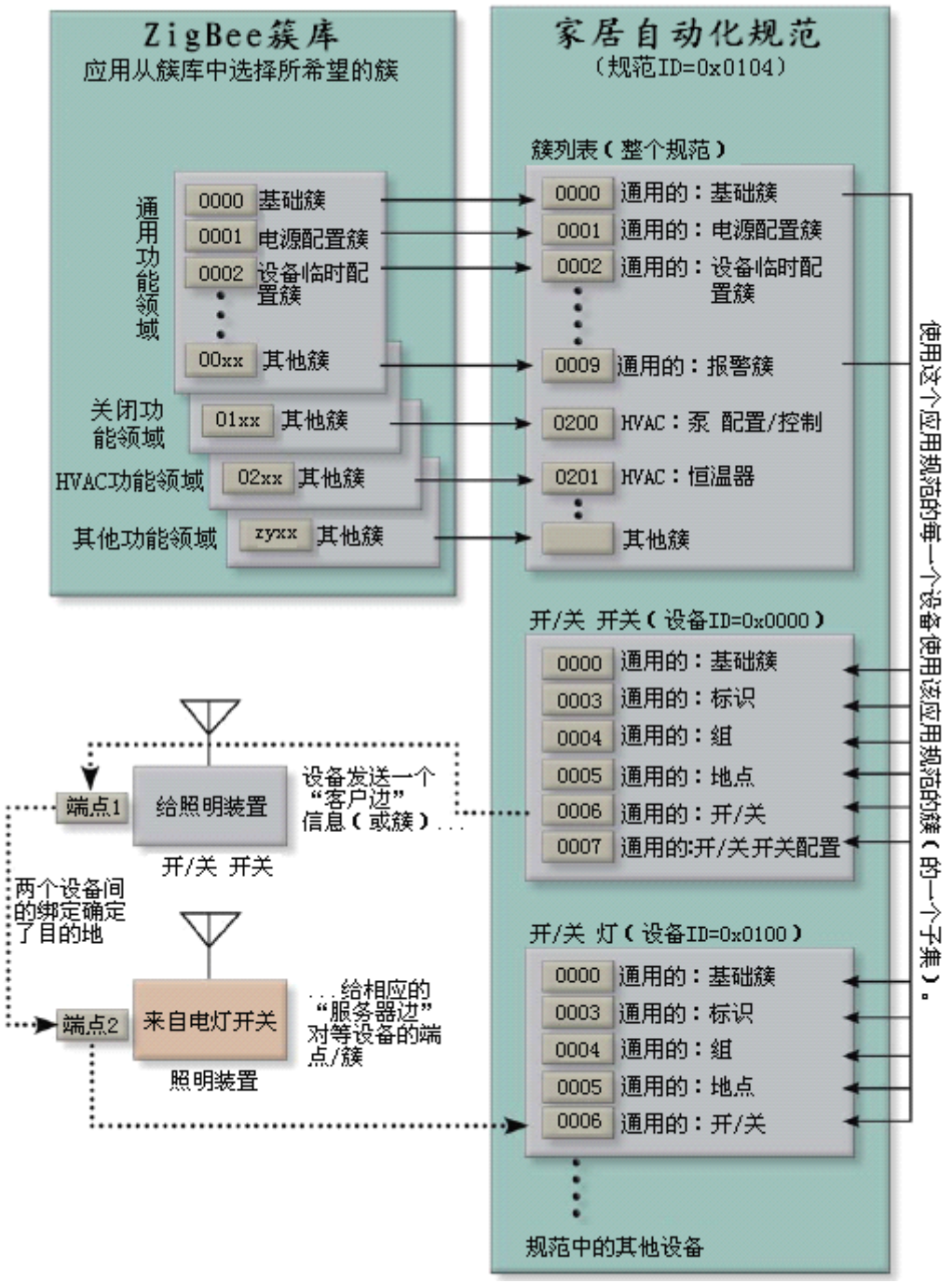
绑定可在具有匹配输入和输出簇（即相同的簇 ID）的个体端点或一组端点之间被创建，例如电灯和开关。ZigBee 设备具有最多 240 个端点，因此每一个物理设备可以支持多个绑定。

绑定可被存储在源设备内，例如，一个远程控制可以存储它需要与之通信的所有应用的地址和端点 ID。这是所谓的直接绑定或源绑定。



绑定信息也可以被存储在一个中间设备的绑定缓存中，它提供了一个查找表来映射所有的源端点和目的端点。这是所谓的间接绑定。


Cluster ID
Profile ID
Src EP
Src Addr
绑定缓存
绑定缓存映射源地址和端点到目的地址和端点。
Cluster ID
Profile ID
Dest EP
Dest Addr


ZigBee簇库
应用从簇库中选择所希望的簇
通用功能领域
0000 基础簇
0001 电源配置簇
0002 设备临时配置簇
00xx 其他簇
关闭功能领域
01xx 其他簇
HVAC功能领域
02xx 其他簇
其他功能领域
zyxx 其他簇
家居自动化规范
(规范ID=0x0104)
簇列表(整个规范)
0000 通用的：基础簇
0001 通用的：电源配置簇
0002 通用的：设备临时配置簇
0009 通用的：报警簇
0200 HVAC：泵 配置/控制
0201 HVAC：恒温器
其他簇
使用这个应用规范的单一个设备使用该应用规范的簇(的一个子集)。
开/关 开关(设备ID=0x0000)
0000 通用的：基础簇
0003 通用的：标识
0004 通用的：组
0005 通用的：地点
0006 通用的：开/关
0007 通用的:开/关开关配置
开/关 灯(设备ID=0x0100)
0000 通用的：基础簇
0003 通用的：标识
0004 通用的：组
0005 通用的：地点
0006 通用的：开/关
规范中的其他设备
端点1
给照明装置
开/关 开关
设备发送一个“客户边”信息(或簇)...
两个设备间的绑定确定了目的地
端点2
来自电灯开关
照明装置
...给相应的“服务器边”对等设备的端点/簇

5. ZigBee 安全

ZigBee 的安全是基于一个 128 位 AES 算法，它被加入到由 IEEE 802.15.4 提供的安全模式中。ZigBee 的安全服务包括密钥建立和传输，设备管理，帧保护。

ZigBee 规范定义了 MAC 层安全，网络层安全和 APS 层安全。应用安全通常由应用规范提供。

5.1 认证中心

认证中心决定是否允许或不允许新的设备加入它的网络。认证中心可周期性地更新并切换到一个新的网络密钥。它首先广播被旧的网络密钥加密后的新的网络密钥。稍后，它通知所有设备切换到新的网络密钥。认证中心通常是网络协调器，但它也可以是一个专用设备，负责如下的安全：

- 认证管理，验证设备加入网络的请求。
- 网络管理，维持并发布网络密钥。
- 配置管理，使能设备间端到端的安全。

5.2 安全密钥

ZigBee 使用三种类型的密钥来管理安全：万能密钥、网络密钥和链接密钥。

5.2.1 万能密钥

这些可选的密钥不是被用来加密帧的。当两个设备进行密钥建立过程（SKKE）来产生链接密钥时，这些可选的万能密钥被用作这两个设备间初始的密码。

源于认证中心的密钥称为认证中心万能密钥，而所有其他的密钥称为应用层万能密钥。

5.2.2 网络密钥

这些密钥提供一个 ZigBee 网络中网络层的安全。一个 ZigBee 网络中的所有设备共享同一个密钥。

高级安全网络密钥必须总是被加密后才能被发送，而标准安全网络密钥可以选择是被加密后发送还是不用加密后发送。注意：高级安全网络密钥只在 ZigBee PRO 中被支持。

5.2.3 链接密钥

这些可选的密钥在两个设备的应用层之间保护单播信息。

源于认证中心的密钥称为认证中心链接密钥，而所有其他的密钥称为应用层链接密钥。

5.3 安全模式

ZigBee PRO 提供了两种不同的安全模式：标准和高级。

特　　性	标准*	高级
网络层安全提供使用一个网络密钥	●	●
APS 层安全提供使用一个链接密钥**	●	●
密钥的集中控制和更新	●	●
从当前激活密钥切换到另一个密钥的能力	●	●
获得两个设备间链接密钥的能力	×	●
实体验证和许可表支持	×	●

* ZigBee 2006 保留

** ZigBee 2006 不支持

5.3.1 标准安全模式

在标准安全模式，设备列表，万能密钥，链接密钥和网络密钥可被认证中心维护也可以由设备自己维护。认证中心仍然负责维护一个网络密钥并控制加入网络的政策。在该模式下，认证中心的存储器需求远少于高级安全模式。

5.3.2 高级安全模式

在高级安全模式，认证中心维护一个设备列表，万能密钥，链接密钥和网络密钥，它需要控制和运行网络密钥更新和网络加入政策。随着网络中设备数量的增长，认证中心的存储器需求将会很大。

6. 试运转

试运转是物理的部署，编址和节点的逻辑绑定以形成一个功能网络。试运行包含很多任务，包括无线电波和物理环境的测量，设备的安置，参数的配置，应用绑定，网络和设备参数的优化以及正确运行的测试和确认。

通常，一些非纯技术上的问题需要被考虑，包括技巧和工作流程，安装实践，设备的简单标识和识别以及接入，与其他共存的无线或有线系统互操作。

对于试运转的考虑通常是把焦点集中在安装和部署上。在开发，测试和制造阶段，容易配置和调试 ZigBee 系统的能力也是同等重要的。

- 在开发和测试过程中，开发者经常必须建立设备的一个系统来进行测试。使用基于标准的通过空中（OTA）的方法快速地调试设备或一个网络可以明显改善生产力。
- 在制造过程中，有可能必须修改设备的参数（也许是为不同的客户群体）来运行基本的制造测试或者甚至对设备的 ZigBee 设置参数进行实际的修改。无需通过固件下载方法就可以修改这些参数给产品生产提供了明显的灵活性。

6.1 试运转工具

在理想状况下，设备应该自我试运转。一个安装者给设备上电，打开开关，然后观察设备。设备自己决定它们应加入哪一个网络，如何在网络中安全地工作，应该绑定到哪一个（哪一些）设备以及它们应该与哪些设备通信。

但在目前的实际情况下，这些工作要由安装者来完成。

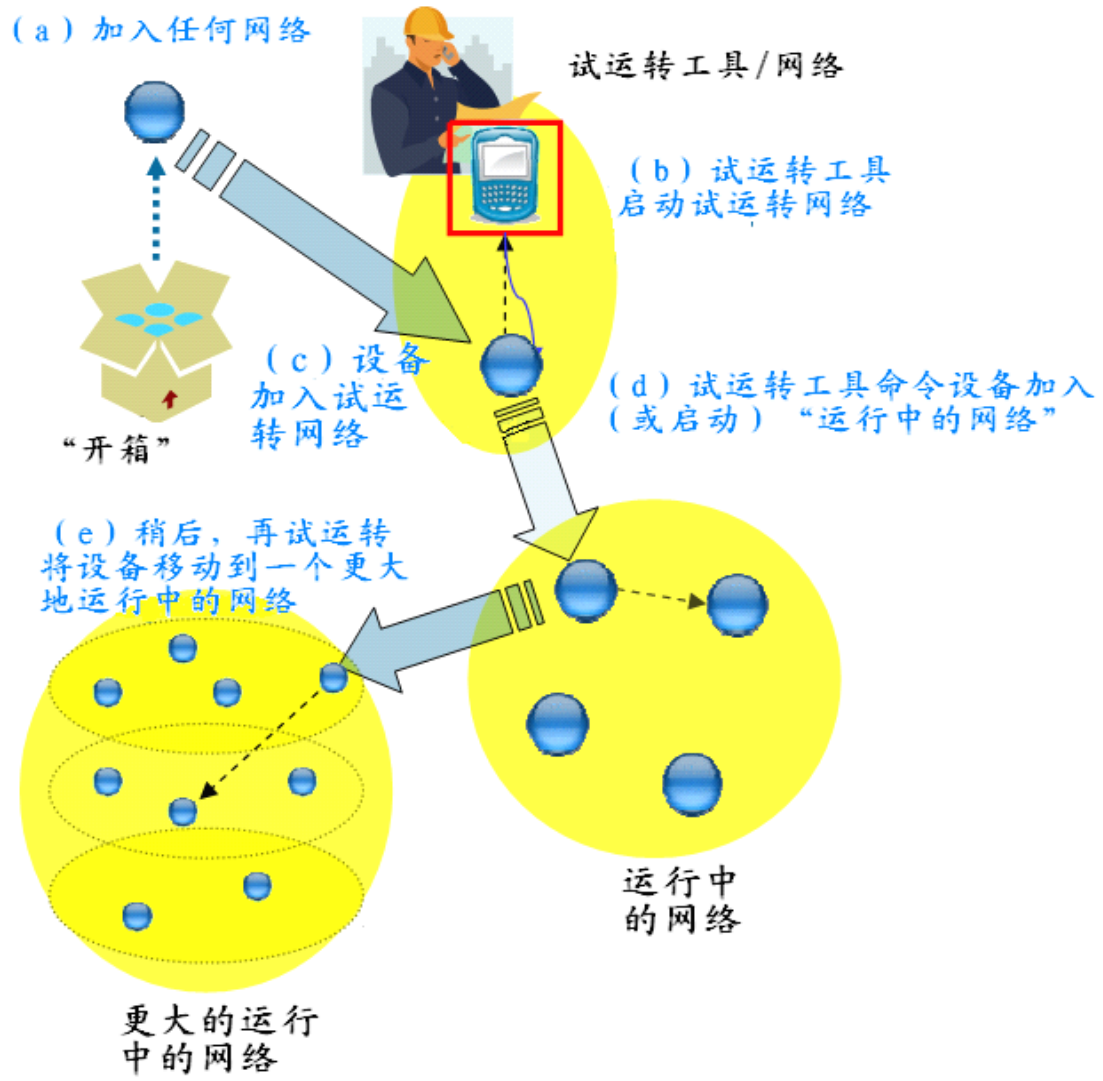
当你看到试运转中所涉及的大量任务时，你可能会意识到使用一个试运转工具的价值。因为安装者喜欢使用一个试运转工具并只需要有限的技术就可以完成这些任务。

这些工具，通常运行在一台膝上型电脑或者个人数字助理（PDA）上。它们提供了一个直观的用户接口，隐藏了下层复杂的技术细节，使得安装者可以可视化地、灵活地配置，调试和管理系统。

试运转工具通常不作为网络后续操作的一部分，它们只是简单地协助试运转、网络后续的维护和管理。

6.2 试运转举例

下面展示了一个典型的试运转情节：



（a）一个新“开箱”设备具有最大的灵活性，它可以加入任何一个网络。

（b）一个安装者使用一个专用工具来启动一个“试运转网络"，通过它设备可被试运转和配置。

（c）新设备加入试运转网络。

（d）安装者使用工具来试运转设备，因此该设备可加入正确的网络，使用正确的安全和绑定。

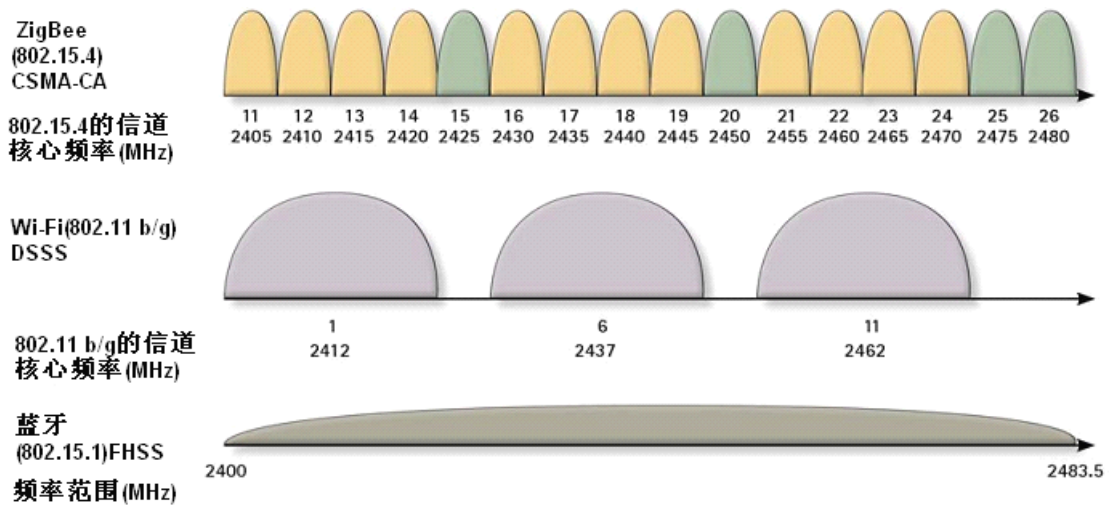
（e）如果需要，已存在的网络可以被试运转，例如，合并几个较小的网络成为一个更大的网络。

7. ZigBee 信道和频率

ZigBee(802.15.4)可用的射频（RF）频谱和信道与 Wi-Fi(802.11 b/g)的重叠。你可以通过选择 ZigBee 信道使用两个相邻 802.11 信道之间的空闲空间以及 25 和 26 号信道来避免冲突 。

7.1 ZigBee，Wi-Fi 和蓝牙的信道

如下图所示，橘黄色表示的 802.15.4 信道与 Wi-Fi 的信道 1，6 和 11 重叠，而灰色表示的 802.15.4 信道与 Wi-Fi 的信道 1，6 和 11 基本不重叠。



信道 1，6 和 11 被建议在美国使用。上图只显示了 2.4GHz 的信道，不同国家有不同的 802.11 b/g 信道。

7.2 信道和频率

信 道 的 逻辑序号	信道编号 （十进制）	信道编号 （十六进制）	频 率 MHz
868MHz 频段			
1	0	0	868.3
915MHz 频段			
1	1	01	906
2	2	02	908
3	3	03	910
4	4	04	912
5	5	05	914
6	6	06	916

7	7	07	918
8	8	08	920
9	9	09	922
10	10	0A	924
2.4GHz 频段			
1	11	0B	2405
2	12	0C	2410
3	13	0D	2415
4	14	0E	2420
5	15	0F	2425
6	16	10	2430
7	17	11	2435
8	18	12	2440
9	19	13	2445
10	20	14	2450
11	21	15	2455
12	22	16	2460
13	23	17	2465
14	24	18	2470
15	25	19	2475
16	26	1A	2480

8. ZigBee 2006 和 PRO

ZigBee 规范的 2006 和 PRO（2007）发行版的详细对比可以访问 www.daintree.net/resources/spec-matrix.php。