Bluetooth™ Wireless Technology meets CAN

Matthias Fuchs
esd electronic system design GmbH, Hannover, Germany

To access mobile and moving CAN fieldbus systems a wireless approach is often a good solution. For short-range wireless communications using RF links the Bluetooth™ Wireless Technology has been introduced. This paper describes the use of the Bluetooth Wireless Technology beyond its primary mission – Bluetooth links as a transport media for CAN data. Furthermore, a practical implementation of a Bluetooth CAN interface is presented.

Introduction

To access autonomous and mobile CAN fieldbus systems a wireless approach is often a good solution. Up to now mostly proprietary radio systems were available and therefore used.
In 1998 the Bluetooth™[1] Wireless Technology was introduced as a standard for short-range wireless communication. The main goal of this technology was to allow instant networking and voice communication between small mobile devices such as cellular phones, PDAs and notebooks. The number of Bluetooth devices is expected to grow up to about one billion units in 2005.
The use of Bluetooth technology in industrial applications is very new and will be highlighted in this paper. After an introduction to Bluetooth technology an overview of CAN-over-Bluetooth scenarios will be given.
A central paragraph discusses various ways to implement a wireless CAN link over a Bluetooth connection with respect to involved Bluetooth protocol layers. At the end an existing Bluetooth CAN interface with its features is presented.

[1] The BLUETOOTH trademarks are owned by Telefonaktiebolaget LM Ericsson, Sweden

1. Bluetooth Wireless Technology

Bluetooth was introduced by Ericsson together with Nokia, IBM, Toshiba and Intel as a replacement for the mechanically vulnerable and inconvenient cable connections between communications products [1]. In order to be as independent as possible of environmental and operating conditions, radio techniques were selected. This made connections possible through textiles and walls, without line-of-sight contact. The named companies founded the Bluetooth™ Special Interest Group (SIG) with the goal to establish a worldwide standard. Meanwhile more than 2000 companies joined the SIG.

Bluetooth allows short distance radio links over approximately 10 meters with an option to extend the distance up to 100 meters. It operates in the free ISM[2] frequency band from 2.402 to 2.480 GHz and uses a Frequency Hopping Spread Spectrum (FHSS) technology in order to achieve some immunity against electronic eavesdropping and interference. Frequency Hopping means that the transmitting frequency is changed 1600 times per second with the

[2] ISM=industrial scientific medical

result that the data traffic is divided into time slots of 625 _s. The signaling speed is always 1 Mbit/s.

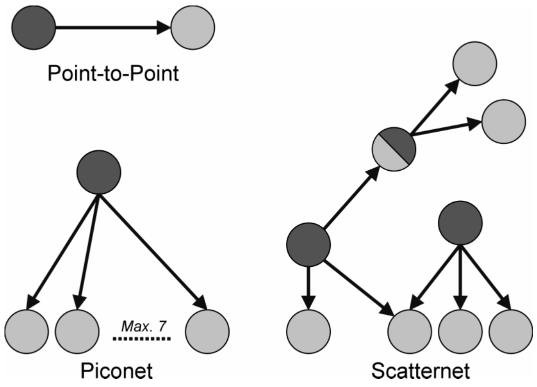

Figure 1 - Topology

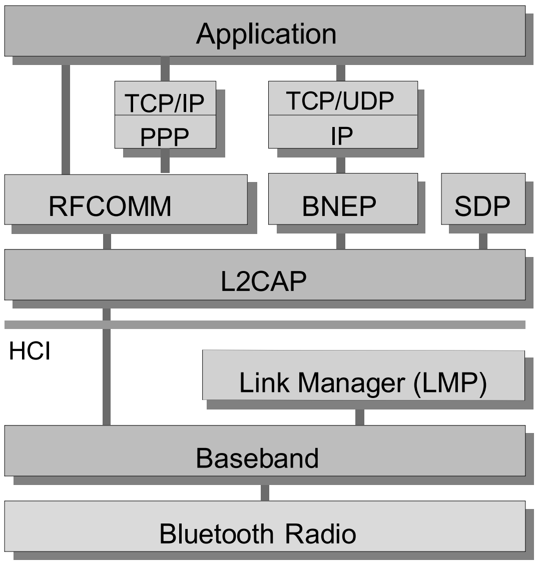

Figure 2 – Bluetooth Protocol Stack

Bluetooth devices are capable of setting up small wireless networks – so-called *piconets* – with other devices that are in-range. Each piconet consists of one master and up to seven slaves (figure 1). The simplest form of a piconet is a point-to-point connection between a master and a single slave. The master of a piconet controls the joining slaves and specifies the hopping sequence. All communication only takes place between the master and a single slave at a time since there is no broadcast mechanism for application data available inside a piconet. Moreover, inter-slave communication is not possible. The master-slave communication follows a TDD[3] scheme, where the master starts its transmission every even and one of the slaves every odd time slot (figure 3).

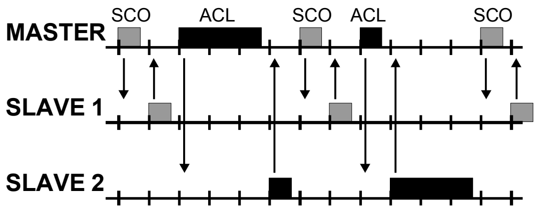

Figure 3 – TDD with ACL/SCO packets

Bluetooth also defines a topology of overlapping piconets – this is called *scatternet*. Some devices in a scatternet scenario participate in more than one piconet. Most of today's Bluetooth implementations are not capable of building scatternets.

Figure 2 shows the simplified Bluetooth protocol stack. It can be seen as two parts: Bluetooth radio (physical layer), Baseband and Link Manager are implemented into Bluetooth chipsets.

The Bluetooth baseband provides two different types of links: *Asynchronous ConnectionLess* (ACL) and *Synchronous Connection Oriented* (SCO) links. The latter are intended for constant bit rate and low latency voice connections. Therefore SCO data is normally not passed to higher layers, but directly injected into the baseband layer. Equidistant time slots will be reserved for SCO connections to guarantee the constant bit rate.

A piconet's master can keep up to three SCO connections at a time. Each connection offers a constant bit rate of 64kbit/s. Since voice channels are intended for telephony applications, normally no error correction is needed.

[3] TDD = Time Division Duplex

It is possible to bundle two or all three SCO channels and to use the extra bandwidth for FEC[4] data.
ACL links were introduced for all kind of data communications with a best-effort approach. All remaining time slots are used for this kind of link. ACL packets contain a checksum (CRC) and are retransmitted automatically from the baseband layer when an error has been detected.
The baseband can also communicate using special packet types that last for three (figure 3) or five time slots. The use of different packet types results in various maximum data rates for data communication:

1.) 5-slot-packets in both directions results in a symmetrical 444kbit/s full-duplex connection.
2.) 5-slot-packets in one direction, while single-slot-packets in the other direction results in a 723kbit/s versus 57kbit/s asymmetrical connection.

The upper Bluetooth protocol stack is mostly implemented on a host CPU. Communication between the lower stack and the host CPU is done through the *Host Controller Interface* (HCI) that in most cases is implemented though an interface like a serial UART or USB.

The lowest upper-stack protocol is *L2CAP*[5], a multiplexing, segmentation and reassembly protocol for higher layers. It transmits and receives data from L2CAP instances of other connected Bluetooth devices.
The presence of the protocols above L2CAP strongly depends on the Bluetooth device type and its supported profiles. Therefore some of them are optional. For further discussion they are shortly described:

[4] FEC = Forward Error Correction
[5] Logical Link Control and Adaptation Protocol

RFCOMM[6] provides virtual serial ports to further upper layer protocols or to an application. These ports are transparently connected to ports on remote Bluetooth devices.

BNEP[7] is one of the newest Bluetooth protocols. It is used for Bluetooth networking applications and replaces physical Ethernet hardware. The BNEP defines a method to transport common networking protocols over Bluetooth media. It encapsulates packets from various networking protocols, which are directly transported over the Bluetooth L2CAP protocol.
The first versions of the Bluetooth specification made use of a higher layered approach for networking over Bluetooth. IP[8] applications should use PPP-over-RFCOMM (see figure 2). But that solution results in a huge protocol overhead and a bad throughput.

SDP[9] is used to discover services and characteristics that are offered by Bluetooth devices. It is not directly involved in data exchange, but it is very important in order to setup any data exchange.

In addition to the used protocols the Bluetooth specification defines *"profiles"*. Profiles specify the use of the Bluetooth protocols for different devices. Some of the defined profiles are "LAN access" (used for wireless access to local area networks), "Dial-Up networking" (defines Bluetooth modem-emulation to setup dial-up connections) and the "serial port profile" that defines how to setup generic virtual serial connections for miscellaneous applications.

[6] RFCOMM = Radio Frequency COMMunication
[7] BNEP = Bluetooth Network Encapsulation Protocol
[8] IP = Internet Protocol
[9] SDP = Service discovery Protocol

2. CAN-over-Bluetooth Scenarios

There are many practical applications where CAN messages need to be transmitted over short distances and where cable connections are either impossible or unwanted.

An example is the moving arm of an industrial robot in a production line. Sensors and activators in this system need to communicate to their environment through CAN. A wireless CAN link removes the need for a cable connection between moving and fixed parts of this system.

A special advantage of the Bluetooth Wireless Technology can be seen in the way Bluetooth devices can start to communicate. Bluetooth nodes only need to get in range. The devices will discover each other and connections can be setup automatically, without any user interaction.
This behavior can be used for autonomous systems in CAN-controlled environments.

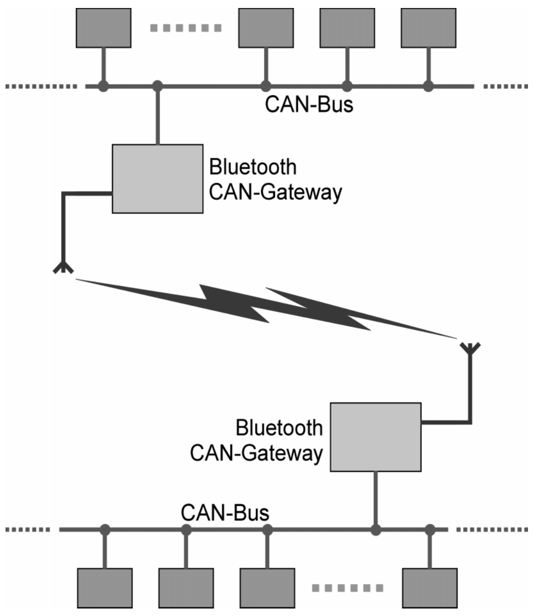

Figure 4 - Bluetooth CAN Gateway

Further applications can be seen in service situations. A CAN network sometimes needs to be analysed by service staff. Instead of a direct connection between the CAN system and the CAN analyzer, a wireless approach should be considered. Therefore a special Bluetooth-equipped CAN node must be present on the bus. The service tool (analyzer) could consist of a simple PDA with a Bluetooth interface and special analyzer software.

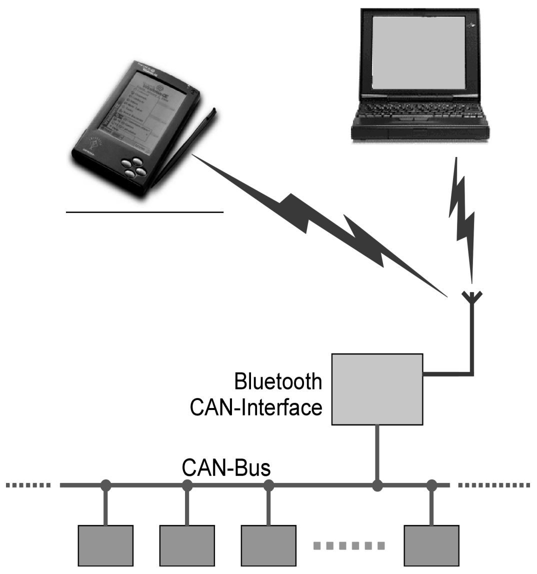

Figure 5 - Bluetooth CAN Interface

The Bluetooth wireless CAN applications can be divided into two main categories:

1.) Wireless CAN-to-CAN bridging using Bluetooth CAN gateways. The goal is to connect two or more independent CAN networks through a Bluetooth link (figure 4).

2.) Wireless access to a CAN network using off-the-shelf Bluetooth-equipped products like PDAs or notebooks (figure 5).

3. Implementation Aspects

This chapter gives an overview about implementation alternatives for CAN-over-Bluetooth. Especially the necessary protocols are discussed with respect to their applicability as transport layer for CAN messages.

The following approaches have in common that they are still compatible with the current Bluetooth specifications. That means that no new protocols are added to the protocol stack.
Depending on the Bluetooth protocol layer that is used for CAN communication across a Bluetooth link, the CAN characteristics are more or less maintained.
Of course the best results can be achieved by some improvements to the Bluetooth radio or baseband layer. But such modifications would never fulfill the existing Bluetooth specification and are therefore not subject of this paper.

	Bluetooth Wireless Technology	***CAN – Controller Area Network***
signaling rate	1 Mbit/s	10 kbit/s – 1 Mbit/s
usable bit rate	723 kbit/s <-> 56 kbit/s, asymmetrical 444 kbit/s, symmetrical	- min. 577 kbit/s (message data, CAN 2.0A) - max. 712 kbit/s (message data CAN 2.0A + identifier + length info) - min. 489 kbit/s (message data, CAN 2.0B) - max. 740 kbit/s (message data CAN 2.0B + identifier + length info)
Transmission	Connection oriented / connectionless	Message oriented
Addressing	Bluetooth address (48 bit)	Message identifier (11 bit or 29 bit)
Media access	Time Division Duplex (TDD), Media access is granted by piconet master	CSMA/CA with priority
Topology	Star / tree (figure 1)	Bus
Network extension	10 m (optional, 100 m)	> 1 km
Latency	- Predictable for SCO connections (units of 1 or 2 timeslots) - Unpredictable for ACL because of retransmissions in baseband	- Predictable - Under 1 ms possible

Table 1 – Comparison between Bluetooth and CAN

A glance to table 1 shows that mapping CAN fieldbus technology onto Bluetooth is only possible with restrictions. The Bluetooth bandwidth can be sufficient considering that a CAN system is normally not used at the top of its capacity. It must not be forgotten, that it is not sufficient to look at the net CAN data rates. To keep the CAN message characteristics, the identifier with 11 or 29 bit length also has to be transmitted.
A result of bandwidth consideration is, that CAN-over-Bluetooth primarily only makes sense in a point-to-point topology.

3.1 CAN-over-SCO-link

This approach uses SCO links for CAN message transmission. Single or multiple queued CAN messages will be transmitted as payload of SCO packets.

The resulting pros and cons are as follows:

+ Predictable and guaranteed latency because of reserved time slots.
+ Simple protocol stack: Only the lower protocol stack is required for CAN data transmission. The upper stack is only essential for SDP.
- Error detection and correction has to be done by the application and correction is only possible if FEC data is available (depends on the SCO connection type).
- Low bandwidth over SCO connections: A single SCO connection offers 64 kbit/s bandwidth; with up to three SCO connections 192 kbit/s are possible.
- A single SCO connection with 192 kbit/s or three 64kbit/s-connections use all available time slots. That means no additional Bluetooth communication is possible with the involved devices.

This approach is acceptable for Bluetooth CAN gateway applications since it offers the best results in latency. The lack of bandwidth must be considered, but should be acceptable for many applications.
All current Bluetooth profiles use ACL links for data communication. This implicates that a new Bluetooth profile for data transmission, especially real-time CAN data, over SCO connections must be specified. This is essential in order to be Bluetooth compatible.

3.2 CAN-over-BNEP

CAN-over-BNEP uses the Bluetooth network encapsulation protocol as a base to transmit CAN messages over a Bluetooth connection.

+ BNEP defines a generic way to transmit various types of packets of a Bluetooth connection. An extension for CAN messages is easily done.
+ The BNEP protocol overhead is very low (1 byte).
- The BNEP layer is situated above L2CAP, which adds additional 4 bytes L2CAP-overhead to each L2CAP packet.
- This approach brings the negative characteristic of data transfer over ACL-links: unpredictable latency due to an ARQ[10] algorithm in the baseband.
- This approach requires some extension to the Bluetooth specification, because the CAN message transfer using BNEP packets is not specified so far.

3.3 Virtual Serial Ports (RFCOMM)

The RFCOMM protocol is used by many Bluetooth profiles. For this reason it is also implemented in many Bluetooth devices. The virtual serial ports that are provided by RFCOMM are easy to use under many Bluetooth protocol stack implementations:

+ Highly compatible with other off-the-shelf Bluetooth devices, because RFCOMM's virtual ports are always available.
+ Easy to implement and highly portable: The CAN-over-Bluetooth application only relies on a virtual serial port.
- Deterministic characteristics are as bad as for BNEP approach.
- High latency because all data must pass many lower protocols.

This approach strongly needs a method for CAN message framing into a serial byte stream, which will be transferred by RFCOMM. This encapsulation mechanism is essential for transmitting CAN messages over serial lines in order to detect message boundaries on the receiving side.

[10] ARQ = Automatic Repeat Request

4. A Practical Approach

The expected growth of the Bluetooth market and its high potential for industrial applications motivated esd to develop a Bluetooth CAN-interface (figure 6) [5]. This section gives an overview about its features and the way the mentioned problems were solved.

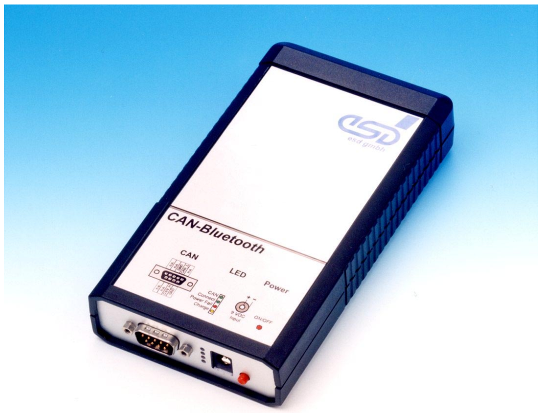

Figure 6 - Bluetooth CAN-Interface

The device is based on an embedded PowerPC platform (PPC405CR, IBM [3]), an integrated Bluetooth module (ROK101007, Ericsson) and a CAN controller (SJA1000, Philips). The device can be powered by an external power-supply or by an internal battery, which allows several hours of mobile operation.
The esd Bluetooth CAN interface uses an embedded Linux as operating system. This decision was led by the fact that an open source Bluetooth protocol stack [4] is available. The device can be used for wireless CAN bridging or as mobile service tool to gain access to a CAN bus from a standard notebook or PDA that will be equipped off-the-shelf with a Bluetooth interface in the very near future.
The Bluetooh CAN-interface can be configured through an embedded web interface. The device implements the Bluetooth LAN access and dial-up networking profile to allow any Bluetooth enabled device to access the web interface using a standard browser. In addition to configuration, a simple CAN access is possible through the web interface.
For high performance CAN-interface-functionality the Bluetooth CAN interface uses the RFCOMM approach (chapter 3.3) with a newly defined encapsulation packet format: *CANBT* (figure 7). Each packet consists of a 2-byte-header and up to 31 bytes of payload. A length field in the header defines the utilization of the payload field. The most important field in the header is called *COMMAND.* It is used to distinguish between different packet types.

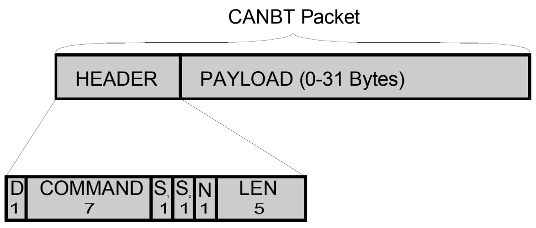

Figure 7 – CANBT packet format

Implemented packet types include:

- Transmission of CAN frames with or without acknowledge and with or without timestamp information
- Configuration of CAN baudrate
- Configuration of acceptance filter
- Status monitoring (heartbeat, CAN bus state, message buffer level)

The bandwidth problem is minimized by a huge buffer memory inside the device. This guarantees no lost CAN messages even on highly loaded CAN systems. Of course this buffering only makes sense for interfacing applications, where no hard real-time requirements exist.
Moreover, the Bluetooth CAN interface implements a configurable local message filtering to reduce the needed bandwidth over the air link.

5. Conclusion

The use of Bluetooth Wireless Technology as a transport media for CAN data in industrial applications is an interesting alternative versus proprietary wireless RF solutions. The expected boom of Bluetooth devices in the next years will push prices for Bluetooth chips down and Bluetooth equipped devices will become usual.
Bluetooth equipped consumer products like PDAs and notebooks together with a Bluetooth CAN interface will find new applications in service situations.
To meet the requirements of a powerful CAN service tool, the esd Bluetooth CAN interface also allows high speed analog capture of the two CAN signal lines with up to 10 Msamples/s.

A direct connection between two or more CAN systems has also been demonstrated with esd's Bluetooth CAN-interfaces in a gateway configuration. But it must be considered what approach is suitable. The newly defined CANBT protocol offers different methods for CAN message transmission over a Bluetooth RFCOMM connection. These methods allow retrieval of information on latency and successful transmission of CAN-messages.

For use in notebooks, esd also offers driver software for common operating systems that make the Bluetooth CAN interface behave like a normal CAN interface.

Literature

[1] Bluetooth Special Interest Group: Bluetooth specification 1.1, www.bluetooth.com

[2] CAN in Automation: CAN specification 2.0, www.can-cia.de/CAN20A.pdf, www.can-cia.de/CAN20B.pdf

[3] IBM PowerPC 405CR: www.chips.ibm.com

[4] Axis Communications AB: OpenBT open source Bluetooth protocol stack, developer.axis.com

[5] esd electronic system design gmbh: Bluetooth CAN-interface, www.esd-electronics.com/english/products/CAN/can-bluetooth.htm

Dipl.-Ing. Matthias Fuchs
matthias.fuchs@esd-electronics.com

Headquarter:
esd GmbH
Vahrenwalder Strasse 207
30165 Hannover
Germany
Phone: +49 511 37298 0
Fax: +49 511 37298 68
Email: info@esd-electronics.com

US office:
esd electronics
7667 W. Sample Road
Suite 127
Coral Springs, FL 33065
Phone: 1-800-504-9856
Fax: 1-800-288-8235
Email: sales@esd-electronics.com

Website: www.esd-electronics.com