Embedded in Thin Slices

Bluetooth Mesh (Part 1)

Alternatives Compared

Wireless mesh networks are being widely deployed in a variety of settings. In this article, Bob begins his series on Bluetooth mesh. He starts with defining what a mesh network is, then looks at two alternatives available to you as embedded systems designers.

By
Bob Japenga

In 2003, our company designed an energy monitor to be used in a load shedding application for electrical customers who were charged for peak usage. The target customers were universities and municipalities, which had multiple sites but a single electric bill that charged them for peak usage as a single user. This common billing paradigm made it important to know the energy usage across multiple sites—some upward to a mile away. Communication between the nodes was done using a proprietary radio to a central hub that passed the data up to the central office where the load shedding algorithms were performed. We weren't involved with the radio portion during this initial stage.

After a few years, the relationship between our customer and the radio designer went south and they came to us to design the radio. Initially we thought it would be a piece of cake. Then we discovered that the radios used a sophisticated mesh network that auto-configured the routes, minimized the hops from node to central hub and self-healed when nodes dropped out. We got gun shy about the complexity of the design as well as our ability to test the vast array of possible scenarios that can take place in a dynamic mesh network. We decided to redesign without changing the radio software at all. But we got our first taste of a mesh network and learned a little about what we didn't know.

A few years later, we got involved in a smart-grid solar project from the ground up. One of the requirements was to provide a ZigBee interface to other devices connected to the grid. So, again, we dipped our toes into the wireless mesh network waters. But it became very clear that we were complete novices about these networks and had a lot to learn.

Over the next few articles, I want to introduce a very popular and potentially powerful entry into wireless mesh networking. In July of 2017, the Bluetooth SIG introduced Bluetooth mesh, which promised to revolutionize IoT. This month I want to introduce two of the competitors to Bluetooth mesh—ZigBee and Thread—and highlight some of the distinctions. These three network protocols are often compared since they all can use a common radio interface and are often implemented on the same chip (**Figure 1**). Although Wi-Fi mesh networks will play a major role in the future of IoT, we won't be comparing it to these three because of its power consumption. In the next set of articles we will drill down into some of the details of deploying a Bluetooth mesh network.

DEFINITION OF TERMS

Node: This is a specific device on the wireless mesh network that can send and receive data on the network.

Hop: As a network data travels from the source to its destination, it traverses between one or more nodes. The transmission between these nodes is called a hop.

Relay: When a node is transmitting some other node's data, it is a relay. (Some notations call this a router.)

Wireless Mesh Network: A wireless mesh network uses a topology where the nodes in the network work together to move the data from the source to the destination. The source RF signal does not need to be received by the destination node for the data from the source to be received by the destination. **Figure 2** shows a simple wireless mesh network. Node A's RF signal cannot be received by Nodes C, D or E. Node B's RF signal cannot be received by Node's D or E. But data from Node A can be sent to Node E with the help of Nodes B, C and D. Transmission from Node A to Node E is a four-hop transmission.

Flooding/Routing: There are two basic mechanisms for a wireless mesh network to get data from the source node to the destination node. One mechanism is called flooding. The source node does not know who is going to relay the data to get the data to the destination. All nodes configured as relays in the mesh that receive the data can forward it on (also by flooding). The other mechanism is called dynamic routing where the source node has a routing table to indicate one or more paths it can take to get its data to its destination. Bluetooth uses what is called managed flooding (more on that in a later article). Thread and ZigBee use dynamic routing tables which self-configure.

OOB Authentication: Out of Band (OOB) authentication and key exchange is where an alternate means of authenticating and exchanging keys happens outside the normal "band" of communications. For example, requiring the user to enter a passcode is an OOB method. Pressing an input on a node device X times could be used as an OOB method. Passing keys through a Near Field Communications (NFC) channel is another. Many of the ICs that are offered with Bluetooth Mesh, Thread and ZigBee built in also offer NFC capability.

TRADE-OFFS AMONG THE THREE

Let's look at some of the trade-offs when choosing a wireless mesh network.

Network Performance: Silicon Labs, which makes parts that support all three mesh networks, released a report documenting a 12-month long study of the network performance of the three mesh networks [1]. The Silicon Labs benchmark showed very little differences between the three protocols on small networks (less than 24 nodes) and small payloads (less than 10 bytes). Thread and ZigBee outperform Bluetooth in both throughput and latency as either the network grows or the payload increases in size. So, if latency and/or larger payloads are important you, may want to stay away from Bluetooth.

Routing/Flooding: This could be a deal breaker for some of your designs. The Bluetooth standard left open the door for

FIGURE 1
The three network protocols—Bluetooth mesh, ZigBee and Thread—are often compared since they all can use a common radio interface and are often implemented on the same chip.

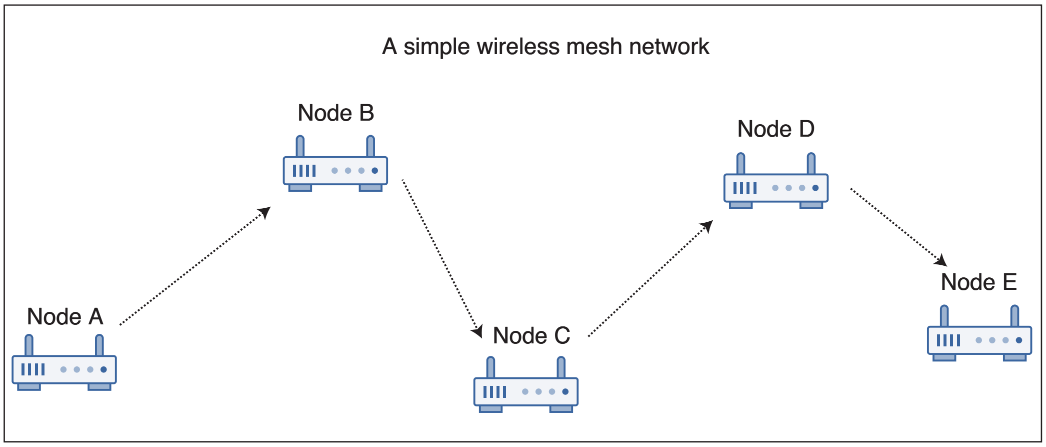

FIGURE 2
Shown here is a simple wireless mesh network. Node A's RF signal cannot be received by Nodes C, D or E. Node B's RF signal cannot be received by Node's D or E. But data from Node A can be sent to Node E with the help of Nodes B, C and D. Transmission from Node A to Node E is a four-hop transmission.

using dynamic routing, but for now doesn't support it. Thread and ZigBee support dynamic routing. Dynamic routing can be a serious problem for mobile nodes since the network could be saturated with attempts to dynamically reconfigure the system as the optimal routes keep changing. The algorithms that scared us away 15 years ago are complex and probably not foolproof especially if the nodes move around. Flooding creates problems if you have to dynamically reduce the number of relays in the system or reduce the radio power because you have too many nodes talking at the same time. For example, if you were selling a system with 150 devices spread out such that at full power and with all nodes configured as relays, most nodes could not hear each other, flooding would work fine. But what if another customer placed your devices such that all 150 nodes could hear each other? The scheme could break down. One Bluetooth mesh software development kit (SDK) says:

> The flooding-based approach to message relaying can cause a lot of redundant traffic on air, which may impact the throughput and reliability of the network. Therefore, it is highly recommended to limit the number of relays in a network to restrict this effect. The rate of relay-enabled devices in the network is a trade-off between message route-redundancy and reliability. It should be tuned according to network density, traffic volumes, network layout, and requirements for reliability and responsiveness [2].

If you are creating large onesie networks and can hand tweak the relays and radio power, or if you're creating a system with a small number of nodes (less than 20), managed flooding is not an issue.

Power Usage: Each wireless mesh network allows you to minimize power with different schemes. Each one has its own advantages and disadvantages that you the designer must understand if power is an issue. With ZigBee, power can be minimized because transmit and receive are synched allowing devices to all sleep at the same time. Therefore, the ZigBee radios can be off most of the time. At the appropriate time slot, all radios wake up and the network is alive. Bluetooth implements a friend feature, which allows the designer to have some devices that do not have power constraints to store messages for its sleeping friends. When the power constrained friend wakes up, it asks its friend for all of its messages. Thread enables the designer to designate some devices as sleepy end nodes that need not be awake for relaying and can keep the radio off until needed.

Maturity: ZigBee has been around since 2003. Thread was first released in 2015 and Bluetooth Mesh in July 2017. For those of you who have experience with the SDKs of major suppliers, you know that it takes time to work out the bugs in these complex software packages. Our experience bears this out with these three types of mesh SDKs. Be prepared to provide more hand holding the newer the technology.

SETTING UP THE NETWORK

Each of these wireless mesh network protocols is very different in how you set up the network. Each offers different options, which should be carefully considered when you choose one protocol over another. There are security trade-offs made with each method. All three provide secure encrypted transmissions. The challenge has to do with jump starting the process. It would be ideal if all this could be done at a secure factory where the network is setup, nodes authenticated and keys exchanged. But most of us don't sell systems like that. Let me try to provide the basics. But I want to emphasize that the devil is in the details of this process.

Bluetooth Setup: To securely install a node on a Bluetooth mesh network, one node must be a provisioner. The provisioner has two primary tasks: 1) Establish a secure link over which the keys can be exchanged to allow the node to talk on the network. 2) Authenticate that the new node is one that you want to add to the network. There are several ways that the specification allows this to happen. To create a secure process is complicated when designing headless nodes (when no OOB methods are available). We will look at this more in depth in our next article.

Thread Setup: Because each device is an IPV6 address on the Internet, installing a Thread node can be done with a smartphone, a tablet or PC. Although simple for a home device (the supporting organization touts Thread for use in home automation), the IoT designer needs to fully understand how this will be done if you are installing hundreds of nodes on a factory floor.

ZigBee Setup: Out of the box, with one

For detailed article references and additional resources go to:
www.circuitcellar.com/article-materials
References [1] through [3] as marked in the article can be found there.

RESOURCES
Nordic Semiconductor | www.nordicsemi.com
Silicon Labs | www.silabs.com

device designated as a ZigBee coordinator, ZigBee nodes can be easily added and can instantly create a mesh network transmitting encrypted data. The problem is the way it is done right out of the box. They use a default Trust Center link key for encryption (published widely on the web). If you use that method in your design, a hacker could easily decrypt the first transmission with this key and decrypt the transmission with the secret keys without any trouble. ZigBee provides the necessary tools to setup the network securely, but they require some OOB method.

SECURITY, RANGE AND MORE

Security: If you can accept the methods prescribed by Bluetooth and Thread, they are more secure (and more complicated) than ZigBee. But Bluetooth and Thread have also not been scrutinized as long. Even the Bluetooth security vulnerability uncovered in July 2018 (Cert Vulnerability #304725 [3]) would not affect a Bluetooth mesh implementation that used OOB methods for setting up the network. So much depends on you the developer following good practices including: 1) Making the device tamper-resistant. (for ZigBee especially, which relies on the Trust Center); 2) Network setup should pass initial security parameters using an OOB method; and 3) Refreshing Keys often. Bluetooth implementations are required to provide a function that does this and the SDK we used had a single function to perform this.

Number of Nodes: If we are thinking big, Thread and ZigBee node size limitation may make Bluetooth a big winner for large networks. Here is the breakdown:

Bluetooth Mesh: 32,767 nodes with a maximum of 126 hops
Thread: 250+ nodes with the maximum number of relays set to 32
ZigBee: 250 nodes

Range: Don't believe the stated range for any of these. One SDK states that for a Bluetooth 5.0 radio to not expect more than 100 feet. However, with a Bluetooth 5.0 radio, we did get some impressive distances through office walls compared to what we get with our Bluetooth 4.2 headsets. Here are the specs nonetheless:

Bluetooth Mesh: 100 feet – 1,000 feet
Thread: 100 feet
ZigBee: 30-300 feet

Gateway Requirements: Because the Thread mesh network nodes are IPV6 addressable, the Gateway (called "Edge Router" in Thread terminology) can be generic (in other words, no special software). This is not true for the ZigBee or Bluetooth gateways.

CONCLUSION

As the Carpenters' song goes: "We've only just begun…" This topic is gigantic. The Bluetooth specification is hundreds of pages of thick prose. But we have started you down the path—of course—in thin slices. Next time we will look in more depth at setting up a Bluetooth mesh network.

ABOUT THE AUTHOR

Bob Japenga has been designing embedded systems since 1973. In 1988, along with his best friend, he started MicroTools, which specializes in creating a variety of real-time embedded systems. MicroTools has a combined embedded systems experience base of more than 200 years. They love to tackle impossible problems together. Bob has been awarded 11 patents in many areas of embedded systems and motion control. You can reach him at rjapenga@microtoolsinc.com.