EE Times Connecting the Global Electronics Community

designlines WIRELESS & NETWORKING

Design How-To

Creating audio applications with Bluetooth

Jason Hillyard, Senior Software Engineer,
Widcomm, Inc. San Diego, Calif..

4/18/2003 01:18 PM EDT

Creating audio applications with Bluetooth

Recently, the Bluetooth community has developed specifications that define how to use streaming audio over a Bluetooth link. This opens up the technology to a whole new class of audio devices, such as wireless stereo headsets, speakers, and portable MP3 players, to name a few.

A frequency hopping spread spectrum (FHSS) radio system operating in the 2.4 GHz unlicensed band, Bluetooth's low power transmissions allow a typical range of about 10 meters. Devices connect to each other to form a network known as a piconet, with up to seven active devices in the piconet. The maximum data throughput between devices is approximately 723 kbit/sec with the data capacity shared between devices on the piconet.

Bluetooth has a protocol stack to transfer data and implement the advanced features required by applications. The protocol stack consists of several different protocols designed for different purposes. The profiles, or applications, reside above the protocol stack. Bluetooth also has a lower protocol stack for link management and baseband control. Bluetooth hardware implementations are typically highly integrated systems consisting of one or two chips.

The 723 kbit/sec throughput of a Bluetooth link is suitable for streaming audio using MP3 or other codec formats. Bluetooth streaming audio is defined by three Bluetooth specifications covering the protocol and profiles: Audio/Video Distribution Transport Protocol (AVDTP), Generic Audio/Video Distribution Profile (GAVDP), and Advanced Audio Distribution Profile (A2DP).

In addition, these Bluetooth specifications there are several ISO/IEC and internet RFC specifications used for Bluetooth streaming audio.

The bulk of the Bluetooth streaming A/V system is implemented in the AVDTP protocol. The protocol is divided into four subsystems: Signaling, stream management, recovery, and adaption.

A device developer implementing Bluetooth streaming audio needs to consider several issues which are not fully covered by the specifications. There are the features required by the AVDTP protocol implementation: Which optional features should be implemented? What

about multiple streams? Then there are issues related to data flow and synchronization which are beyond the scope of the specifications.

In a typical hardware implementation for streaming audio playback, data received over the Bluetooth link is processed by the protocol stack and passed to an application, which takes the audio stream data and sends it over a hardware interface to the audio IC. The audio IC decodes the digital audio and converts the signal to analog.

Implementing AVDTP with the minimum required features is about the same complexity as implementing RFCOMM; that is to say, somewhat complex. Implementers should consider supporting multiple streams and multi-point now. For simple streaming audio device examples in many consumer applications, optional features such as recovery, reporting, header compression, and multiplexing are not required; devices can perform adequately without them.

In most Bluetooth implementations the data flow problem boils down to this: maintaining a data transfer with a constant bit rate on a Bluetooth link. If data is sent to slowly the audio decoder will run out of stream data to process, causing an audible error. Lost data packets may also cause the same problem. On the other hand if data is sent to quickly then data will buffered up at the audio decoder, eventually causing congestion or data loss when the device runs out of buffer space. Since there is no flow control mechanism built into AVDTP or L2CAP other mechanisms must be used to prevent data loss.

The mechanism used by the audio source, or device sending the stream, depends on the type of source. If the source is "live" and audio stream data is provided by an audio encoder, then the encoder itself will provide the constant bit rate. If the source is from a file, then a timer must be used to maintain a constant bit rate.

In an MP3 application, suppose the device is sending an MP3 stream from a file encoded at 128 kbit/sec and 48 kHz sample frequency. This means an MP3 audio frame 384 bytes long is sent every 24.0 msec. So, if the device simply sets a periodic timer for 24.0 msec and sends a packet when the timer expires the constant bit rate will be maintained.

Timing, size options

If the SBC frames are small with a short period, some devices may have problems using timers or processing data at such short intervals. This suggests that rather than send a small packet containing a single frame at very short intervals we would rather send a larger packet containing several frames at longer intervals. Also, notice the maximum size of MP3 frames. This gives us an idea of how large to set the L2CAP MTU of the AVDTP transport channel such that audio frames do not need to be fragmented across AVDTP packets.

What if the timer isn't so accurate and the packet actually gets sent at 20 msec or 29 msec? If a packet arrives late the audio decoder runs out of data and we hear a glitch. So, even a little inaccuracy can cause big problems if we are relying on every packet to be sent right on time.

A better approach would be to give ourselves some slack in the data flow. Assuming the device receiving our stream can buffer up at least a few packets, we can send a number of packets as fast as possible when streaming starts. This helps with timer inaccuracy and data delayed by lost packets as well. But how many packets can we buffer up? This depends on the implementation of the device receiving the stream.

The device receiving the stream can also improve the data flow. Regardless of how fast or slow, the peer is sending the stream, the device receiving the stream can smooth out the

flow by delaying playback until a number of packets have been received. This helps with timer inaccuracy and data delayed by lost packets.

When more than one stream is transferred between Bluetooth-connected devices the stream playback must be synchronized. Consider the example wireless PC speakers. The PC sends each speaker a Bluetooth audio stream. There are actually two synchronization problems in this example. First, the audio playback of the two speakers needs to be synchronized with each other. Second, the audio playback needs to be synchronized with the display on the PC. Although the Bluetooth specifications do not cover synchronization issues, there are some features of the system we can use to help solve these synchronization problems.

Every Bluetooth device has a free-running system clock which determines the timing of the frequency-hopping transceiver. In a piconet, the slave devices are synchronized to the master's system clock. The speakers will both be synchronized to the Bluetooth clock timing of the PC. Depending on the implementation of the Bluetooth chip, it may be possible for an application to derive a timer based on this clock. This clock can be used in conjunction with the RTP presentation timestamp in the packet to synchronize the playback. Therefore it is possible to use the piconet timing as a synchronization source between the two speakers.

The second part of the synchronization problem boils down to how much delay is present from when the PC sends the audio stream to when the speakers play it back. Studies show that a delay larger than 80 msec can be noticeable in situations like this. However, it may be desirable for the device receiving a stream to buffer up a few audio frames before playback to help maintain a constant data rate on the link. This 80 msec limit can be an upper bound of how many frames to buffer. For example, an MP3 stream sampled at 44.1 kHz has a frame period of 26.122 msec. Therefore, no more than three frames should be buffered to keep delay under than the limit.

This article was excerpted from ESC paper 540, titled "Creating Audio Applications with Bluetooth A/V."

See related chart

More Related Links

Security Outlook Darkens at RSA

Radio Over Fiber Paves Way for Future 5G Networks

Wireless Researcher Snags Prize

SDR's Hard Side Shown in DARPA Hackfest

Web Giants Want Optical, Flash Advances