Bluetooth’s Next Challenge

Greg Burns

With the frenzy of interconnecting everything to everything, the next iteration of BT definitely has its work cut out for it.

BT chips are now shipping at a rate of more than three million per week, more than all other short range RF wireless chips combined (as of November 2004).

BT Wireless Technology defines an amazing collection of interoperability standards, known as profiles, for application scenarios ranging from file transfer, to imaging, printing, network access, remote control and streaming audio and video. Products currently incorporating BT include cell phones, headsets, automobiles, consumer electronics, printers, cameras and medical equipment from manufacturers around the world.

As with most emerging technologies, BT took longer to become mainstream than originally forecasted. But also like most successful emerging technologies, BT is having a much greater effect on the market and is reaching into many more products than originally anticipated.

BT was originally positioned as a high-end feature on cell phones, eliminating the tangle of wires between phones and headsets, phones and hands-free units and phones and PCs.

But from the beginning, visionaries at Ericsson, along with IBM, Intel, Microsoft, Motorola, Nokia, Toshiba and others, recognized that BT was a technology that could be applied to many more applications as a wireless connection of “everything to everything.”
However, there are barriers to the widespread adoption of BT in devices outside the mobile phone space. Two of which are *complexity* and *cost* — not so much the complexity of adding the BT radio, because the hardware interfaces are standardized, but the complexity of integrating the required BT protocol software stacks and in developing BT applications.

In a mobile phone platform, the BT software development cost can be easily amortized across hundreds of millions of handsets. However, this is not the case for the lower volumes of consumer electronics devices, audio/video receivers for example, or in vertical markets, such as medical devices. Furthermore, software integration requires extensive expertise in embedded systems and a deep

familiarity with BT specifications. Finally, the product must be tested for interoperability and compliance with the appropriate BT specifications to obtain BQB certification.

Overcoming these barriers requires a new set of powerful software tools that enable rapid integration of BT into new and existing designs that up to this point haven’t been offered to the marketplace. These tools eliminate the need for in-depth familiarity with the BT specification and the mysteries of RF design, along with providing guarantees for reliability and compliance. These tools would need to enable the lay engineer to integrate BT rapidly and affordably into virtually any device, truly empowering the vision of “everything connected to everything.”

BT Development Platforms — Protocol Stack and Profiles

While there are a number of proprietary products that are available for software stack development, most current offerings are a bit too complex. There is a convincing argument that manufacturers needed an easier way to integrate BT technology into their products. To address this issue, the BT development platform, BHAPI, was defined. While still a proprietary platform, its interface makes BT development much easier for inexperienced BT developers.

A BT protocol stack is a collection of layered software components that interface with a BT radio module. A BT radio module is one or more chips that implement what is commonly referred to as a BT baseband. The baseband provides low-level functionality for managing radio communication links. The BT protocol stack typically communicates with the baseband over a standard interface, the HCI (see “BT Host Controller Interface” box). The protocol stack implements the protocols required for determining the capabilities of other BT devices and establishing communication links between those devices. BT profiles define application-specific protocols that are layered on top of the BT protocol stack. The protocol stack and profiles have APIs that are used to develop BT applications.

Interestingly enough, the BT specification does not define APIs. Instead, the specification defines protocols and the functions that must be available to applications. Companies that provide protocol stacks are free to define the APIs as they choose.

Creating a True BT Development Platform

BHAPI partitions BT functionality into a server and a client that provide a clean separation of the BT and application functions.

The server, BHSRV, encapsulates the BT protocol stack and profiles. And the BHAPI client, BHCLI, provides a high-level API that greatly simplifies the development of BT applications.

The BHAPI architecture is designed so that BHCLI and BHSRV can run on separate CPUs, although this is not required. In particular, BHSRV can be integrated directly with a BT baseband so that the baseband, BT protocol stack and profiles all run on the same chip. Other advantages of the BT Development Platform are:

- The communication protocol between BHCLI and BHSRV is a command/ event protocol that can be compared to the HCI command/event protocol defined by the BT specification. The significant difference is that the BHAPI protocol operates at the

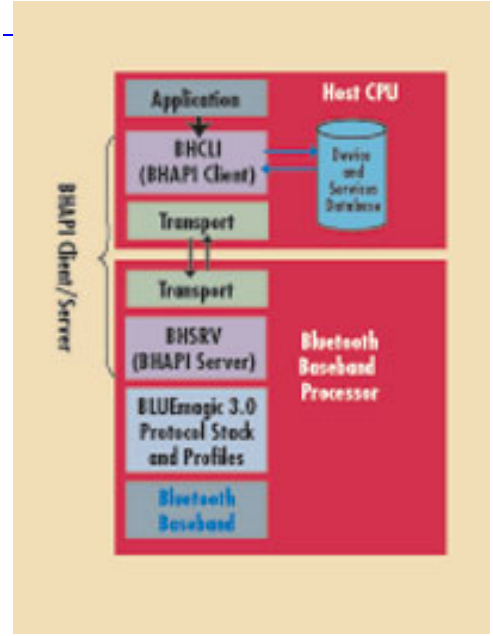
level of BT profiles, whereas the HCI is a device driver type interface.

- C and Java programming versions of the BHCLI API.
- APIs for managing an integrated device and service database and other "housekeeping" functions. These are functions that most application developers need and would otherwise have to implement themselves.

There are at least a half-dozen BT chip manufacturers that produce BT baseband processors capable of running BHSRV. Baseband processors that implement BHSR are interchangeable because such devices all implement the same BHAPI protocol.

A motivation for the design of BHAPI was to facilitate BT integration into existing products without major hardware redesigns (see Figure 1). In such cases, product development engineers may need to bypass the BHCLI API and code directly to the BHAPI protocol. For these engineers, full documentation is available for the BHAPI protocol, as well as sample code for a very lightweight BHAPI client.

Small applications that do real work can be implemented in only a few hundred lines of code. For example, support for the BT AVRCP could be added to a consumer electronics device with minor modifications to the control firmware and the incorporation of a BT module running BHSRV.



[1]

BT Development Platforms Ease Product Development The current alternative is to integrate a BT protocol stack into the software design. A complete BT protocol stack with profiles, is around 100,000 lines of code, It would require between 64 kB and 128 kB of flash and between 4 kB and 32 kB of RAM, depending on the specific profiles and application scenarios. The engineering effort to integrate a BT protocol stack is significant. It requires a high level of familiarity with BT specifications and experience in the nuances of RF engineering. In low-volume applications, engineering costs alone can easily exceed the cost of the BT hardware.

By contrast, the BT development platform client is much thinner — approximately

10,000 lines of code that include APIs for all the profiles — and support for a device and service database. This smaller code base and added functionality greatly simplify application level integration.

Another key advantage to the product developer is that a BT development platform server implemented directly on a BT module can be treated as an engineered black box: plug them in and they work immediately.

Interoperability testing and qualified protocol stack and profile implementations will have already been performed on that module, eliminating a major testing burden. The product developer is also insulated from hardware and software changes to the module, even from updates to the BT specifications. With a Development Platform server supported on multiple vendors’ BT modules, the product developer is also not tied to one hardware supplier.

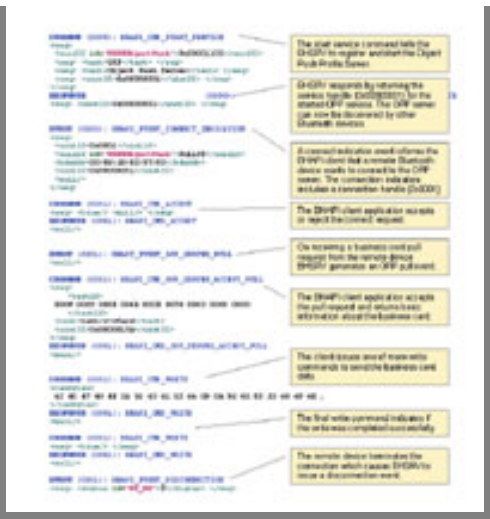
BHAPI for CE Engineers

A BHSRV module is an integrated component that includes a BT radio, baseband, protocol stack and profiles, BHSRV and support for one or more transports. The standard transport interface between a BHCLI and a BHSRV module is a 3-wire UART interface. The BHAPI platform protocol has no real-time performance requirements, so the UART interface can be configured to operate from 9600 b/s to 1 Mb/s. For BT profiles such as file transfer or basic imaging, higher data rates are required to achieve maximum throughput. Other profiles can run well with a low speed link. Streaming audio applications may impose some real-time constraints of their own.

The BHSRV module is a black box that requires no expertise in BT radios, protocols or profiles, so for the product engineer tasked with quickly adding BT capabilities to an existing product design, a downside is that black boxes can be difficult to debug. BHAPI allows any application can be prototyped and debugged on a Windows or Linux PC before being integrated into the product. This can be done using a BHSRV module or a PC implementation of BHSRV. A PC connected via serial cable to the product development platform can operate as a virtual BHSRV module. On the PC, debug and trace tools are included in the platform and give the developer full visibility into the BHAPI protocol and BT protocols using an integrated BT packet sniffer.

The BHAPI Platform supports all of the currently specified BT profiles, so the same BHSRV module can be used in many different product applications. This also allows BT capabilities to be added to a product incrementally without requiring a complete hardware redesign.

As an example, the AVRCP might be integrated into a set-top box to enable remote control via BT. Later, the HID profile might be added to allow keyboard input from a BT keyboard. Then, support for BIP might be added to enable the set top box to receive images and display slideshows via BT. All of this can be accomplished using the BHAPI Platform with only minimal software or firmware changes.



[2]

In an example such as this, adding BT to a product previously meant integrating a protocol stack and profiles before even starting on application development. Bringing 100,000 lines of foreign code into a product is daunting, even for companies that specialize in software development. Companies whose core expertise is in consumer electronics, automotive systems or industrial control usually deal with much smaller bodies of code. These companies do not always have the software engineering skills or the budget allocated to individual products available to deal with the complexities of BT.

In medical applications, such as patient monitoring, where the advantage of fewer wires seems obvious, regulatory requirements may make it difficult to incorporate external code, such as a BT protocol stack and profiles, into a product.

However, the BHAPI BT development platform is the catalyst needed for widespread adoption of BT in product categories outside of the mobile phones and PCs. A lightweight BHAPI client can be implemented in a few hundred lines of code. In many cases, BHAPI can be integrated with a product’s existing firmware and without any serious design modifications. By eliminating the technical barriers, the cost to add BT becomes essentially the cost of the BHSRV module. The economics of this model work even for low volume products where the engineering overhead is not amortized across millions of units.

In highly competitive markets such as consumer electronics, the need to keep products fresh and deliver the latest features limits product lifetimes. Aggregate volume for a product category may be high, in the tens of millions, but volume for a given product design may be relatively small. The key to success is rapid design cycles based on standard platforms and components.

Advantages of Developer Platforms and Economies of Scale

Medical: Medical patient monitoring device — In this market, volumes are typically low relative to other products that use BT technology. Regulatory issues complicate product development. A standard BHSRV module that can be taken through the

approval process and reused for multiple applications is attractive to medical product companies.

Profiles that might be incorporated into a medical monitoring device are:
- File transfer profile — enables download of data files from the device to a PDA.
- Human interface device host — enables keyboard input of patient information.
- Basic printing profile — enables wireless connection to a printer.
- Basic imaging profile — enables download of graphs or images.
- Serial port — enables connection to BT thermometers, blood pressure cuffs and pulse oximeters.

If the monitoring device is for use in the home, the Dialup Network Profile might be added to enable data to be sent via a mobile phone modem.

Consumer Electronics: DVD player — In this application scenario, the initial requirement is to add BT remote control capabilities and roadmap additional functions for the future. The initial profiles package:
- Audio/video remote control — enables wireless BT remote control functions.
- Human interface device host — enables mouse and keyboard input.

The desire is to add these functions with minimum effect upon the existing design through firmware changes to the DVD's microcontroller. In this case, the BLST is used, and the application is modified to send only the required BHAPI commands to enable the two supported profiles.

Support will be added to a DVD player to upload images from the BT device into a memory device and display a slide show, or use BIP to control a slide show from a remote device such as mobile phone. Other potential features will eventually support:
- Basic imaging profile — enables image upload and slide shows.
- Advanced audio distribution profile — enables wireless surround sound speakers.

Convergence Products: Smart-phones — Smart-phones merge the mobile phone, MP3 player, PDA and digital camera into a single device. BHAPI provides the BT features needed for wirelessly enabling all of these capabilities. In this application scenario the key profiles provided by BHAPI are:
- Headset audio gateway – enables use of BT headset.
- Hands-free audio gateway — enables use with an automotive hands-free kit.
- Object push profile — enables business card exchange.
- File transfer profile — enables music and image file transfer.
- Dialup networking profile — enables laptop/PDA to use phone as a modem.
- Serial port profile — enables all for "other" applications.
- Basic imaging profile — enables image capture, upload, download, print.
- Advance audio distribution profile — enables streaming to stereo headset.

The Smart-phone has a capable host processor and via Linux or other powerful operating system standards would be capable of hosting a full BHAPI C or Java client. The BHAPI server could run directly on the Smart-phone CPU or could run embedded on a BT baseband processor.

An Innovative Approach

The BHAPI BT development platform defines a high-level protocol that sits above, and is integrated with, a certified BT software stack and profiles. Used as a component, It inherits the BT qualification status of the underlying protocol stack and in many cases will not require any further certification from the BQB.

The high level protocol provides an abstraction layer on top of the BT stack and profiles and permits the application and protocol stack to run in separate memory spaces or on separate CPUs.

The abstraction layer simplifies the application interface to the BT profiles by defining consistent representations of devices, services and connections. Where practical, operations that are similar are unified, for instance, many profiles share the BHAPI “write” command even though the actual implementations of the write function are profile-specific. Generic “start” and “stop” commands register and de-register profiles so that resources are only allocated when needed.

Running the application and protocol stack in separate memory spaces improves robustness and reliability while helping to ensure adherence to the BT specification. Running on separate CPUs allows the execution of the BT profile execution to be offloaded, reducing resource requirements on the host CPU. This advantage is most prominent when the BHAPI server is integrated directly onto a BT baseband processor.

In this scenario, commands originate at the application and events originate from the protocol stack and profiles. For this reason, the application, the sender of commands and recipient of events, is considered to be the “client,” and the protocol stack, the recipient of commands and sender of events, the “server.”

- The BHAPI server responds to all commands with an acknowledgment, some data or an error status.
- The BHAPI server also generates asynchronous events such as connect requests or PIN code requests.
- For streaming data, the BHAPI protocol uses a credit-based flow control method. This is mapped by BHSRV to whatever flow control mechanism is provided by the profile.
- Commands and events carry a “connection handle,” which identifies a virtual connection between a local and remote BT service.

This provides a clean unification of all the different types of BT profile connections. The application can use the connection handle to de-multiplex command responses, events and data associated with different profiles. BHAPI allows multiple simultaneous connections using different profiles to different BT devices. The only constraints are the resources available to the specific BHAPI server implementation.

BHAPI provides access to all the important BT functions and has commands for putting BT links into low power modes, enabling and disabling device discoverability and connectivity, as well as fully supporting all three BT security modes and supporting name string localization.

For documentation purposes, the BHAPI protocol is described in XML format. The same format is generated in trace logs during testing and debugging. The actual protocol is a compact binary representation based on the "data element" format defined in the BT specification of the SDP.

- When the BHAPI server is running on a BT baseband processor, the BHAPI protocol runs over a serial transport, currently on a UART hardware interface.
- The BHAPI protocol assumes it has a reliable transport; Open Interface currently has two implementations, the OISP optimized for performance, and the BLST that trades off small code size for lower data throughput.
- Both of these are 3-wire serial protocols designed specifically for BHAPI that provide a reliable transport and manage their own flow control.
- When the BHAPI server and client are both hosted on the same processor the BHAPI protocol uses an IPC mechanism provided by the underlying operating system. For example, on Windows or Linux BHAPI uses "pipes."

Although the emphasis with BHAPI has been on providing a high level BT Development Platform, the devil is in the details. BHAPI provides C and Java APIs that enable rapid application development on many capable host platforms. In many cases, functions in the BHAPI client API are thin layers on top of the BHAPI protocol, but the API includes support for common housekeeping functions such as device and service discovery and management of a discovery database.

The BHAPI client API is a synchronous API typically used in a multi-threaded application environment. The API supports C and Java application development.

Conclusion

With a BT development platform now available to product designers and CE engineers, BT is becoming a mainstream technology ready and waiting for active development in a wide range of vertical applications as well as mass market CE devices.

BT chips are affordable, reliable and readily available from multiple suppliers. Unfortunately the cost and complexity of BT application development to this point has prevented the true potential of BT from being realized outside of the single, well-funded handset market. The BHAPI BT development platform makes BT application development available to the masses.

And as they say, if you build it, they will come…

WD&D

About the Author

Greg Burns is Chief Technology Officer at Open Interface, the principle architect of BLUEmagic 3.0 and BHAPI. He has more than 25 years of software development experience. Burns previously was Group Program Manager at Microsoft, during which time he defined mobility features for Windows XP, prototyped applications for Windows-based embedded devices, drove key networking initiatives in Windows 98 and Windows 2000 and was an early pioneer in interactive TV, home networking and advanced consumer technology. He can be reached at greg.burns@openinterface.com ***[5].***

Links:
[1] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0506/wd56coverstory_1_lrg.jpg
[2] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0506/wd56coverstory_protocol_lrg.jpg
[3] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0506/wd56coverstory_screenshots_lrg.jpg
[4] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0506/wd56coverstory_client-code_lrg.jpg
[5] mailto:greg.burns@openinterface.com