

Tips for Designing ZigBee Applications

Originally published on [Wireless Net DesignLine](#) site
(04/25/2006 7:07 PM EDT)

Many engineering disciplines are needed to define, architect, design, test and ready the final product for sale.

By John Sawyer, Indesign LLC

So you think you want to develop a ZigBee application, but you have some questions. What's important in a ZigBee development? What tools do I need? What steps are involved? How much time will it take? What kind of investment is required? What are the unforeseen questions?

The tips presented here can't answer all of these questions for everybody but they will provide a tutorial for the development of your ZigBee application.

The first and perhaps most important step in developing your application is to determine whether ZigBee is appropriate for your product. The ZigBee radio standard boasts many advantages over other wireless options.

With a public ZigBee profile and conformant platform, interoperability with other vendor devices is assured. ZigBee was defined to support extremely low power and remarkably long battery life. But the ZigBee data rates are low relative to other wireless options. And while substantially smaller than Bluetooth or 802.11/WiFi, the ZigBee stack is not a simple piece of software.

Partnering

After deciding that you want to develop a ZigBee application, you must choose a ZigBee provider partner. A ZigBee partner will supply the ZigBee stack software along with a radio chip and baseband microcontroller.

Historically, these have been two chip solutions. Recently, though, vendors have been introducing single chip ZigBee devices with plenty of additional resources to support your application code. A quality ZigBee partner will also supply you with continual updates on the most recent features.

Several ZigBee providers are available, and each has invested many person-years of development to arrive at their ZigBee solution. In choosing a partner, you should consider technical aspects (the cost vs. capability of their chipset, supported features) and business aspects of the provider (company history, licensing costs, technical support).

The ZigBee providers will also want to know about your information such as your previous product development experience and the number of devices that you plan to sell. When this exchange of information is complete, you should rather quickly be able to find a ZigBee provider willing to collaborate on your application.

After you have chosen a ZigBee provider, you must then decide how to implement the ZigBee radio as a module connected to the rest of the system or as a chip integrated into the circuitry of your board.

Using a ZigBee radio module offers many advantages. A module usually leads to a shorter development cycle. The RF design is already done for you so you don't need to become an RF expert. In addition, a module may carry FCC Modular Approval, which means you won't have to take your product through the FCC's intentional radiator approval process.

Disadvantages to using a ZigBee radio module versus an integrated ZigBee chip include a higher per-unit product cost. A module design also requires more physical space inside the housing and imposes more constraints on the industrial design of the product. What's more, a module may have limited antenna options, which may be unsuitable for your product.

The final architecture task is to completely define your ZigBee network structure. The ZigBee standard supports multiple topologies for you to choose from. These include mesh, star, and cluster tree network configurations, which are shown in Figure 1.

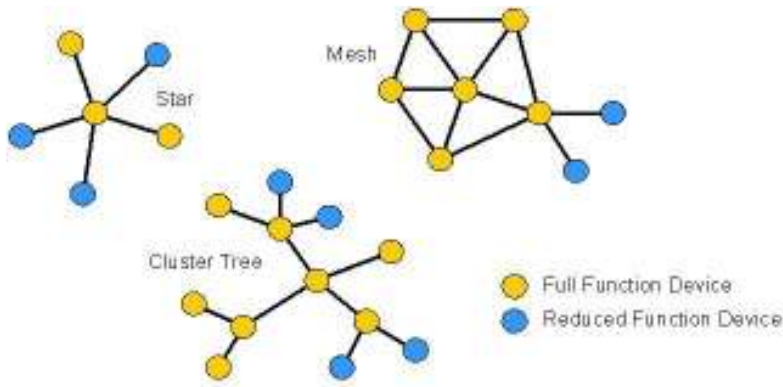


Figure 1: Network structure diagram showing mesh, star and cluster tree network configuration options.

The typical ZigBee network configuration is comprised of low power reduced function end point devices, generally called sensors. The sensors communicate with full function network control devices that handle the routing of packets over the network.

ZigBee radios

ZigBee radios can be implemented in several RF bands. Those operating in the 2.448 GHz band are the most common since this is the only RF band that is usable worldwide. In North America, 915 MHz is available for ZigBee radios and offers a few advantages over 2.4 GHz, such as slightly better range.

Fewer chip providers are available who offer the lower frequency range, because antennas tend to be larger and the over-the-air data rate is lower. The 868 MHz version of ZigBee also offers valuable features, though it is available for use only in Europe.

Now that all the architecture decisions have been made, it is time to open the development system provided by your ZigBee partner and dig in. An example is shown in Figure 2.



Figure 2: Sample ZigBee development platform including multiple command/debug interfaces, integrated sensors, GPIO access, with an integrated ZigBee radio chip.

The development system will have sample application code to reference. Your final application will likely resemble this sample because you will apply the same Application Programming Interface (API) that the sample uses. You may also use the same operating system, system calls, and interrupts. Typical API commands are "FormNetwork()", "JoinNetwork()", "SendMessage()", and "ZigBeeStackTick()". Learn to love this API - you will live with it for the next couple of months.

During the development phase, you will face many design challenges. Many will involve typical embedded development issues, while others will arise because you've added a new design element - the ZigBee stack.

Power consumption optimization

ZigBee advertises ultra low power. This is great for the radio - but you have to expand that to include your entire system design. To fully optimize power usage and battery life, the firmware, electrical, and RF teams must devote a good deal of time to optimizing power consumption. As part of this effort, microcontroller sleep modes must be defined and implemented. Be aware that your development system may not work properly while the system is in sleep mode. You may need to use the old-fashioned "GPIO toggle and scope tracing" debug techniques.

If your system will be battery-powered, keep in mind that ZigBee radios do not violate the laws of physics. ZigBee radios use the current required to transmit at the selected power level - typically in the range of 20 to 50 mA. Achieving multi-year battery life is highly dependent on the usage scenario. Pay particular attention to how frequently the radio needs to wake up, how long it takes to wake up, and how much current it consumes while asleep. You may find that a ZigBee provider's data sheets are only helpful to a limited extent in this area, and you will end up making your own measurements on current consumption.

In general, the principal tradeoff you need to make is data latency versus battery life. It seems counterintuitive; however, since ZigBee transmissions are at low RF power - in the range of a milliwatt - battery power usage during transmit and receive modes are similar. Don't assume that a radio mode will use little power if it spends most of the time in receive mode.

The ZigBee development system you start with will probably not match your final hardware. This means you will be designing new interfaces to your devices - buttons, displays, memory, etc. You may also need to add a communication channel to another system, such as a system monitor PC program.

This is where taking the time to thoroughly understand the API and your development system's starting point helps, because you will need to merge these new interfaces with the rest of the existing design.

Interoperability

In order to verify interoperability of your device, you may need to swallow your pride and purchase a competitor's product. Use of another ZigBee approved system during development may help you discover how your product's performance can best be optimized. Another option to test interoperability is to participate in one of the ZigBee Alliance's quarterly interoperability events called ZigFests. To participate in a ZigFest your company must be at least an Adopter class member of the ZigBee Alliance. There are also certification companies that can provide a pre-certification testing environment. Remember that, fortunately, the ZigBee radio is not a frequency hopper like Bluetooth. Before the network forms, the network coordinator scans the available channels to find the "clearest" one. A function to perform this automatic frequency selection is usually included with the ZigBee stack you purchase, but is easy to implement if not. The network coordinator can then be programmed to periodically test the network to determine if the selected frequency remains the best option. If not, the network coordinator can move the network to a different channel without operator intervention. This insures the network will perform optimally at all times. If data security is required for your application, ZigBee includes provisions for strong data encryption. ZigBee data security is based on the 128-bit AES algorithm. If you are using a public ZigBee profile then the security decisions have already been made and are pre-defined in the profile.

Data security

For a custom application that requires data security, you have options regarding where the security is applied - in the application code or in the lower layers of the ZigBee stack. If your application needs the strongest security possible, secure it in your application code. The ZigBee stack defines

optional security in the lower layers of the stack, which can be used to validate each data packet that is exchanged on your ZigBee network. Your development will most likely include the development of two ends of a system - the sensors, which may be reduced functionality ZigBee end devices, and the devices with which they want to communicate. Each end of the system will present its own design challenges. Power optimization is generally the largest design issue for the sensors, while message management is the largest for the ZigBee routers.

Message management is an important part of the application code. Your application code will communicate with the ZigBee stack by sending messages to the stack by calling stack functions and receiving messages from the stack through callback functions. The application code will likely need to monitor these messages and may need to perform tasks, such as timing messages and purging "lost" messages, on top of its normal network management task.

As part of the application development planning process, be sure to include time to write test code. During your integration phase, test code will help identify and verify boundary conditions of the feature operations. Specialized test code may be required to put the device in perpetual transmit mode during hardware compliance testing. At the factory, properly designed test code can quickly and thoroughly verify operation of each device as it comes down the line.

One tool vital to designing a ZigBee application is the ZigBee RF sniffer/protocol analyzer. Even if your design uses a ZigBee radio module and a public ZigBee profile, you will eventually need to examine commands as they are sent over the air. Many protocol sniffers are currently available with a wide range of capabilities and cost. Choosing the sniffer will depend on your experience with protocol analyzers, the depth of protocol analyzing required, future expected ZigBee work, and the cost of each unit. The most logical path is to start with a basic unit and upgrade later if you need additional capabilities.

Time-to-market

If time-to-market is a major requirement for the system, use of a ZigBee radio module is almost a given. A module-based design will provide the fastest development cycle.

Many projects require fast time-to-market with a small number of devices to introduce a technology or a product line. Often, higher volume production comes at a later date. If that is the case, it may be possible for a two-phase development - the "fast" phase using a ZigBee radio module and a "cost-reduced" integrated design phase to follow. This cost reduction phase often involves merging functionality from modules into a single circuit board. The timing of a two-phase approach also makes sense, because expertise is built with the technology before merging the radio chip on the circuit board. Plan ahead, and the optimal market rollout is achievable.

Like any other electronic device that is brought to market, your ZigBee application must meet regulatory standards. Since a ZigBee device is an intentional radiator, the device must meet global standards.

As the owner of the design, you will be responsible for submitting the product for compliance testing, adjusting the design as needed to pass, and filing the final design with the appropriate agency. Plan for several rounds of testing as early as possible in the design phase. This includes a pre-screening test on the radiated emissions.

Since most ZigBee designs transmit at the low RF power of about one milliwatt, the fundamental RF emissions are not likely to be problematic. However, requirements on emission levels - particularly the second and third harmonics that fall into FCC "restricted bands" - need to be carefully monitored. You must also screen less technical compliance areas, such as product labeling requirements and antenna limitations.

To advertise your product as being ZigBee compliant, your design must go through ZigBee certified product testing. To submit the product for ZigBee testing, your company must be a member of the ZigBee Alliance. Joining the ZigBee Alliance is an inexpensive proposition and should be done immediately because membership will allow access to too many ZigBee-related documents and discussions. Visit their website at www.zigbee.org.

For each and every ZigBee product, you will need to generate a unique 802.15.4 MAC address. The 802.15.4 MAC address is an eight-byte (64 bit) value. The first three bytes (the Organizationally

Unique Identifier or "OUI") are licensed to your company by the IEEE Standards Association for an annual fee. Your company is then free to uniquely assign the remaining five bytes.

Finally, depending on the factory assembly and testing process, it may be desirable to have a software-based manufacturing test provide the MAC address to program into each device. In this case, your application code may need to support a non-violate memory write function that may not otherwise be necessary.

Conclusion

ZigBee is a well-defined and feature-rich radio standard; however, developing a ZigBee product is not a trivial exercise. Many engineering disciplines must work together to define, architect, design, test and ready the final product for sale. Following the suggestions set forth in this discussion will enable you to prepare a practical plan towards creation of a ZigBee radio application.

About the author

John Sawyer is a Principal Software Engineer at Indesign, LLC, an engineering design services company. His projects have included many different wireless technologies including ZigBee, Bluetooth, WiFi and proprietary systems.

About Indesign

Indesign provides design engineering services as well as full turn-key product design, from product concept to ready-to-manufacture electronic product design. Offering complete engineering design capabilities involving multiple engineering disciplines and an ISO certified product development process, Indesign can provide requirements analyses, architecture development, mechanical design, electrical design, software design, verification and validation testing, and other product development capabilities. More information is located on the World Wide Web at www.indesign-llc.com.