Understanding 802.15.4™ and ZigBee™ Networking

Abstract

This paper provides an introductory description of 802.15.4 and ZigBee networking. Each section provides background information on aspects of the communications infrastructure. Topics include network topology, routing and application service transport.

Introduction

802.15.4 and ZigBee are standards-based protocols that provide the network infrastructure required for wireless sensor network applications. 802.15.4 itself defines the physical and MAC layers, whereas ZigBee defines the network and application layers.

For sensor network applications, key design requirements revolve around long battery life, low cost, small footprint and mesh-networking in supporting communication between large numbers of devices. These have driven the specifications for 802.15.4 and ZigBee to deliver a simple yet relatively resilient, large-scale, multi-hop wireless network with the ability to support many different applications.

This whitepaper provides an introduction to 802.15.4 and ZigBee data networking. It covers, in three major areas, networking aspects of network topology, routing and application services.

Network Topology

Network Formation

An 802.15.4 network begins with network formation. A device that is capable of being a coordinator identifies a channel that is relatively free of interference through energy scans, and then sets itself up to be a coordinator. In ZigBee, this is called the PAN coordinator. Subsequent to this, devices that wish to join the network will do so by first issuing beacon requests to solicit beacons from devices that could potentially allow them to join the network. Initially, only the PAN coordinator will respond. In addition to the PAN coordinator, any device capable of allowing other devices to join the network is a ZigBee router. What ensues is a series of message exchanges that will determine whether a device may join the network. In 802.15.4, this process is called "association". A key factor in such a determination is a router's capacity to accept additional devices as its child.

Where 802.15.4 differs from other wireless technologies, such as 802.11/Wi-Fi, is in permitting a hierarchy of associations, rather than a single parent-children structure. For instance, a device joining the coordinator could itself be a router, and that device could permit other devices to join it. As a result, multiple levels of associations can be achieved.

A small network is shown in Figure 1. In this case, the PAN coordinator (which always has network address "0000") has three devices associated to it, and of these, two of them (with network addresses "0001" and "071e"), acting as routers, have a device associated to them.

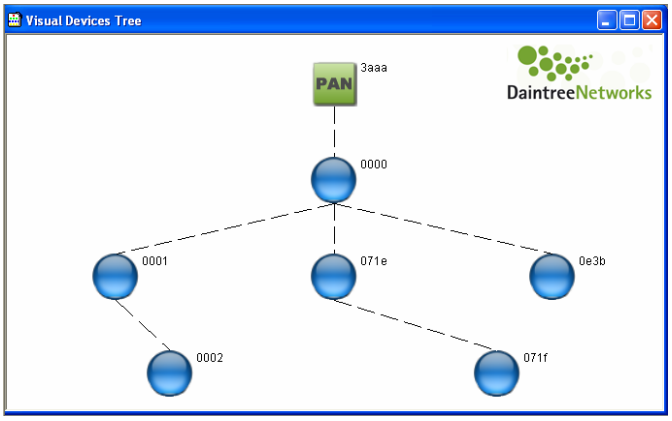

Figure 1 Network Topology

Such a hierarchy of devices serves to provide true wireless networking. In contrast, 802.11/Wi-Fi offers what is fundamentally a wireless access point technology – a wireless connection is made between the laptop, say, and the wireless router. Thereafter, communication is made on wires (for instance, on a wired-LAN to other wired devices or another 802.11 wireless router). 802.15.4 is able to deliver packets completely in the wireless domain across multiple hops, for instance, from a child to its parent, and then its parent to its parent, and so forth.

The Stack Profile

ZigBee offers a stack profile which defines network, application services and security parameters for the entire network. The stack profile is selected by the ZigBee coordinator, and is chosen on the basis of application areas, such as Home Controls, Building Automation or Plant Control. Each such stack profile will then define parameters such as the overall dimensions of the network, supported routing algorithms, routing and application binding table sizes and the nature of the security services for this network.

Of particular interest to the current discussion on the network topology is the specification of three parameters – the maximum depth of the network (nwkMaxDepth), the maximum number of children per router (nwkMaxChildren) and the maximum of children routers per router (nwkMaxRouters). Collectively, these will determine the overall structure of the network.

Addressing

All 802.15.4 devices have a 64-bit (long) IEEE address, which uniquely identifies the device. In order to extend battery life, shorter addresses are used to shorten the packet sizes and hence the time a device is actively communicating. ZigBee requires that all communications after joining the network be made on a 16-bit (short) network address. The 16-bit network address of a newly joining device is assigned by its parent during the association message exchange mentioned earlier.

ZigBee specifies an algorithm (often called "Cskip") that provides address ranges to routers and coordinators, to be assigned to joining devices based on their location within the network hierarchy. While these address ranges vary depending on the Stack Profile parameters mentioned earlier, the address ranges are always the same for a given set of Stack Profile parameters. For instance, the network addresses allocated in Figure 1 reflect the Stack Profile parameter value of nwkMaxDepth = 7, nwkMaxChildren = 5 and nwkMaxRouters = 3. For these same three parameter values, the first router that joins the coordinator will always be allocated network address 0001, the second 071e, and so forth. However, example networks used later (for instance, in Figure 8) involved different Stack Profile parameter values, resulting in different allocated network addresses.

Types of Networks

The exact structure of the ZigBee network cannot be pre-determined in most situations due to variable dependencies such as location of devices and radio propagation during network formation. However, the three stack profile parameters mentioned earlier will dictate the rough overall structure of the network.

For instance, if the application has devices physically placed in a linear fashion (such sensors placed on a pipeline, or a single transceiver per house along a street), setting the nwkMaxChildren and nwkMaxRouters to 1 will create a linear structure as shown in Figure 2. Similarly, a star network can be achieved by setting nwkMaxDepth to 1 and nwkMaxRouters to 0. In between, where many networks are likely to fall, are breadth and depth values that provide customizable network structures, on a per-application basis.

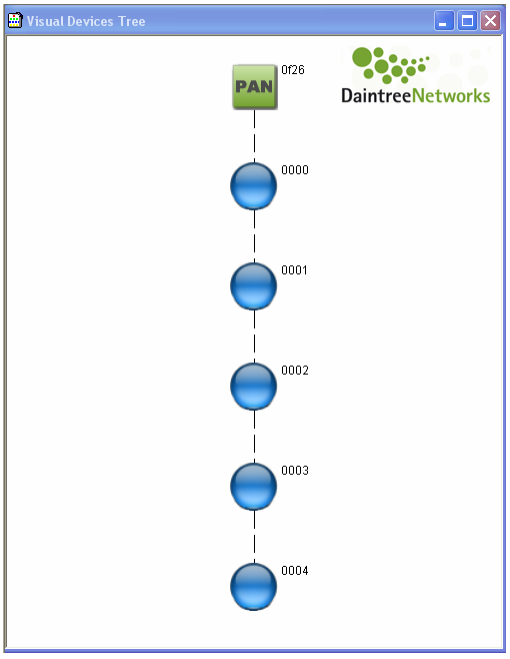

Figure 2 Network Topology (Linear)

Network Dynamics

In addition to joining a network, other activities could cause a change to the network structure. Devices could, for instance, leave (or be forced to leave) a

network, or potentially re-associate elsewhere within the network (for instance, if the device restarts).

Figure 3 demonstrates an example of re-association. In this example, the device with a short address of "0e3b" re-joins the network as "097d" and subsequently as "0260". At each stage, this device joins a new parent, which allocates an address from the range of addresses it has at its disposal.

When one combines a large network of devices, with ongoing dynamics of devices associating, re-associating and disassociating to the network, tracking devices and the network topology can become a challenge.

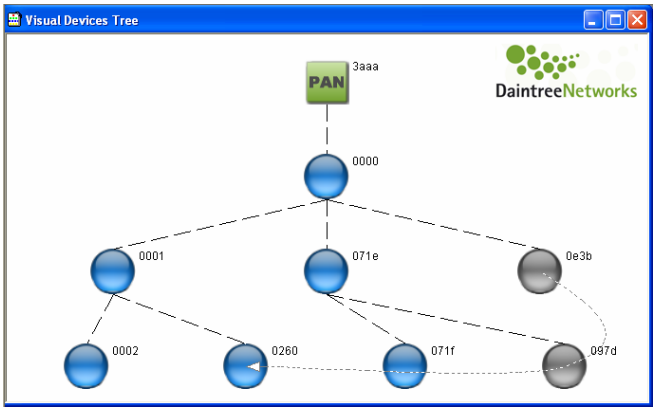

Figure 3 Re-associations Causing Topology Changes

Routing

Tree Routing

In addition to managing network joining and address assignment, the network formation process also provides a routing algorithm called "tree routing". The "Cskip" address allocation algorithm provides address ranges that allows any device to quickly identify whether a particular network address belongs to a descendant of that device (and through which of its children), or elsewhere in the device tree. As a result, any device could make simple routing decisions based on passing a packet up or down the device tree.

Continuing with the network introduced earlier, a tree route is shown in Figure 4. In this case, a packet is generated by device "071f" and is destined for "0002". Using tree routing, this packet is transmitted unicast from "071f" to its parent "071e" and to its parent "0000", the coordinator, and then down the tree through "0001", and ultimately arriving at "0002".

The most significant benefit with tree routing is its simplicity and its limited use of resources. By having a simple algorithm to determine whether an address is a child or a descendant of a child, or elsewhere on the tree, any router can make a routing decision simply by looking at the destination address. In these cases, a router simply decides to route a packet to one of its children or to its parent. As a result, precious memory resources need not be used to store routing information. Hence, very low cost devices can be deployed without routing capability, but can still participate in any ZigBee compliant network.

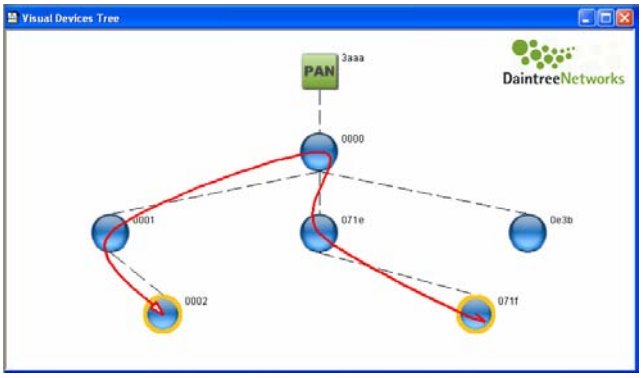

Figure 4 Tree Routing

Route Discovery (AODV)

Although tree routing is simple and uses very little resource, it can be quite inefficient. Depending on how the network forms, two devices, in close proximity could join the network on remote branches of that network, resulting in packets between these two devices traversing through many hops, when a single hop transmission between them would have been possible.

Figure 5 shows both tree routing and route discovery, a routing algorithm used in ZigBee. Tree routing was originally used, and the packet flow from device "0002" to device "071f" was transmitted up and down the tree (through "0001", "0000" and "071e"). Subsequently, route discovery was initiated; device "071f" was able to respond to the route request message sent by device "0002" and a direct route was established.

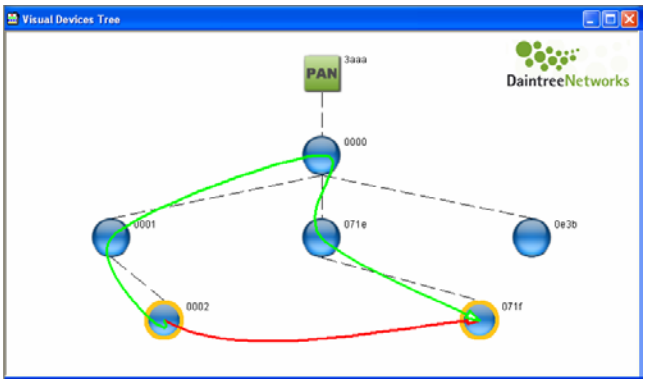

Figure 5 Tree Routing and Route Discovery

In addition to providing for more efficient transmission, route discovery also provides greater resiliency. When a device fails or is temporarily unavailable to forward packets, all its descendants on the tree will be isolated from the rest of the tree if only tree routing is used. For instance, in Figure 5, if device "0001" failed, device "0002" would not be able to communicate with the rest of the network (and vice-versa). However, with route discovery, a direct route from "0002" to "071f" (or elsewhere on the tree) could be found. Route discovery provides the means to achieve mesh networking, allowing all devices on the network to communicate to all other devices using routes that are updated dynamically.

So, how does route discovery work? Route discovery uses the Ad hoc On Demand Distance Vector (AODV) routing algorithm. This algorithm is specified in the Internet Engineering Task Force (IETF) as an experimental RFC (3561) and has been adapted for use within ZigBee networks. The basic algorithm works in the following way – when device S sends a packet to device D, device S can initiate the route discovery process by issuing a broadcast message, called a Route Request, to request a route to device D. Any device which received that Route Request would also propagate the request on by sending another Route Request message. Eventually, a device that receives a Route Request is able to reach device D (for instance, if device D is in its neighbor table, or perhaps the device is D itself), and would respond with a Route Reply to the device it received the Route Reply from. That Route Reply will be propagated back along the original Route Request path that was taken until it arrives back at device S. This series of back-propagated Route Replies forms the forward routing path for future communication of packets from S to D. Route Reply messages also include a cost value which accumulates as the Route Reply messages propagate back to source. The cost mechanism allows the source, and nodes along the path, to select the lowest cost path to the destination, in the case where multiple nodes respond to the original Route Request.

Route Discovery in Greater Detail

In this more in-depth look at Route Discovery, we use a specific example to demonstrate route discovery. Consider again an attempt to discover a route from device S to device D.

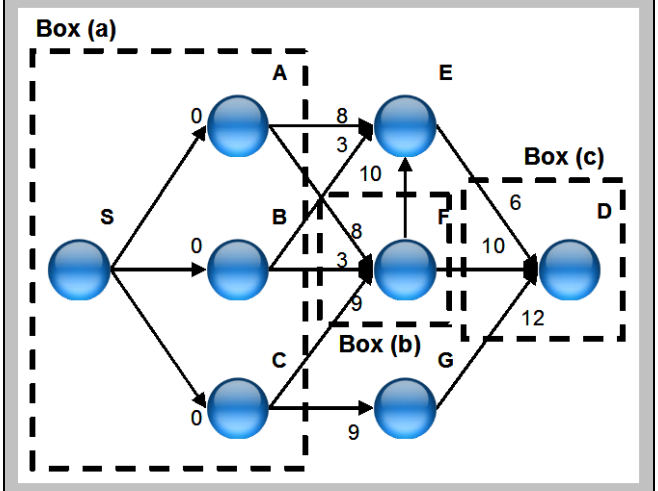

Figure 6 Route Request

Using Figure 6, we observe that between these two devices are ZigBee Routers A, B, C, E, F and G. Looking at box (a) on the left of Figure 6, we observe that device S initially broadcasts the Route Request with a path cost of 0 (the default setting for the initial transmission), represented by the number at the head of the arrow. Devices A, B and C detected this initial broadcast, and then broadcast the Route Request after adding an additional cost, based on the received signal strength from device S. For instance, device C adds to its own broadcast, a value of 9 to the cost for a total of 9, based on the received signal strength.

We observe in box (b) in the middle of Figure 6, that three Route Requests were observed by device F. Device F then adds the cost based on received signal strength, selects the lowest cost of these and broadcasts the Route Request. Device F also stores the device that provided the lowest cost route to it (after adding the cost from the received signal

strength). If the Route Reply returns through F, F will then send its Route Reply to this device.

In box (c), device D then adds the cost based on received signal strength of all incoming Route Requests, observes the lowest path cost of each of these, and then selects the route with the lowest cost. For this example, let us assume that the lowest cost route came from E. This will be the selected route, and device D will then send a Route Response back through device E, which then propagates the Route Response back along its identified lowest cost route, ultimately yielding the Route Reply path shown in Figure 7. Thus, the forward route will be S-B-E-D.

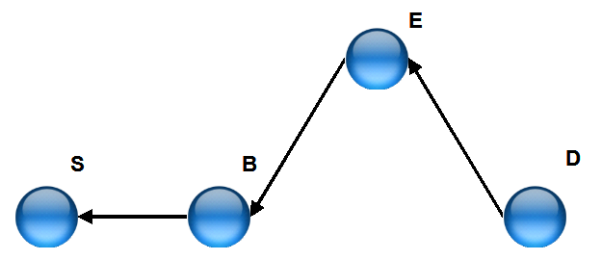

Figure 7 Route Response

Application Service Transport

The end-goal for 802.15.4 and ZigBee networking is to enable the provisioning of application services. Examples are wide-ranging, from home control to building and industrial automation.

To explain the basics of how application services are offered, consider a specific example – a home remote control that provides the means for turning on the lights, controlling the heating system, or displaying the current temperature from thermostats. Each type of service is uniquely identified by a profile ID – for instance, Home Control Lighting (HCL) has been allocated profile ID 0x0100. Within each profile are specific features or controls – these are identified by a cluster ID. Continuing with the HCL example, cluster ID 0x01 is provides ON/OFF control for Dimmer Remote Controls.

The following sections provide an overview on how such application services are transported on ZigBee networks. In particular, we will examine how ZigBee's Application Support Sub-Layer (ASSL) offers transport-layer connectivity for applications, akin to Transmission Control Protocol (TCP) in IP networks (although without the flow control features that TCP offers).

To achieve this, the ASSL defines endpoints, which are equivalent to TCP ports, and it is on these endpoints that application services are carried. Returning to the HCL example above, endpoint 1 on a remote control could be associated with profile 0x0100 (HCL). By doing this, endpoint 1 on the remote control will provide the means to control a specific light globe. A corresponding endpoint (which does not have to be endpoint 1) is assigned on the light globe itself and the communication between these two endpoints using cluster ID 0x01 provides the means for turning that light on or off. Other services such as thermostat temperature settings would be provided on other endpoints, allowing the same remote control to control other devices.

The application layer itself contains a substantial number of features that involve application profile-based control, device management and security. These are beyond the scope of this document, which focuses largely on transport and networking issues.

ZigBee Endpoints

Every ZigBee node or device can contain up to 241 endpoints. An endpoint is analogous to a TCP port – each endpoint can provide a distinct application service. Endpoint 0 itself is reserved for device management, and is called the ZigBee Device Object (ZDO). Endpoint 0 provides the means to control and manage devices and endpoints. The remaining 240 endpoints can be allocated for any required services.

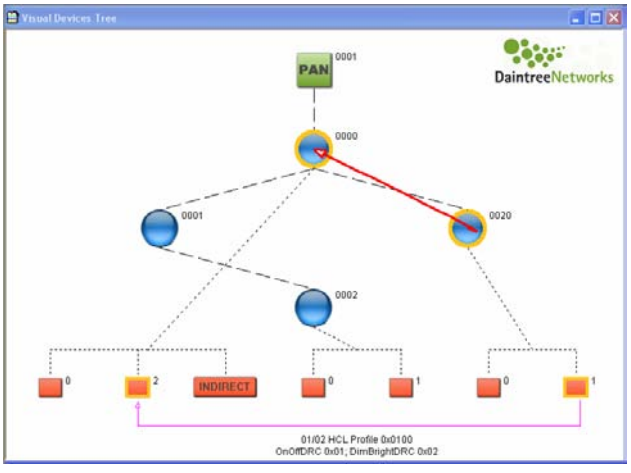

Figure 8 ZigBee Application Endpoints

Consider the example presented in Figure 8. In this example, three devices were identified as having active endpoints – the endpoints are represented by squares at the bottom of the window. All three have active endpoint 0, or ZDO. They have all been identified as also having active endpoint 1 or 2. In this diagram, a message exchange between endpoint 1 of device "0020" and endpoint 2 of device "0000" was detected. These were Home Control Lighting (HCL) application profile transactions, involving turning lights on and off. The details of this message exchange are beyond the scope of this document.

End Device Binding

The message exchange shown in Figure 8 requires device "0020" to store information about the device it needs to communicate to, such as the target device's address and its endpoint number, and a few other pieces of information. The storage of this kind of information may require more RAM than is available in device "0020". This issue is amplified in cases where a single source endpoint may need to communicate to multiple destination endpoints.

ZigBee offers an alternative solution to deal with this issue. Rather than require all devices to store information about their target device, PAN coordinators may store that information on behalf of end devices. The process is called Indirect Addressing. In this case the coordinator maintains a binding table on behalf of the end devices. When indirect addressing is used, the source device will forward its transactions to the coordinator rather than sending packets directly to the destination. Upon receipt of an indirectly addressed packet the coordinator will look up the source endpoint in the binding table, and then forward the packets on to the required destination endpoint/s. The binding table can be populated based on an explicit request to the coordinator, known as a bind request, or using a process known as End Device Binding.

End Device Binding is a process where devices advertise to the coordinator that they would like to be bound. The end device bind request is typically sent based on some user driven event such as a button press. The most common application of end device binding is the installation of a light and corresponding switch in a Home Control Lighting installation. During installation, an engineer will press a button on the light and the switch simultaneously. If the end device bind requests from each device reach the coordinator within a pre-specific period of time of each other, then the coordinator will form a binding and enter that binding in the binding table.

The End Device Bind process is demonstrated in Figure 9. We observe device "0020" and device "0002" initiating an End Device Bind Request to the PAN coordinator. After matching the requests and populating the binding table, the coordinator subsequently responds with an End Device Bind Response to each device. These exchanges occur on endpoint 0, or ZDO, which, as mentioned earlier, provides the means to achieve device management. Figure 9 demonstrates the End Device Bind request and response messages. Figure 10 shown the complete set of transactions required to complete the End Device Bind process.

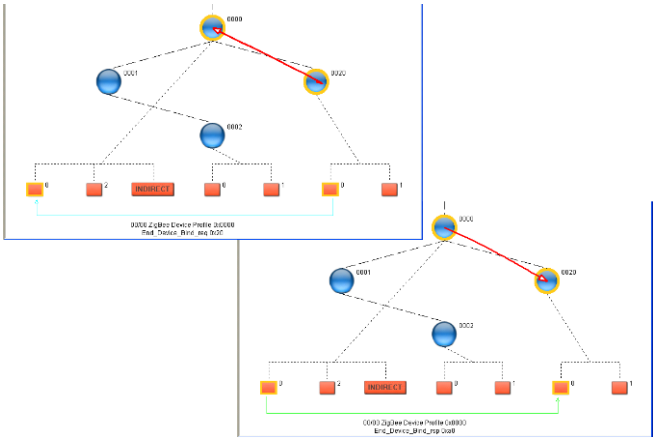

Figure 9 End Device Binding

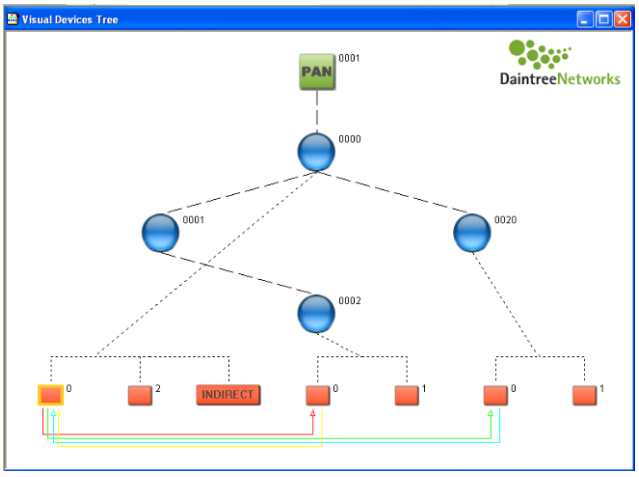

Figure 10 End Device Binding – All Transactions

The result of these exchanges is the creation of a binding between endpoint 01 on device "0002" and endpoint 1 on device "0020". Subsequent to this, packets need only be generated by device "0002" from endpoint 1 to the PAN coordinator. The PAN coordinator then indirectly sends that message on to endpoint 1 on device "0020", as shown in Figure 11. By doing this, device "0002" does not need to store and remember its target device, reducing the resource requirements for that device.

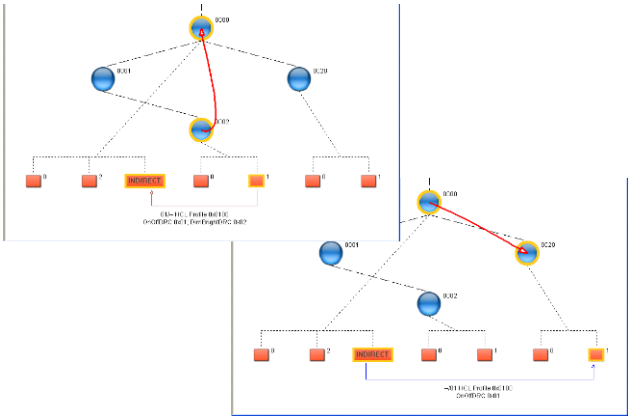

Figure 11 Indirect Addressing Following End Device Bind

Concluding Remarks

This whitepaper has provided an introduction into the key networking and transport mechanisms employed by 802.15.4 and ZigBee networks. Examples of screenshots from captures of live networks were used to demonstrate network formation, routing and application transport.

While most of the explanations provided in this whitepaper revolve around simple and small networks to enable clarity of explanation, wireless sensor networks are more likely to involve much larger networks. An example of an 18-node network is shown in Figure 12. Here, a combination of re-associations, 5 routes, and of course, network formation is presented.

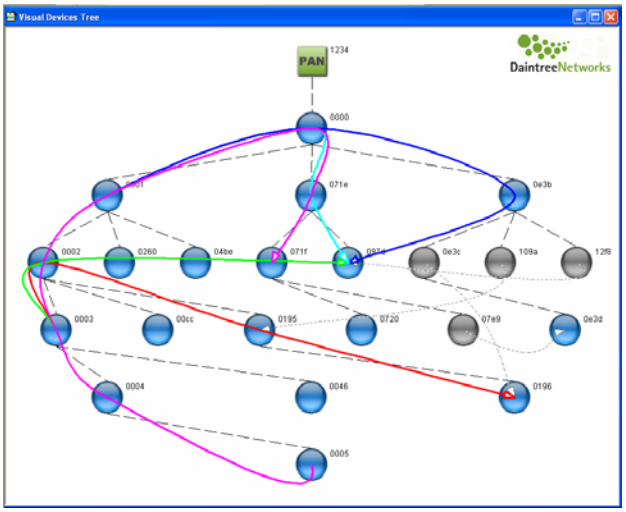

Figure 12 Large Network with Routing

About Daintree Networks

Based in Mountain View, California, Daintree Networks is a clean technology company that provides wireless control solutions for commercial buildings. Daintree has a strong background in wireless sensor and control mesh networking, with extensive knowledge and experience gained through its industry-standard design verification and operational support tool, the Sensor Network Analyzer (SNA). In addition to wireless embedded expertise, Daintree has put together a team of seasoned professionals from the lighting, telecommunications and networking worlds. Daintree's expertise and knowledge is now being focused on the development of cost-effective building automation systems. These provide benefits including reduced energy consumption, costs and carbon footprint, compliance with new "green" building regulations, and cost savings available through government rebates and the ability to take advantage of demand response programs.

Daintree's Wireless Lighting Control Solution (WLCS) allows lighting manufacturers to speed their time to market, and enables them to deliver powerful, comprehensive, flexible, and reliable wireless lighting control systems for commercial buildings. For more information, visit www.daintree.net or email sales@daintree.net

Daintree Networks Inc
1503 Grant Road, Suite 202
Mountain View, CA 94040 U.S.A

(w) www.daintree.net
(e) sales@daintree.net
(p) +1 (650) 965-3454