Targets and Envisaged Applications

- **Ad-hoc Networking and Cable Replacement**

- **Low Price (USD 5.00)**

Targets and Envisaged Applications

Ad-hoc Networking and Cable Replacement

Most importantly, Bluetooth shall provide for ad-hoc networking. That is, equipment with included Bluetooth equipment shall be able to connect more or less seamlessly to each other, once that the two devices are close enough.
The range of any Bluetooth equipment obviously depends on the receiver's sensitivity and on the transmitter's output power. In that respect, the currently available Bluetooth devices provide for ranges of maximum 10 m but distances of 2 – 5 meters are more realistic.
In any case, the interconnected Bluetooth devices together form a so called pico network (⇔ Bluetooth expression) as illustrated in the figure.

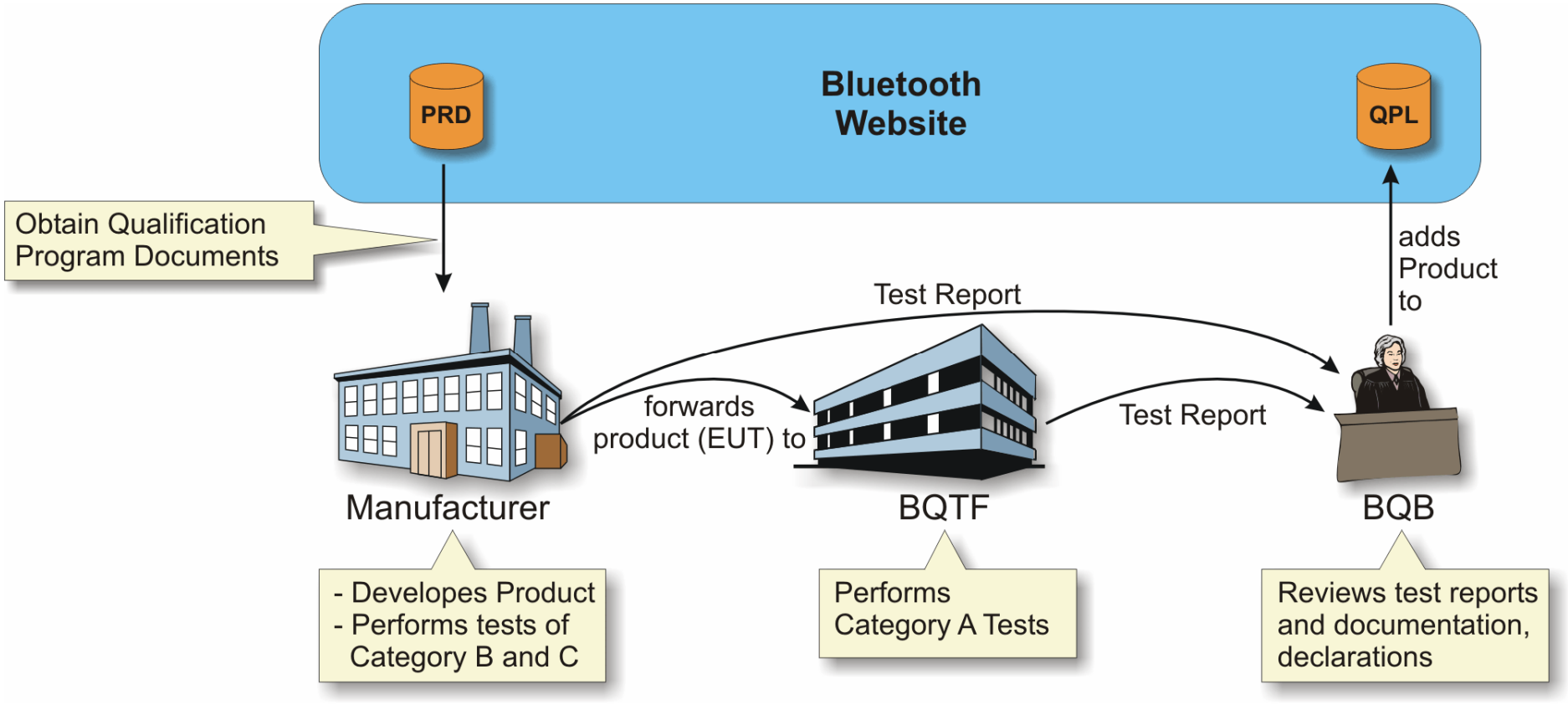
Note:

- In a Bluetooth pico network, there is always one master (in the example the desktop in the center) and up to seven slaves to this master.
- Slaves can only communicate with and through the master; communication between slaves is not possible.
- The figure also includes the mobile phone which is connected not only to the desktop computer but also to a Bluetooth headset. In fact, the mobile phone and the headset form a second Bluetooth pico network in which either the headset or the mobile phone are the master.
- Accordingly, the master of a Bluetooth pico network can be slave in other Bluetooth pico networks and a slave in one Bluetooth pico network can be master in a second Bluetooth pico network.
- However, no device may be master in more than one Bluetooth pico network

Low Price (USD 5.00)

The original idea of Bluetooth was plain cable replacement, particularly in the mobile phone area. The typical price for such a cable is around USD 10.00 – USD 20.00 which means that a competing wireless technology must not cost more to be successful. Sharing this price between the two devices to be connected means that Bluetooth should not cost more than USD 5.00 – USD 10.00 per unit.

The Bluetooth Qualification Process



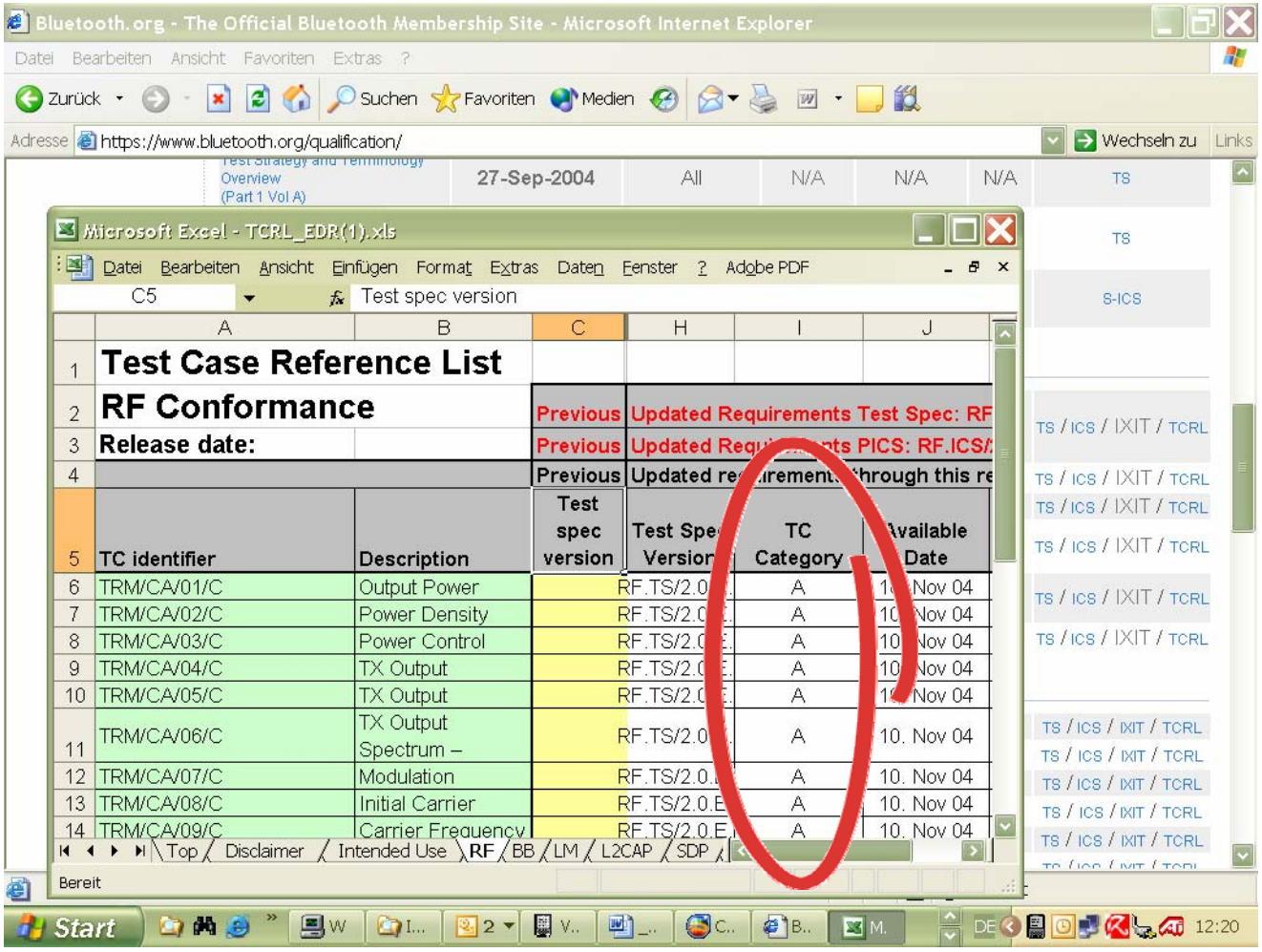
The Bluetooth Qualification Process

- ⇒ The Bluetooth qualification program is based on the definition of qualification rules in the PRD (Bluetooth Qualification Program Reference Document) which can be downloaded from the website in the bottom of this page.
- ⇒ The Bluetooth qualification starts with the device manufacturer developing the Bluetooth product. After successful product design and internal acceptance, the manufacturer is required to pass the Bluetooth qualification program before this product can be added to the QPL (Qualified Products List).
- ⇒ As the figure illustrates, the manufacturer may conduct the testcases which are categorized B and C himself. However, the manufacturer may prefer to request a registered test-house (⇔ BQTF) to perform these testcases.
- ⇒ In any case, the manufacturer has to request a BQTF to perform the testcases which are in category A. To do so, the EUT (Equipment Under Test) has to be forwarded to the test-house.
- ⇒ Test results for all testcases of categories A, B and C have to be relayed to a notified body (called BQB) to be reviewed before the respective Bluetooth device is finally added to the QPL.

Note that in addition there are so called unplug fests for Bluetooth which are strongly recommended by the SIG to increase product interoperability. For more information please refer to this website: http://programs.bluetooth.org/upf/ or to this document: “UnPlugFest New Participants_r6.pdf” which can be downloaded from the same website.

[http://qualweb.bluetooth.org/]

How to determine the Testcase Category

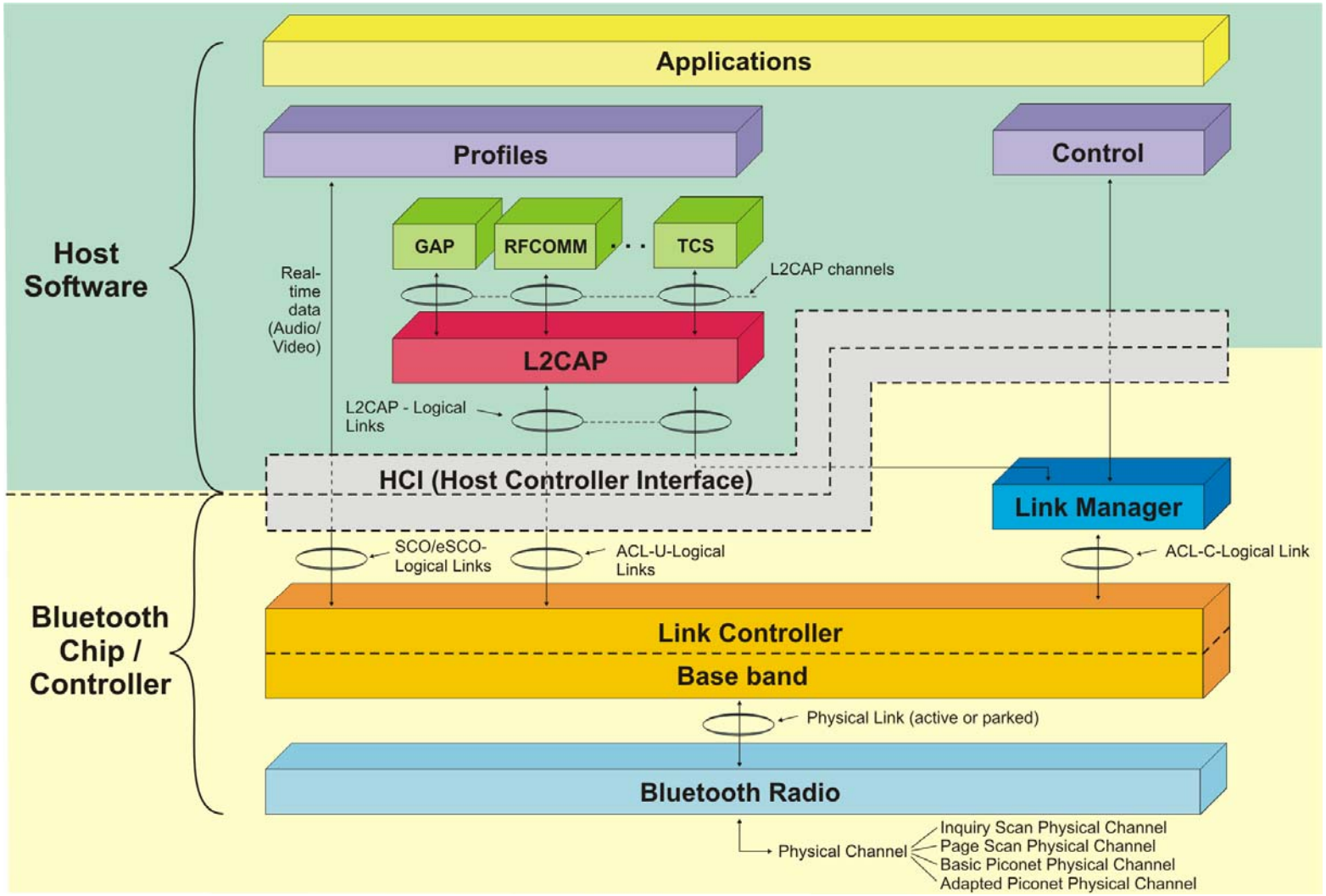


How to determine the Testcase Category

The screenshot illustrates how the category of a specific testcase can be determined:
The Bluetooth website contains a testcase reference list (TCRL) which lists among other things the category of all testcases.

[https://www.bluetooth.org/qualification/]

The Bluetooth Protocol Stack



The Bluetooth Protocol Stack

The figure illustrates the structure of the Bluetooth protocol stack and the relationship and interconnections among the different layers. Basically, the Bluetooth protocol stack can be differentiated into the following major parts:

Bluetooth Chip / Controller

Within the specification, this part is frequently also called the controller. It comprises the radio part, the baseband layer together with the link controller and the LM (Link Manager).

Host Controller Interface (HCI)

The HCI (Host Controller Interface) is there to provide rules for communication, setup and messaging between a remote Bluetooth host and the Bluetooth device on another chip. There is no HCI implementation, if the Bluetooth host software is on the same chip as the underlying software (⇔ LM, Baseband …).

To save cost for memory and software in the Bluetooth chip and makes sense to extract the upper layers from the Bluetooth chip. These upper layers can be operated e.g. on a PC's CPU which has plenty of resources available. Note that the Bluetooth specification also provides guidelines for different intermediate transport systems between Bluetooth host and Bluetooth device like for instance USB (Universal Serial Bus) or UART (Universal Asynchronous Receiver and Transmitter).

Host Software

The host software contains all higher layer protocols of the Bluetooth protocol stack as well as the Bluetooth profiles. In particular, this relates to the L2CAP-protocol which most importantly multiplexes information from multiple upper layer applications towards the link controller / baseband layer. The L2CAP-layer also takes care of segmentation of this information to meet the lower layer requirements.

In addition to L2CAP, the host software includes important protocols as the GAP (Generic Access Profile) or RFCOMM for RS-232 serial port emulation.

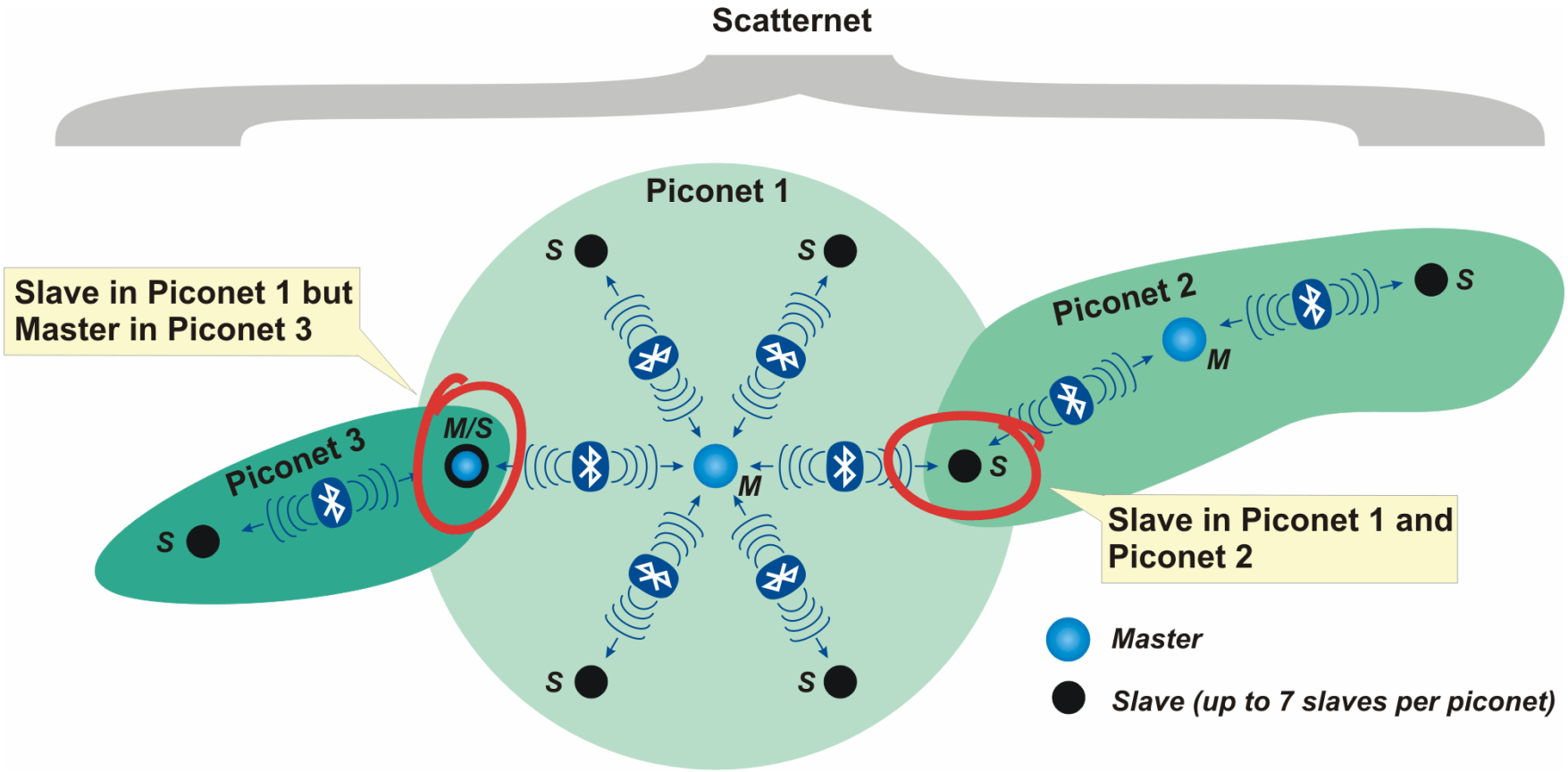
Profiles

Bluetooth uses the concept of profiles to have equipment of different manufacturers compliant to each other. To be more precise: The generic Bluetooth protocol stack provides various different features that may or may not be supported by a specific implementation. Through the use of profiles, a given application like a Bluetooth telephone headset is provided a set of commands to and responses from the generic part of the Bluetooth protocol stack.

Channels and Links

Please note the position and relationship of the various physical and logical links and channels that will be dealt with in more detail later.

Bluetooth Ad-Hoc Networking



Bluetooth Ad-Hoc Networking

The figure illustrates all possible relationships among Bluetooth devices which are possible. Most importantly, Bluetooth introduces two new terms, piconet and scatternet which meaning shall be explained here:

Piconets

A Bluetooth piconet consists of at least two Bluetooth devices. One of these Bluetooth devices will be the master of this piconet while the other devices will be slaves to this master.

Note:

- There may be no more than seven slaves in a given Bluetooth piconet.
- Within any Bluetooth piconet, communication is only possible between master and slave.
- However, no communication is possible between two slaves of the same Bluetooth piconet.

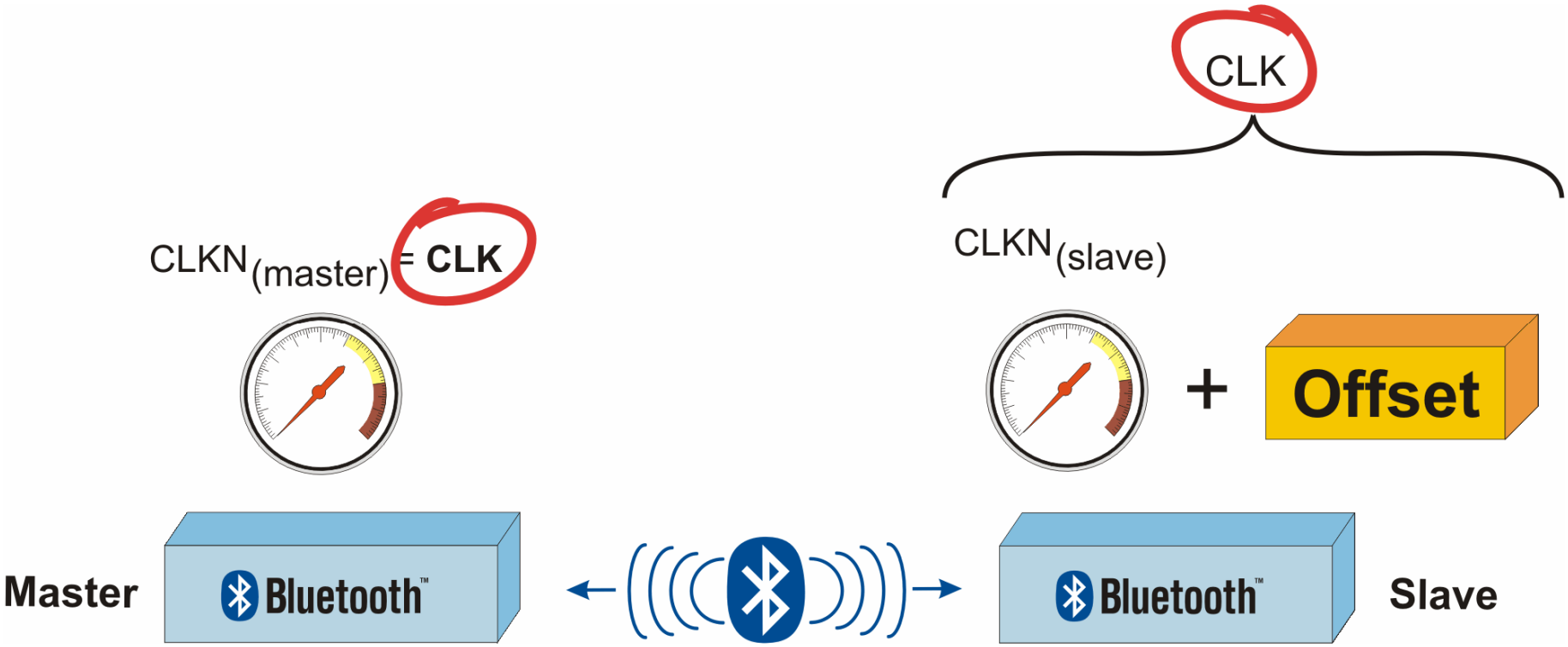
Scatternets

A scatternet is formed when two Bluetooth piconets are combined. As the figure illustrates, such “combining” is achieved when one Bluetooth device is slave of more than one Bluetooth piconet or when one Bluetooth device is master of one piconet and slave in one or more additional Bluetooth piconets. Note that a given Bluetooth device may never be master in more than one Bluetooth piconet.
The difficulty of a scatternet is the fact that the combining Bluetooth devices need to time-share among the different piconets. This obviously decreases their performance in a single piconet.

Despite the fact that scatternets are a legitimate network form in Bluetooth, there is no provision for any routing function between any two piconets within a scatternet.

[Bluetooth Core Spec Version 2.0, Volume 1, Part A (Architecture), (4.1)]

Clock Synchronization in a Piconet

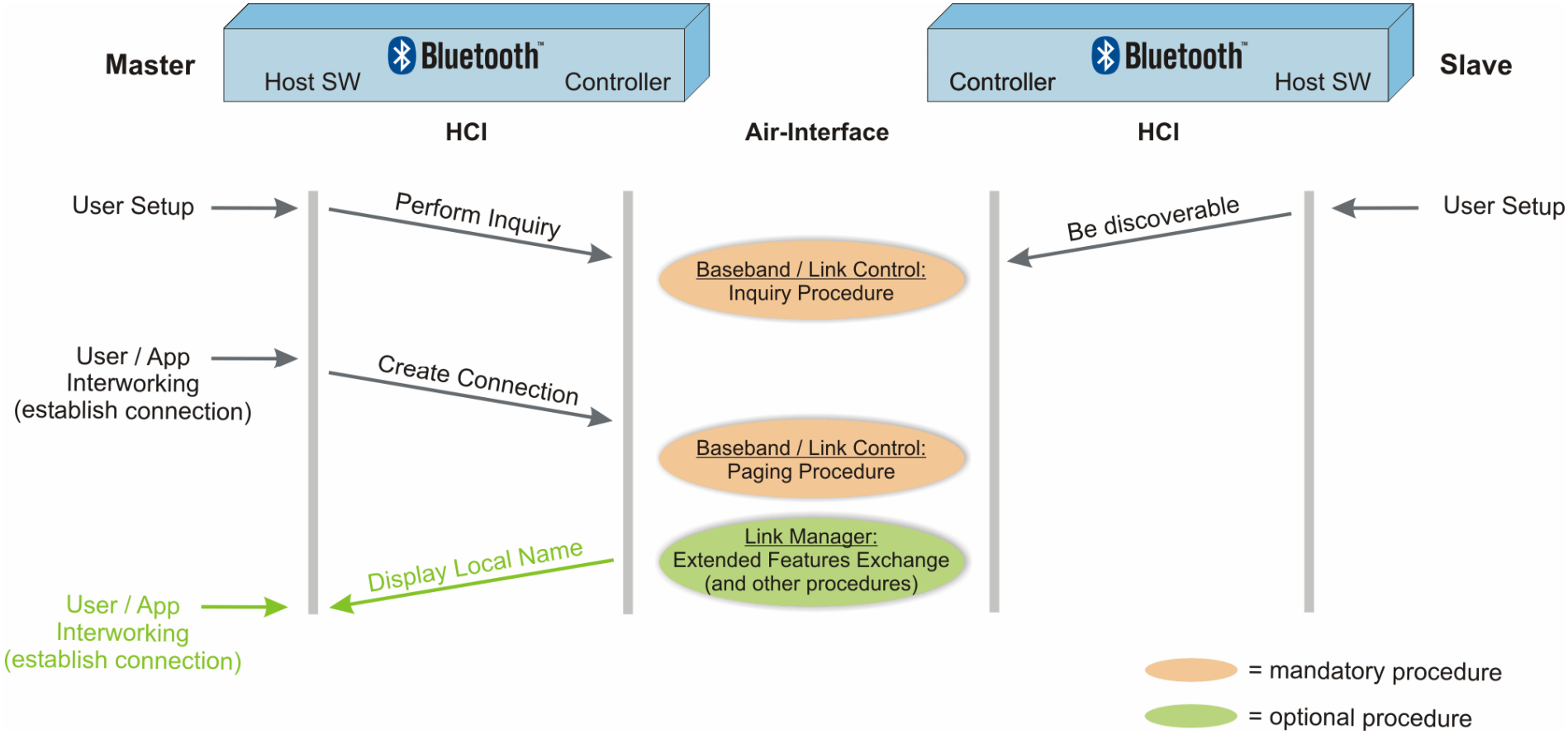


Clock Synchronization in a Piconet

In every piconet, the master clock is the reference clock for all slaves. As the figure illustrates, each slave shall synchronize to the master clock by adding a necessary offset to its own CLKN.
Accordingly: CLK = CLKN + Offset.

[Bluetooth Core Spec Version 2.0, Volume 2, Part B (Baseband), (2.2.4)]

(1) Interconnecting Bluetooth Devices – Procedural Overview



(1) Interconnecting Bluetooth Devices – Procedural Overview

The figure on this slide and the two following slides highlights the interworking within a Bluetooth device and between two Bluetooth devices that shall be interconnected.
The internal communication relates to HCI-related messaging between the Bluetooth chip / controller on one hand and the host software on the other hand. Obviously, this communication won't be visible in a one-chip solution.

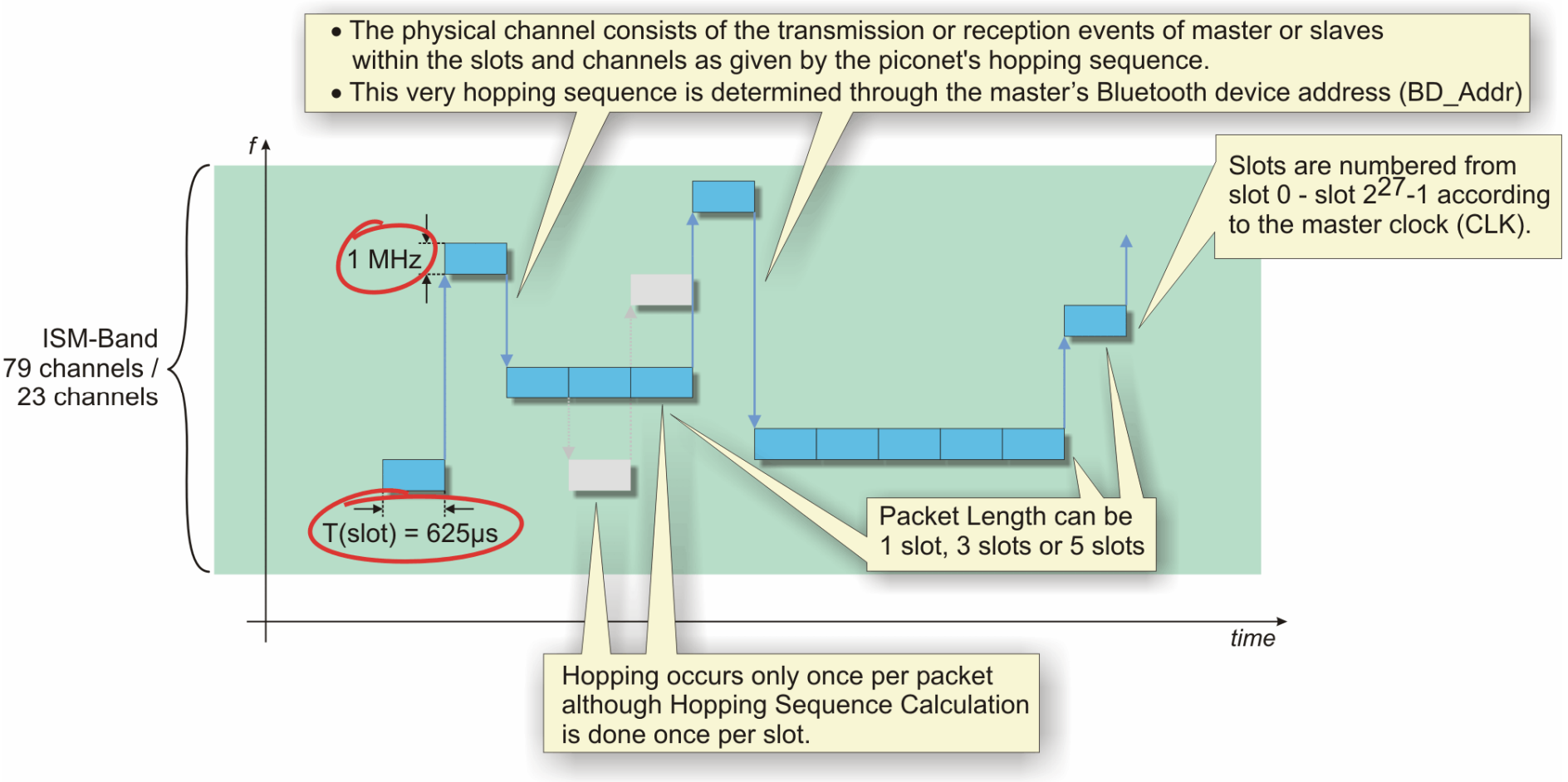
Part 1: Inquiry Mode and Being Discoverable

It is the user that tells the host software to configure the Bluetooth chip to be discoverable and / or to perform inquiries. As a result of the inquiry procedure, the detected (responding) devices are usually displayed to the (human) user. Note that at this stage, no local name can be displayed but only the device class and BD_ADDR which are included in the Baseband: FHS-packet.

Part 2: Paging Procedure

If the user decides to establish a connection to the found device, the host will send the respective commands through the HCI towards the Bluetooth chip which in turn will perform a Baseband: paging procedure. Note that the device that actively pages another device is the default master of the upcoming piconet although the two devices may swap their function. Following this paging procedure, the two devices may perform a whole set of optional LMP-procedures like exchange of "local name" or the exchange of supported features.

Details of the Basic and Adapted Piconet Physical Channels



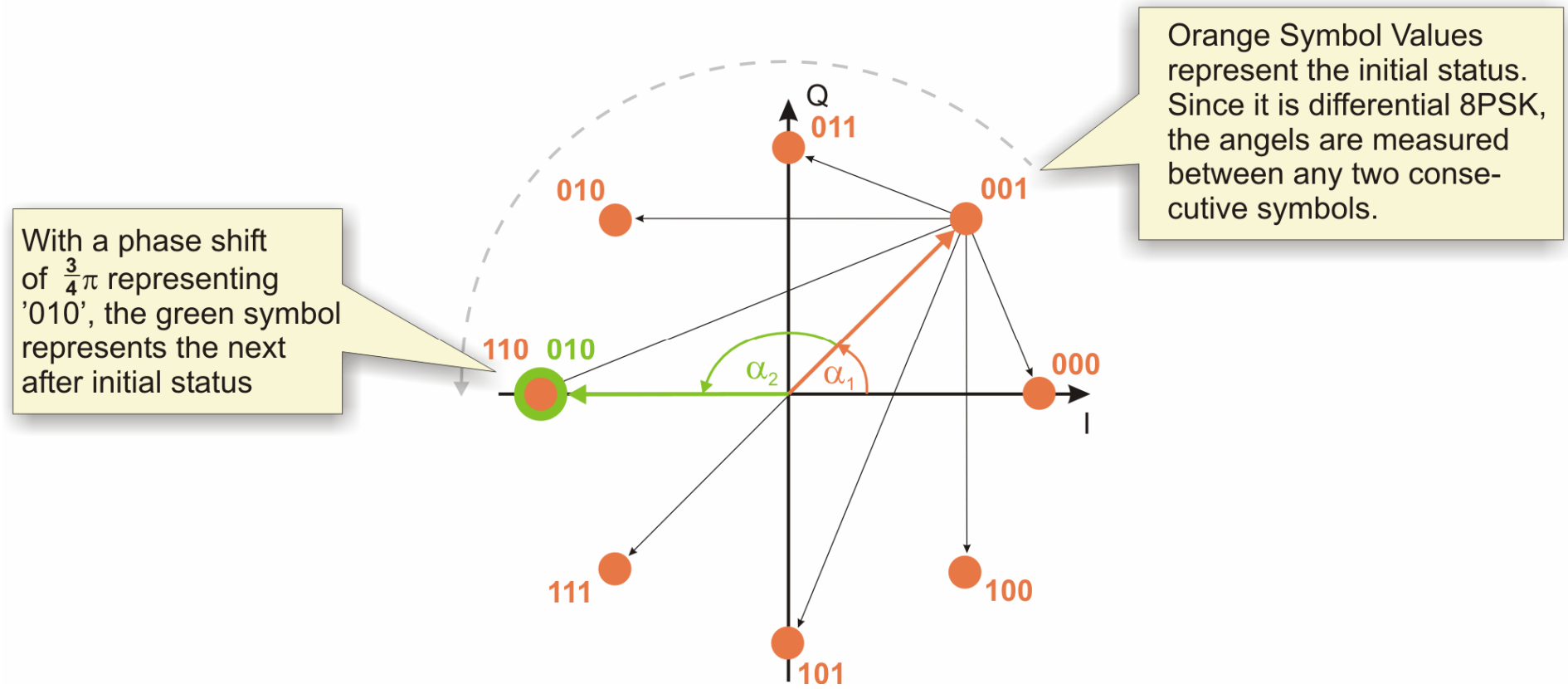
Details of the Basic and Adapted Piconet Physical Channels

The figure illustrates how the physical channel constitutes itself. The following important facts apply:

- **On the basic and adapted physical channels, packets can have a length of 1, 3 or 5 consecutive timeslots.**
- **Each slot has a duration of 625 µs and the channel bandwidth is 1 MHz.**
- **Slots are numbered according to the most significant 27 bits of the Bluetooth master clock (CLK) and rank from slot No 0 – slot number 2^{27}-1.**
- **The term "Adapted physical channel" relates to a reduced number of 23 frequencies. Note that the minimum number of channels available is N(min) = 20. [Bluetooth Core Spec Version 2.0, Volume 2, Part B (Baseband), Chapter 2.3.1]**
- **Frequency Hopping is an essential part of the Bluetooth technology. It is very important that master and slaves need to perform synchronized hopping and it is important to recognize that hopping occurs only once per packet and not once per slot.**

[Bluetooth Core Spec Version 2.0, Volume 2, Part B (Baseband), Chapter 2.2, 2.4]

8DPSK (8-phased Differential Phase Shift Keying)



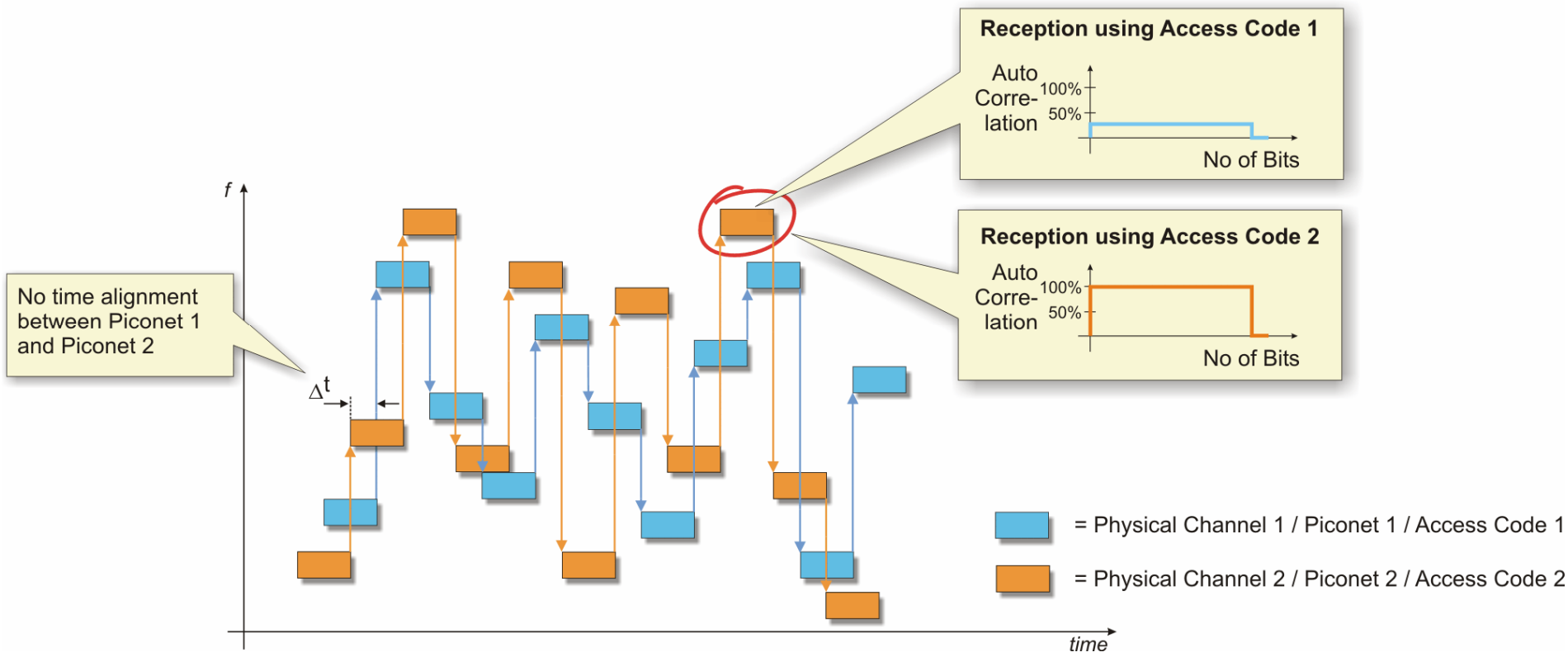
8DPSK (8-phased Differential Phase Shift Keying)

- ⇒ 8DPSK supports 3 bits per symbol. Accordingly, with a symbol rate of 1 Mega-symbols per second the bit rate is increased to 3 Mbit/s. Again, we like to emphasize that this is only a theoretical value
- ⇒ 8DPSK is a differential modulation scheme. Accordingly, the “Zero”-phase is re-initialized with every new symbol. We illustrate an example in the figure: Initially, the “Zero”-phase correlates with the I-plane. Then the first symbol is ‘001’ (⇔ $\alpha(1) = 45°$). The next symbol is ‘010’ which is represented by a phase shift of $\alpha(2) = 135° = 3/4\ \pi$. However, since we operate with 8DPSK rather than with plain 8-PSK, the actual signal vector jumps to a phase shift of $180° = \pi$.

[Bluetooth Core Spec Version 2.0, Volume 2, Part A (Radio), Chapter 3.2.1]

Details of the Access Code

- **Meaning of the Access Code**



Details of the Access Code

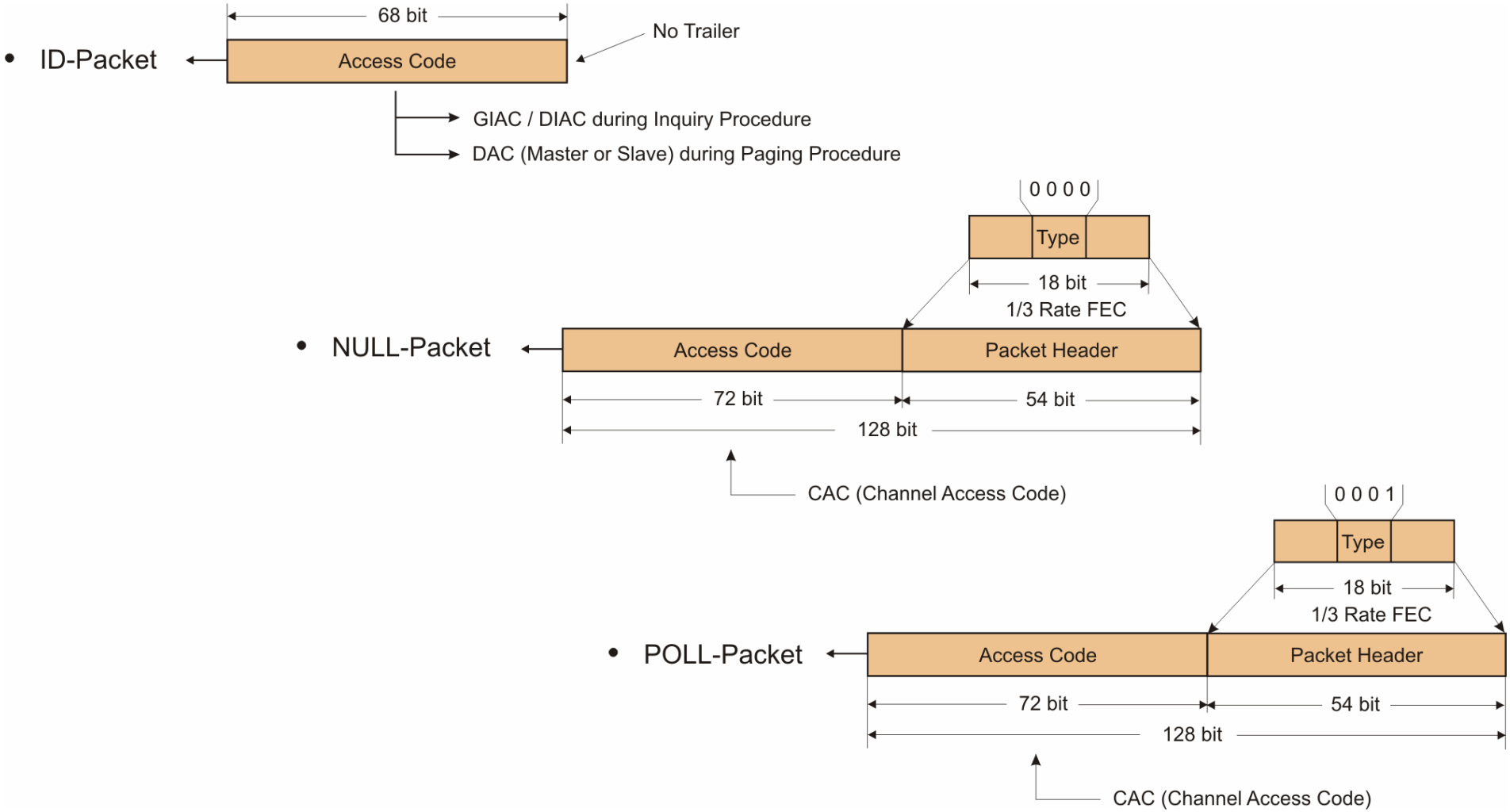
Meaning of the Access Code

Most importantly, the access code allows for an easy detection of packets which belong to the own piconet as the access code has strong autocorrelation capabilities.

The figure shall highlight the importance of the access code for piconet identification. In that respect, the figure illustrates the circumstances for two piconets which are collocated (but obviously not time aligned).
The two piconets should barely interfere if the hopping sequence provides for an infrequent match of the hopped-on frequency.

[Bluetooth Core Spec Version 2.0, Volume 2, Part B (Baseband), Chapter 6.3]

Common Baseband Packet Types



Common Baseband Packet Types

ID-Packet

The ID-Packet only consists of the access code. It has neither a packet header nor a payload section. The ID-packet is used during the Baseband: Inquiry- and Baseband: Paging-procedures.

[Bluetooth Core Spec Version 2.0, Volume 2, Part B (Baseband), (6.5.1.1)]

NULL-Packet

The NULL-packet consists only of the access code and the packet header. It is used by a device to provide ARQ-information (⇔ ARQN) and flow control information (⇔ FLOW-bit) to a peer. It can also be used as response for a received POLL-packet.

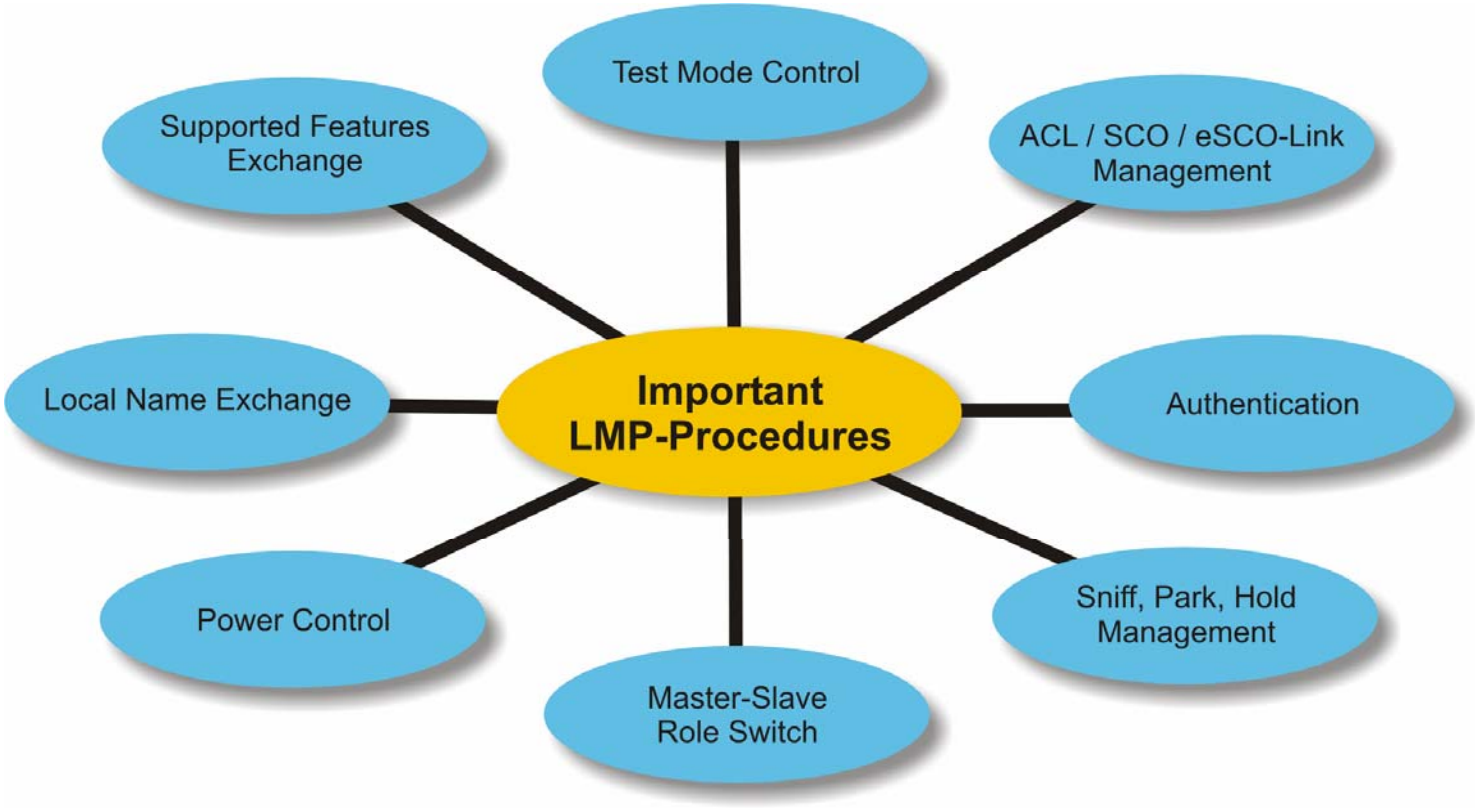
[Bluetooth Core Spec Version 2.0, Volume 2, Part B (Baseband), (6.5.1.2)]

POLL-Packet

The POLL-packet consists only of the access code and the packet header. It shall only be used by a piconet master to request packet transmission from a slave when the master has no other information to transmit.

[Bluetooth Core Spec Version 2.0, Volume 2, Part B (Baseband), (6.5.1.3)]

Important Link Manager Procedures



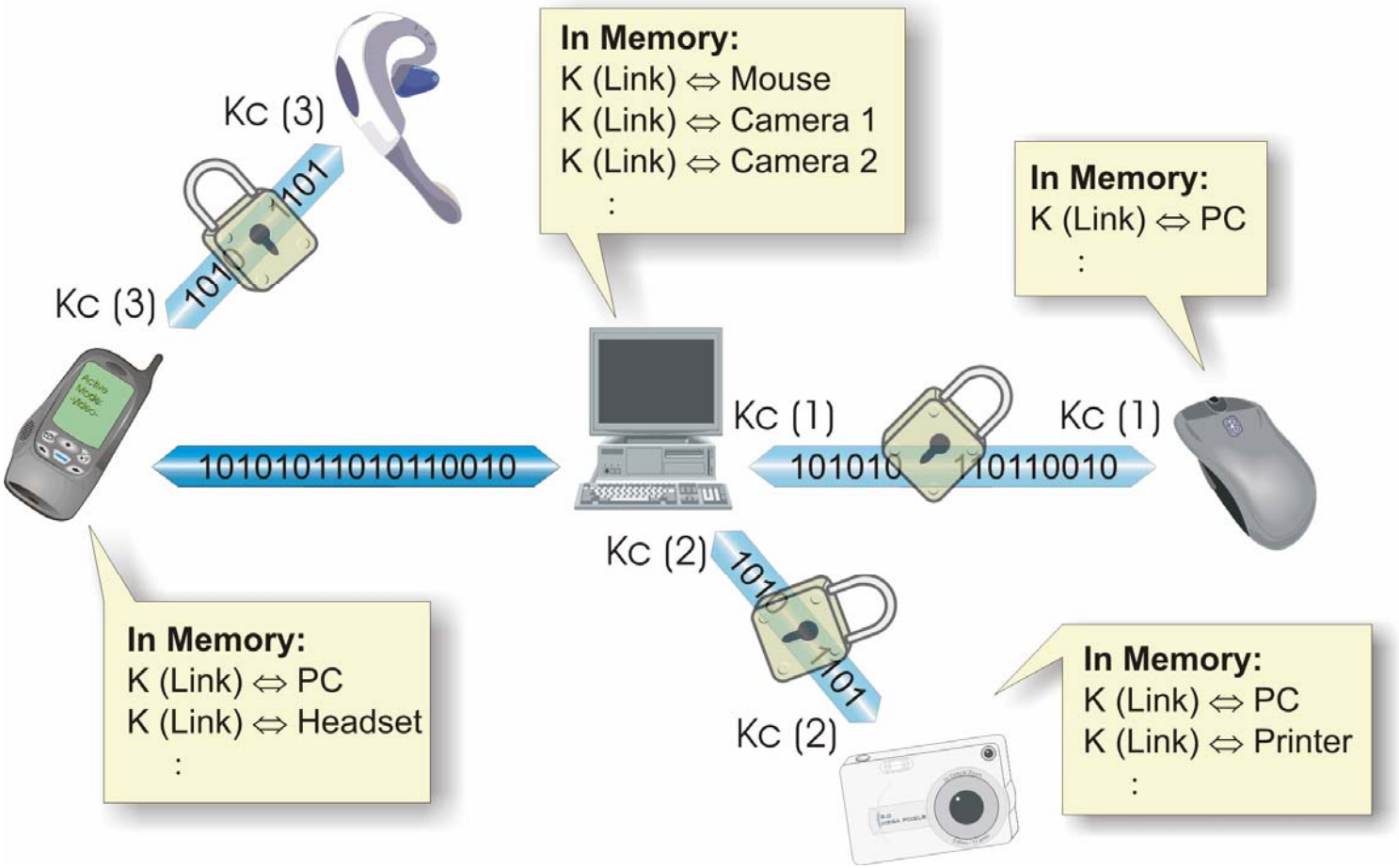
Important Link Manager Procedures

Intentionally left blank

[Bluetooth Core Spec Version 2.0, Volume 2, Part C (Link Manager), (4)]

Authentication and Encryption

- **Overview**



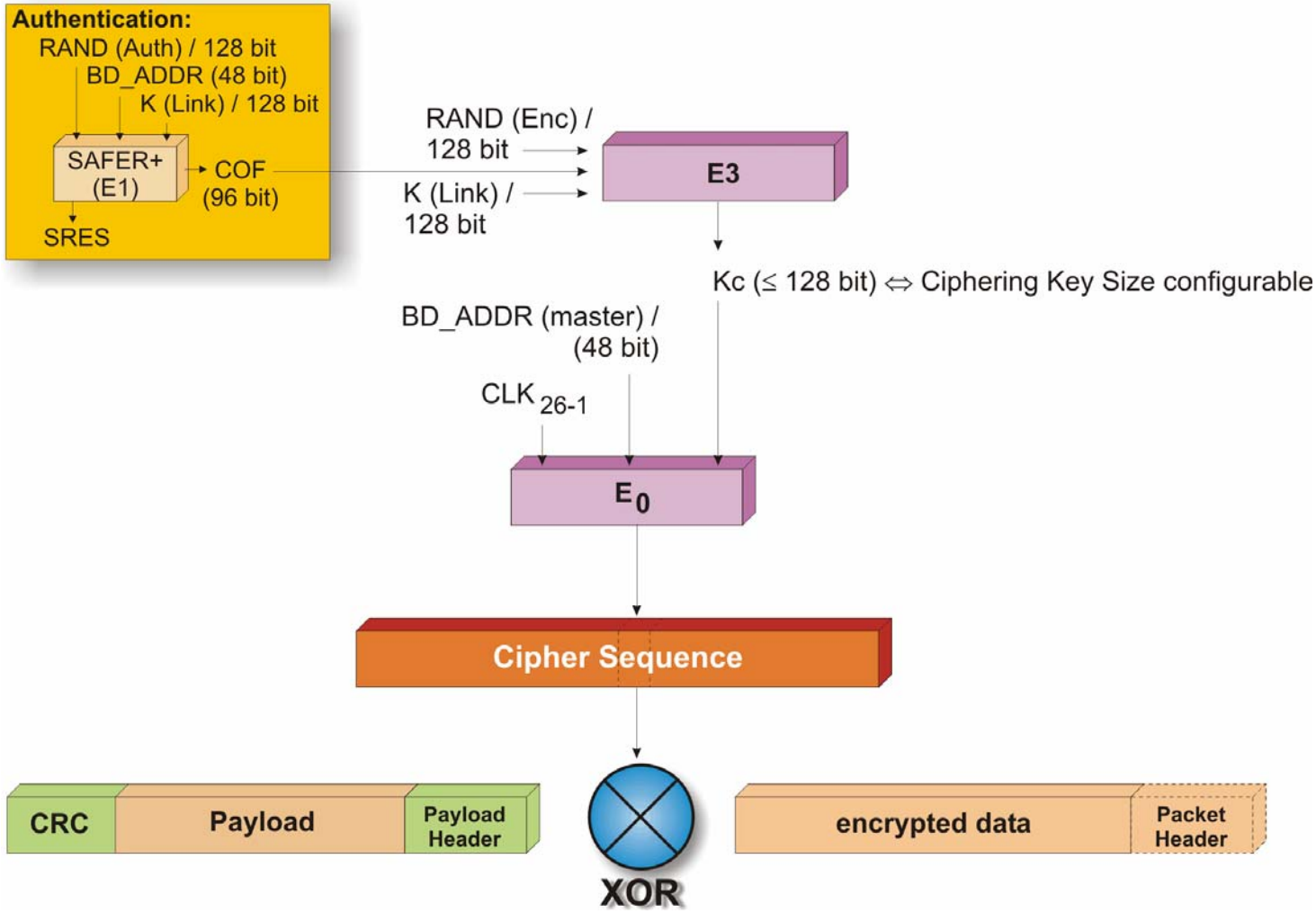
Authentication and Encryption

Overview

- **The figure illustrates a typical Bluetooth piconet and emphasizes the security relevant issues:**
- **Most importantly, the master (the PC in the center) needs to apply different ciphering keys Kc to each slave.**
- **Each device typically stores the link key for all bonded devices in a non-volatile buffer to allow for an easy reconnection.**
- **Encryption is optional. The link between the PC and the mobile phone is not encrypted.**
- **However, the mobile phone is master in a second piconet with the Bluetooth headset. This connection is ciphered.**

[Bluetooth Core Spec Version 2.0, Volume 2, Part H (Security Specification)]

The Encryption Process



The Encryption Process

The encryption process itself is illustrated in the figure. Note that initially, the ciphering key Kc needs to be calculated:

Calculation of COF (Ciphering Offset Value)

The COF is calculated during the authentication procedure and is one input for the E3-algorithm.

Calculation of Kc

The variable length ciphering key Kc is calculated based on the link key K(link) which has been calculated during the authentication process, on an encryption specific variable RAND(enc) that is relayed to the peer when ciphering is being switched on and on the already mentioned COF.

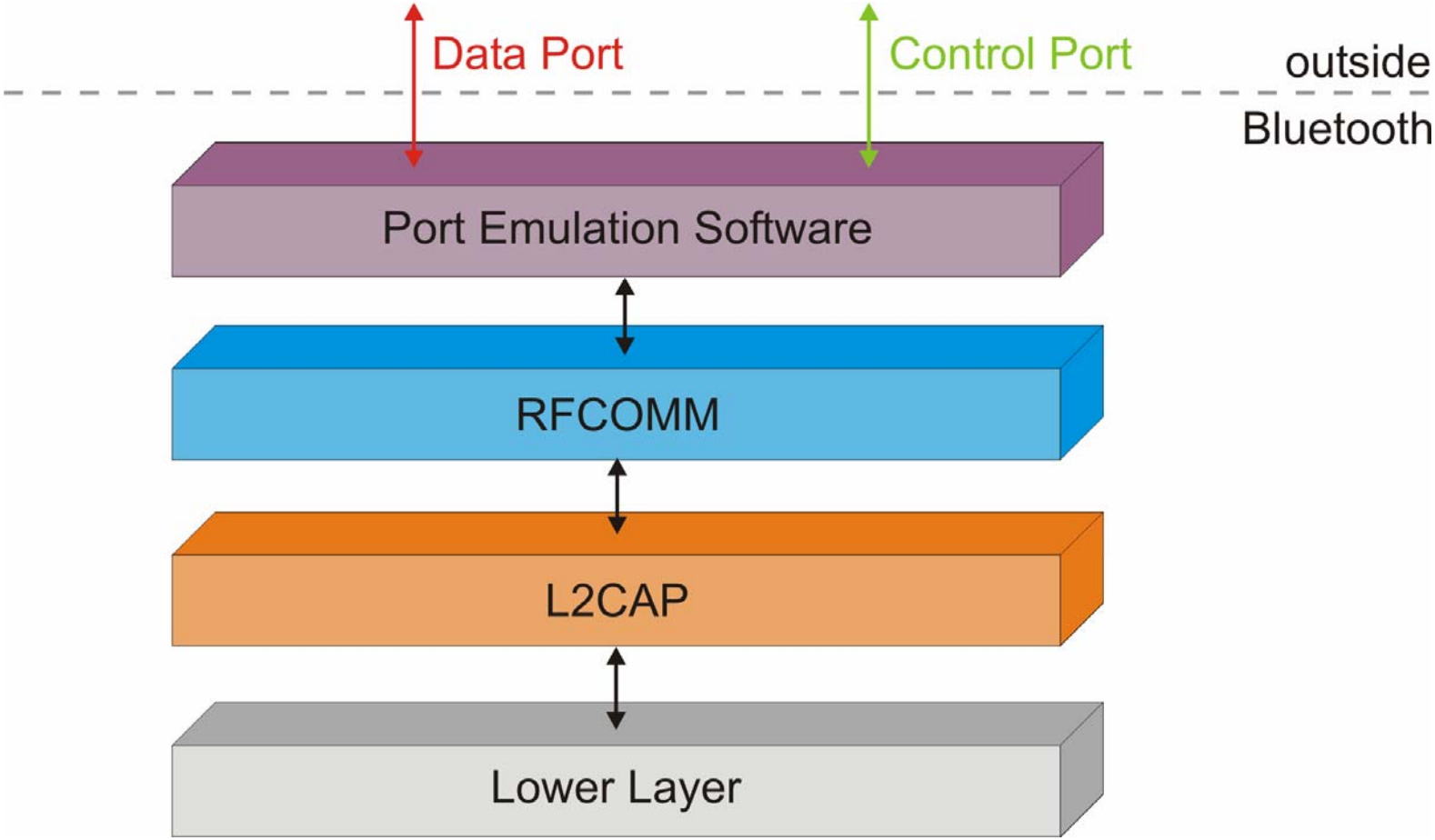
The Bluetooth SIG allows for variable length Kc-values to suit national regulations that do not allow for 128 bit long ciphering keys.

Ciphering

Ciphering itself is based on the E0-algorithm that used the BD_ADDR of the master as input as well as the ciphering key Kc. The use of the 26 bits CLK(26-1) of the master's clock guarantee that the created ciphering sequence changes with every packet to be encrypted.
As illustrated in the figure, the entire payload including payload header and CRC is ciphered but not the packet header.

[Bluetooth Core Spec Version 2.0, Volume 2, Part H (Security Specification), (4)]

The RFCOMM-Protocol



The RFCOMM-Protocol

- ⇒ The RFCOMM-protocol resides on top of L2CAP. Its main task is the emulation of RS232 serial ports towards applications. RFCOMM is usually implemented with operating-system specific port emulation software that serves as interface for control information and data towards the application.
- ⇒ The control information may be based on AT-commands.
- ⇒ RFCOMM has not been invented by the SIG. RFCOMM has rather been inherited from ETSI, to be more precise it copies many capabilities of the specification GSM 07.10 (now called 3GTS 27.010).
- ⇒ Both RFCOMM and 3GTS 27.010 are used to emulate the RS-232 port and its connections. As such, RFCOMM is capable to relay information like RTS (ready to send) and similar through L2CAP and the remaining Bluetooth stack towards a peer.
- ⇒ The functions of RFCOMM shall remain transparent towards the application.

[Bluetooth Core Spec Version 2.0, RFCOMM Specification, www.3gpp.org (3GTS 27.010)]