Using a Wireless Network Analyzer to Debug a ZigBee Protocol System

by Kim Otten, Microchip Technology Inc.

Messages and beacons that appear when a device joins a ZigBee Protocol network can reveal some interesting facts. The short range wireless networking market is set

[1]

for explosive growth. In-Stat says that the market for IEEE 802.15.4&#153 wireless Personal Area Networking (PAN), via the ZigBee&#153 protocol specification and other proprietary protocols, could grow by as much as 200%, with annual shipments surpassing 150 million units by 2009.

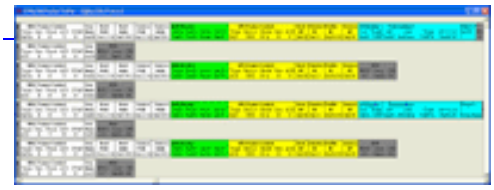
For wireless designers, there are many Radio Frequency (RF) transceivers and ZigBee protocol stacks available today for implementing ZigBee protocol functionality into their applications. Regardless of transceiver and stack selection, design engineers should look at devices and tools that will enable them to evaluate and develop their ZigBee protocol applications quickly. Given the sophistication of the ZigBee protocol, it is also necessary that designers use the appropriate development tools to help them design and debug their applications. This article assumes that the reader has basic familiarity with the ZigBee protocol.

There are several wireless network analyzers, or sniffers available. Obtaining one when you first begin your wireless development is a critical investment, and will save you hours of frustration. Some analyzers have graphical interfaces, while others have a more text-based interface. Be sure to select an analyzer with an interface that is easy for you to read. In the examples below, we will be using Microchip's ZENA&#153 Wireless Network Analyzer, which displays messages in a graphical format. But no matter what analyzer you chose, being able to quickly decipher the key portions of the ZigBee protocol messages will enable you to develop and debug your system efficiently.

Network Formation

The messages that appear when a device joins a ZigBee protocol network tell us a lot about the network. When a device tries to join a network, it issues a beacon request. Nearby coordinators and routers will respond by issuing beacons. The new

device will select a beacon and send that device an association request, asking to join



[2]

the network. After a short delay, the new device will send a data request, asking for a response. The network device will then send an Association Response, indicating whether or not the new device has been accepted on the network. These messages might appear as illustrated in Figure 1.

The first message is from the new device, requesting beacons. The next two messages are beacons. We can tell some interesting things about the devices from these beacons. First, we can tell that the second beacon is from the coordinator, because the source address is 0x0000, and the depth specified in the beacon payload is 0. The first beacon is from a router, because its source address is nonzero. We can also tell that this router is joined to the network through the coordinator, because it is on the same network (the source PANs are identical), and the beacon payload Depth is 1. We also know that at the application layer, both of these devices are accepting new devices to join the network, because the superframe specification Assoc bits are set. However, if we look at the beacon payloads, we can see that the coordinator has no capacity for any additional routers or end devices, because the RtrCap and DevCap bits are both 0. The router, meanwhile, still has capacity for both routers and end devices.

In the fourth message, the new device has decided to try to join the network. By examining the destination address of the association request, we can tell what network device the new device is trying to join to &#151 the router. The association request tells us a lot about the new device. The critical elements are the Dev and RxOn bits. The Dev bit indicates that the device is trying to join as an end device. Also, we know that this device will always have its transceiver on, so its parent will not have to buffer messages for it.

This bit is critical for later communication. If RxOn is set to off, then the parent will assume that the new device keeps its transceiver off, and will buffer messages for it. If the new device never requests messages, then the parent will never send messages to it.

Finally, we can see that the association request was successful. We now have a new FFD (Full Function Device) end device on the network, with a short address of 0x1AF9.

Acknowledged Transmissions

The ZigBee protocol is a highly acknowledged protocol. There are three levels of acknowledge: MAC (Medium Access Control), APS (Application Sub-support), and

[3]

AF (Application Framework). Not only does this improve the reliability of the system, but it also gives us insight as to where the fault lies when problems occur. In Figure 2, device 0x0001 is telling device 0x0000 to set an attribute to a specific value using all three levels of acknowledge.

The first message is the command to device 0x0000 to set an attribute. This is followed by a short MAC acknowledge. This acknowledge indicates that the transceiver of the receiving device correctly received the message. These are often generated automatically by the transceiver. The ZigBee protocol specifies that all messages must request a MAC acknowledge.

In the APS frame-control field of the first message, there is a bit that indicates whether or not an APS acknowledge is requested. If this bit is set, then the APS layer of the recipient's ZigBee stack will automatically generate an APS acknowledge, as indicated by the third message. This acknowledge indicates that the destination device's stack received the message. The APS acknowledge will generate another MAC acknowledge, as shown by the fourth message.

Looking at the transaction decoding of the first message, we see that the Set with Acknowledge command was used. This is an indication to the application layer that the application layer must acknowledge the command with a Set Response command, as indicated by the fifth message. This acknowledge indicates that the application layer received the message, and the transmitted error code will indicate if any problems occurred when setting the attribute to the requested value. The AF acknowledge will generate another MAC acknowledge, as shown by the sixth message. In addition, the AF acknowledge was sent with an APS acknowledge requested, which will generate the seventh and eighth messages.

These multiple layers of acknowledge generate quite a bit of air traffic, but also allow us to locate problems more easily if there is a break in the message train, as shown by Table 1.

Topology Issues

After a wirelessly networked system has been developed, a wireless network analyzer can assist with installation and topology analysis. Some wireless network analyzers,

Table 1: Determining ZigBee Protocol Root-Cause Problems

First Missing Message	Possible Problems
1. Original Command	-The source device is not a member of a network -The source device's stack has overflowed its processing capabilities -The source transceiver is having trouble transmitting -If the destination device is a child of the source device and the child device's transceiver is off when idle, the child has not requested messages -The source device and the sniffer are operating on different channels
2. MAC ACK	-The wrong destination PAN ID or address was used -The destination device is out of range -The destination device is not powered -Radio interference at the destination -The transceiver has overflowed
3. APS ACK	-The destination device's stack has overflowed its processing capabilities
4. MAC ACK	Same as message number 2
5. AF Acknowledge	-The application does not have response capability -There was an error with the command and the application does not have complete error handling, for example, incorrect endpoint, invalid attribute ID, data type or value
6. MAC ACK	Same as message number 2
7. APS ACK	Same as message number 3
8. MAC ACK	Same as message number 2

[4]

including the ZENA Wireless Network Analyzer, have the capability to graphically display air traffic as the messages traverse the network. The devices are represented as circular nodes, and the messages are represented as lines connecting the nodes. Additionally, if a bitmap that describes the physical layout is loaded, topology issues can become clearer.

Suppose we have a network with four devices: one coordinator, two routers and one FFD end device. Although all devices are full-function devices, we've seen through our packet analysis that messages between the FFD end device and one of the routers are being sent through the other router. If we open the Network Configuration Display window, load a descriptive bitmap, and drag the nodes to the location on the bitmap that describes their physical location, we can quickly see what is happening.

Figure 3 shows the Network Configuration Display window with the network traffic in question. We can see that the two devices are not within radio range of each other, most likely due to the walls within the building. The messages must be routed through another device to reach their destination. So, while the behavior might have originally been worrisome, we can now see that the system is behaving exactly as it should.

Note that a single sniffer may not be able to see all of the nodes on the network. In order to observe all network traffic, it may be necessary to obtain multiple message captures from different locations, or use multiple sniffers positioned in different locations.

Conclusion

The ZigBee protocol is simpler than many wireless protocols, but the right tools are still required to develop applications efficiently and effectively. From confirming that the correct data values are being sent, to understanding network behavior, a wireless network analyzer is an essential tool for anyone doing wireless network development.

About the Author

Kim Otten is a principal applications engineer for the Advanced Microcontroller Architecture Division of Microchip Technology Inc., 2355 W. Chandler Blvd., Chandler, AZ; (888) 628-6247.

Source URL (retrieved on *03/27/2015 - 2:20am*):
http://www.wirelessdesignmag.com/product-releases/2007/02/using-wireless-network-analyzer-debug-zigbee-protocol-system

Links:
[1] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0702/cov1_lrg.gif
[2] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0702/cov2_lrg.gif
[3] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0702/cov3_lrg.gif
[4] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0702/cov4_lrg.gif