13 Wireless Communications Using Bluetooth

Oge Marques and Nitish Barman

CONTENTS

13.1 Introduction ................................................................................................308
13.2 Overview ....................................................................................................309
  13.2.1 Masters and Slaves ......................................................................310
  13.2.2 Frequency Hopping Spread Spectrum (FHSS) and
          Time-Division Duplexing (TDD) ................................................310
  13.2.3 Piconets and Scatternets ..............................................................310
13.3 Protocol Stack ............................................................................................311
  13.3.1 The Radio Layer..........................................................................313
  13.3.2 The Baseband Layer....................................................................314
      13.3.2.1 Device Addressing......................................................314
      13.3.2.2 Frequency Hopping....................................................315
      13.3.2.3 Link Types (ACL and SCO) .......................................315
      13.3.2.4 Packet Definitions.......................................................316
      13.3.2.5 Logical Channels........................................................319
      13.3.2.6 Channel Control .........................................................320
      13.3.2.7 Error Checking and Correction..................................320
      13.3.2.8 Security.......................................................................321
  13.3.3 The LMP Layer ...........................................................................322
  13.3.4 The L2CAP Layer .......................................................................325
      13.3.4.1 L2CAP Channel Management ...................................327
  13.3.5 The SDP Layer ............................................................................327
13.4 Bluetooth Profiles Specification.................................................................329
  13.4.1 GAP..............................................................................................329
  13.4.2 SDAP...........................................................................................330
  13.4.3 SPP...............................................................................................330
  13.4.4 GOEP...........................................................................................331
13.5 Additional Considerations..........................................................................331
  13.5.1 Power Management .....................................................................331
  13.5.2 Security........................................................................................332
13.1 INTRODUCTION

Bluetooth is a wireless communications standard that allows compliant devices to exchange information with each other. The technology makes use of the globally available, unlicensed ISM (Industrial, Scientific, and Medical) band. Although it was initially developed as a cable-replacement technology, it has grown into a standard that is designed to support an open-ended list of applications (including multimedia applications). As a short-range, low-power technology with data rates of up to 720 kbps, it is ideally suited for use in establishing ad hoc personal area networks (PANs).

The Bluetooth specification emerged from a study undertaken by Ericsson Mobile Communications in 1994 to find alternatives to using cables to facilitate communications between mobile phones and accessories. As this study grew in scope, other companies joined Ericsson’s efforts to utilize radio links as cable replacements. In 1998 these companies – Ericsson, Intel, IBM, Toshiba, and Nokia – formally founded the Bluetooth Special Interest Group (SIG). In July 1999 this core group of promoters published version 1.0 of the Bluetooth specification.

Shortly after the specification was published, the group of core promoters was enlarged further with the addition of four more companies: Microsoft, Agere Systems (a Lucent Technologies spin-off), 3COM, and Motorola.

In addition to the core promoters group, many hundreds of companies have joined the SIG as Bluetooth adopter companies. In fact, any incorporated company can join the SIG as an adopter company by signing the Bluetooth SIG membership agreement (available on the Bluetooth Web site). Joining the SIG entitles an adopter company to a free license to build Bluetooth-based products, as well as the right to use the Bluetooth brand.

The list of adopter companies continues to grow in part because there is no cost associated with intellectual property rights, but primarily because there are so many potential applications and usage models for Bluetooth:

1. Cordless desktop: In this cable-replacement usage model, all (or most) of the peripheral devices (e.g., mouse, keyboard, printer, speakers, etc.) are connected to the PC cordlessly.
2. Ultimate headset: This usage model would allow one headset to be used with myriad devices, including telephones, portable computers, stereos, etc.
3. Automatic synchronization: This usage model makes use of the hidden computing paradigm, which focuses on applications in which devices automatically carry out certain tasks on behalf of the user without user intervention or awareness. Consider the following scenario: A user attends a business meeting, exchanges contact information with other attendees, and stores this information on a PDA. Upon returning to the user’s office, the PDA automatically establishes a Bluetooth link with the user’s desktop PC and the information stored on the PDA is automatically uploaded to the PC. All this happens without the user’s conscious involvement.
There are many other usage models that have been proposed by the Bluetooth SIG as well as other contributors. With increasing numbers of applications being developed around Bluetooth, it is highly likely that the technology will be well accepted by the market.

The remainder of this chapter is organized as follows. Section 13.2 provides a broad overview of the Bluetooth protocol stack and introduces some concepts and terminology. Section 13.3 gives a more-detailed explanation of the key features of the various layers of the Bluetooth protocol stack. Section 13.4 introduces the second volume of the Bluetooth specification, the Profile specification. It serves as a basic introduction to the concept of Bluetooth profiles, and gives a brief description of the fundamental profiles. Section 13.5 discusses security and power management issues, two items that are critical for the acceptance of Bluetooth in the consumer market. Finally, Section 13.6 concludes this chapter with some anecdotal evidence of the interest evinced in Bluetooth application development.

13.2 OVERVIEW

The Bluetooth specification comprises two parts:

1. The core specification, which defines the Bluetooth protocol stack and the requirements for testing and qualification of Bluetooth-based products
2. The profiles specification, which defines usage models that provide detailed information about how to use the Bluetooth protocol for various types of applications

The Bluetooth core protocol stack consists of the following five layers:

1. Radio specifies the requirements for radio transmission – including frequency, modulation, and power characteristics – for a Bluetooth transceiver.
2. Baseband defines physical and logical channels and link types (voice or data); specifies various packet formats, transmit and receive timing, channel control, and the mechanism for frequency hopping (hop selection) and device addressing.
3. Link Manager Protocol (LMP) defines the procedures for link set up and ongoing link management.
4. Logical Link Control and Adaptation Protocol (L2CAP) is responsible for adapting upper-layer protocols to the baseband layer.
5. Service Discovery Protocol (SDP) – allows a Bluetooth device to query other Bluetooth devices for device information, services provided, and the characteristics of those services.

In addition to the core protocol stack, the Bluetooth specification also defines other layers that facilitate telephony, cable replacement, and testing and qualification. The most important layers in this group are the Host Controller Interface (HCI) and RFCOMM, which simply provides a serial interface that is akin to the EIA-232
(formerly RS-232) serial interface. HCI provides a standard interface between the host controller and the Bluetooth module, as well as a means of testing/qualifying a Bluetooth device.

The Bluetooth protocol stack is discussed in greater detail in Section 13.3. In the remainder of this section we will introduce key concepts and terminology that are helpful for an understanding of Bluetooth technology.

13.2.1 **Masters and Slaves**

All active Bluetooth devices must operate either as a master or a slave. A master device is a device that initiates communication with another Bluetooth device. The master device governs the communications link and traffic between itself and the slave devices associated with it. A slave device is the device that responds to the master device. Slave devices are required to synchronize their transmit/receive timing with that of the masters. In addition, transmissions by slave devices are governed by the master device (i.e., the master device dictates when a slave device may transmit). Specifically, a slave may only begin its transmissions in a time slot immediately following the time slot in which it was addressed by the master, or in a time slot explicitly reserved for use by the slave device.

13.2.2 **Frequency Hopping Spread Spectrum (FHSS) and Time-Division Duplexing (TDD)**

Bluetooth employs a frequency hopping technique to ensure secure and robust communication. The hopping scheme is such that the Bluetooth device hops from one 1-MHz channel to another in a pseudorandom manner – altogether there are 79 such 1-MHz channels defined by the SIG. Typically, each hop lasts for 625 μs. At the end of the 625-μs time slot, the device hops to a different 1-MHz channel. The technology ensures full-duplex communication by utilizing a TDD scheme. In the simplest terms this means that a Bluetooth device must alternate between transmitting and receiving (or at least listening for) data from one time slot to the next. The Bluetooth specification facilitates this TDD scheme by (1) using numbered time slots (the slots are numbered according to the clock of the master device), and (2) mandating that master devices begin their transmissions in even-numbered time slots and slave devices begin their transmissions in odd-numbered time slots (Figure 13.1). Both these techniques will be further discussed in Section 13.3.

13.2.3 **Piconets and Scatternets**

Bluetooth technology furthers the concept of *ad hoc* networking where devices within range of each other can dynamically form a localized network for an indefinite duration. *Ad hoc* Bluetooth networks in which all member devices share the same (FHSS) channel are referred to as piconets. A piconet can consist of up to eight devices: a single master and up to seven slave devices. The frequency hopping sequence of the piconet is determined by the Bluetooth device address (BD_ADDR) of the master device. At the time that the piconet is established, the master device’s address and clock are communicated to all slave devices on the piconet. The slaves
use this information to synchronize their hop sequence as well as their clocks with that of the master’s.

Scatternets (Figure 13.2) are created when a device becomes an active member of more than one piconet. Essentially, the adjoining device shares its time slots among the different piconets. Scatternets are not limited to a combination of only two piconets; multiple piconets may be linked together to form a scatternet. However, as the number of piconets increases, the total throughput of the scatternet decreases. Note that the piconets that make up a scatternet still retain their own FH sequences and remain distinct entities.

Scatternets offer two advantages over traditional collocated ad hoc networks. First, because each individual piconet in the scatternet retains its own FH sequence, the bandwidth available to the devices on any one piconet is not degraded by the presence of the other piconets (although the probability of collisions may increase). Second, devices on different piconets are able to “borrow” services from each other if they are on the same scatternet.

13.3 PROTOCOL STACK

The five layers of the Bluetooth core protocol are divided into two logical parts (Figure 13.3). The lower three layers (Radio, Baseband, and LMP) comprise the
Bluetooth module, whereas the upper layers (L2CAP and SDP) make up the host. The interface between these two logical groupings is called the Host Controller Interface (HCI). Note that the HCI is itself a distinct layer of the Bluetooth protocol stack. The reason for separating the layers into two groups stems from implementation issues. In

\[\text{FIGURE 13.2} \text{ Scatternet examples: (a) piconet 1 and piconet 2 share a common slave device; (b) a device acts as slave in piconet 1 and master in piconet 2.}\]
many Bluetooth systems the lower layers reside on separate hardware than the upper layers of the protocol. A good example of this is a Bluetooth-based wireless LAN card. Because the PC to which the LAN card connects typically has enough spare resources to implement the upper layers in software, the LAN card itself only contains the lower layers, thus reducing hardware complexity and user cost. In such a scenario the HCI provides a well-defined, standard interface between the host (the PC) and the Bluetooth module (the LAN card).

13.3.1 The Radio Layer

Bluetooth devices operate in the 2.4-GHz ISM band. In North America, Europe, and most of the rest of the world, a bandwidth of 83.5 MHz is available in the ISM band. In France, however, the ISM band is much narrower. The Bluetooth SIG has actively lobbied the French regulatory bodies to release the full ISM band; France is expected to comply by 2003. Although the Bluetooth specification defines a separate RF specification for the narrower (French) band, we will not consider it here.

The ISM band is littered with millions of RF emitters ranging from random noise generators, such as sodium vapor street lamps, to well-defined, short-range applications, such as remote entry devices for automobiles. Microwave ovens are particularly noisy and cause significant interference. Clearly, the ISM band is not a very reliable medium. Bluetooth overcomes the hurdles presented by the polluted environment of the ISM band by employing techniques that ensure the robustness of transmitted data. Specifically, Bluetooth makes use of frequency hopping (along with fairly short data packets) and adaptive power control techniques in order to ensure the integrity of transmitted data.

FIGURE 13.3 Bluetooth protocol stack.
The Bluetooth Radio specification defines a frequency hopping spread spectrum radio transceiver operating over multiple RF channels. The specification divides the 83.5-MHz bandwidth available in the ISM band into 79 RF channels of 1 MHz each, a 2-MHz lower guard band, and a 3.5-MHz upper guard band. The channel frequencies are defined as

\[ 2402 + k \, \text{MHz} \]

where \( k = 0, 1, \ldots, 78 \).

In order to maximize the available channel bandwidth, the data rate for the RF channels is set at 1 Mbps. Gaussian frequency shift keying (GFSK) is used as the modulation scheme, with a binary 1 defined as a positive frequency deviation from the carrier frequency and a binary 0 defined as a negative frequency deviation.

In addition to specifying the modulation scheme, the Radio specification defines also three power classes (Table 13.1) for Bluetooth devices. Each power class has associated with it a nominal transmission range and a maximum power output. The nominal power output for all three classes is 1 mW (0 dBm).

### TABLE 13.1
Bluetooth Radio Power Classes

<table>
<thead>
<tr>
<th>Power Class</th>
<th>Range of Transmission (meters)</th>
<th>Maximum Power Output (mW/dBm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>1/0</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>2.5/4</td>
</tr>
<tr>
<td>3</td>
<td>100</td>
<td>100/20</td>
</tr>
</tbody>
</table>

13.3.2 THE BASEBAND LAYER

The baseband layer is by far the most-complex layer defined in the Bluetooth specification. It encompasses a wide range of topics, not all of which can be discussed in this short introduction to the Bluetooth technology. What follows is a brief overview of the key aspects of the baseband layer.

13.3.2.1 Device Addressing

The radio transceiver on each Bluetooth device is assigned a unique 48-bit Bluetooth device address (BD_ADDR) at the time of manufacture. Portions of this address are used to generate three types of access codes: the device access code (DAC), the channel access code (CAC), and the inquiry access code (IAC). The DAC and the IAC are used for paging and inquiry procedures. The CAC is specified for the entire piconet and is derived from the device address of the master device. It is used as the preamble of all packets exchanged on that piconet.

With regard to slave devices on a piconet, there are two other addresses of interest. The first is the active member address (AM_ADDR). This is a 3-bit address
that is assigned to an active slave device on the piconet. Because the piconet may have seven active slaves at any given time, a slave’s AM_ADDR uniquely identifies it within its piconet. The second address of interest is the parked member address (PM_ADDR). This is an 8-bit address assigned to slave devices that are parked, i.e., devices that are not active on the piconet but remain synchronized to the piconet’s master. An active slave device loses its AM_ADDR as soon as it is parked. Similarly, a parked device that becomes active immediately relinquishes its PM_ADDR and takes on an AM_ADDR. Note that the reactivated device may or may not be assigned the same AM_ADDR that it was assigned before it was parked.

13.3.2.2 Frequency Hopping

In order to ensure secure and robust data transmission, Bluetooth technology utilizes a frequency hopping spread spectrum technique. Under this scheme, Bluetooth devices in a piconet hop from one RF channel to another in a pseudorandom sequence. Typically, a hop occurs once at the beginning of every time slot. Because each time slot has a 625-μs duration, the nominal hopping frequency is 1600 hops per second. All devices that are part of the same piconet hop in synchrony. The hopping sequence is based on the master’s BD_ADDR and consequently is unique for each piconet.

Because there can be multiple piconets operating in close proximity to each other, it is important to minimize collisions between the devices in different piconets. The specification addresses this issue in two ways. First, the pseudorandom hop selection algorithm is designed to generate the maximum distance between adjacent hop channels. Second, the duration of the time slots is kept very small (625 μs). Time slots of such short duration ensure that even if a collision occurs it will not last long. A collision is unlikely to recur during the next time slot because the piconet will have hopped to a different RF channel.

Strictly speaking, the baseband specification defines a different hopping sequence for different types of operations. The earlier description is that of the channel hopping sequence, which utilizes all 79 RF channels and defines a unique physical channel for the piconet. However, the other hopping sequences (associated with device discovery procedures such as paging and inquiry) utilize only 32 RF channels.

One final note: It was previously stated that the nominal hop rate is 1600 hops per second. This is true only for packets (discussed later) that occupy a single time slot. For packets that occupy three (or five) time slots, the same RF channel is used for transmission during all three (or five) time slots.

13.3.2.3 Link Types (ACL and SCO)

The baseband specification defines two types of links between Bluetooth devices.

1. Synchronous Connection Oriented (SCO) link: This is a bidirectional (64 kbps each way), point-to-point link between a master and a single slave device. The master maintains the SCO link by using reserved time slots
at regular intervals. The slots are reserved as consecutive pairs: one for transmissions from the master to the slave and the other for transmissions in the opposite direction. The master can support up to three SCO links simultaneously. A slave device can support up to three simultaneous SCO links with one master or — if it is the adjoining device on a scatternet — it can support up to two simultaneous SCO links with two different masters. SCO links are used for exchanging time-bound user information (e.g., voice). Packets sent on these links are never retransmitted.

2. Asynchronous Connectionless (ACL) link: This is a point-to-multipoint link between the master and all the slaves on a piconet. An ACL link can utilize any packet that is not reserved for an SCO link. The master can exchange packets with any slave on the piconet on a per-slot basis, including slaves with which it has an established SCO link. Only one ACL link can exist between any master–slave pair. ACL links are used for the exchange of control information and user data, therefore packet retransmission is applied to ACL packets received in error.

### 13.3.2.4 Packet Definitions

The baseband layer defines the format and structure of the various packet types used in the Bluetooth system. The length of a packet can range from 68 bits (shortened access code) to a maximum of 3071 bits. A typical packet (Figure 13.4), however, consists of a 72-bit access code, a 54-bit packet header, and a variable length (0 to 2745 bits) payload.

Every packet must begin with an access code (see Section 3.2.1). The access code is used for synchronization and identification at either the device level (DAC/IAC) or the piconet level (CAC).

The packet header contains link control information, so it is important to ensure its integrity. This is accomplished by applying a 1/3-rate FEC scheme to the entire header. The packet header consists of six different fields:

1. **AM_ADDR** (3-bit field): This field contains the active member address of the slave device that transmitted the packet (or, if the master sent the packet, the AM_ADDR of the slave device to which the packet was addressed).
2. **TYPE** (4-bit field): This field identifies the type of packet, as explained later in this section.
3. **FLOW** (1-bit field): This field is used for providing flow control information on an ACL link. A value of 0 (STOP) indicates a request to the sender to stop sending information, whereas a value of 1 (GO) indicates that the receiver is once again ready to receive additional packets. If no packet is received from the receiver, a GO is implicitly assumed.

![FIGURE 13.4 Packet structure of a typical Bluetooth packet.](image)
4. ARQ (1-bit field): This field is used for acknowledging payload data accompanied by a CRC (cyclic redundancy check). If the payload was received without error, a positive acknowledgment (ACK) is sent; otherwise, a negative acknowledgment (NACK) is sent. An ACK corresponds to a binary 1 and a NACK corresponds to a binary 0. Note that NACK is the default value for this field and is implicitly assumed.

5. SEQN (1-bit field): This field is used for a very simple sequential numbering scheme for transmitted packets. The SEQN bit is inverted for each new transmitted packet containing a payload accompanied by a CRC. This field is required for filtering out retransmissions at the destination.

6. HEC (8-bit field): This field, the header error check, is used for ascertaining the integrity of the header. If the header is received in error, the entire packet is ignored.

The payload portion of the packet is used to carry either voice or data information. Typically, packets transmitted on SCO links carry voice information – with the exception of the DV packet type, which carries both voice and data – and packets transmitted on ACL links carry data. For packets carrying voice information, the payload length is fixed at 240 bits (except DV packets). Packets carrying data, on the other hand, have variable length payloads that are segmented into three parts: a payload header, payload body, and a CRC.

The payload header is either one or two bytes long. It specifies the logical channel on which the payload is to be carried, provides flow control information for the specified channel, and indicates the length of the payload body. The only difference between the one-byte and two-byte headers is that the two-byte headers use more bits to specify the length of the payload body. The payload body simply contains the user host data and the 16-bit CRC field is used for performing error checking on the payload body.

13.3.2.4.1 Packet Types

The packets defined for use in the Bluetooth system (Table 13.2) are identified by the 4-bit TYPE field in the packet header. As is evident from the length of the TYPE field, 16 possible packet types can be defined altogether. Bluetooth packet types can be categorized in four broad groups: control packets, single-slot packets, three-slot packets, and five-slot packets. Single-slot packets require only one time slot for complete transmission. Similarly, three-slot and five-slot packets require three and five time slots, respectively, for complete transmission. The single-slot, three-slot, and five-slot packets are defined differently for the two different link types (SCO and ACL), while the control packets are common to both links.

The four control packets defined in the specification are:

1. NULL: This packet type is used to return acknowledgments and flow control information to the source.

2. POLL: This packet is used by the master to poll slave devices in a piconet. Slave devices must respond to this packet even if they have no information to send.
3. FHS: The frequency hop selection packet is used to identify the frequency hop sequence before a piconet is established or when an existing piconet changes to a new piconet as a result of a master–slave role switch. This control packet contains the BD_ADDR and clock of the sender (recall that the master’s device address and clock are used to generate a pseudo-random frequency hop sequence). In addition, the packet contains also the AM_ADDR to be assigned to the recipient along with other information required in establishing a piconet.

4. DM1: The DM1 (data–medium rate) packet is used for supporting control information on any link. This is necessary, for example, when synchronous information (on an SCO link) must be interrupted in order to supply control information. In an ACL link, this packet can be used to carry user data. The packet’s payload can contain up to 18 bytes of information. The payload data is followed by a 16-bit CRC. Both the data and the CRC are encoded with a 2/3-rate forward error correction (FEC) scheme.

The remaining 12 packet types are defined differently for each link. Some packet types have not yet been defined or have been set aside for future use.

### 13.3.2.4.2 SCO Packets

All packets defined specifically for the SCO link are single-slot packets. They all have a 240-bit payload and do not include a CRC. The first three SCO packets are designed to carry high-quality voice (HV) information (64 kbps) and are defined as follows:

<table>
<thead>
<tr>
<th>Packet Group</th>
<th>Type Code</th>
<th>Packet Name on SCO Link</th>
<th>Packet Name on ACL Link</th>
<th>Number of Slots</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control group</td>
<td>0000</td>
<td>NULL</td>
<td>NULL</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0001</td>
<td>POLL</td>
<td>POLL</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0010</td>
<td>FHS</td>
<td>FHS</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0011</td>
<td>DM1</td>
<td>DM1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0100</td>
<td>Undefined</td>
<td>DH1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0101</td>
<td>HV1</td>
<td>Undefined</td>
<td>1</td>
</tr>
<tr>
<td>Single-slot group</td>
<td>0110</td>
<td>HV2</td>
<td>Undefined</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0111</td>
<td>HV3</td>
<td>Undefined</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>DV</td>
<td>Undefined</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1001</td>
<td>Undefined</td>
<td>AUX1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1010</td>
<td>Undefined</td>
<td>DM3</td>
<td>3</td>
</tr>
<tr>
<td>Three-slot group</td>
<td>1011</td>
<td>Undefined</td>
<td>DH3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>1100</td>
<td>Undefined</td>
<td>Undefined</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>1101</td>
<td>Undefined</td>
<td>Undefined</td>
<td>3</td>
</tr>
<tr>
<td>Five-slot group</td>
<td>1110</td>
<td>Undefined</td>
<td>DM5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>1111</td>
<td>Undefined</td>
<td>DH5</td>
<td>5</td>
</tr>
</tbody>
</table>
• HV1: This packet contains 10 bytes of data and is protected by 1/3-rate FEC. Because one packet can carry about 1.25 ms of speech, HV1 packets need to be transmitted once every two time slots.

• HV2: This packet contains 20 bytes of data and is protected by 2/3-rate FEC. Because one packet can carry about 2.5 ms of speech, HV2 packets only need to be transmitted once every four time slots.

• HV3: This packet contains 30 bytes of data and is not encoded with an error correction scheme. Because one packet can carry about 3.75 ms of speech, HV3 packets need to be transmitted only once every six time slots.

The last packet type defined for the SCO link is the DV (combined data and voice) packet. This packet contains a 10-byte voice field and a data field containing up to 10 bytes of data. The data field is encoded with 2/3-rate FEC and is protected by a 16-bit CRC. It is interesting to note that the voice and data information contained in this packet are treated completely differently. The voice information – which is synchronous – is never retransmitted, whereas the data information is retransmitted if it is found to be in error. So, for example, if the data field of a packet is received in error, the packet is retransmitted (i.e., the data field contains the same information), but the voice field contains new (different) information.

13.3.2.4.3 ACL Packets
Six different packet types have been defined specifically for the ACL link. All the ACL packets are designed to carry data information and are distinguished from one another by two basic criteria: (1) whether they are encoded with a FEC scheme, and (2) how many time slots they require for complete transmission. The packet types protected with an FEC scheme are referred to as medium data rate (DM) packets, whereas the unprotected packets are referred to as high data rate (DH) packets.

There are two DM packet types defined: DM3 and DM5. Both these packet types are similar to the DM1 control packet in that they are protected with a 2/3-rate FEC scheme. The difference is that unlike DM1, which requires one time slot for complete transmission, DM3 and DM5 require three time slots and five time slots, respectively. Additionally, as one might expect, DM5 packets can carry more data than DM3 packets, which in turn can carry more data than DM1.

Three different DH packet types are defined for the ACL link: DH1, DH3 and DH5. All these packets can carry more data than their DM counterparts because the DH packets are not encoded with an error correction scheme. As with the DM packet types, DH3 and DH5 carry more information than DH1.

The sixth packet type defined for the ACL link is the AUX1 (auxiliary) packet type. This packet type is very similar to DH1 except it does not contain a CRC code.

The remaining packet types have not been defined as yet and have been set aside to accommodate packet types that may be required in the future.

13.3.2.5 Logical Channels
Bluetooth specification defines five logical channels for carrying either link control and management information or user data.
The two channels that carry link control information are the link control (LC) channel and the link manager (LM) channel. The LC logical channel is mapped to the header of every packet (except the ID packet) exchanged on a Bluetooth link. This channel carries low level link information such as flow control, ARQ, etc. The LM channel is typically carried on protected DM packets and it can utilize either link type (SCO or ACL). A packet used for the LM channel is identified by an L_CH value of 11 in the packet’s header. The LM channel carries LMP traffic and therefore takes priority over user data channels.

Three logical channels are defined for conveying user data: the user asynchronous (UA) channel, the user isochronous (UI) channel, and the user synchronous (US) channel. The UA and UI channels are normally mapped to the payload of packets transmitted over an ACL link, but they can be mapped also to the data field of the DV packet on an SCO link. The US channel carries synchronous user data (e.g., voice communication) and is mapped to the payload field of packets transmitted over an SCO link.

13.3.2.6 Channel Control

Channel control deals with the establishment of a piconet and adding/removing devices to/from a piconet. The specification defines two primary states for a Bluetooth device: standby and connected. The standby state is the default, low-power state. A device is in the connected state when it is a member of a piconet, i.e., when a device is synchronized to the hop sequence of a piconet.

Devices do not transition directly from the standby state to the connected state, and vice versa. In fact, the specification describes several substates that a device must transition through in order to move from one primary state to the other. These substates are associated with inquiry and paging procedures that are required for slave devices to join or leave the piconet. Figure 13.5 depicts the state transitions involved in going from the standby state to the connected state and back. The observant reader will note that there is a direct transition from the connected state back to the standby state. This direct transition occurs only in case of a hard reset. Normally, devices must methodically detach from the piconet, which requires transitioning through different substates.

The Bluetooth system makes provisions for connected devices to be put into one of three low-power modes while remaining synchronized to the hop sequence of the piconet. These modes (discussed further in Section 3.3) can be considered as substates of the connected state (Figure 13.5).

13.3.2.7 Error Checking and Correction

Bluetooth provides a variety of error checking and error correction mechanisms. Bluetooth packets can be checked for errors at three levels. When a packet is received, the receiver immediately checks its channel access code (recall that the CAC is the preamble for all packets). If the CAC embedded in the packet does not match the CAC of the piconet, then either the packet was received in error, or the packet was destined for another piconet. In either case, the packet is discarded. If the CAC
matches the CAC for the piconet, the next error checking mechanism is employed. This involves checking the header error check. If the HEC indicates that the header data has irrecoverable errors, the packet is again discarded. If the packet passes this second error-checking test, the final error checking mechanism involves checking the CRC to check the integrity of the packet payload.

Bluetooth also makes use of three different error correction schemes. The first two schemes are 1/3-rate FEC and 2/3-rate FEC. Using these two schemes, it may be possible to recover from bit errors without having to retransmit the packet. However, if the packet payload cannot be recovered despite the use of these two correction schemes or if no FEC was used, then an ARQ scheme is used to request retransmission of the packet. For more information on either FEC scheme, please refer to the Bluetooth specification.2

13.3.2.8 Security

Security is of paramount importance in any wireless (RF) communication. It is even more important in the case of Bluetooth, which aims at becoming a ubiquitous, de facto standard for wireless communications. With this in mind, the specification provides for security mechanisms in more than one layer of the protocol stack. At the baseband (lower link level), the facility for link security between two devices relies on two different procedures:
1. Authentication: Procedure used to verify the identity claimed by a Bluetooth device
2. Encryption: Procedure used to encode user data so that it is unintelligible until decoded using a specific key

The security algorithms defined in the baseband specification utilize the following four parameters:

1. BD_ADDR: Publicly known 48-bit address of the device
2. Authentication key: 128-bit secret key preconfigured with the device
3. Privacy key: Variable length (4 to 128 bits) secret key that is preconfigured also with the device
4. Random number: 128-bit number used for device authentication

In addition to these key responsibilities, the baseband layer addresses other issues, including transmit/receive timing, data whitening, and encoding of audio information for transmission over Bluetooth links.

13.3.3 The LMP Layer

The Link Manager Protocol is responsible for establishing and maintaining ACL and SCO links, and establishing, managing, and terminating the connections between different devices. LMP provides a mechanism for the link managers (LMs) on separate devices to exchange messages containing link management information with each other. These messages are sent as LMP protocol data units (PDUs) and are carried in the payload of single-slot packets (DM1 or DV) transmitted on the LM logical channel.

Because LMP PDUs are used for link management, they have been assigned a very high priority. Consequently, LMP PDUs may be transmitted during slots reserved for an SCO link. The PDU contains three fields:

1. A 1-bit transaction ID field, which specifies whether the link management transaction was initiated by the master (0) or the slave (1) device
2. A 7-bit OpCode field, which identifies the PDU and provides information about the PDUs payload
3. A payload field, which contains the actual information necessary to manage the link

Many PDUs have been defined for the different transactions defined in the specification. Two of the most general response PDUs are LMP_Accepted and LMP_Not_Accepted. Although many link management transactions have been defined in the specification, not all of them are mandatory. However, the LM on every Bluetooth device must be able to recognize and respond to any of the specified transactions. In case the LM on the receiving device does not support a particular transaction, it must still respond to the requesting device’s LM with an LMP_Not_Accepted PDU.
In the remainder of this section we will briefly mention some of the more-important link management transactions defined in the specification. These transactions are grouped into three broad categories: link control, power management, and security management.

Link management transactions used for link control include:

- **Connection establishment**: Before two devices can exchange L2CAP information, they must first establish a communications link (Figure 13.6). The procedure for establishing a link is as follows: the LM of the device requesting the connection sends an LMP_Host_Connection_Req PDU to the receiving device. The LM of the receiving device may then either accept or reject the request. If the request is accepted, the LMs of the two devices further negotiate the parameters of the link to be set up. When this negotiation is completed, the LMs of both devices send an LMP_Setup_Complete PDU to each other and the link is established.

- **Link detachment**: When a device wishes to terminate its link to another device, it issues an LMP_Detach PDU. The receiving device cannot reject the detach request and the link between the two devices is immediately terminated.

- **Clock and timing information**: LMP PDUs in clock-and-timing-related transactions are used by devices to obtain clock offsets and slot offsets from each other. Support for many of these PDUs is mandatory because link timing cannot be established or negotiated without them.

- **Master–slave role switch**: This transaction plays an important role in the establishment of a scatternet.\(^5\) The device that wants to switch roles
initiates the process by sending an LMP_Switch_Req PDU. If the request is accepted, the role switch occurs after the two devices exchange some timing information.

- Information exchange: Transactions of this type allow devices to request information from each other. As an example, a (local) device may want to inquire another (remote) device about the optional link manager features supported by the receiving device. In that case, the local device sends an LMP_Features_Req PDU to the remote device. In addition to requesting a list of features supported by the remote device, the PDU contains information about the optional features supported by the local device. When the remote device receives the request, it sends an LMP_Features_Res PDU back to the local device. The LMP_Features_Res PDU contains information about the optional features supported by the remote device. At the end of the transaction, both devices are aware of the optional services supported the other.

Power management transactions allow Bluetooth devices the flexibility to conserve power when they are not actively exchanging information. The transactions in this category include:

- Sniff mode: While an active slave must monitor every even-numbered time slot for transmissions from the master, a slave device in sniff mode has to monitor far fewer slots. Note that slots reserved for an SCO link are not affected by sniff mode. A master may force a slave device into sniff mode by sending an LMP_Sniff PDU to the slave. Alternatively, either the master or the slave may request that the slave device be placed in sniff mode by issuing an LMP_Sniff_Req PDU.

- Hold mode: In an established piconet, there may be times when a slave device will not be addressed for a significant duration. In such instances, it is highly desirable to place the slave device in sleep-like mode for the duration of time that it will not be addressed, in order to conserve power. In hold mode, the ACL link between the master and slave is temporarily suspended (i.e., there is no ACL traffic between the master and slave). The duration for which the link is suspended is called the hold time and is stipulated at the time the link is suspended. Note that any SCO links between the master and slave device are unaffected. As with sniff mode, the master may either force a slave into hold mode (LMP_Hold PDU) or either the master or the slave may request that the slave be placed in hold mode (LMP_Hold_Req).

- Park mode: In the sniff and hold modes, the slave device is considered an active member of the piconet (i.e., it is still one of the seven active slave devices on the piconet). In park mode, however, the slave device is no longer active on the piconet; however, it still remains synchronized to the piconet. The advantage is that if the device needs to rejoin the piconet, it can do so very quickly. Interested readers are referred to the LMP specification for further information.
Security is addressed in more than one layer of the Bluetooth protocol stack. In the LMP layer, the key security management transactions are device authentication and link encryption; while the former is a mandatory feature of all Bluetooth devices, the latter is optional.

- **Device authentication:** This transaction is based on a challenge response scheme. The transaction begins by one device (the verifier) sending an LMP_Au_Rand PDU to another device (the claimant). The PDU contains a 128-bit random number. The claimant uses this number as the input to an encryption function. The output of this function is then transmitted back to the verifier. If the value received by the verifier matches the value it expects, the claimant is authenticated. At the end of this transaction, the verifier and claimant may swap roles in order to authenticate the link in the opposite direction. The authentication transaction is far more complex than the simple procedure outlined here. A much more detailed explanation can be found in the Bluetooth LMP specification.

- **Link encryption:** Link encryption is often necessary to protect the privacy of the data transmitted over a Bluetooth link. This transaction is initiated when a device issues an LMP_Encryption_Mode_Req PDU. The encryption mode determines whether the encryption is to be applied to a point-to-point link only or to broadcast packets as well. If the request is accepted, the devices negotiate the size of the encryption key to be used by exchanging LMP_Encryption_Key_Size_Req PDUs. Once the encryption key size has been determined, the devices begin the encryption process by issuing an LMP_Start_Encryption PDU.

### 13.3.4 The L2CAP Layer

The L2CAP layer serves to insulate higher-layer protocols (including non-Bluetooth-specific protocols such as TCP/IP, PPP, etc.) from the lower-layer Bluetooth transport protocols. In addition, it supports protocol multiplexing with respect to higher-layer protocols. Another key feature of the L2CAP layer is that it facilitates segmentation and reassembly of higher-layer packets. Note that L2CAP itself does not segment (or reassemble) packets. It simply provides packet length information to the higher-layer protocols, which allows those protocols to segment (and reassemble) the packets submitted to (and received from) the L2CAP layer. L2CAP supports quality-of-service (QoS) features by allowing the exchange of QoS information between the L2CAP layers on different devices.

The L2CAP layer messages are sent in packets known as L2CAP PDUs. These PDUs are carried on ACL packets transmitted on the user asynchronous logical channel. Because these PDUs may be carried on multislot packets, the L_CH field in the payload header of the ACL packets is used to indicate whether the current packet is the start of the L2CAP PDU (denoted by an L_CH value of 10) or a continuation of the L2CAP PDU (denoted by an L_CH value of 01). The L2CAP layer does not itself guarantee the reliable transmission of L2CAP PDUs. It assumes that the packet retransmission facility provided by the baseband layer is sufficient for ensuring a reliable communication channel.
All communication between the L2CAP layers on different devices occurs over logical links referred to as channels (Figure 13.7). These channels are identified by the two endpoints (one on the first device, the other on the second device) of the link. Each endpoint is assigned a unique 16-bit channel identifier (CID). The CIDs are assigned locally, i.e., the endpoint local to a particular device is assigned its CID by the L2CAP layer of that same device. These endpoints are in turn uniquely associated with some recipient (e.g., higher-layer protocol) to which the payload of the L2CAP PDU is delivered. The CIDs are administered locally and the scheme for assigning CIDs is left up to the implementer. The only exception is that some CIDs have been reserved (Table 13.3) for specific channels.

![Figure 13.7](image-url)
The Bluetooth specification defines three types of L2CAP channels:

1. Connection-oriented (CO) channel: This is a persistent channel that supports bidirectional communication and requires the exchange of signaling information before it can be established.

2. Connectionless (CL) channel: CL channels are unidirectional, are not persistent, and are typically used for broadcast messages. If a device wants to respond to a L2CAP transmission on a CL channel, it must send its response on another channel. Also, CL channels allow the L2CAP layer to provide protocol multiplexing (refer to the specification for additional details).

3. Signaling channel: This is very similar to the CO channel except that its CID (and other channel parameters) are fixed, and therefore no signaling information is exchanged in order to establish the channel. Indeed, the signaling channel is itself used for exchanging signaling information.

### 13.3.4.1 L2CAP Channel Management

In order for a CO channel to be established or terminated, signaling information has to be exchanged between the local and remote devices. This signaling information is exchanged by means of a request-and-response transaction mechanism. The signaling commands used in this transaction are L2CAP_Connection_Req, L2CAP_Connection_Res, L2CAP_Configuration_Req, L2CAP_Configuration_Res, L2CAP_Disconnection_Req, and L2CAP_Disconnection_Res. There are additional signaling commands defined in the specification, but they will not be discussed here.

The procedure for establishing a CO channel is as follows: the local device (i.e., the device that wants to establish the CO channel) sends an L2CAP_Connection_Req command to the remote device. The command contains the source CID (i.e., the CID of the endpoint on the local device) and other signaling information. If the remote device accepts the connection request, it sends an L2CAP_Connection_Res command back to the local device. The response command contains the destination CID (i.e., the CID of the endpoint on the remote device), as well as other signaling information. Once the CO channel has been established, it has to be configured. The local and remote devices negotiate channel configuration by exchanging L2CAP_Configuration_Req and L2CAP_Configuration_Res commands. If the two devices cannot agree on configuration parameters, the CO channel is either terminated or the default configuration parameters (implementation dependent) are used. The most important of the configuration parameters negotiated between the two devices are the QoS parameters.

### 13.3.5 The SDP Layer

Bluetooth technology was developed to support mobility and facilitate the formation of ad hoc networks. This scenario is qualitatively different from “static” networks in which member devices do not constantly leave/join the network. In an ad hoc network, devices are free to join or leave the network at whim. This presents a
complex challenge, because a device leaving a network may deprive the network of some service that only that device provides. On the other hand, a device that joins a network (piconet) may provide a service that was previously not available on any other device in that network. So how does any device on the network know what services are available on the network at any given time? This is precisely the issue addressed by the SDP.

SDP utilizes the concept of a client/server model. Clients are devices that are searching for services, whereas servers are devices that provide services. These roles are entirely interchangeable, i.e., any device can be either a client or a server, depending on whether it is using a service provided by another device or providing a service to another device. SDP requires that all devices (in their role as servers) maintain a service registry. The service registry is a collection of service records that provide information (service attributes) about the services provided by the device.

SDP specifies two types of service attributes: (1) universal service attributes, which could apply to any class of service (e.g., printing, telephony, LAN access, etc.); and (2) service-specific attributes, which are meaningful only in the context of a particular service class. It is important to note that the universal service attributes are not necessarily mandatory. In fact, the specification mandates only two such attributes: the service class attribute, which identifies the class or type of service, and the service record handle, which is a pointer to the service record of that service.6

Service discovery is accomplished by means of SDP transactions. SDP transaction information is communicated using SDP PDUs. In order for service discovery to work, it is necessary that the services are represented in a standard format, one that both the client and the server can use to refer to the service. SDP uses universally unique identifiers (UUIDs) to represent the services. Typically, two transactions are required in order for a client to obtain service information from the server: a service search transaction, and a service attribute transaction. The first transaction is initiated by the client when it sends an SDP_Service_Search_Req PDU to the server. Upon receiving this request, the server responds to the client with an SDP_Service_Search_Res PDU. The response PDU returns a list of service record handles that match the service requested by the client. Of course, if the server does not support the requested service, it still responds to the client, but the list of service record handles is empty. Following the response from the server, the client begins the second transaction by sending an SDP_Service_Attribute_Req PDU to the server. The parameters of this PDU are the service record handle (for the service of interest) and the attribute IDs of the attributes desired by the client. When the server receives the SDP_Service_Attribute_Req PDU from the client, it responds with an SDP_Service_Attribute_Res PDU that contains a list of attributes (attribute ID and the corresponding value) retrieved from the service record specified by the service record handle supplied by the client. At the end of the second transaction, the client should have enough information to connect to the service. It is important to note that SDP does not actually provide the client with the requested service. If the client wants to connect to the service, it utilizes some other protocol to do so.

There is a third type of transaction defined in the SDP specification, but it is simply a composite of the first two transactions, and is not discussed here. Interested readers may refer to the specification itself for additional details.
13.4 BLUETOOTH PROFILES SPECIFICATION

As previously mentioned, the Bluetooth specification includes not just the core protocol specification, but also a second volume referred to as the Profiles specification. The Bluetooth SIG made a conscious decision to make this second volume a part of the specification. This makes sense considering that one of the key features of Bluetooth technology is the emphasis on universal interoperability. Bluetooth devices from different manufacturers are expected to work seamlessly with each other. In order to facilitate this seamless interoperability, the Bluetooth specification provides usage models for the many different types of foreseen applications built around Bluetooth technology. These usage models are termed profiles in the specification. It should be noted that this part of the specification does not limit implementers to the profiles defined therein; it simply provides guidelines for using Bluetooth technology for different application types. Indeed, as more applications are devised for Bluetooth, it is inevitable that new usage models will be added to the profiles specification.

The Bluetooth profiles specification defines 13 different profiles (Figure 13.8), which are logically divided into two types. The first are fundamental profiles, which are essentially building-block profiles for the other type of profiles, the usage profiles. All usage profiles inherit from at least one of the fundamental profiles. The four fundamental profiles are briefly discussed in the following sections.

13.4.1 GAP

The generic access profile (GAP) is the fundamental Bluetooth profile. All other profiles stem from GAP. GAP defines the key features necessary for Bluetooth devices to successfully establish a baseband link. The features defined in GAP are:
• Conformance: Every Bluetooth device that does not support some other profile must at least conform to GAP. Essentially, this means that GAP specifies certain features that must be implemented in all Bluetooth devices.
• Discovery procedures: The minimum set of procedures required for a Bluetooth device to discover another Bluetooth device.
• Security procedures: Procedures required for using the different security levels.
• Link management facilities: Facilities that ensure that Bluetooth devices can connect to each other.

In addition to these features, GAP defines also the mandatory and optional modes of operation for a Bluetooth device. Please refer to the specification for further information. Finally, GAP defines a standard set of terminology that is to be used with respect to user interfaces developed for Bluetooth devices. Defining standardized terminology ensures that users of the technology will recognize Bluetooth functionality across different user interface designs.\(^5\)

13.4.2 SDAP

The service discovery application profile describes, in general terms, how applications that use the SDP should be created, and how they should behave. Fundamentally, SDAP specifies which services an SDAP-based application should provide to its users. These services are defined as “service primitive abstractions,” and four such primitives are defined:

1. ServiceBrowse: Used by a local device when conducting a general search for services available on a set of remote devices
2. ServiceSearch: Used by a local device when searching for a specific service type on a set of remote devices
3. EnumerateRemDev: Used by a local device to search for remote devices in its vicinity
4. TerminatePrimitive: Used to terminate the operations initiated by the other three primitives

13.4.3 SPP

The serial port profile is concerned with providing serial port emulation to two devices that want to utilize a Bluetooth link for serial communication. As can be expected, SPP uses the RFCOMM protocol for providing serial port emulation. The key feature of the SPP is that it outlines the procedures necessary for using the RFCOMM protocol to establish a serial link between two devices. The overarching goal of the SPP is to ensure transparency, i.e., an application using the emulated serial link should not be able to distinguish it from a physical serial link. By defining the SPP profile, the SIG has assured that legacy applications (that make use of serial communication links) will not have to be modified in order to use Bluetooth.
13.4.4 GOEP

The generic object exchange profile is based on the object push/pull model, as defined in IrDA’s OBEX layer. GOEP distinguishes between a client device and a server device. The client device is the device that initiates the object exchange operation by requesting the OBEX service from the server. The client either pushes a data object onto or pulls a data object off of the server device. The server device is the device that provides the client device with this push/pull object exchange service. Note that there is no correlation between master/slave (in the context of Bluetooth) on the one hand, and client/server in the context of OBEX on the other. In the simplest terms, GOEP defines the primitives that allow objects to be exchanged between a client and server. The two most important of these primitives are object push and object pull. An interesting side note regarding GOEP is that it was originally developed to provide Bluetooth devices with a synchronization capability, but during the course of development, it grew into the concept of IrDA interoperability.

The remaining nine profiles are usage profiles. As stated earlier, these profiles inherit features from at least one of the four fundamental profiles. Although we will not discuss the usage profiles, they are listed below for reference:

1. Cordless telephony
2. Intercom
3. Headset
4. Fax
5. Dial-up networking
6. LAN access
7. Object push
8. Synchronization
9. File transfer (FTP)

13.5 ADDITIONAL CONSIDERATIONS

13.5.1 Power Management

Bluetooth technology is motivated by the desire to provide a universal interface to battery-driven portable devices. As such, the technology makes several provisions for effective power management in order to conserve power. Some of the key power management features are implemented at the micro level. The first such feature is identified with respect to frequency hopping. The mechanism for frequency hopping is defined such that no dummy data has to be exchanged in order to keep the master and slave devices synchronized to each other. This obviates the need for the transceivers on both the master and slave devices to periodically wake up and transmit dummy packets. The second feature is inherent in the packet format. Because Bluetooth packets begin with the access code of the piconet for which they are intended, a device that is listening on the piconet’s hop frequency during its receive slot can quickly ascertain whether the packet carried on the current hop frequency is intended for its piconet. If the device determines that the packet was not intended
for its piconet, the device’s receiver can go to sleep for the remainder of the time slot. Moreover, the packet header contains information about the length of the payload. So, if the payload is very small, the device does not have to keep its receiver on for the entire duration of the time slot. It can put its receiver to sleep as soon as the entire payload has been received. Aside from the micro-level power management features discussed here, Bluetooth provides support for macro-level power management in that it allows devices to be put in any one of three power saving modes: hold, sniff, and park (see Section 13.3.3).

### 13.5.2 Security

Because Bluetooth is an RF-based wireless technology, data exchanged between Bluetooth devices can be easily intercepted. Clearly, there is a need to protect personal and private data from would-be eavesdroppers. The Bluetooth SIG has made a conscious effort to provide various security mechanisms at many different levels of the specification. To begin with, the fact that the technology employs a frequency hopping spread spectrum technique to establish a channel for communication itself provides a certain degree of security. Consider that without knowing the hop sequence of the piconet, the eavesdropper will not know which 1-MHz channel to listen to during the next 625-μs time slot. But there are other intentional security mechanisms provided in the Bluetooth specification.

At a macro level, the specifications provide three security modes for a Bluetooth device. Devices in security mode 1 never initiate any security procedures. Additionally, devices in this mode are not required to support device authentication. In security mode 2, devices do not need to initiate security procedures until an L2CAP channel is established. Once the L2CAP link is established, the device can decide which security procedures to enforce. Security mode 3 is the most stringent of the three security modes. In this mode, security procedures are initiated at the link level, i.e., before any link is established between devices.

There are two key security procedures defined in the specification: device authentication and link encryption (see Section 13.3). In addition to the security mechanisms provided by Bluetooth the lowest levels, more-advanced security mechanisms can be employed at higher layers.4,7,8

### 13.6 Concluding Remarks

Although the Bluetooth profiles specification defines usage models for several foreseen applications of Bluetooth technology, these are by no means the only applications that can be built around Bluetooth. In fact, new applications are constantly being defined. One source for innovative new applications is the Computer Society International Design Competition.9 In 2001 and 2002, this international competition, sponsored by the IEEE Computer Society, focused on Bluetooth technology. The competition required student teams to build applications around Bluetooth technology. Some of the applications envisioned by the student teams have prompted members of the SIG to consider new profiles (e.g., a profile designed specifically
for medical devices). All this development activity bodes well for the acceptance of Bluetooth technology by the consumer market.

In this chapter we have endeavored to present a salient, albeit cursory overview of Bluetooth technology. It must be borne in mind, however, that Bluetooth is a feature-rich and well-defined wireless communications standard. Indeed, the Bluetooth specification spans more than 1500 pages. Interested readers are encouraged to refer to the specifications,\textsuperscript{2,3} as well as several publications that discuss the technology in much greater detail.\textsuperscript{5,10}

**ACKNOWLEDGMENT**

Nitish Barman gratefully acknowledges the support and encouragement provided by his family (especially his brother) and the generous assistance (proofreading and feedback) provided by his good friends and colleagues, Ms. Shefat Sharif and Mr. Scott Bowser.

**References**
