Applying Embedded Wireless Networks

Embedded Systems Conference, San Francisco, 2004
Class #349

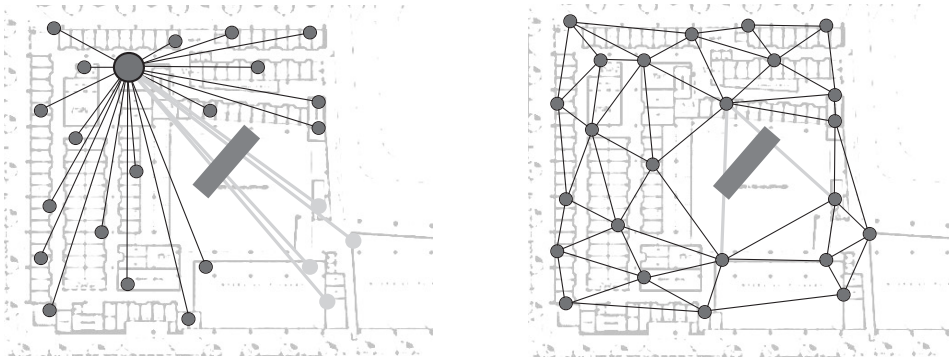
Cliff Bowman
Ember Corporation
Boston, Massachusetts, USA

Background and Scope

There is tremendous market interest in embedded wireless devices today. In part, this interest stems from the emergence of inexpensive RFICs, but the development of networking software specifically for embedded applications has played an equally important role. As the networking requirements for embedded systems become more complex, the combination of "chip plus stack" has emerged as a standard offering.

The significance of the networking stack quickly becomes obvious during development. The accountant may think that low-cost radios make large-scale network applications (*e.g.* automated meter reading, building management, battlefield threat detection, irrigation) look good, but the engineer has to find a way to get hundreds or thousands of devices to work together under the constraints of his embedded processor. The new RFICs may whittle down the BOM cost of a home application, but unless the software makes the product easy to install the product will never sell. I've seen several teams try to develop networking functionality as part of the application, but for the same reasons why people don't usually write their own operating systems this approach rarely makes sense. Fortunately, there are multiple vendors offering scalable, feature-rich, and field-tested networking software specifically designed for small processors.

Some of the most advanced embedded networking software on the market today implements a type of organizational structure called mesh networking. In mesh networks, nodes relay messages on behalf of their neighbors; this direct interaction among nodes leads to a web-like network of interconnections as shown in Figure 1. While a traditional star network organization can also be used with RFICs, there are many features of mesh networks that complement the characteristics of today's low-cost radios. Consequently, we will discuss mesh networks below as an important supporting technology.



Star Network

Mesh Network

Figure 1 - Network Topologies

This paper examines new application areas for embedded wireless communications. Point-to-point and rigid one-to-n data communications that have traditionally been associated with embedded wireless devices are not considered here. While remote car starters, wireless game controllers, pet containment fences, and the like all benefit from the availability of low-cost RFICs, these designs are already well understood.

We turn instead to applications where greater network functionality is required. Most of the functionality is already available in the popular networking stacks, but understanding what that functionality is and something about its use benefits anyone designing these applications. In addition, wireless nodes behave differently than traditional wired devices, and these differences are particularly apparent in larger networks. Consequently, we place particular emphasis on these differences and the challenges engineers face when using wireless communications in embedded systems.

Enabling Technology - Radio

The most obvious technological advance behind the new embedded wireless applications is the reduction in radio cost. The emergence of viable sub-$3 single-chip data radios, with sub-$1 radios on the horizon, makes it possible to put wireless connectivity into entirely new applications. Light switches, thermostats, smoke detectors, and a host of other consumer products are taking advantage of these inexpensive radios. Military and law enforcement agencies have commissioned battery-powered, disposable sensors for chemical threat detection, landmine replacement, and perimeter security; similar sensor networks are being developed for environmental monitoring applications in a number of industries. Disposable sensors that can be shipped in-the-box with pharmaceuticals and other perishable cargoes, “smart locks” that detect tampering with shipping containers, and active tags that track the maintenance and service history of high-value assets are all in various stages of development. The common requirement among these diverse applications is that the addition of communications be cheap and non-intrusive—an apt description of wireless technology based on the new RFICs.

But what does one typically get in a “cheap” single-chip radio? After dissecting past generations of garage door openers, pet containment collars, cordless phones, and other low-end RF devices, I have to confess that my expectations weren’t high. I feared poor sensitivity and selectivity, primitive modulation strategies, and touchy and inflexible designs. Happily, the performance of popular RFICs is quite good, and they offer a far richer set of features than many would imagine.

Table 1 lists some of the characteristics of three popular radio devices, but it doesn't really capture some of the new functionality. While the fulfillment of traditional requirements such as frequency flexibility, robust modulation or spread-spectrum support, and minimal power draw is sufficient to distinguish many of the new radio devices, much of their popularity stems from completely new feature categories. For example, both the ZMD44101 and the EM2420 integrate many traditional baseband functions, with the EM2420 incorporating its own spreading logic, message buffers, encryption engine, automatic packet validation and acknowledgement generator, and link quality assessment logic.

Characteristic	Chipcon CC1020	ZMD 44101	Ember EM2420
Frequency Band(s) MHz	403 / 868 / 928	868 / 928	2400
Data Encoding	ASK, FSK, GFSK	DSSS	DSSS
Max Data Rate	153.6	40	250
Power Draw Sleep/Peak @+0 dBm	<.6µW/<62 mW	<8µW/<48 mW	<2 µW/< 36 mW
Typical LOS Range	300 m	100 m	100 m
MAC Support	-	802.15.4 "Thin" MAC	802.15.4 Low-Level MAC

Table 1 – Three Inexpensive RFICs[1]

A further trend is greater RF circuit integration within the RFIC, thus minimizing total component count and final manufacturing cost. Each of the devices in Table 1 is a highly integrated radio requiring minimal supporting circuitry. In fact, the reference designs for each of these RFICs contain only a handful of parts – merely a crystal for frequency generation and a few passives for decoupling and impedance matching.

Activity in the field of low-cost RFICs is accelerating. Startup companies like Chipcon and Ember continue to innovate with breakthroughs like their all-CMOS, 0.18µm designs. Traditional silicon powerhouses like Motorola and Intel have announced major projects to enter the field. New players seem to be emerging daily, while serious standards efforts like IEEE 802.15.4 are driving the market towards greater interoperability. All signs point to the continued vitality and advancement of this technology.

[1] These data were taken from the manufacturer's websites in January, 2004. Please refer to the manufacturers for up-to-date information.

Enabling Technology - Network Stack

Inexpensive single-chip radios may lead to the replacement of wired communications in some embedded applications, but cost reduction alone wouldn't open all the new application areas being explored today. Networks of more than a handful of embedded devices would still be impractical without the recent advances in network technology. If embedded designers were forced to create a custom networking strategy for each application, or if existing networking software demanded powerful processors and memory-rich systems for their implementation, the rapid emergence of large-scale "sensor networks" would remain a mere academic conjecture.

Fortunately, network software designed specifically for memory- and processor-constrained embedded devices is available from multiple vendors. These stacks support the latest RFICs and provide an interface from popular microprocessors. They're typically offered in the form of a library that's compiled into a user application. An example of this architecture is shown in Figure 2.

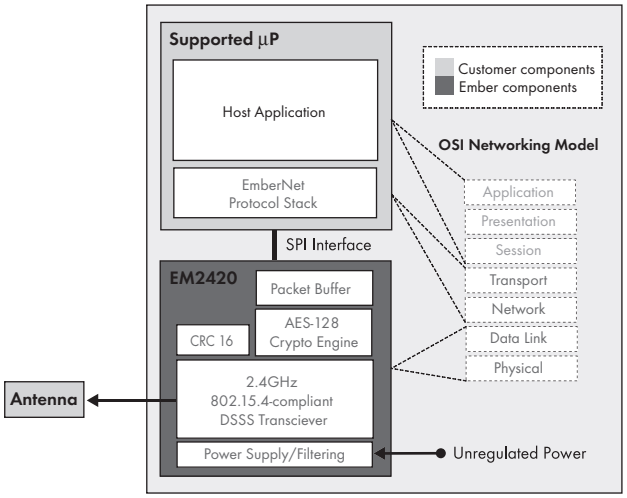


Figure 2 - Example Stack Interface

Although popular network stacks are designed specifically to have a small footprint, they provide a surprising range of "big network" functions. Beginning with sophisticated MAC functionality for large-scale collision avoidance, link-level retry, link-level acknowledgement, multi-hop routing, and fault detection, popular stacks support all the basics of network operation. Some implementations additionally support TCP-like connections, provide tools for dynamic key exchange, and offer low-level network diagnostics. Dynamically adjusted output power, synchronization for battery-management and simultaneous measurement, automatic network discovery and organization, and over-the-air updating are additional features that are available today.

One advance in embedded networking software that's receiving a lot of attention is the implementation of mesh networks on 8-bit microcontrollers. Seen in the organization of the Internet, mesh networks offer the kind of scalability, organizational flexibility, and fault-tolerance required in many industrial and performance-critical applications. Until recently the routing algorithms upon which these networks were built required a lot of memory and hefty computational platforms for their implementation; the breakthrough came a few years ago when DARPA-sponsored research at MIT and Berkeley made embedded implementations a reality.

The military saw the utility of large networks of sensors that could function with very little manual intervention. Such a network might be airdropped from a C-130 or fired from artillery pieces into a hostile area. Wherever the sensors landed, the network would need to organize itself and find a routing strategy to bring threat information back to a command station. If individual nodes failed, whether by being captured or being destroyed during deployment, the network would need to recover from these failures and continue to function. The hope was to create low-cost sensor units that could be deployed in large numbers to replace human patrols and landmines.

The result of this research included a new family of "reverse gradient" routing algorithms that required much less memory and processing power than previous mesh networks. Efforts to further develop and commercialize these algorithms have been extremely successful over the past three years, and field-tested network stacks based on these algorithms are available today. In addition, a standards effort by the Zigbee alliance is bringing the two most popular reverse gradient algorithms together, along with the best results of commercial research. The result will create a new embedded PAN standard built on IEEE 802.15.4 RFICs sometime in 2004.

A number of vendors offer the entire networking solution: an RFIC paired with stack libraries. Sometimes the networking capability is built into a packaged device, whether it's an integrated sensor module or a more generic device with standard hardware interfaces. Other vendors offer a chip-level product that can be designed into whatever device the designer needs. Either way, the tools to create wireless embedded applications more cheaply, more conveniently, and on a larger scale than ever before are available today.

Using Wireless - Propagation

When an embedded designer first begins to use wireless technology, he quickly discovers that the associated engineering considerations are very different from those of wired communications. The most glaring difference is the issue of signal propagation, which is hardly a consideration in wired systems unless extreme distances or data rates are required. In a wireless system, propagation is a primary concern that affects all aspects of design.

The first lesson about propagation is that it changes over time. Links in a factory that appear strong and reliable during deployment may be rendered completely unusable after the installation of a new machine. Telemetry at a retail gas station may flow flawlessly whenever a repair technician visits the site but fail when a delivery truck blocks the access point. Even if the nodes in a network could be placed to avoid all current and future interference, continuous variations in signal quality still can present challenges. The simple fact is that propagation fluctuates moment by moment due to changing natural and manmade conditions.

Sometimes propagation varies because of changes in the noise floor. All communication depends upon separating the useful signal from the background noise, and when there is a lot of background noise, it is difficult to get one's signal through. This ambient noise level is known as the "noise floor" and is due primarily to atmospheric conditions and local noise sources. In actual deployments, the noise floor is varying constantly, and the intensity of the noise can rise and fall by tens of decibels. Very often, this is the principal cause of changes in link quality.

Propagation can vary because of obstacles moving in and out of proximity with the network. Sometimes the role of an obstacle is obvious, such as when a delivery truck blocks an access point. At other times, the role of obstacles is more insidious, such as when reflective surfaces create multiple pathways for the radio waves. Such a condition is called "multipath," and it can degrade link quality through destructive interference of the signal traveling via two paths of different lengths.

Changes in path attenuation can lead to variations in propagation. The performance of outdoor networks, for example, can change significantly from sunny weather to rain or from summer to winter. Networks set up in modern office buildings can vary dramatically when cubicles are rearranged, temporary partitions are moved, or doors open and close. Finally, I've seen networks perform well at night when the building was empty but marginally when the workers arrived.

It is always helpful to know some of the reasons why propagation changes over time, but fluctuation in signal quality is not something that can be prevented. Sometimes designers use strategies like reducing link distances, increasing transmitter power, or using better antennas to provide some margin against changes in link quality. Other designers exploit the often highly localized nature of propagation changes and design in alternate message pathways. No matter what strategy is used, designing with propagation changes in mind is the best way to ensure reliable communications, and using one of the popular networking stacks is a convenient way to do this.

Network Characteristics – Synchronization

In many applications, synchronization among nodes is an important consideration. The application might require a set of sensors to make a simultaneous measurement, for example. Other systems might run on battery power, so coordinating the sleep/wake cycle of the devices becomes important. Finally, some wireless networks use timing critical techniques such as TDMA (a medium access control strategy based on timed transmissions) or frequency-hop spread-spectrum modulation. In each of these examples, some or all of the devices in a network will need to synchronize their timing.

Synchronization isn't much of a challenge when all nodes can communicate directly. Such a situation is similar to a wired buss, and if one node broadcasts a periodic signal the entire network can synchronize from it. Unfortunately, things become more complicated when the network grows because relaying takes time. If messages must be relayed from one section of a network to another, the synchronization signal will arrive at different nodes at different times.

The problem is a hard one, and synchronization across a large network remains a topic of active research. Even so, there are practical strategies for achieving "enough" synchronization for each of the examples mentioned above. Most of these focus upon localized synchronization, where a node keeps time with its neighbors, or correcting a synchronization signal with a delay value at each relay point. Using a proven networking stack is again a good way of ensuring that the final implementation behaves as desired.

Network Characteristics – Deployment Issues

No design can be successful if it is difficult to deploy. Deployment problems drive up the total solution cost and usually have a negative effect on system reliability. Poorly designed wireless systems require more time and skill on the part of the installer, and complex deployments can be a significant barrier to the acceptance of a design.

When someone installs a wireless system, there are typically two deployment challenges he must face. The first is node placement, which is the positioning of each device so that communications links are strong and reliable. The second is commissioning, where each node is somehow identified and configured for its particular function in the network.

Some networks are fairly rigid in their organization: certain nodes must be able to communicate directly with certain other nodes. An example might be where sensor nodes are assigned a "data aggregation point" to which they must directly report their data. In these cases, we would like to place the sensor nodes well within range and in clear line of sight of the access point; where this is not possible, we would want to test connectivity thoroughly to prevent later problems. In these types of applications, a site survey to determine optimal node placement is often an important deployment step.

Other networks allow a bit more flexibility in node placement. For example, nodes in a mesh network automatically relay for one another. This means that sensors reporting to an aggregation point need not communicate with the point directly; instead, their reports merely need to travel to a neighbor who has a neighbor who can send to the aggregation point. Site surveys and precise node placement are less critical in these types of networks because other nodes can "help out" by relaying messages. Furthermore, even when the natural placement of nodes fails to provide a relay path, additional nodes functioning solely as repeaters can be added as needed.

The second deployment issue is commissioning. For example, devices in a lighting control application must be interchangeable when they ship from the factory. This means that the installer receives the devices in an unconfigured state, and he associates switches and the lamps they control during installation. This association process is called commissioning, and the usefulness and market acceptance of designs hinges on the ease with which they can be commissioned.

In practice, there are numerous strategies for commissioning devices in a network. The best strategy depends upon the application, but there are common elementary functions from which most strategies can be built. For example, commissioning a large-scale network often begins by identifying nodes of a certain type (e.g. all the lights in a particular room) and then arranging them into functional groups. Functions like these can be developed as part of each new application, of course, but this approach is rarely a good use of development resources. Because current stack implementations provide commissioning support in a ready-made, pre-tested library, many designers find it convenient to use these functions to implement their commissioning strategies.

Using Wireless – Network Topology

One of the first decisions to be made about a wireless application is what the basic flow of messages will be. Will a number of sensors report their data to an aggregation point? Will wall switches, light sensors, timers, and occupancy sensors work together to control banks of lights? What happens if a message destination is out

of range of the message originator? How will messages be relayed? Will devices move in the network? How will the network recover from the loss of a route? All of these questions have bearing on message flow, and efficient message flow is an expression of the network's organization or topology.

The fundamental types of network topology are star and mesh. Star networks have a central node that all of the other nodes interact with; this node is sometimes called an "access point," "aggregation point," or "master," and it arbitrates all communications within the network. Since the master must interact with every node, the master's radio limits the geographic spread of the network. In mesh networks, there is no innate hierarchy among nodes, and nodes interact freely with one another. Mesh networking software is based on the idea that nodes relay messages for one another, so the network can extend beyond the range of any individual radio.

Practical networks are rarely purely star or mesh. Star networks, for example, may be linked together by a backbone network that has its own topology (*e.g.* when multiple 802.11 access points are used to cover a large area). Mesh networks may be used to support multiple data aggregation points for a sensor array, with the aggregation points linked together with a console by another network tier. These hybrid networks provide flexibility in applying each type of network where it functions best, and large-scale applications are usually based upon them. Fortunately, each subsection of a hybrid network functions like its corresponding fundamental networks type, so large-scale systems can be understood piece by piece.

Star networks have the advantage of being simple to implement and to characterize. Because the master node arbitrates all communications within the star, routing and multiple-access problems become very simple to manage. Timing control is centralized, so it's easy to analyze. Nodes transmit directly to the master, so properly-designed applications can utilize most of the bandwidth available on the radio.

Mesh networks are complicated to implement from scratch, but they offer many advantages in networks based upon low-cost radios. Because nodes relay on behalf of their neighbors, the radios' low power output doesn't limit the network's coverage. Automatic relaying means that there is no distinction between network infrastructure and the network itself, so nodes can be installed incrementally and without a lot of up-front cost. Mesh networks support path redundancy, so the failure of any single node or link should not cause the network to fail. Finally, because the nodes in a mesh network merely have to communicate with a neighbor, mesh networks are easier to deploy than star networks where connectivity with an access point is required.

Mesh Network Design Issues – Bandwidth Management

Mesh networks are built upon the principle of nodes relaying on behalf of their neighbors. This relaying is usually performed in store-and-forward fashion in embedded networks, and this has two important implications for design. The first is that each "hop" adds latency, so that it may take a significant amount of time for a message to travel the breadth of a large network. The second is that receiving and re-transmitting a message makes a portion of the network busy, and it can handle no other messages until the first one is a few hops away. Figure 3 illustrates this point.

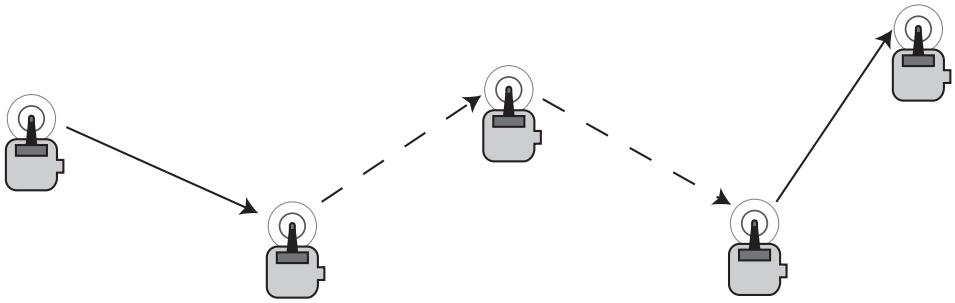


Figure 3 - Message Relay

If the node on the left originates a message and transmits that message to the second node, both nodes are "busy" during transmission. The second node forwards the message to the third node, thus occupying both of those nodes. Note that because it can hear the second node, the first node is busy during the retransmission as well. When the third node transmits the message to the fourth node, this transmission ties up the second node (which can hear the transmission) along with the third and fourth node. The first node is not busy, but it cannot initiate a new message along the same path because its relay, the second node, is busy. Finally, when the original message is making its fourth hop, the first node can transmit again. Note that this waiting has the effect of reducing the bandwidth available to the first node; in effect, it takes three times as long for a message to clear the originators area as it does for the message to be transmitted. Thus the effective bandwidth in a mesh network is at best 1/3rd of the radio data rate.

In practice, the actual effective bandwidth for multihop messages is about 1/6th to 1/10th of the radio's data rate. This means that a mesh network built upon the 250 Kbps CC2420 actually supports point-to-point throughputs of 25 to 40 Kpbs, depending upon the stack and application. Different radios will offer different performance, but the point is the same: bandwidth is a limited commodity in mesh networks, and it needs to be managed carefully through stack selection and careful application design.

Whatever the stack that is used, making the most of the available bandwidth is easier when the application takes a distributed approach. In my ESC paper last year, I described some basic "rules of thumb" to help ensure efficient utilization of a mesh network. These were: distribute tasks, shorten message paths, and use a MAC-savvy protocol. As I showed in that paper,[2] applications that follow these guidelines can accomplish much more than a simple comparison of data rates would suggest.

Mesh Network Design Issues – Global Updates

Another important practical feature is the support of global updates. These updates may be as simple as changing radio channels or a complex as dynamically changing security keys. Sometimes software updates are required, or perhaps parameter settings need to be changed. Regardless of the update, keeping a path open for the update messages is a challenge in mesh networks.

As a consequence of having neighboring nodes relay, the order in which individual nodes are updated is important. Ideally, the update would work itself across the network from the furthest nodes to the nearest; unfortunately, determining global placement from local data isn't easy. Once again, library software provides field-tested basic functions from which a complex update strategy can be built.

Conclusion

New low-cost RFICs have sparked interest in entirely new applications for embedded systems using wireless technology. Many of these applications require significant network functionality, and developing this functionality is a major undertaking. Fortunately, multiple vendors offer field-tested networking libraries paired with a single-chip radio, and engineers can build upon these libraries as they develop their applications.

Wireless communications have different characteristics from wired communications. Many of the tools needed to work successfully with wireless communications are built into the network libraries offered today. Understanding the characteristics like propagation, relay latency, and bandwidth utilization is key in the evaluation of stacks and the design of good wireless applications.

[2] "Architechting Communications in a Wireless Mesh Network," (Class 523) Proceedings of the Embedded Systems Conference, San Francisco, 2003.

A particularly useful network organization strategy for low-cost radios is mesh networking. By having nodes relay for one another and by creating multiple message pathways, mesh networks are ideally suited for the limited range of embedded wireless devices. Mesh networking is supported by some of the available network stacks, and with an understanding of mesh characteristics, a designer can build an efficient, embedded wireless application using these tools.