Introduction to Embedded Bluetooth

This document is still a work in process. Please contact us if you have any comments, questions, or complaints. Revision NC March 2, 2005

This document is an introduction to the use of Bluetooth for embedded control applications. Although there are many resources that discuss Bluetooth for wireless headphones, keyboards, mice, and similar uses, this document is confined to the advantages and potential pitfalls of using Bluetooth for embedded control.

In general, Bluetooth can be very useful for certain embedded applications, but applications that require either very high data reliability or guaranteed data availability will have to be designed very carefully since these are both weaknesses in Bluetooth.

Bluetooth Overview

Bluetooth is a radio communication system that is intended to replace any cable in any short-range communication link. It easily meets this goal in non-demanding applications such as connecting a keyboard to a PC or headphones to a cell phone, but will have great difficulty replacing an EIA-485 link with more than eight nodes. Several dozen potential applications have been identified by the Bluetooth SIG, and are supported with specific protocols. One of the intended applications is the replacement of legacy EIA-232 links. Bluetooth works well for EIA-232 applications that don't push the envelope of the EIA-232 specification.

Bluetooth uses the ISM band starting at 2.4GHz. Bluetooth divides this band into 79 sub-bands and hops among them as described below.

Piconet

Bluetooth devices connect to each other in a network called a piconet. A piconet may consist of from two to eight devices. In a piconet, one device is always the master, and the remaining devices are all always slaves. Master-slave roles are never exchanged because the message and channel timing are derived from the clock and Bluetooth address of the master. A device may participate in more than one piconet at a time, and may be a master in one piconet and a slave in another piconet at the same time. A device may not be the master in more than one piconet at the same time.

Since Bluetooth shares its bandwidth with many competing services, a piconet attempts to reduce interference by hopping from one frequency to another at frequent intervals. Since there may be more than one Bluetooth piconet in an area at the same time, the hop pattern that a piconet uses is partially determined by the unique BT address of the master device. The Bluetooth devices themselves negotiate this frequency hopping; the user or designer does not have any role.

Unicast links are bi-directional transmissions that exist between exactly two devices irrespective of how many devices are in the piconet. Messages may be sent in either direction. A connection process must be executed before any messages can be sent.

Broadcast links are one-way links that contain a master and any number of receivers, including zero. No connection process is ever used. All messages go from the master out to whatever devices can receive them. There is no mechanism to resend in case of errors, so the usual solution is to send every message several times and hope for the best. Better performance can be achieved by using message based error recovery methods such as convolution.

Bluetooth does not support multi-casting. (Define multi-casting REVISIT THIS LATER) A piconet with more than two devices communicates by having the master exchange data with one slave at a time.

Message Structure

Data coming into a Bluetooth link passes through several layers, all of which add a time delay to the message traffic. For an EIA-232 link, this time is measurable, and may be significant. For one particular application, a message exchange that requires about 60 ms over a wire line takes at least 120 ms at close range. At greater distances, where the link is weakening anyway, the delay can be seconds. In fact, it is possible to place two devices at a distance such that the link will be maintained but no data can be exchanged. This is more noticeable for longer messages than for shorter ones.

Data rates

The Bluetooth standard says that the data rate is 1 Mega "symbols" per second. After subtracting overhead, this leaves about 721K symbols per second. For Bluetooth 1.1 and regular Bluetooth 1.2, a symbol is one bit, but for Bluetooth 1.2 EDR (Enhanced Data Rate) a symbol contains 3 bits for a data rate of 2.1 Mbps.

Data rates up to 128Kbps are ensured by the Bluetooth specification, while support for higher data rates is optional. For EIA-232 replacement, the limiting factor is generally the EIA-232 level converters. In this case, the maximum useful bit rate is 10K – 20K Bytes/sec.

The difference between 128Kbps and 721Kbps or 2.1Mbps is not discussed in the Bluetooth specification.

Data is sent in packets that may vary from 17 to 339 data bytes. The size of each packet is determined on the fly by the transmitting and receiving devices, the user and programmer have no influence over the size.

Range

Apparently, at one time the data rate was going to be adjustable for longer range. There are older documents that show tremendous ranges at very low data rates. Whatever the situation was, the date rate is now fixed as described above and the range is limited to around 10 meters. The range is somewhat shortened by stucco walls, and shortened a lot by block walls. Metal objects, like vehicles, don't seem to affect the range at all.

Data Reliability

The Bluetooth specification does take some measures to ensure the reliability of the data. However, keep in mind that for the vast majority of Bluetooth applications, the odd dropped bit doesn't really matter and in most cases won't even be noticed. Even for keyboards, the user can easily back up and fix the error. For several applications, like headphones and cell phones, the data ages very quickly and becomes useless. For other applications, the data must get through intact. This is controlled by a quality of service setting in one of the data transport layers. The setting may adjusted to indicate that the data has a limited lifetime after which it becomes invalid, or that the data should be delivered within a given time period, or that the data is reliable and should be delivered without error, regardless of how long it takes.

Bluetooth uses forward error correction (FEC), header error check (HEC), and cyclic redundancy checking (CRC) to detect most errors. If errors are detected, the higher quality of service settings will resend the data. However, Bluetooth uses the CCITT generator polynomial for CRC. This polynomial has a higher rate of undetected errors than, for example, the generator polynomial used by CAN. Unlike CAN, Bluetooth is not intended for applications with severe noise.

Broadcast mode, where one master is sending the same message to several or many slaves, sends error check data, but never resends data in response to an error. Although the data is usually sent several times, it is nevertheless described in the specification as inherently unreliable.

If very high reliability is required, then the application must take additional measures on its own. Convolution can be used for a new application, but of course, it can't be readily retrofitted to a legacy application.

Security

Unlike reliability, the Bluetooth specification takes great pains to provide very good security. Security measures are provided at both the application layer and the link layer. All of the security processes are optional, and can be ignored where less security is required.

Even though eavesdropping on a Bluetooth link is not particularly likely, it is theoretically possible, and some applications will need to take additional measures. For the most part, this is left up to the individual application to establish. Before discussing security, some definitions are in order.

Authentication is a procedure for verifying the identity of a remote device. The two devices share a secret key and use a challenge-response scheme to check each other's keys. In this scheme, the verifier sends a random number to the claimant, which processes the number with the key and returns it to the verifier. The master is not required to be the master, and in some applications, both devices may challenge the other using a different random number generated with the same key. After a failed attempt, a waiting interval is imposed before the verifier will start a new authentication on the same claimant. This interval increases for every failure to reduce the usefulness of simply trying a large number of keys. Bluetooth can generate and store the keys required. Although the specification devotes considerable space to a very thorough discussion of authentication, it does not ever explain where authentication takes place: in the application, or in one of the transport layers. The generation and management of keys is so complex that leaving it to each application would seem to be counterproductive, but there is no indication that anything else is in effect.

Bonding is the procedure for performing the first authentication, where a common link key is created and stored for future use.

Pairing starts with a mutual authentication, regardless of the security level, and then requires a user created PIN from both devices. After a successful pairing, the remote device is considered to be a trusted device.

Authorization is a procedure where a device grants a specific level of access to a specific remote device. It assumes that the remote devices identity has already been verified through authentication. Bluetooth does not handle authorization; it is left to the application.

Bluetooth supports security mode 1, also known as "non-security" mode, and security mode 2, also known as "service-level enforced" security. It does not handle security mode 3 "link-level enforced" security.

Bluetooth uses encryption derived from the Massey and Rueppel summation stream cipher method. This method has been widely researched and its strengths and limitations are well documented. Bluetooth uses a high resynchronization frequency to disrupt the more common methods of attack. The encryption key is changed for every data packet.

Power Consumption

Low power consumption is not one of Bluetooth's strongest points. The current drain is highest while waiting for a connection and is about 30 times higher than idle mode. If the application doesn't need the quickest response to a page or inquiry, then the scan duty cycle can be reduced to as low as 0.5% at the expense of response time.

Masters and slaves exchange packets during timeslots. Each timeslot is 625 microseconds. The master sends data during a master timeslot, and the slave must accept or reject the data during the next timeslot. Thus, a master always knows when a packet will be sent and when the acknowledgement will arrive. This means that the master can reduce power without losing data or time. The slave does not know when data will arrive and must assume that data will arrive on every master time slot. It must leave its receiver powered up for most of the master slot duration. The result is higher power consumption for the receiver than for the transmitter. In fact, when no data is being transmitted, the receiver will draw around five times as much current as the transmitter.

The Bluetooth SIG dealt with this by creating low power modes called sniff, park, hold and deep sleep. In sniff mode, the slave spends less time listening for master transmissions. This requires the master to only send in a specified ratio of the total time slots, but communication does continue. In park and hold modes, the slave stays synchronized to the channel, but it does not participate in message exchanges. This reduces the slave power consumption to about that of the master, but that is still nearly 2 mA.

Buy vs. build

Bluetooth is very complex. The current revision of the specification is 1200 pages, which does not include over two-dozen protocols for specific services.

A few companies have produced Bluetooth modules that are intended to be drop-in replacements for RS232 links. They are successful in less demanding applications, which covers more than 99% of all RS232 applications, but links that require high data reliability or guaranteed data availability will have to be tested very carefully. These modules are available for under $70 in small quantities. At this time, they can be built in-house for considerably less, but this product is perfect for manufacture in Asia at dirt-cheap prices. When the market gets large enough to attract Chinese manufacturing, it won't be worthwhile to build your own. CSR in particular has tried to jump-start this process by producing a complete RS232 replacement design guide, including schematics, parts list and a list of the required certification tests.

At this time, there are no commercially available drop-in replacements for communication formats other than RS232. Bluetooth could be considered for low speed, low message rate RS485 networks with eight or fewer nodes at comparatively short distances.

Embedded Bluetooth is probably most common in USB HID products such as keyboards and mice. Bluetooth cannot support the data rates or guaranteed latency requirements for either Full Speed or High Speed USB and so is not a candidate for replacing these links.

At this time, there are three Bluetooth IC providers: CSR, Broadcom, and National Semiconductor.

CSR claims to have 2/3 of the market including Apple Computer, AIRCable, Brainboxes, Dell, Sony, IBM, and TDK among many others. As explained above, CSR has gone to great lengths to make first time Bluetooth designs successful.

Broadcom has devices specifically aimed at particular markets, such as cell phones and keyboards. The keyboard device includes key interface hardware, and key scanning logic in addition to the Bluetooth parts. Broadcom's web site doesn't appear to offer any design aids of any sort for any of its products.

National Semiconductor has a page on its web site devoted to Bluetooth in general, and a Bluetooth IC in particular. However, none of its distributors has any information on it, and inquiries to NSC went unanswered.

Protocols

Of the more than 60 protocols that have been published so far, only a few are of any interest for embedded applications. They are listed here. Even so, most can be ignored for most applications. Only RFCOMM and perhaps SPP require any attention from the application.

L2CAP Logical Link Control and Adaptation Protocol
LAP Lower Address Part
LCP Link Control Protocol
LMP Link Manager Protocol
OBEX Object Exchange Protocol
RFCOMM serial cable emulation protocol
SPP Serial Port Profile

References

Broadcom web site, www.broadcom.com
CSR web site, www.csr.com
National Semiconductor web site, www.nsc.com

BlueCore Bluetooth Qualification Summary, CSR April 2002
RS232 Cable Replacement Example Design for BlueCore2-External May 2003 CSR
Specification of the Bluetooth System version 1.2 November 2003
TDK blu2i Module User Guide