4. Bluetooth

Bluetooth is a **Wireless Personal Area Network** which defines a class of wireless networks providing connectivity between **mobile** or **immobile devices** in the operating space of a person (with a radius of approx. 10 m around the person).

Further Information

Online sources:

- Bluetooth Specifications, **http://www.bluetooth.com/** - **http://www.bluetooth.org/**
- many “white papers” about Blutooth available

Bluetooth security:

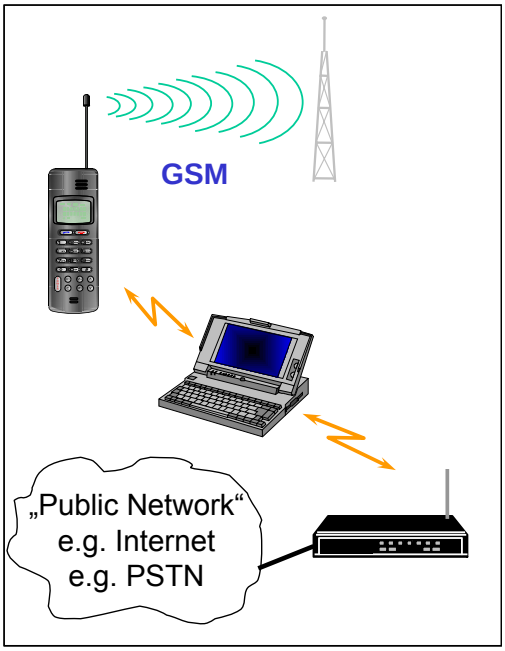
- M. Jakobsson and S. Wetzel, „Security Weaknesses in Bluetooth“, Bell Labs, 2001, http://www.cs.stevens-tech.edu/~swetzel/publications/bluetooth.pdf
- M. Schmidt, „Blauzahnlücken, Angriffsmöglichkeiten bei Bluetooth“, c‘t 11/03, p. 176

4.1. Wireless Personal Area Networks (WPANs)

With the advent of small electronic devices, the need for communication in the vicinity of a person arises.

„Wireless Personal Area Networks“ (WPANs) were (and are) developed to fulfill these communication needs. These networks should:

- **Replace cables**
  (cables are clumsy to handle, have various incompatible connectors),
- **Replace infrared communication (IrDA)**
  (WPANs have a higher data rate and require no direct line of sight),
- Enable **seamless communication** of computers, peripherals, handhelds, PDAs, cell phones, ...
  (includes the integration of voice and data),
- Communicate with devices within the **personal operating space** (POS, radius approx **10 m**)
  (for some applications: up to 100 m),
- Have **low power** requirements
  (to be powered with batteries),
- Have functionality similar to a **LAN**
- Support various types of **access points**



Bluetooth

The **Bluetooth* SIG** (**Special Interest Group**) is an association of industry leaders in telecommunication driving the development of the Bluetooth WPAN technology.

Bluetooth is

- **optimized for mobile, portable devices**
- **globally usable** (operates in the license-free 2.4 GHz band),
- **robust,**
- **low-cost,**
- **short ranged** (approx. 10 Meter).

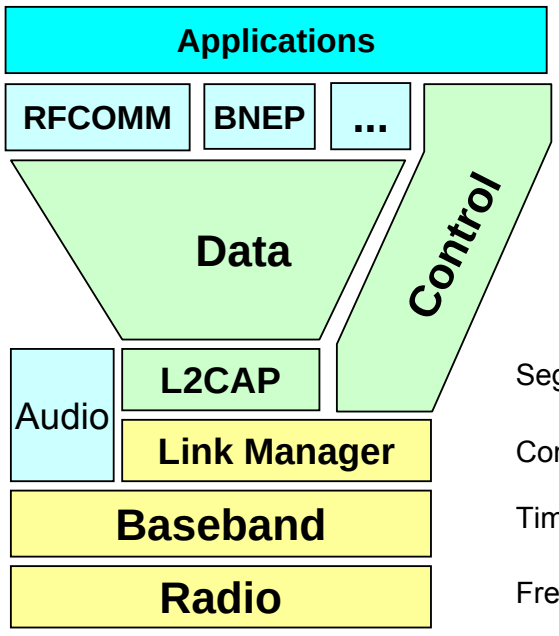
*The Bluetooth technology was code-named after the 10th century Danish king Harald Blaatand (Bluetooth), approx. 950-986, son of the first Danish king Gorm the Old. Harald Blaatand erected a rune stone in Jelling. The stone‘s inscription says that Harald christianized the Danes and controlled Denmark and Norway. Although originally intended as a code name for the technology, the name stuck.
More info e.g. at http://www.answers.com/topic/bluetooth

4.2 The Bluetooth Specification

4.2.1. Bluetooth Specification: Overview (1)

Bluetooth Core Specification:

- Defines the **protocol stack**
- **Physical** definitions (radio) as well as **definition of protocols**
- Specifies **test interfaces** and „**compliance requirements**“



- **RFCOMM:** Emulation of a serial cable
- **BNEP:** Bluetooth Network Encapsulation Protocol (transport of Ethernet frames)
- **SDP:** Service Discovery Protocol
- **...**

Segmentation/reassembly, multiplexing, ...

Connection and link management, encryption, ...

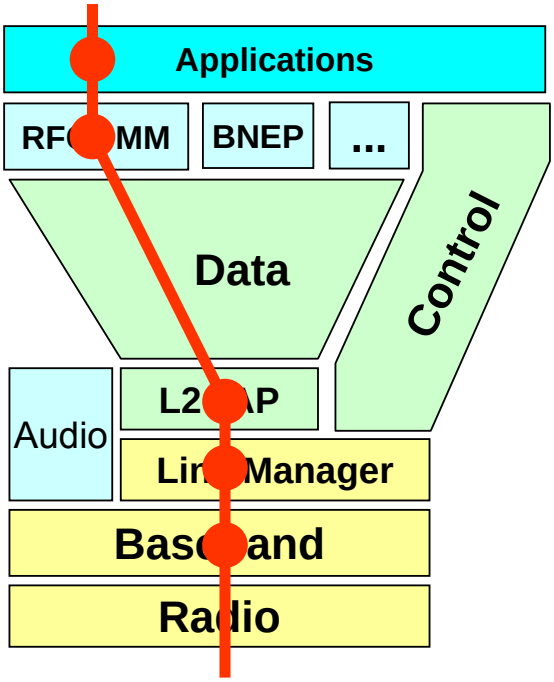
Timing, Framing, Medium Access, ARQ, Flow Control

Frequencies, modulation, transmit power, ...

Bluetooth Specification: Overview (2)

Bluetooth Profiles:

- Standardized solutions for specific **use cases**, e.g.
  - Personal Area Networking Profile
  - File Transfer Profile
  - Serial Port Profile (different from the RFCOMM protocol!)
- Each device supports one or more profiles
- Profiles define vertical cuts through the protocol stack:
  - Which **protocols** are used?
  - What are the **requirements** to be fulfilled?
  - How can devices tell whether another device supports a profile and which features of the profile it implements?
- Profiles are the basis for **interoperability** of Bluetooth applications and for the **certification** of Bluetooth solutions



4.2.2. Bluetooth Radio

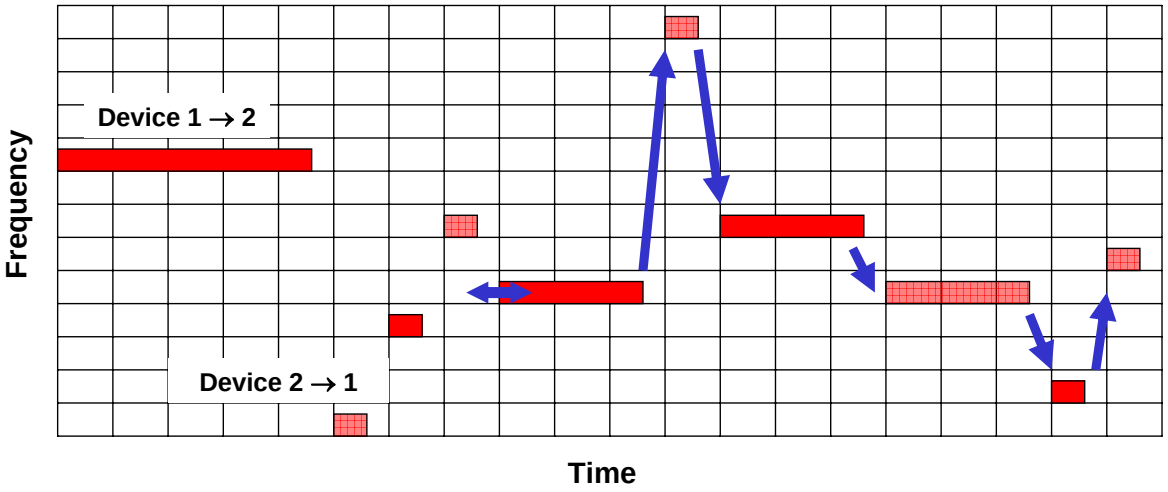
Properties of Bluetooth Radio:

- Bluetooth uses the **2.4 GHz ISM band** (ISM = Industrial, Scientific, Medical)
- Three output power classes are defined:
  - Class 3: max. 1 mW
  - Class 2: max. 2,5 mW
  - Class 1: max. 100 mW, power control required

  Range approx. 10 m to 100 m
- Within the frequency band, **79 channels of 1 MHz width are defined**
  - **Frequency Hopping** Spread Spectrum System (up to 1600 hops/s):
    - Sequence of channel changes realizes one virtual channel per piconet
    - Provides resilience against interference und frequency selective fading
  - Raw data rate 1 mbit/s (Modulation: Gaussian Frequency Shift Keying)
  - Max. **net data rate** 723 kbit/s for data
  - Max. 3 simultaneous full-duplex 64 kbit/s **voice channels**
- **Simple design** to reduce costs: frequency hopping, modulation, receiver characteristics, ...

Frequency Hopping

Communicating devices „hop“ through the frequency space in a coordinated fashion:



Communicating devices must agree in:

- **Hopping Sequence**
- **Timing**

Devices must be in **interference range** and use the **same frequency** at the **same time** on order to provoke a **collision**!
⇒ A few devices using **different hop sequences** do not cause much harm to each other

4.2.3. Baseband: Bluetooth Piconet

Each device in a Bluetooth network has a (globally unique) IEEE 802 **48 bit address (BD_ADDR)** and a **clock (CLK)** running at 3200 Hz.

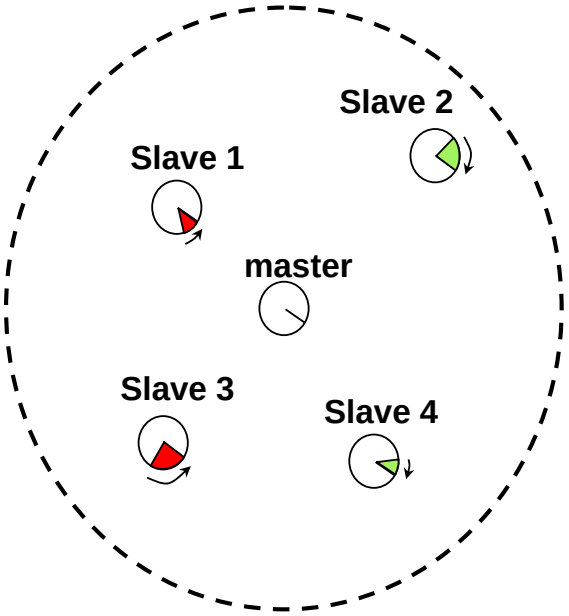
A Bluetooth network (a Piconet) consists of:
- a **Master** and
- at least one **Slave**

synchronized to each other

The master determines the frequency hopping sequence from:
- its BD_ADDR
- its clock

→ **Pseudo random** hop sequence

Slaves calculate the same sequence using:
- BD_ADDR of the master
- Difference of own and master‘s clock: **clock offset**



Properties of a Piconet:
- **Master/Slave communication** only (no direct communication between slaves)
- Up to 7 active slaves
- Additional inactive “parked” slaves (up to 255)

Bluetooth Medium Access Control (MAC)

In a piconet:

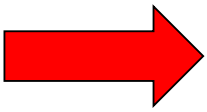
- All devices are **synchronized** to the master
- At each **point in time** the hop sequence determines a **frequency** to send or receive on

Only **one device** of a piconet may send at a time, otherwise the devices interfere with each other!
A **medium access method** is needed in order to use the medium efficiently!

Local Area Networks often use random access methods (e.g. CSMA/CA, CSMA/CD) that may cause **collisions** on the medium.

Due to collisions, random access methods cannot guarantee packet delivery within a given time bound.

But: Bluetooth was designed to transport **voice calls**. Given the restricted overall bandwidth of Bluetooth, it is hard to fulfill the hard requirements on **delay** and **jitter** using random access methods.



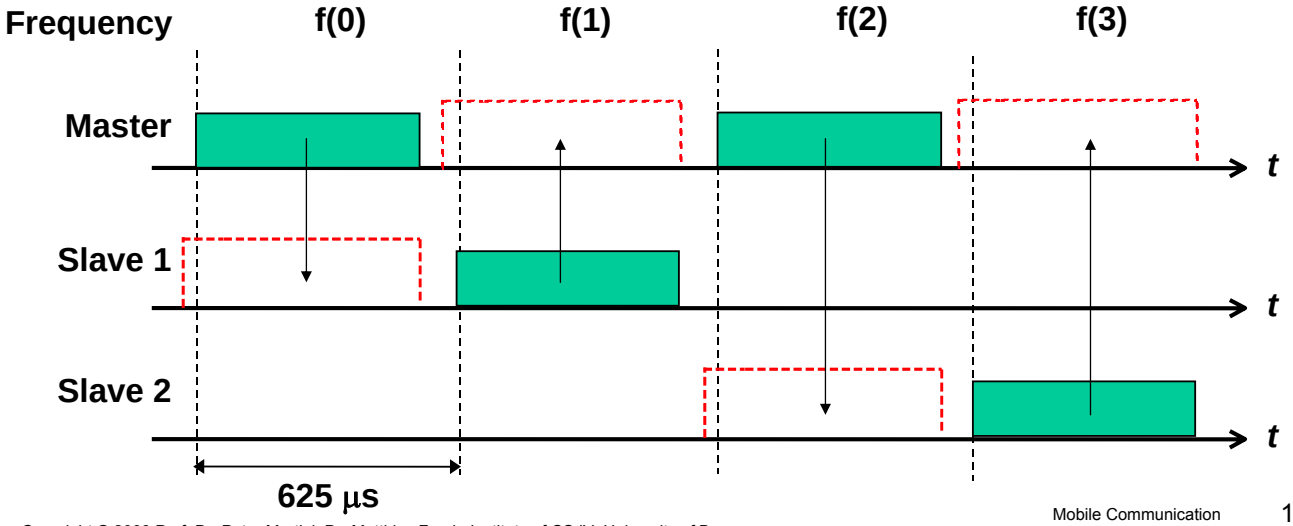
Bluetooth uses a simple, contention-free „Polling“-MAC:

Time-Division-Duplex (TDD)

Time-Division-Duplex (1)

- The master determines **slots with fixed length** (625μs). Additionally, each slot is assigned a frequency using the hop sequence
- The **master** may send a unicast packet to a specific slave or a broadcast packet to all slaves in the **even slots only**
- In the **odd slot** subsequent to a master transmission, the **slave addressed by the master** may respond

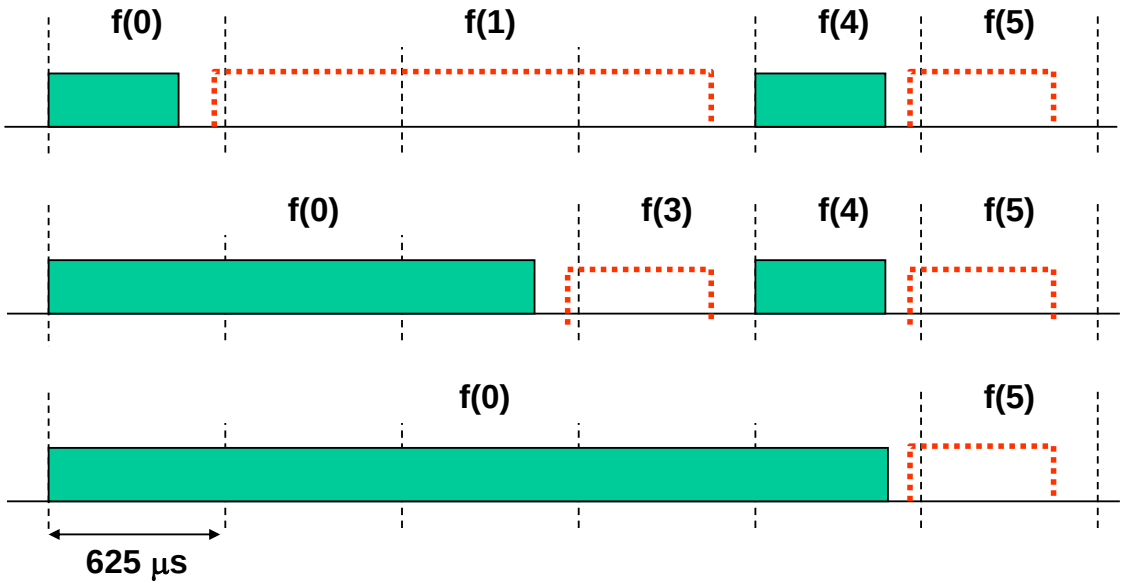
A slave may **never** send on its own, it always has to be **polled** (using some packet) by the master!
It is the duty of the master to ensure that slaves are polled at times.



Time-Division-Duplex (2)

In order to increase efficiency, packets longer than a single slot were introduced:

- The odd/even slot rule only allows packets spanning an odd number of slots
- Bluetooth uses **1, 3 and 5 slot packets**



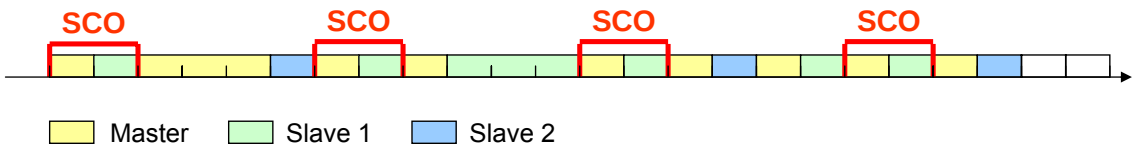
Frequencies cannot be switched instantly. In order to avoid performance losses due to frequency switching, packets are always sent on a **single frequency**.

Types of Links

Bluetooth supports two types of links:

- **Synchronous Connection Oriented (SCO) link** (typically used for voice)
  - **Point-to-point full duplex connection** between master and slave
  - Master **allocates** slot pairs using a fixed period
  - May use a code insensitive to random bit errors (CVSD, Continuous Variable Slope Delta Modulation)
- **Asynchronous Connectionless (ACL) link** (typically used for data)
  - **No reservation of slots** (master decides which slave to address)
    - Slots reserved for SCO have priority
  - **TDD scheme**: A slave may only send when polled by the master
  - **Packet header** indicates which slave is addressed

In case of bad link conditions, it may be beneficial to secure packets against transmission errors by applying **Forward Error Correction** (**FEC**).



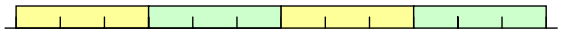
Packet Types

Depending on the link type, the following packet types are defined:

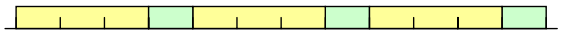
- **Asynchronous Connectionless (ACL) link**
  - **DM1, DM3, DM5**, 2/3 Rate FEC, CRC, 1, 3 and 5 slot packets
  - **DH1, DH3, DH5**, no FEC, CRC, 1, 3, and 5 slot packets
  - **AUX1**, no FEC, no CRC, 1 slot packet
  - Max. data rates in kbit/s:

	Symm.	Asymmetric	
DM1	108,8	108,8	108,8
DH1	172,8	172,8	172,8
DM3	258,1	387,2	54,4
DH3	390,4	585,6	86,4
DM5	286,7	477,8	36,3
DH5	433,9	723,2	57,6

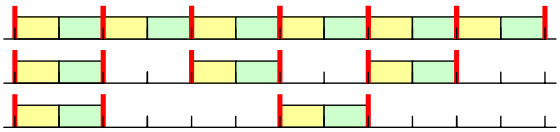
Symmetric: DM3/DM3



Asymmetric: DM3/DM1

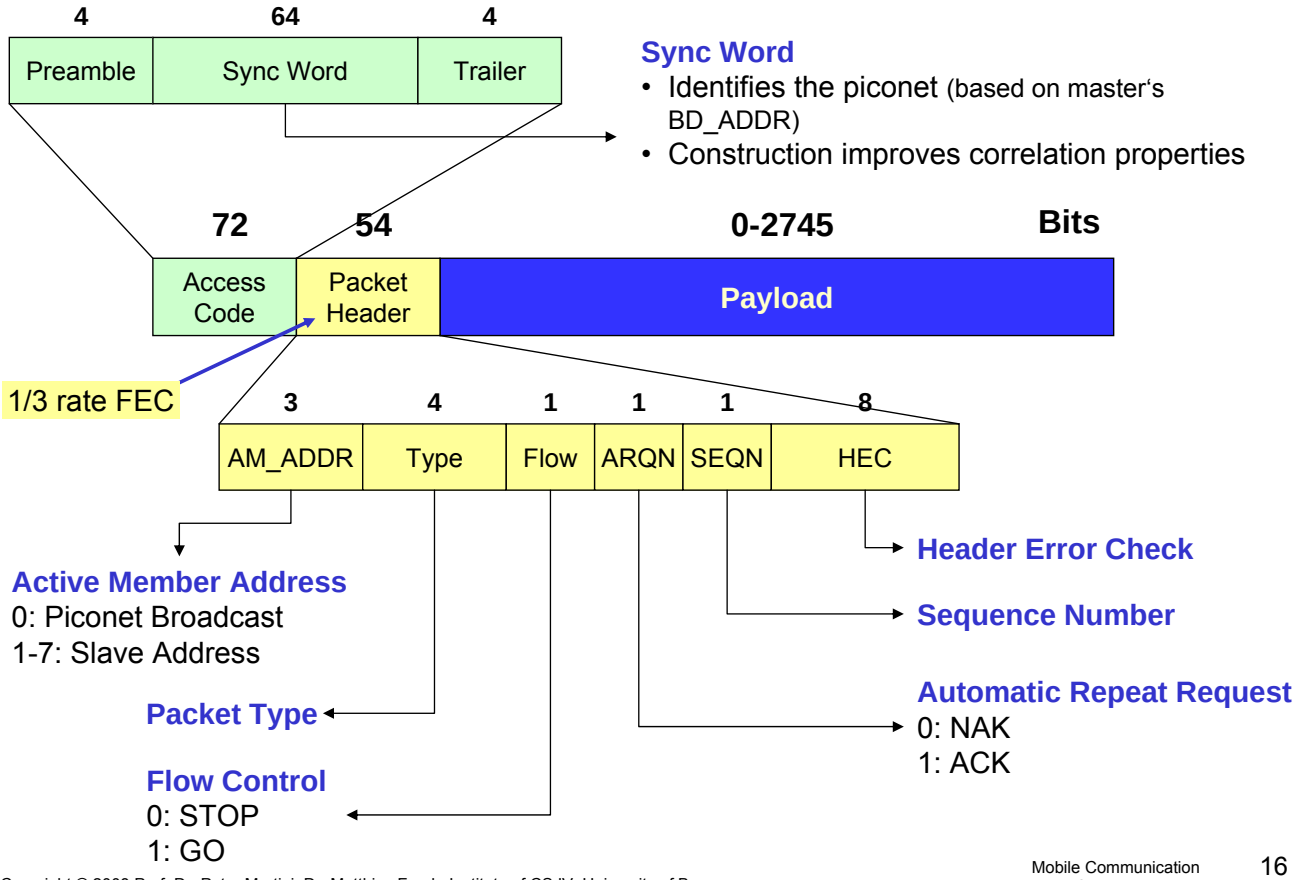


- **Synchronous Connection Oriented (SCO) link**
  - **HV1**, 1/3 Rate FEC, 64 kbit/s full duplex
  - **HV2**, 2/3 Rate FEC, 64 kbit/s full duplex
  - **HV3**, no FEC, 64 kbit/s full duplex



  - **DV**, 64 kbit/s audio (SCO) full duplex + 57,6 kbit/s data (ACL) symmetric
  - **DM1**, used for link management

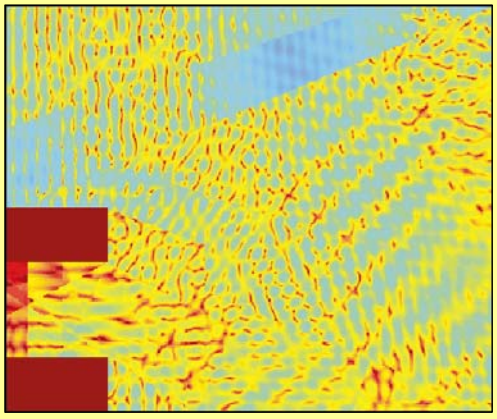
Packet Format



Automatic Repeat Request (ARQ)

Error rates in wireless communication systems are much higher:

- **Interference**
  - Other piconets in interference range may use the same frequency by chance
  - 2.4 GHz Band is unlicensed:
    - Microwave ovens
    - 802.11 Wireless LAN
    - Other applications like video transmissions, etc.
- **Complex propagation characteristics**
  - Small changes in position may lead to huge variations in signal strength (**small scale fading**)



Source: IEEE 802.11 Standard

Automatic Repeat Request (ARQ)

What happens if a packet is not received at all or is not received correctly?

Common practice for low error rates experienced in wired networks (e.g. Ethernet):

- Discard packets with bad CRC
- Higher layers are required to correct the error (e.g. link layer or transport layer)

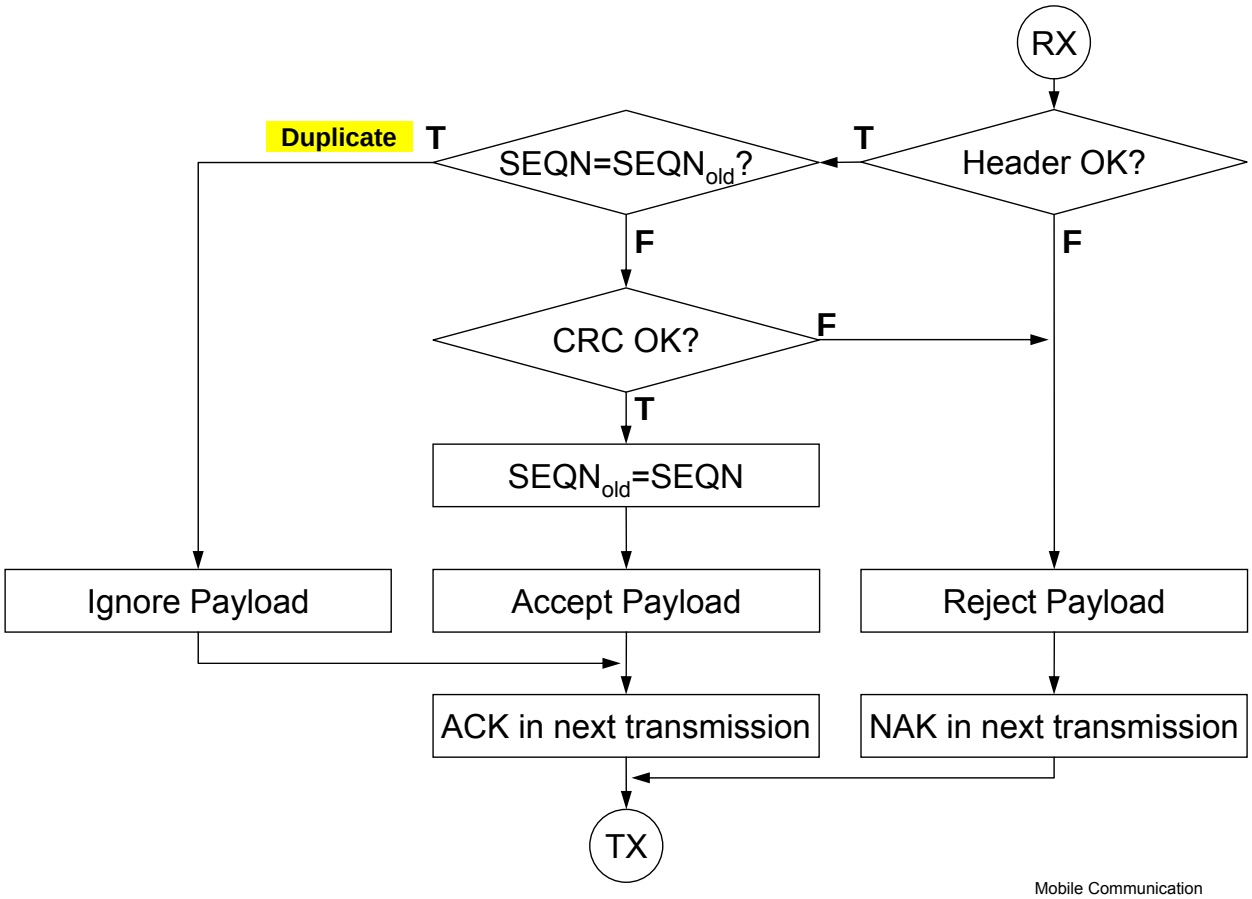
Error rates in wireless communication system are much higher:

- Higher layer correction is too slow in many cases
- Higher layers (TCP in particular) may assume that packet losses are caused by congestion instead of packet errors
  → Congestion avoidance reduces data flow erroneously

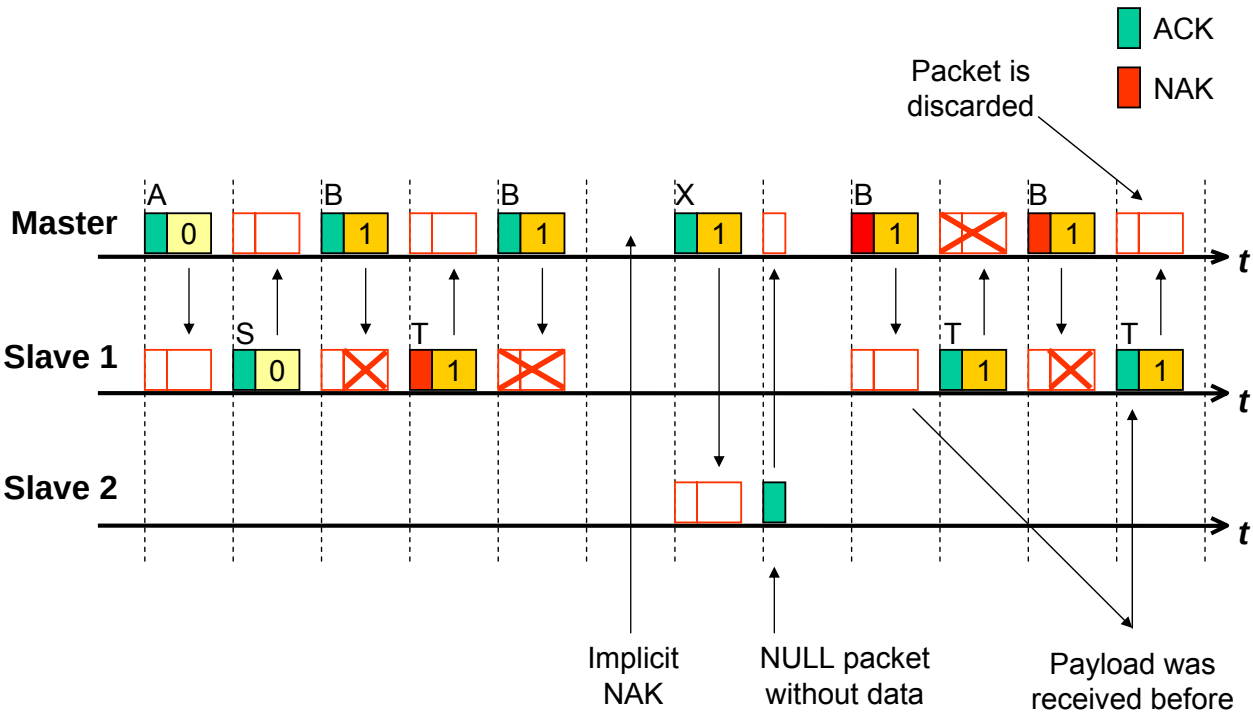
Bluetooth baseband uses a **fast automatic repeat request** (**Fast ARQ**) for ACL data packets:

- **Piggy-backed** ARQN-Bit in the header of the return packet
  - If there is no data to send, a short packet without data (NULL or POLL) is used
  - If the slave sends no return packet, **NAK is implicit**
- Acknowledgement occurs as **fast** as possible
  - Master-to-slave: ACK in the next odd slot following the current transmission
  - Slave-to-master: ACK at next poll of the slave
- Duplicate packets filtered by **alternating sequence numbers**
  - Sequence number switches ($0 \rightarrow 1$, $1 \rightarrow 0$) for each new packet

The Bluetooth Receive Protocol for Data Packets



ARQ: Example



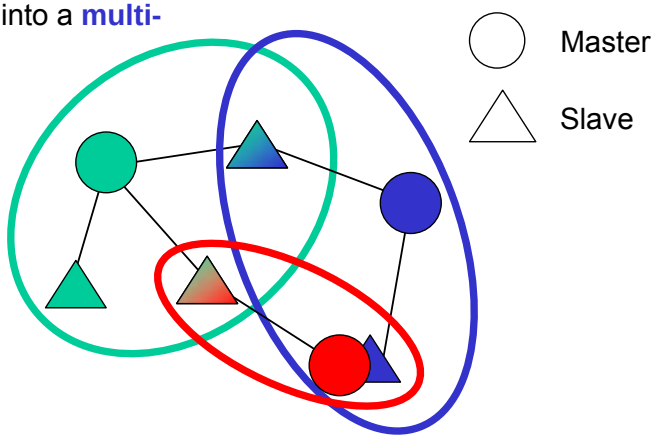
Multi-Hop Bluetooth Networks: Scatternets

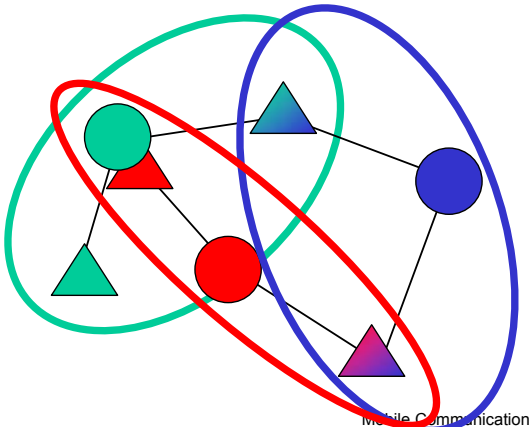
Multiple Bluetooth piconets may be combined into a **multi-hop Bluetooth network**, a „**scatternet**“.

A Bluetooth device connected with several other devices may be:

- A master to all connected devices (slaves)
- A slave to all connected devices (masters)
- A combination thereof: master to connected slaves and slave to different masters

- The same physical topology can be achieved using different master/slave assignments.

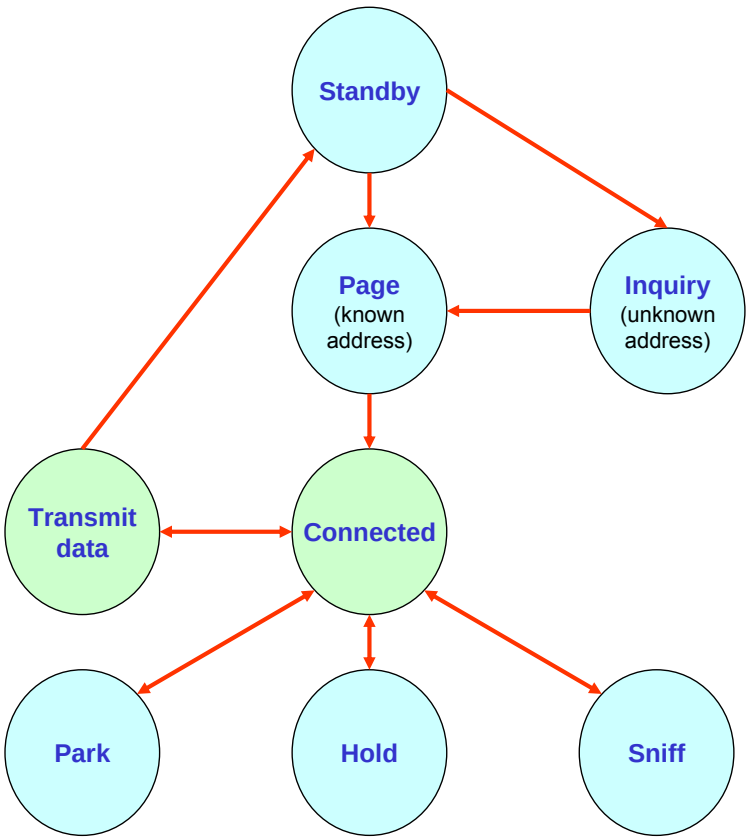




Each device can only participate in one piconet at each point in time

Devices switch between piconets using a coordinated time division multiplex scheme

Connection Management in the Piconet



Not connected

Connection establishment

Active

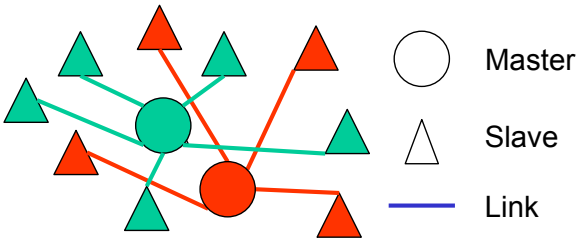
Power saving

Connection Establishment (1): Page

Being in communication range is not sufficient to establish a link!

Master and slaves of a piconet must agree in:

- **Hop Sequence**
- **Timing**



New slaves must get to know these parameters from the piconet's master!

Page Scan (passive role, future slave):

- Each **1.28 s** (or 2.56 s), a device scans **one** of its 32 **page frequencies** for approx. 11 ms.
- The slave responds if it is hit by a paging device during the scan: initial synchronization
- The **32** page frequencies are based on the device's **BD_ADDR.**

Paging (active role, future master):

- The master sends **page trains** using **16** of the 32 page frequencies of the slave.
- The page train is **repeated 128** (or 256) **times**, this corresponds to the **1.28 s** (or 2.56 s) needed to **hit** the slave's **scan window**.
- If no answer is received, the **remaining 16** frequencies are tried.
- The **master/slave roles** may be **switched** after a successful connection establishment.

Connection Establishment (2): Inquiry

Paging only connects to devices that are already known:

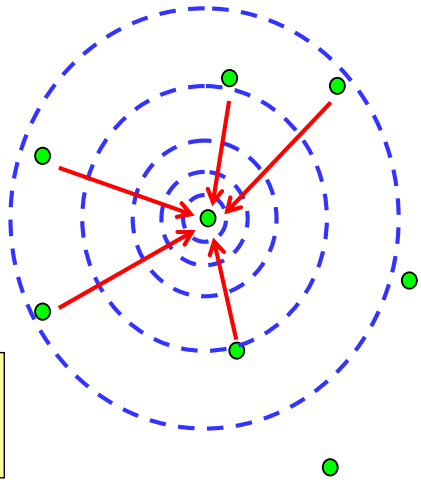
Paging needs the BD_ADDR of the slave

Using **inquiry**, a device may search for other Bluetooth devices in range:

- Similar to page procedure
- Detectable devices periodically enable **inquiry scan**
- Devices searching other devices switch between two **inquiry train repetitions**

An inquiry response contains:

- **BD_ADDR** (used for paging)
- Device **clock** (speeds up paging)
- **Class of device** (e.g. Access Point, Audio, Telephony, ...)

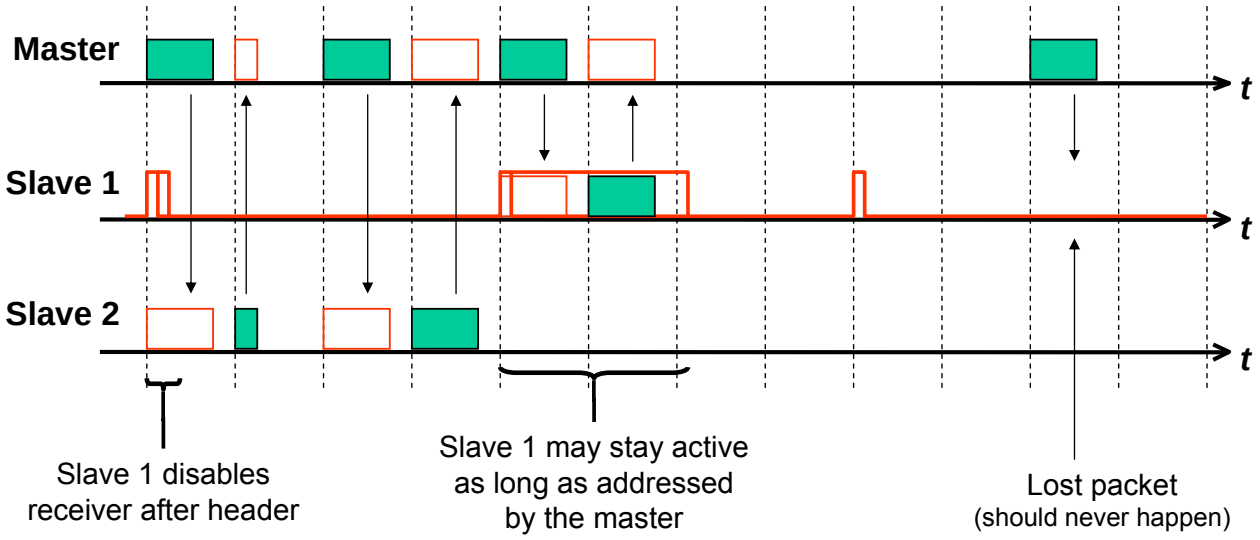


Inquiry does not establish a connection!
Devices must page the BD_ADDR obtained from the inquiry response if a connection is desired.

Power Saving: SNIFF Mode

In order to save power, an **active slave** minimizes its listen time:
- It listens for a **valid access code** only at the beginning of each **even slot**.
- If an access code is received, it has to listen to the **packet header** in order to figure out the intended recipient. Otherwise, disable the receiver till the next even slot.
- If the **packet** is for the slave, receive the packet. Otherwise, disable the receiver for the duration of the packet.

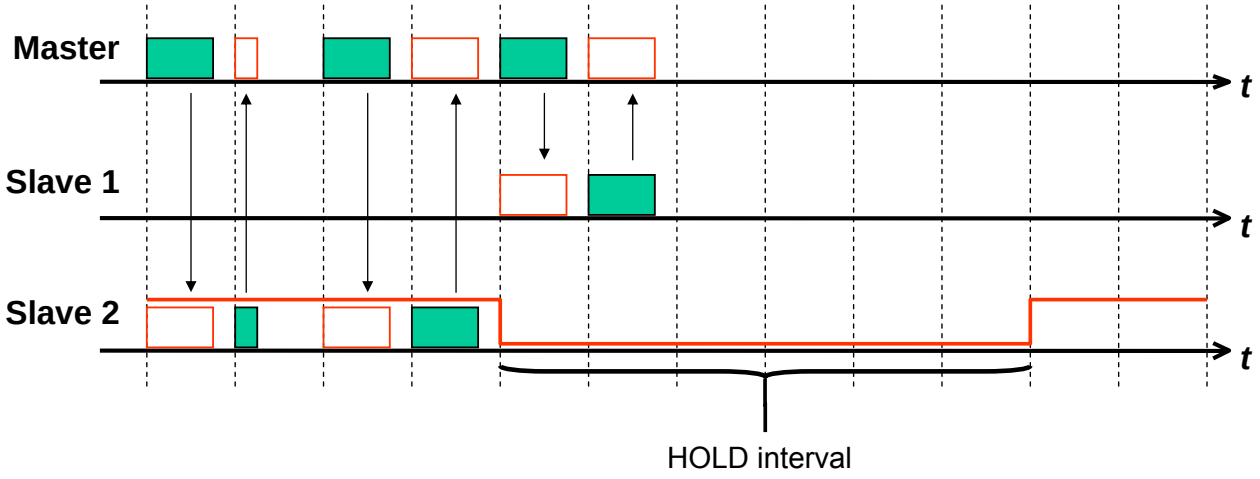
SNIFF mode reduces the duty-cycle even further: Master and slave agree on a periodic subset of even slots in which the slave listens.



Power Saving: HOLD Mode

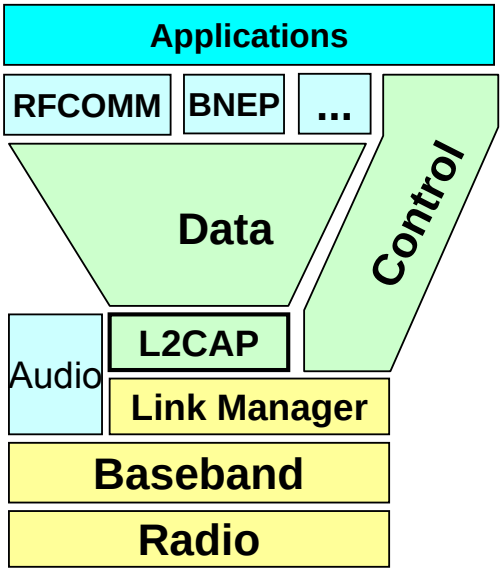
A slave may want to cease activity in a piconet for a specific time because
- it needs bandwidth for an **inquiry or page,**
- it wants to be active in **another piconet** (in a scatternet), or
- it wants to **save power.**

The slave and the master agree on a **HOLD interval**. After the interval has passed, the slave **resynchronizes** to the master.



4.2.4. L2CAP (1)

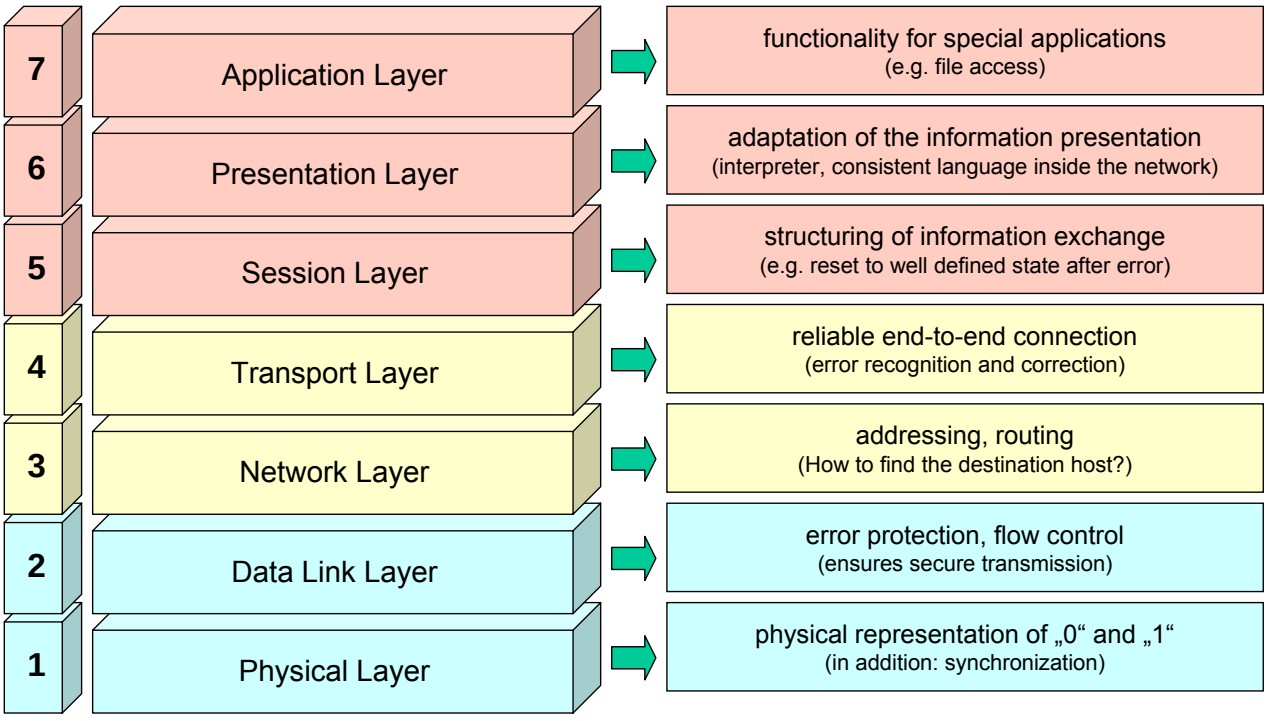
Common network protocols (e.g. IP) are not optimized for directly interfacing baseband:



- **Small packets**
  - Largest baseband packet (DH5): **341 data bytes**
  - Largest Ethernet packet: **1500 data bytes**
  - Use of packet types may depend on link quality:
    - **Good link**: 1500 bytes in 4 x DH5 + 1 x DH3
    - **Average link**: 1500 bytes in 7 x DM5
- No provisions to use more than one user protocol on top of baseband
- Protocols know nothing about **master and slave** roles or Bluetooth **connection establishment**

Logical Link Control & Adaptation Protocol (L2CAP) has two main purposes:
- **Logical Link Control** (i.e. link layer protocol)
- **Adaptation**: provide larger packets, abstraction of master/slave principle, …

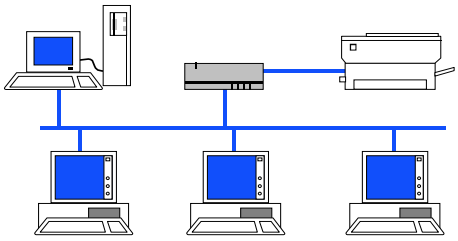
Reminder: OSI 7 Layer Model



application oriented: layers 5 to 7
transport oriented: layers 3 and 4
technology oriented: layers 1 and 2

Reminder: IEEE LMSC Layer Model

LAN/MAN Standards Committee der IEEE ("IEEE 802")

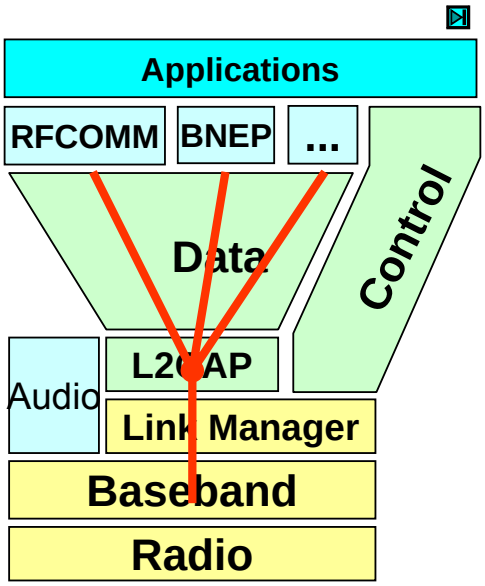


How to control the access to the media?

L2CAP (2)

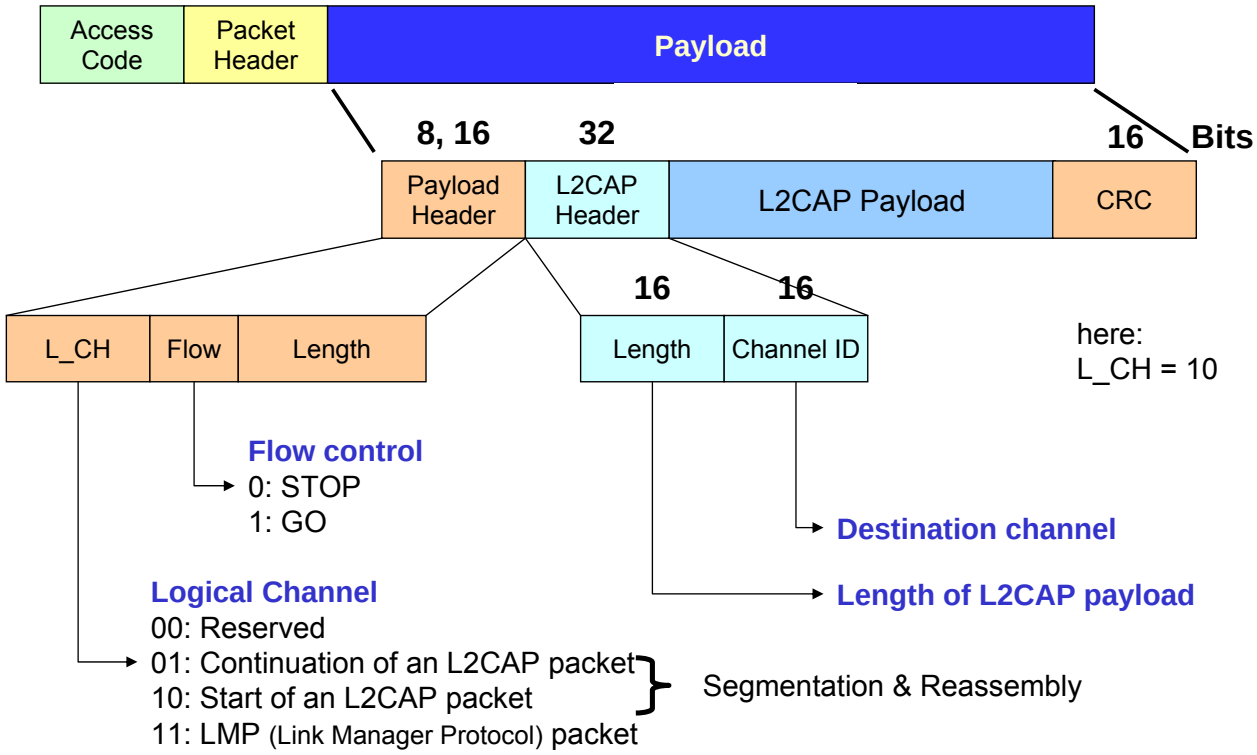
Baseband provides:
- Reliable channel between master and slave (fast ARQ)
- Piconet broadcasts
- Payload header

L2CAP provides:
- Protocol **multiplexing** for higher layers
  - Connection oriented
- **Segmentation and Reassembly** (SAR)
- **Group abstraction** (connectionless, unreliable)
- **Parameter negotiation**
  - Maximum transfer unit
  - „Flush Timeout" (discard payload after a specified time if not yet delivered)
  - Simple Quality of Service model



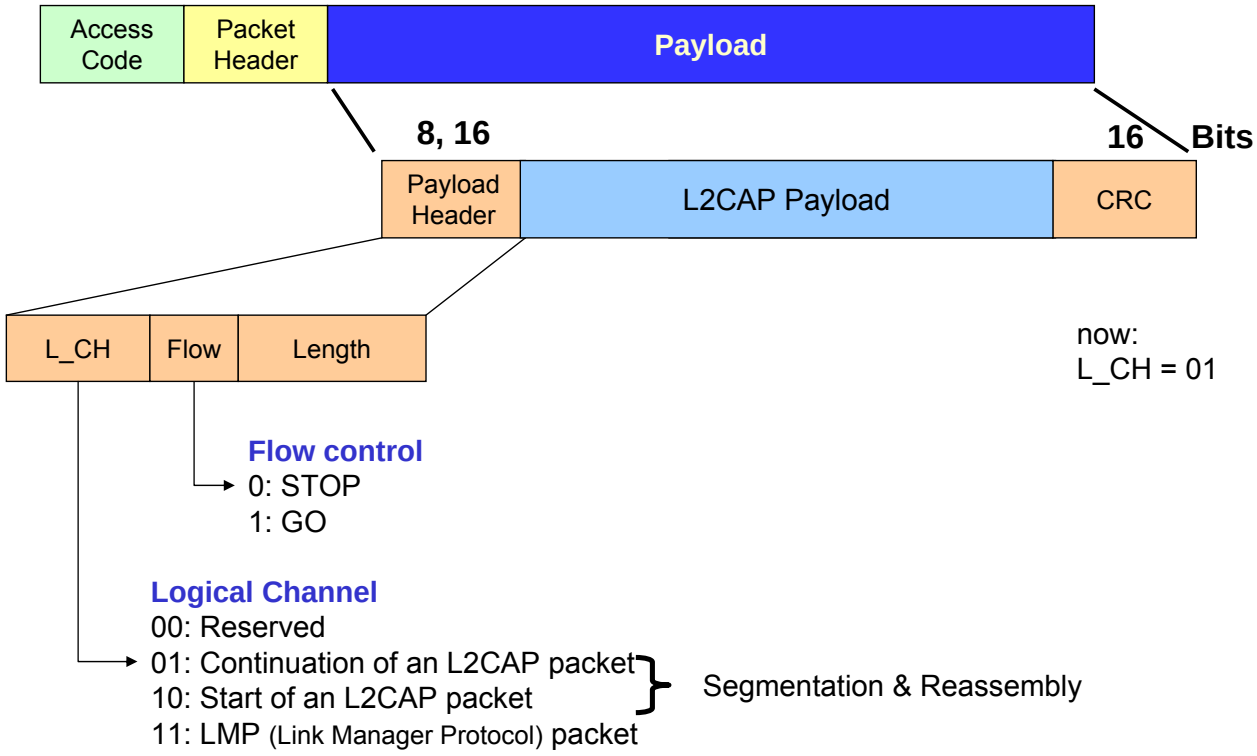
L2CAP was kept simple:
- No packet retransmission mechanisms and checksums
  → provided by baseband
- No SCO support

Baseband Packet Format of L2CAP Packets



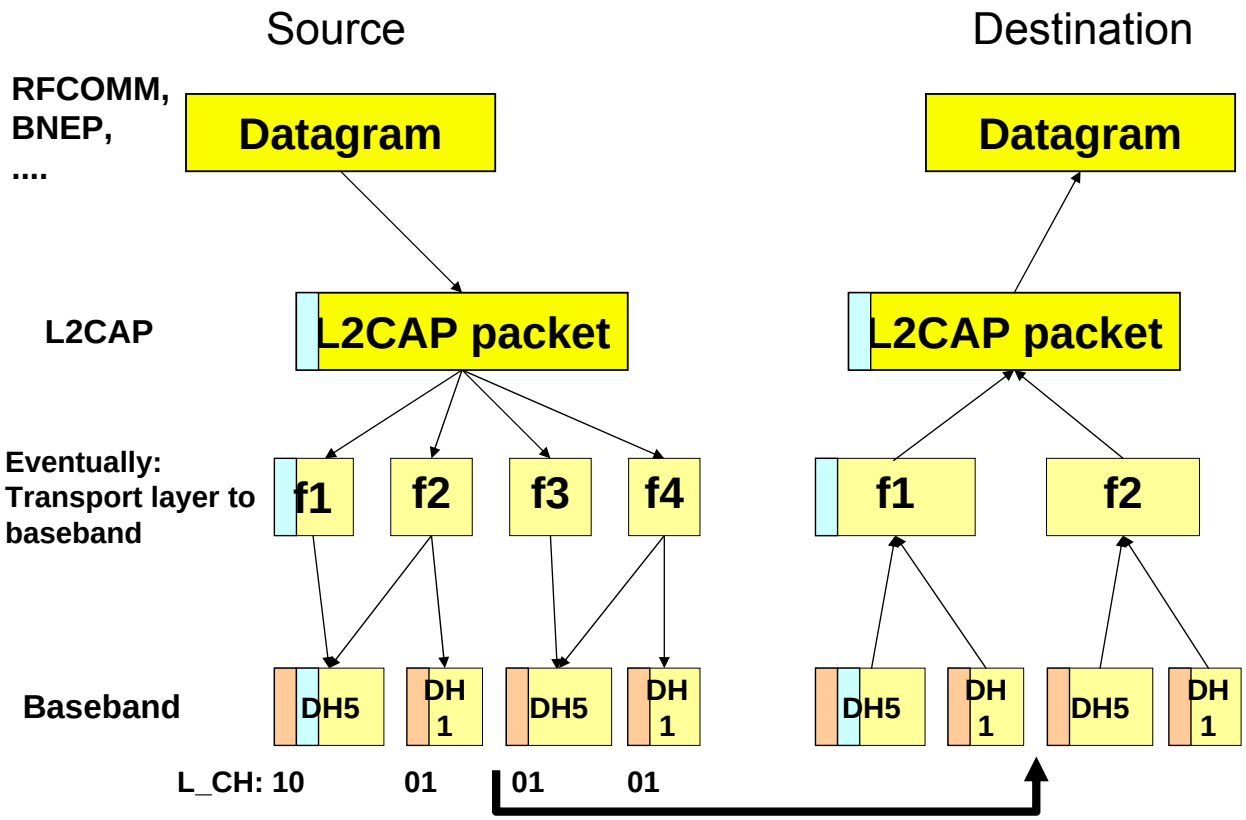
- **Channel IDs** assigned at L2CAP connection establishment
- Only the **first fragment** of an L2CAP packet contains an **L2CAP header** (L_CH=10)

Baseband Packet Format of L2CAP Packets



- **Channel IDs** assigned at L2CAP connection establishment
- Only the **first fragment** of an L2CAP packet contains an **L2CAP header** (L_CH=10)

Segmentation & Reassembly



4.2.5. Additional Bluetooth Protocols

Service Discovery Protocol (SDP):

- **Inquiry** categorizes devices into broad classes
- Detailed information about **supported services** and their **parameters** exchanged by **SDP**
- Service discovery was enhanced by **ESDP** profile, which uses **UPnP** (Universal Plug and Play)

RFCOMM:

- **Emulates** a **serial** connection (supports a large base of legacy applications)
- **Not** bound to **speed limitations** of a physical serial connection
- Reuses the **TS 07.10** specification from the **GSM** standard (protocol that allows to multiplex several "virtual" serial connections using a single physical serial connection)

Telephony Control Specification – Binary:

- Defines **signaling** for voice calls (e.g. call setup)
- Provides the base for the **telephony profiles**

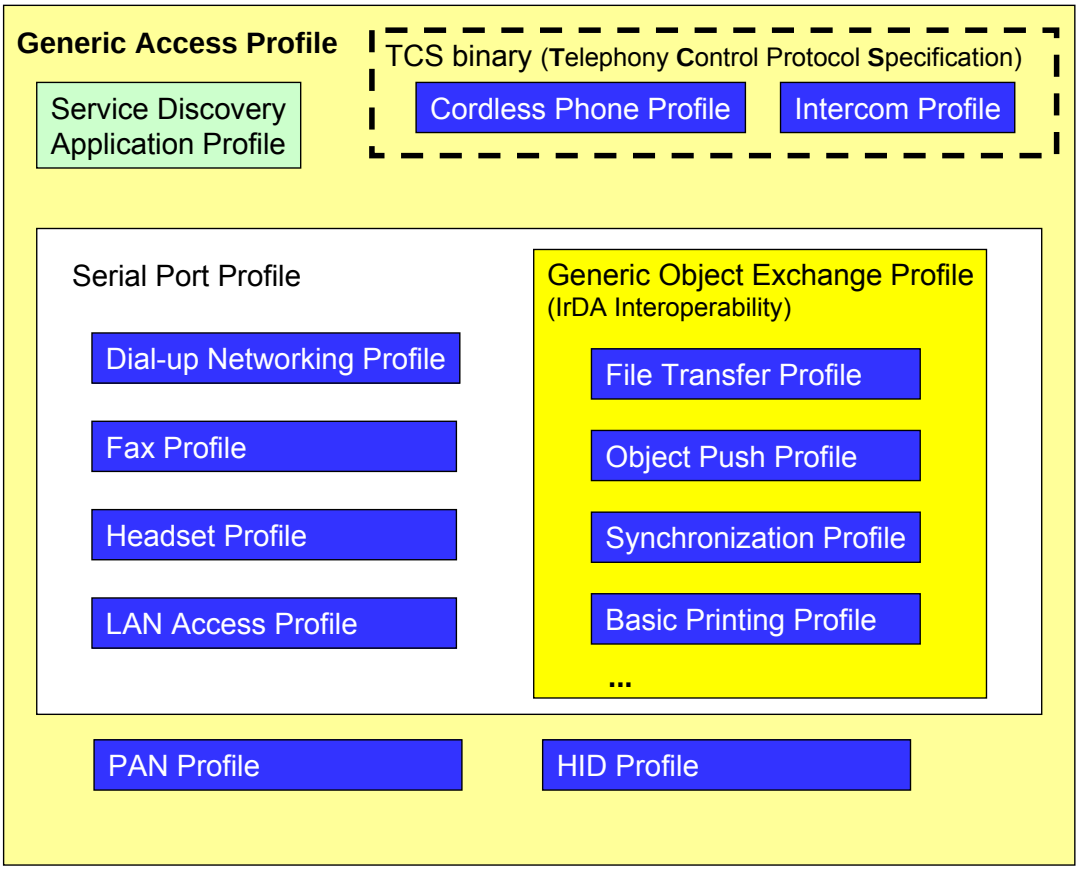
IrDA Interoperability:

- Adaptation of **IrDA** („Infrared Data Association") protocols to Bluetooth
- Allows most infrared applications to use Bluetooth (e.g. exchange of business cards)
- Protocols rely on **RFCOMM** or **TCP/IP**

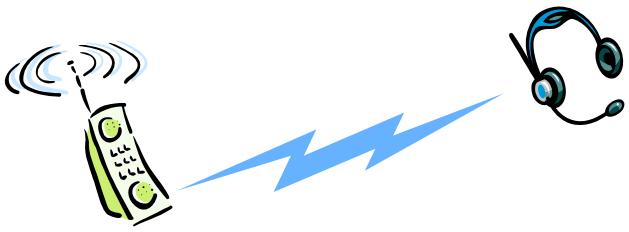
4.3. Bluetooth Profiles

Bluetooth Profiles are standardized **solutions** for specific **use cases**. They are the base for **interoperability** of Bluetooth applications and for the **certification** of Bluetooth solutions.

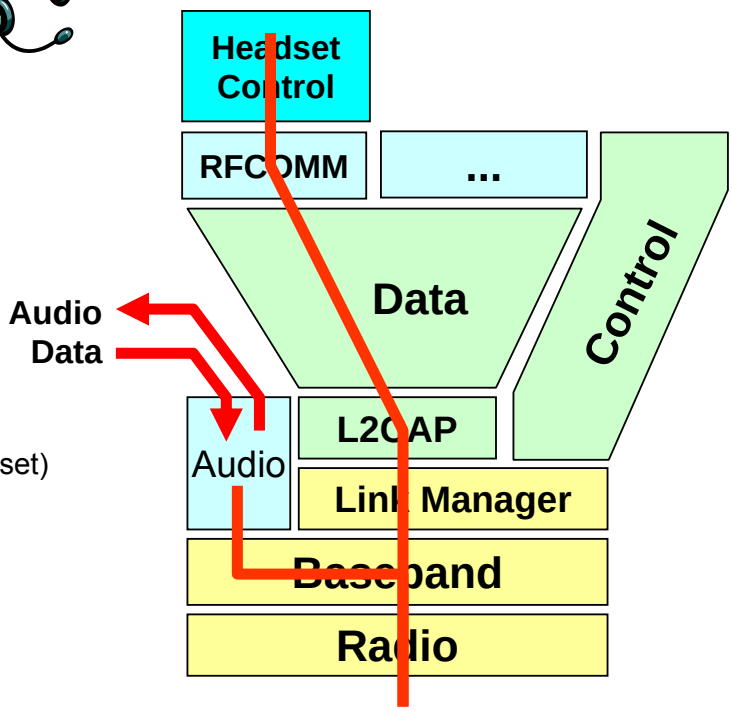
4.3.1. Overview of the Bluetooth Profiles



4.3.2. Headset Profile



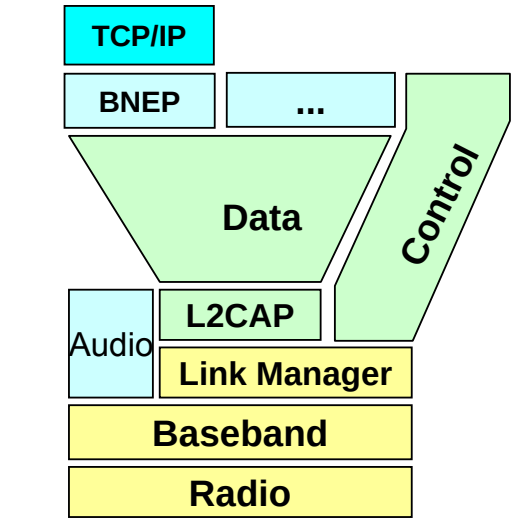
- **Full duplex audio (SCO)**
  - Incoming and outgoing calls
- **Control using RFCOMM**
  - AT commands (modem command set)
  - Call indication
  - Supports keys on headset
  - Volume control by the remote side

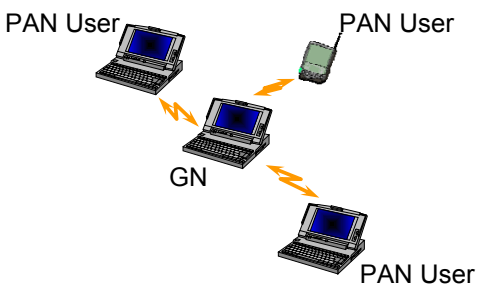


4.3.3. PAN Profile

The **PAN profile** supports:
- Using **access points**
- **Ad-hoc networking** in a Bluetooth piconet

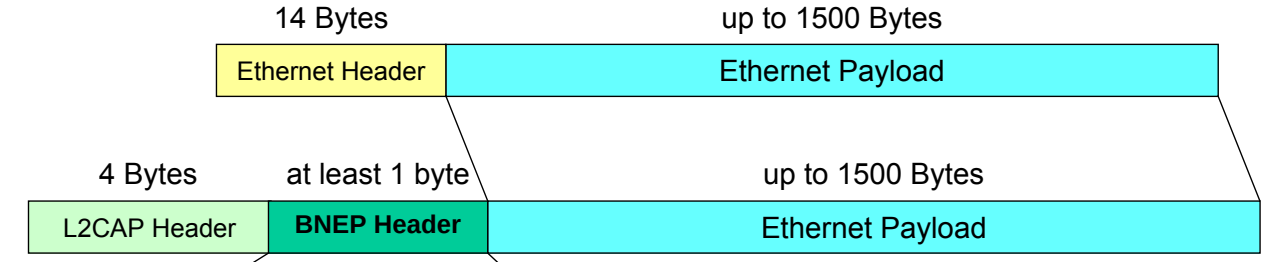
Implementation:
- BNEP transports **Ethernet payloads**
  - **Ethernet header** is compressed
  - Uses L2CAP connections
- Master of a piconet works as a **bridge**
  - Standalone: **Group Ad-Hoc Network (GN)**
  - Infrastructure: **Network Access Point (NAP)**
  - Master has to forward packets between slaves
- Support of **TCP/IP**:
  - Logically behaves like Ethernet: IP over Ethernet
  - Additionally: Support for **autoconfiguration** of IP addresses



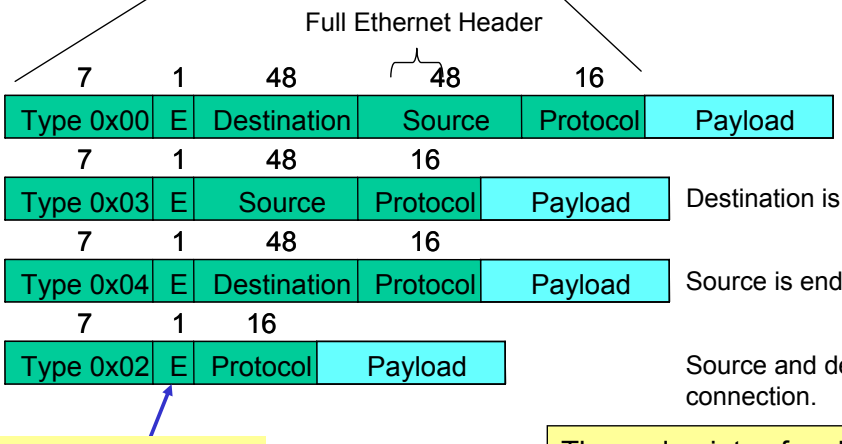


Bluetooth Network Encapsulation Protocol (BNEP)

An **Ethernet** packet is converted into a **BNEP** packet and encapsulated in an **L2CAP** packet:



Typical BNEP frames:



The end points of an L2CAP connection are known: **BNEP omits known addresses**

4.4. Bluetooth Security

Frequency Hopping already provides some protection against passive eavesdropping. An eavesdropper has to know the hopping sequence of the devices.

Bluetooth provides additional security features allowing **authentication** of devices and **encryption** of data.

4.4.1. Authentication and Encryption

Bluetooth authenticates **devices**, not users!

Example: Mobile phone and headset are able to authenticate each other. Both cannot recognize who is using the headset.

Bluetooth is able to encrypt **all** user data on ACL or SCO links as soon as the devices successfully authenticated each other. Each master/slave pair may establish a secure connection of its own.

Bluetooth defines two basic key types:

- **Link key**: Used to authenticate devices (using challenge-response authentication)
- **Encryption key**: Symmetric encryption key
  - Renewed each time encryption is enabled
  - Based on link key

Why are different keys needed?

- The encryption key may be restricted in length due to legislation
  - Such restrictions have no influence on the security of authentication
- Allows encryption key to have a shorter life time

4.4.2. Generating the Authenticating Link Key

Devices establish a baseband connection

Devices do not know each other

Link key generated from PIN (so-called **initialization key**).

PIN may be:
- manually entered on both devices
- manually entered on one device, fixed on the other (in case of simple devices like headsets)

Devices do know each other

Authenticate using the saved **combination key** as link key

After successful authentication using the initialization key:
Compute and save a common link key to be used for future authentication (so-called **combination key**)

In the ideal case, the PIN is only required once!

4.5. Bluetooth Version Overview

Bluetooth 1.0, 1.0B – December 1999

• first version(s) – many interoperability problems

Bluetooth 1.1 – February 2001

• many errors of 1.0/1.0B fixed
• RSSI - Received Signal Strength Indicator

Bluetooth 1.2 – November 2003

• Adaptive Frequency-hopping spread spectrum (AFH)

IEEE WPAN Group 802.15
IEEE Standard
802.15.1 – June 2002
(based on Bluetooth version 1.1)

Bluetooth 1.0 – 1.1 – 1.2
working with **nominal**
data rate of 723 kbit/s

Bluetooth 2.0 – November 2004

• Enhanced Data Rate (EDR) of **up to 3.0 Mbps (nominal)**
• Lower power consumption, improved BER (bit error rate) performance

Bluetooth 2.1 + EDR – August 2007 (aka Lisbon Release)

• new features: secure simple pairing, Quality of Service

Future of Bluetooth:
- **codename "Seattle":** adopting Ultrawideband UWB radio tech.
- High Speed Bluteooth, Ultra Low Power Bluetooth

Annex: Pointer to SS03: 4.4. Bluetooth

The subsection 4.4. on Bluetooth as used in the lecture in Summer 2003 contained an additional subsection 4.4.5.:

Interested students are referred to our WWW archive of lecture slides.

The following 4 slides roughly present a summary of the interference study.

4.4.5. A Study of the Interference Behavior of Bluetooth

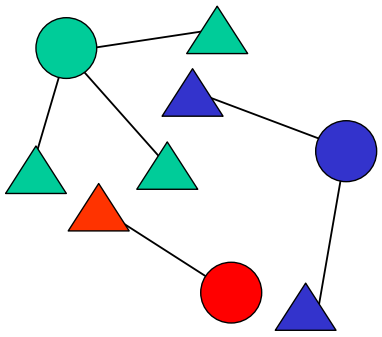
When building **scatternets** out of Bluetooth piconets or when using a large number of piconets in **close proximity**, the effect of **interference** between independent Bluetooth piconets becomes noticeable.

In the following, a **worst-case** model for the interference effects that honors the properties of the **Bluetooth MAC** is introduced.

4.4.5.1. Scenarios and Models

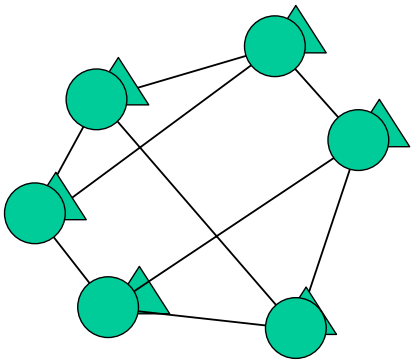
Two scenarios of devices in close proximity are analyzed:

Independent Piconets:



- Master has **central role** (e.g. Access Point)
- Traffic: Mostly **downlink** (master to slave)

Scatternet:



- All purpose ad-hoc network
- Traffic: Mostly **bidirectional**

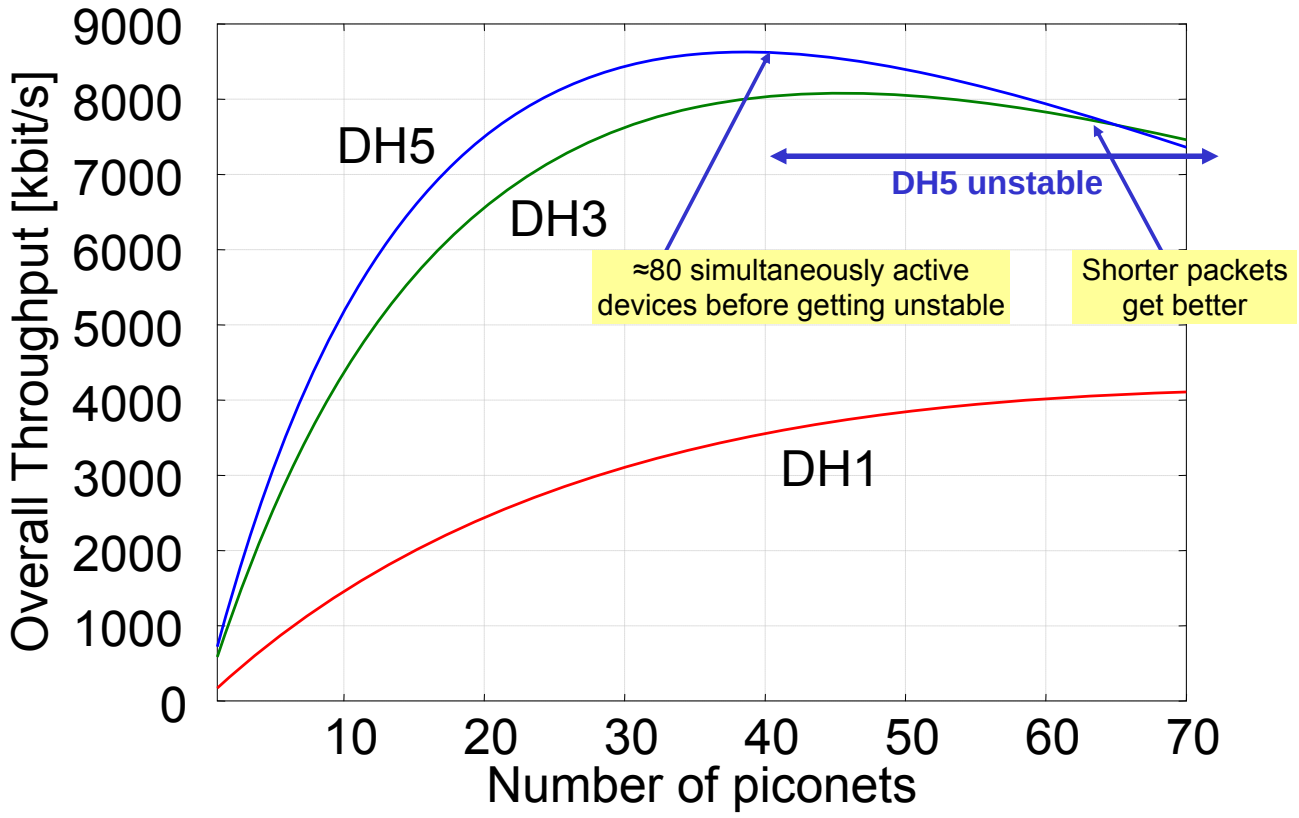
In both cases: Independent, unsynchronized Piconets!

What is the expected throughput in these scenarios?

Outcome: Average rate of payload deliveries per baseband frame r_{suc}

expected throughput = maximal throughput • r_{suc}

Overall Throughput: Downlink Traffic (Analytical Model)



Overall Throughput: Bidirectional Traffic (Simulation)