# Mobile communications

**Bluetooth**

(WPAN)

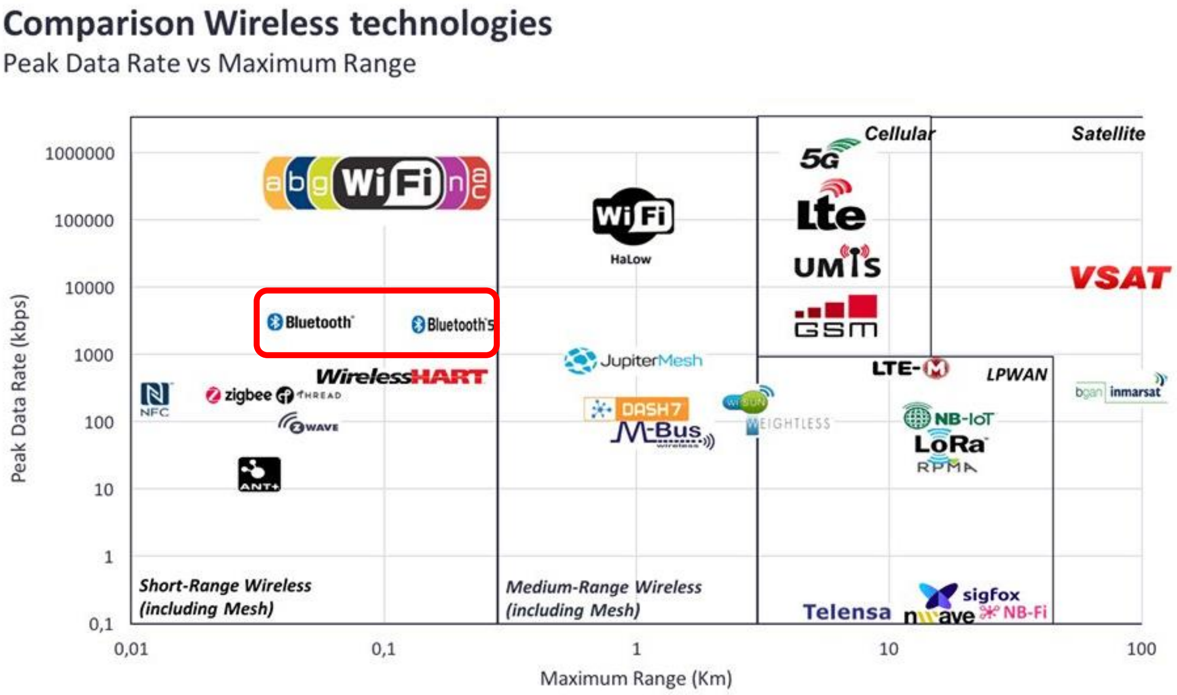
# Outline

- Bluetooth networks
- Piconet operation
  - Inquiry
  - Paging
- Bluetooth stack
- Profiles and security
- BT 4.0 BLE

# Outline

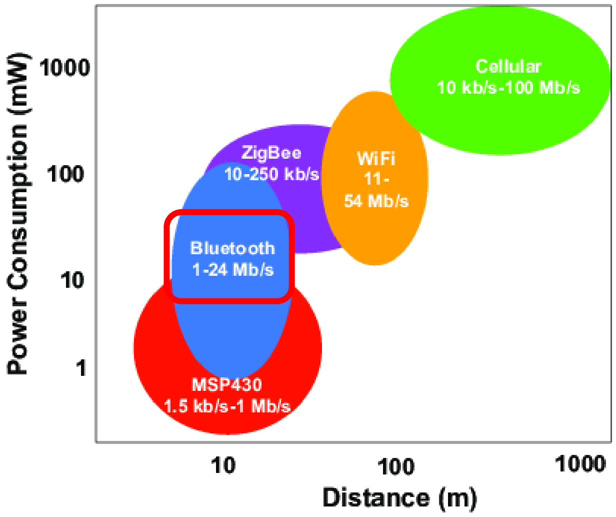
- Bluetooth networks
- Piconet operation
  - Inquiry
  - Paging
- Bluetooth stack
- Profiles and security
- BT 4.0 BLE

# Comparison Between Wireless Technologies



Comparison Wireless technologies

Peak Data Rate vs Maximum Range



*Ahmed, Mobyen & Björkman, Mats & Causevic, Aida & Fotouhi, Hossein & Lindén, Maria. (2015). An Overview on the Internet of Things for Health Monitoring Systems.*

Tradeoff between data rate, range and energy

# Personal Area Networks

- Target deployment environment: communication of personal devices working together
  - Short-range
  - Low Power
  - Low Cost
  - Small numbers of devices

- PAN Standards
  - Bluetooth – Industry consortia (**Bluetooth SIG**)
  - IEEE 802.15.1 – ”Bluetooth” based
  - IEEE 802.15.2 – Interoperability and coexistence
  - IEEE 802.15.3 – High data rate WPAN (UWB)
  - IEEE 802.15.4 – Low data rate WPAN (Zigbee,...)
  - IEEE 802.15.5 – Mesh Networks
  - IEEE 802.15.6 – Body Area Network
  - IEEE 802.15.7 – Visible Light Communication

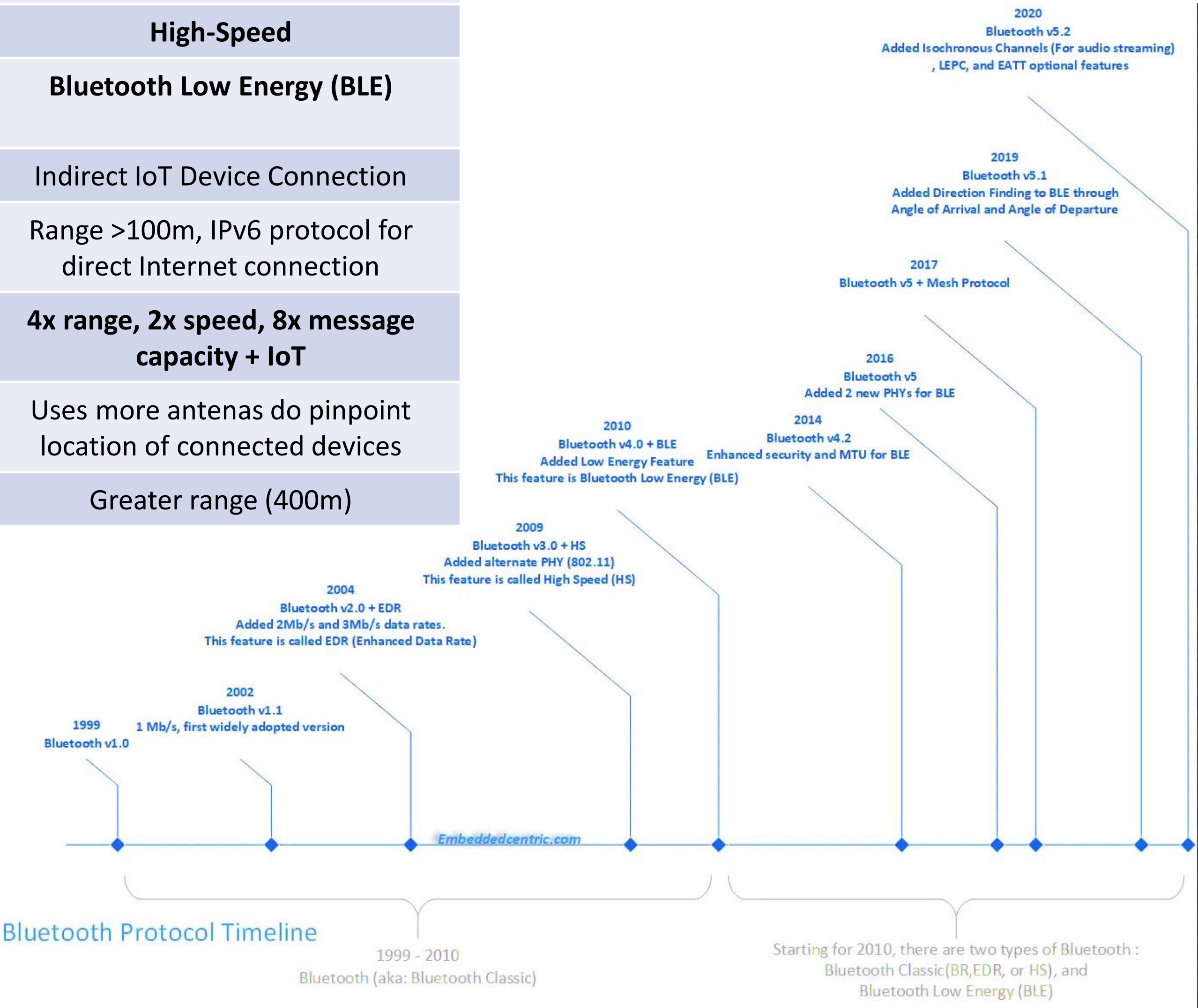
# Bluetooth

- Created by Ericsson (1994)
- Maintained by the Bluetooth SIG (https://www.bluetooth.com/)
- Originally for replacing “USB”, not “Ethernet”
  - Cable replacement technology
  - Later also used as Internet connection, phone or headset
- PAN - *Personal Area Network*
  - Started with 1 Mbps connections
  - Includes synchronous, asynchronous, voice connections
  - Piconet routing
- Small, low-power, short-range, cheap, versatile radios (3 classes)
- Master/slave configuration and scheduling

# Bluetooth Versions

| Version | Data rate | Feature |
|---|---|---|
| 1.1 | 1 Mbps | First widely adopted version |
| 2.0 + EDR | **3 Mbps** | **Enhanced Data Rate (EDR)** |
| 3.0 + HS | **24 Mbps** | **High-Speed** |
| 4.0 | 24 Mbps/ **1 Mbps** (BLE) | **Bluetooth Low Energy (BLE)** |
| 4.1 | 25 Mbps | Indirect IoT Device Connection |
| 4.2 | 25 Mbps | Range >100m, IPv6 protocol for direct Internet connection |
| 5.0 | **50 Mbps** | **4x range, 2x speed, 8x message capacity + IoT** |
| 5.1 | 50 Mbps | Uses more antenas do pinpoint location of connected devices |
| 5.2 | 50 Mbps | Greater range (400m) |

Now in 5.4, with some additional improvements


2020
Bluetooth v5.2
Added Isochronous Channels (For audio streaming)
, LEPC, and EATT optional features
2019
Bluetooth v5.1
Added Direction Finding to BLE through
Angle of Arrival and Angle of Departure
2017
Bluetooth v5 + Mesh Protocol
2016
Bluetooth v5
Added 2 new PHYs for BLE
2014
Bluetooth v4.2
Enhanced security and MTU for BLE
2010
Bluetooth v4.0 + BLE
Added Low Energy Feature
This feature is Bluetooth Low Energy (BLE)
2009
Bluetooth v3.0 + HS
Added alternate PHY (802.11)
This feature is called High Speed (HS)
2004
Bluetooth v2.0 + EDR
Added 2Mb/s and 3Mb/s data rates.
This feature is called EDR (Enhanced Data Rate)
2002
Bluetooth v1.1
1 Mb/s, first widely adopted version
1999
Bluetooth v1.0
Embeddedcentric.com
1999 - 2010
Bluetooth (aka: Bluetooth Classic)
Starting for 2010, there are two types of Bluetooth :
Bluetooth Classic(BR,EDR, or HS), and
Bluetooth Low Energy (BLE)

Bluetooth Protocol Timeline

# WLAN vs. Bluetooth

| | Bluetooth | WLAN / WiFi |
|---|---|---|
| Specifications authority | Bluetooth SIG | IEEE, WiFi Alliance |
| Year of development | 1994 | 1991 |
| Bandwidth | Low (50 Mbps) | Very High (2 Gbps 802.11ax) |
| Hardware requirement | Bluetooth adaptor on all the devices connecting with each other | Wireless adaptors on all the devices of the network, a wireless router and/or wireless access points |
| Cost | Low | High |
| Power Consumption | Low | High |
| Frequency | 2.4 GHz | 2.4/5 GHz |
| Security | It is less secure | It is more secure |
| Range | 10 meters | 100 meters |
| Primary Devices | Mobile phones, mouse, keyboards, office and industrial automation devices | Notebook computers, desktop computers, servers |
| Ease of Use | Fairly simple to use. Can be used to connect up to 7 devices at a time. It is easy to switch between devices or find and connect to any device. | It is more complex and requires configuration of hardware and software |

# Bluetooth features (I)

- Radio network, on the **2.4 GHz**, world-wide
    - **ISM (**Industrial, Scientific and Medical**)**; Unlicensed but regulated

- FH (**Frequency Hopping**) **Spread Spectrum**:
    - **79** channels of 1 Mhz in the 2.402 GHz to 2.480 GHz range

- Defines a **Master**
    - Synchronizes everyone to his hop-pattern

- **TDD** (Time Division Duplex)
    - Data is transmitted in one direction at a time with transmission alternating between two directions (**Master transmits in even** timeslots and **receives in odd ones**)

# Bluetooth features (II)

- Defines two types of networks:
  - **Piconets** (has 1 Master)
  - **Scatternets** (joining multiple piconets via common Master or Slaves)

- Maximum **8 active devices** per piconet
  - 1 Master + 7 Slaves

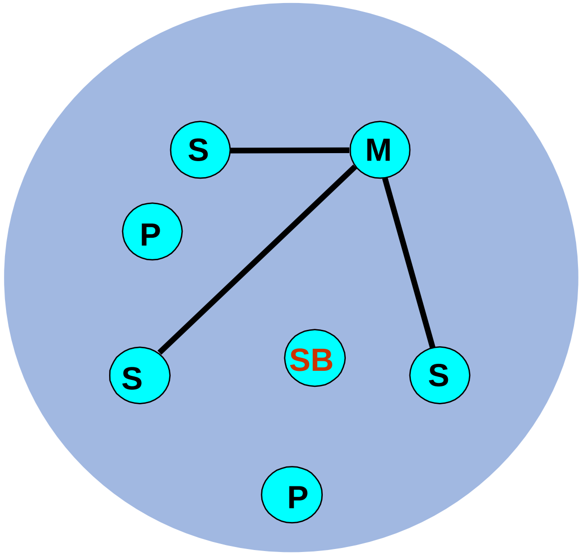
- Two main types of connections
  - **SCO** (Synchronous Connection Oriented), voice link
    - FEC (forward error correction), no retransmission
    - Connection explicitly set up prior to transmitting
  - **ACL** (Asynchronous Connection Less), data link
    - Asynchronous, packets must be acknowledged

# Frequency Hopping Spread Spectrum (FHSS)

- Signal broadcast over pseudo random series of frequencies
- Receiver hops between frequencies in sync with transmitter (1600 hops per second, every 625uS)
- Spreading code determines the hopping sequence
  - Must be shared by sender and receiver (e.g. standardized)
- Eavesdroppers hear unintelligible blips
- Jamming on one frequency affects only a few bits

# Piconets (I)

- Bluetooth devices connected in an “ad-hoc” cell
- There is a Master with up to 7 active Slaves and several hundreds parked
  - Slaves only communicate with master
  - Slaves must wait for permission from master
  - Communication can be 1-to-1 to 1-to-many
  - No direct communication between slaves
- Each station (Master or Slave), has a 48-bit fixed device address

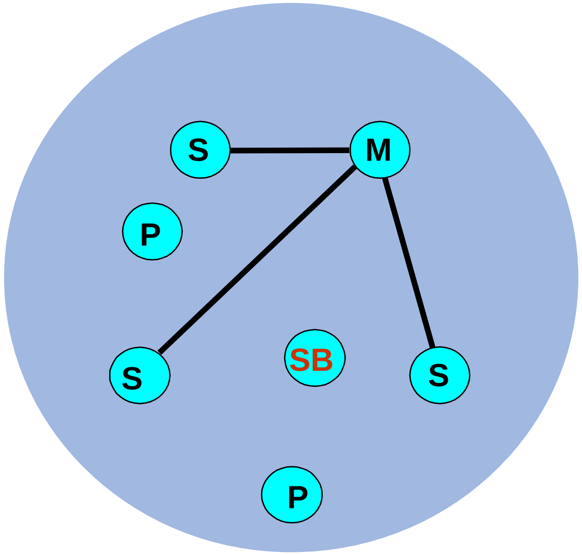


**M = Master**
**S = Slave**

**P = Parked**
**SB = Standby**

# Piconets (II)

- Master defines radio parameters (”clock” and “deviceID”)
  - Channel, hopping sequence, timing, …
- Each Piconet has a unique FH pattern (and a single ID)
- Each piconet has a maximum bandwidth
- A node in one **Piconet** can also be part of another Piconet, either as a Master or as a Slave, creating a **Scatternet**



M = Master
S = Slave

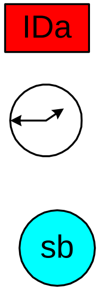
P = Parked
SB = Standby

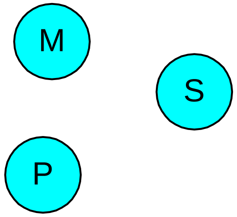
# Outline

- Bluetooth networks
- Piconet operation
  - Inquiry
  - Paging
- Bluetooth stack
- Profiles and security
- BT 4.0 BLE

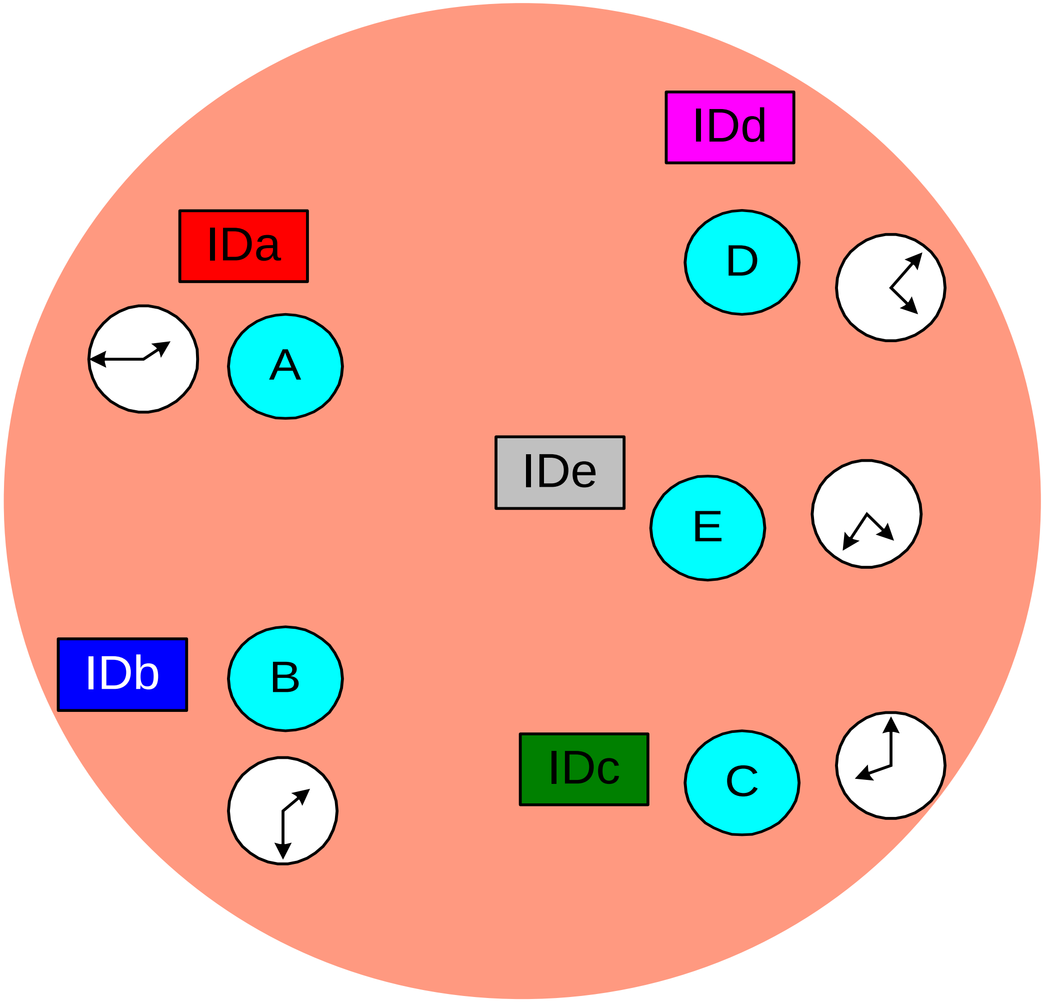
# Piconet operation

- FHSS: all devices must share the same hopping pattern:
  - *Master* provides clock and deviceID such that:
    - The unique deviceID (48-bits) defines hopping pattern
    - Clock defines phase inside the pattern
- If a device is inside a piconet, and is not connected, it must be in *standby*
- There are two types of piconet addresses
  - *Active Member Address* (AMA, 3-bits, 7 addresses)
  - *Parked Member Address* (PMA, 8-bits, 255 addresses)

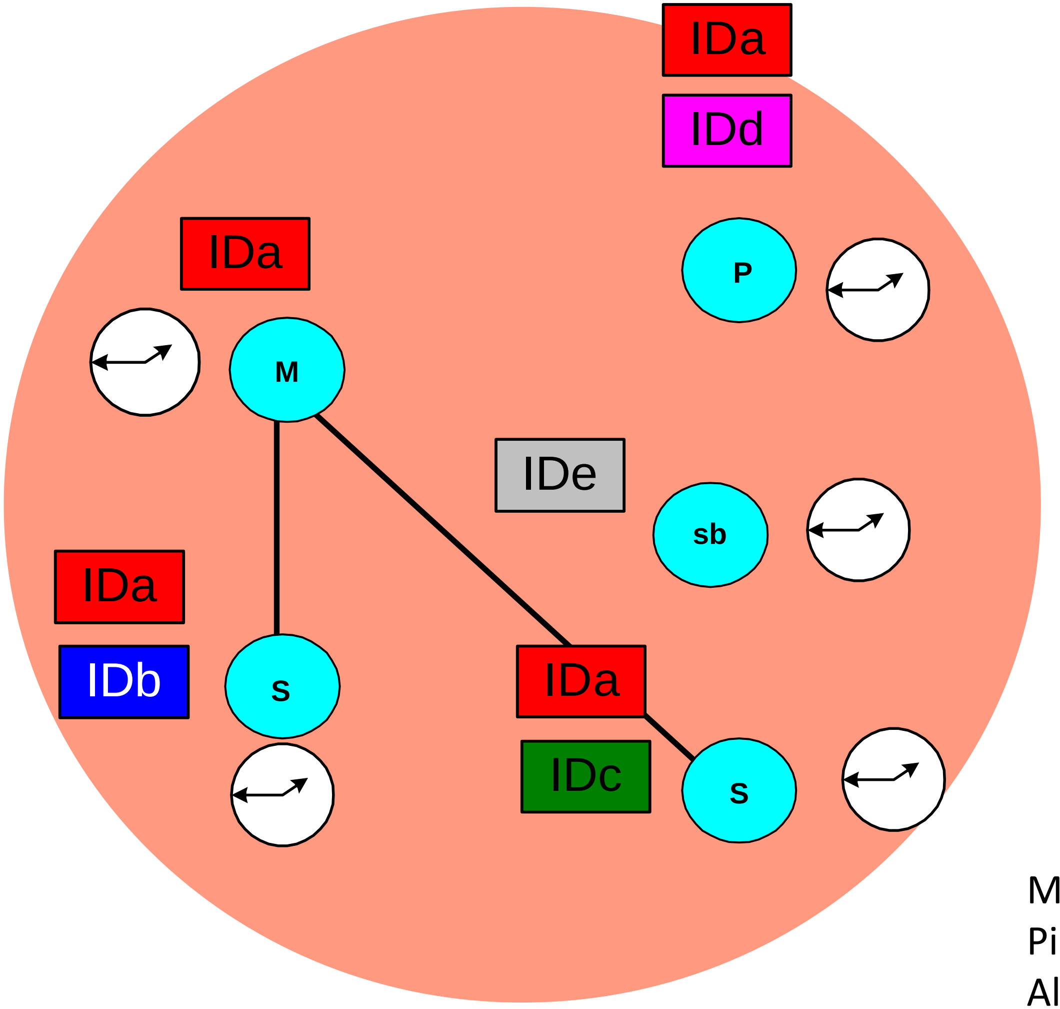




# Piconet before setup



# Piconet in operation

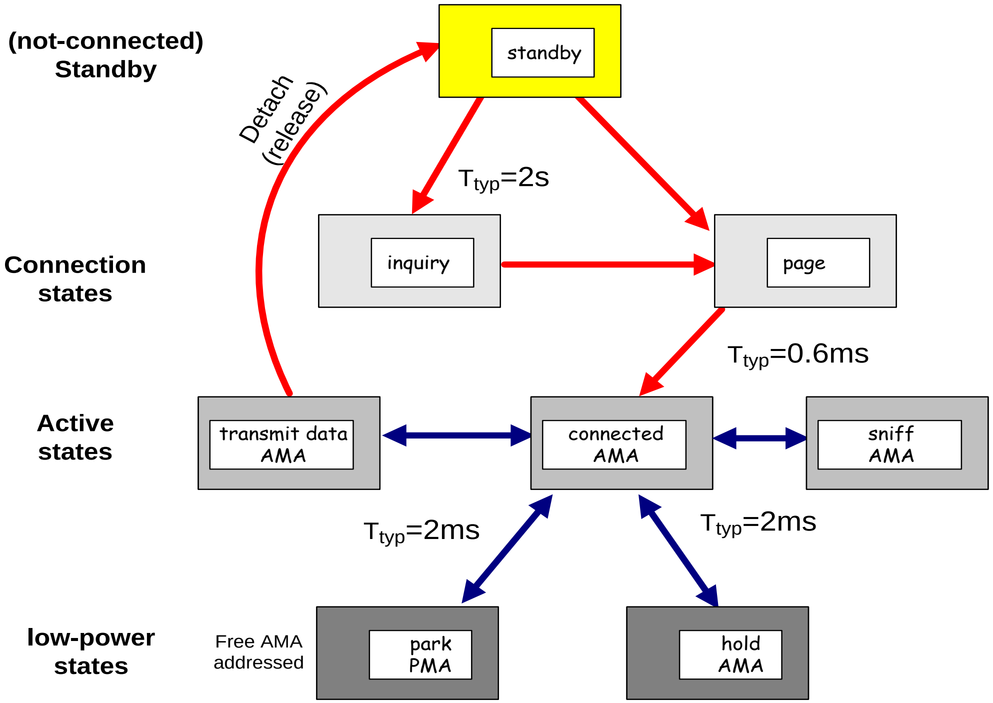


**Piconet built!**

Master know all slaves
Piconet ID shared
All in sync

# Device states

- Standby
  - Do nothing; waiting to join a piconet
- Inquire
  - Search for other devices (discover nodes)
- Page
  - Connect to a specific device
- Connected
  - Active on a piconet (Master or Slave)
- Park/Sniff/Hold
  - Low Power connected states



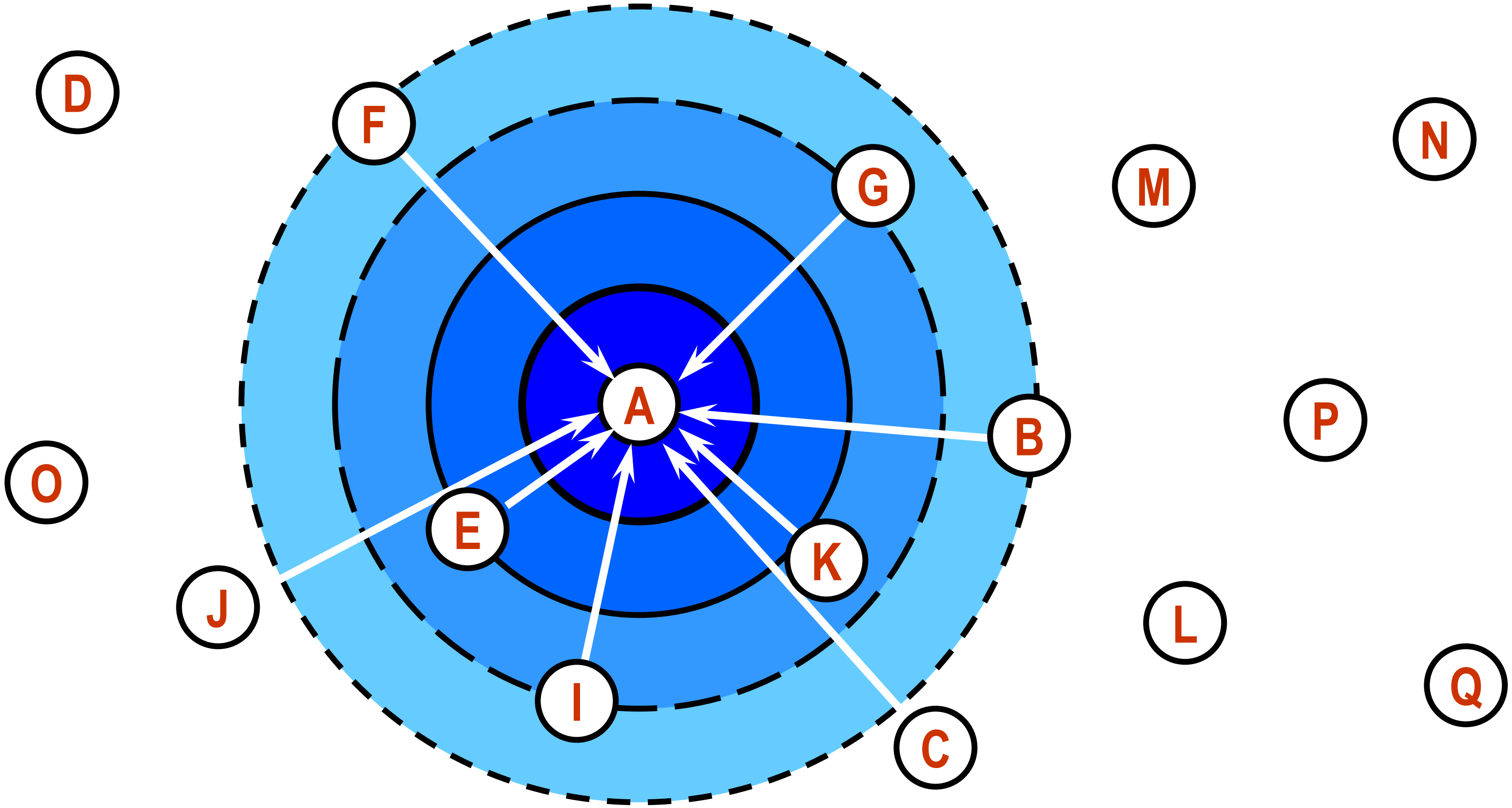
**Park**: release AMA, get PMA
**Sniff**: listen periodically, not each slot
**Hold**: stop ACL, SCO still possible, possibly participate in another piconet

**AMA**: Active Member Address
**PMA**: Park Member Address

# Low-Power Operation in BT classic

- 3 modes (Slaves):
  1. **Sniff**
     - Low-duty cycle mode
     - Wakes up periodically to talk to master
     - Fixed “sniff” intervals
  2. **Park**:
     - Very low power state
     - Used to admit more than 7 slaves in piconet
       - Slave gives up its Active Member Address (AMA)
       - Receives “Parked” Member Address (PMA)
     - Wakes up periodically listening for broadcasts which can be used to “unpark” node
  3. **Hold**
     - Node sleeps for specified interval
     - Master can put slaves in hold while searching for new members, attending another piconet, etc.
     - No ACL packets (*Asynchronous Connection-Less*) → general data packets
       - But SCO (*Synchronous Connection Oriented*) possible → Audio

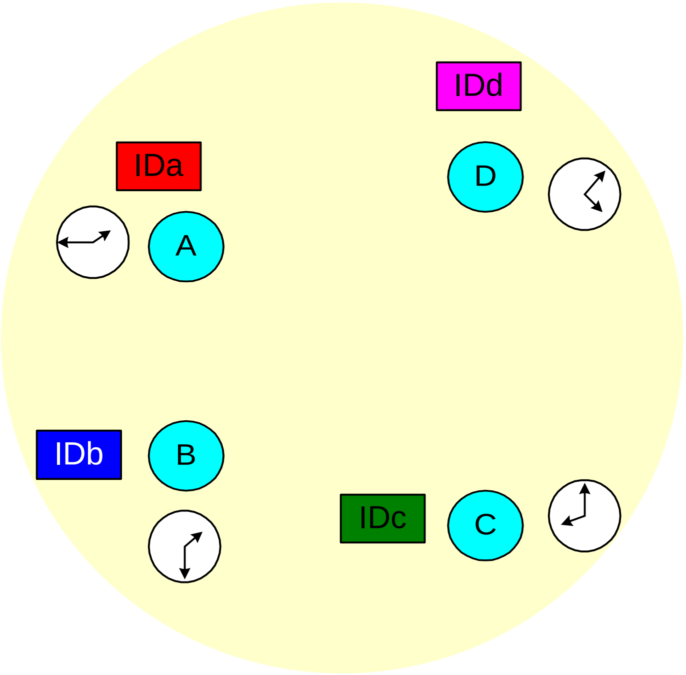
# Device Discovery Illustrated



**10 meters**

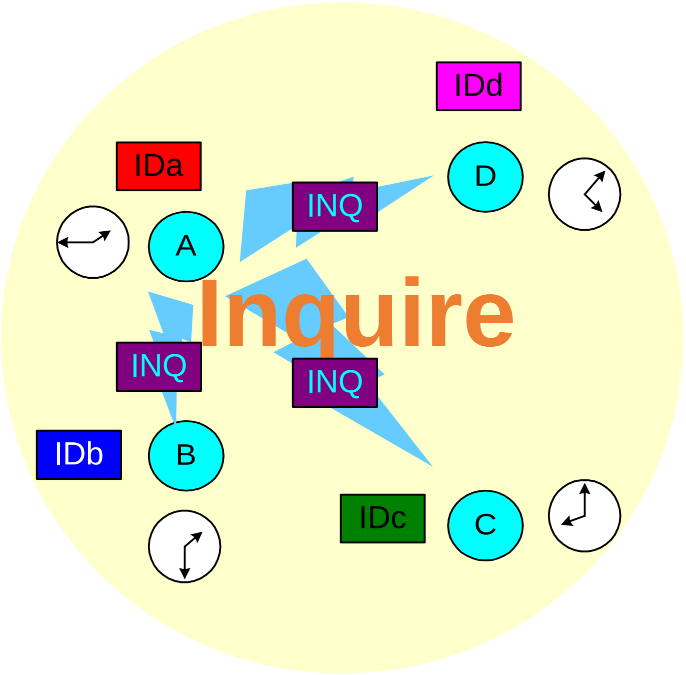
**After inquiry procedure, A knows about others within range**

# Scanning units



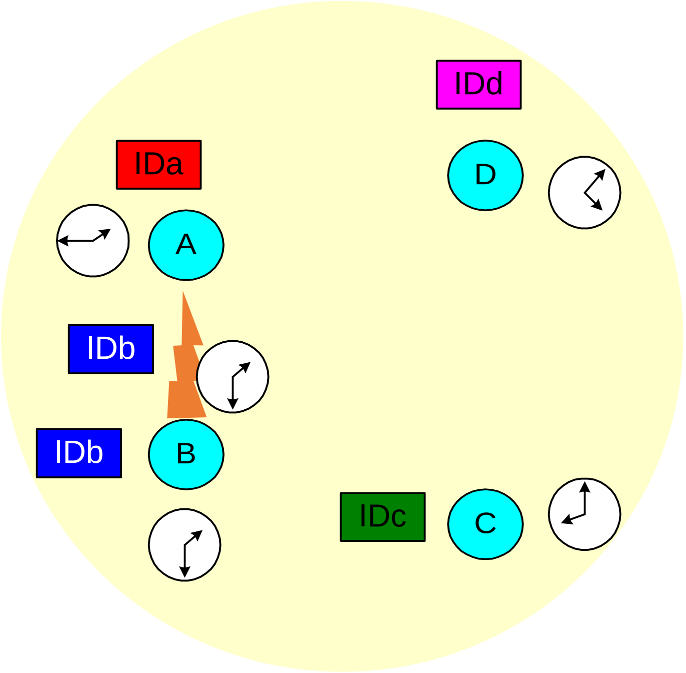
- Device A wants to search for stations

# Scanning units



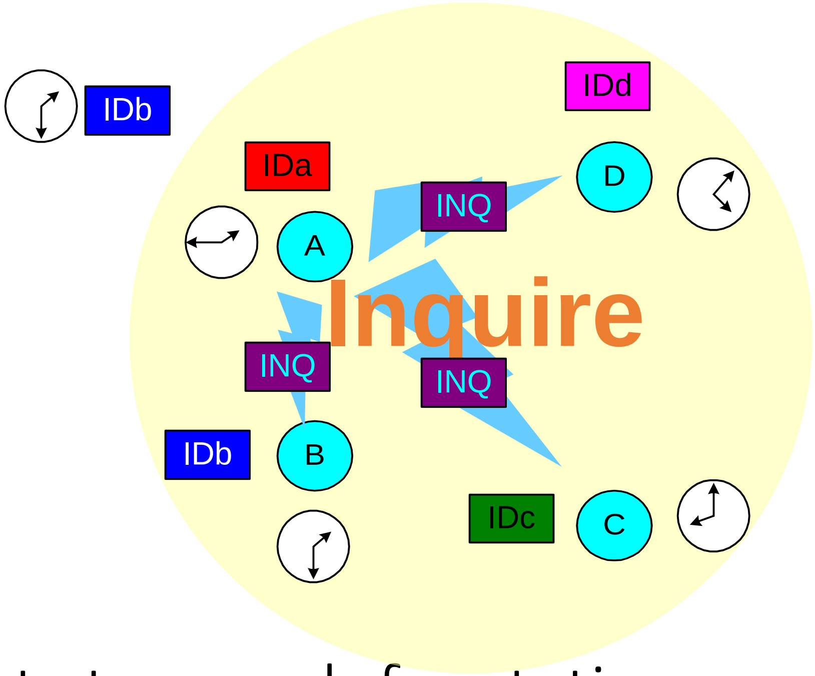
- Device A wants to search for stations
- A does an inquire (page with ID 000)
  - Devices B,C,D are doing an inquire scan

# Scanning units



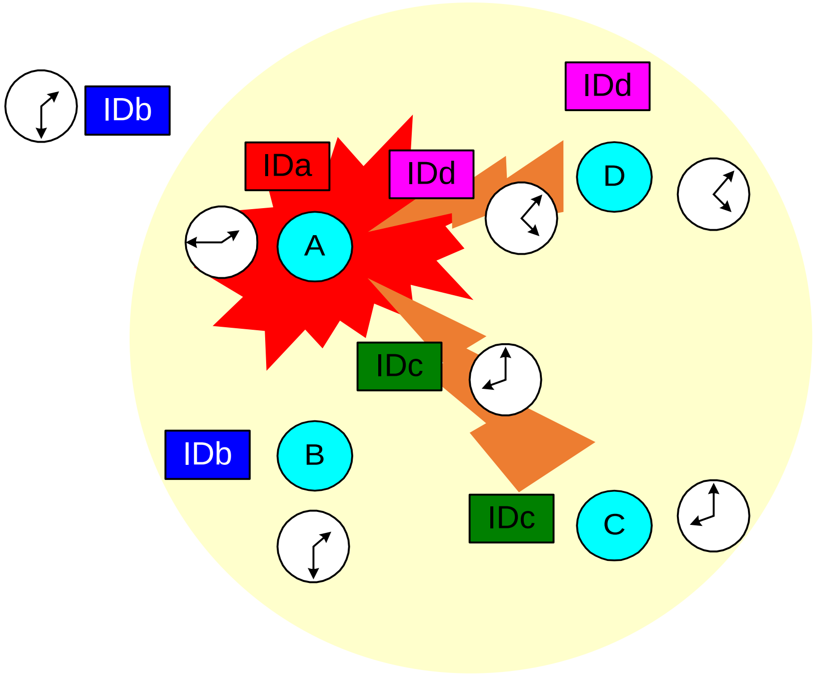
- Device A wants to search for stations
- A does an inquire (page with ID 000)
- B answers with FHS packet
  - Contains *DeviceID* and *Clock*

# Scanning units



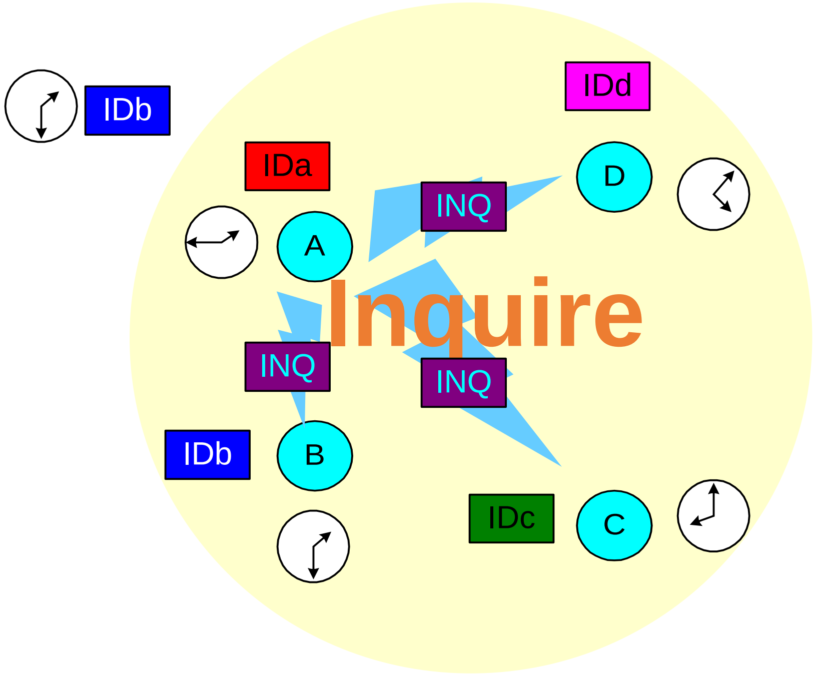
- Device A wants to search for stations
- A does an inquire (page with ID 000)
- B answers with FHS packet
  - Contains DeviceID and Clock
- A does an inquire again

# Scanning units



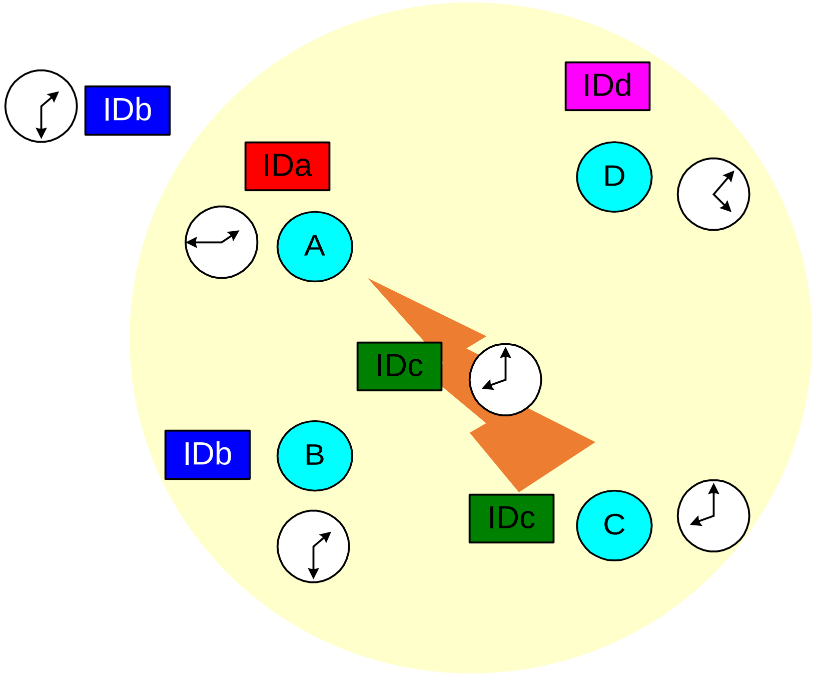
- A wants to search for stations
- …
- A does an inquire again
- C e D answer at the same time with FHS packet
  - Packets are corrupted
  - A does not answer
  - C and D will wait a random number of slots

# Scanning units



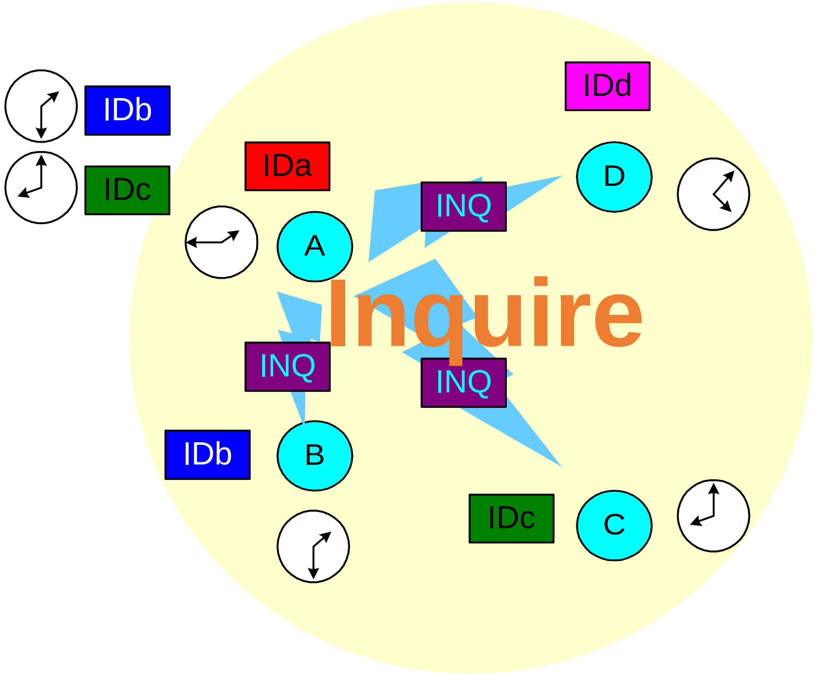
- A wants to search for stations
- …
- A does an inquire again

# Scanning units



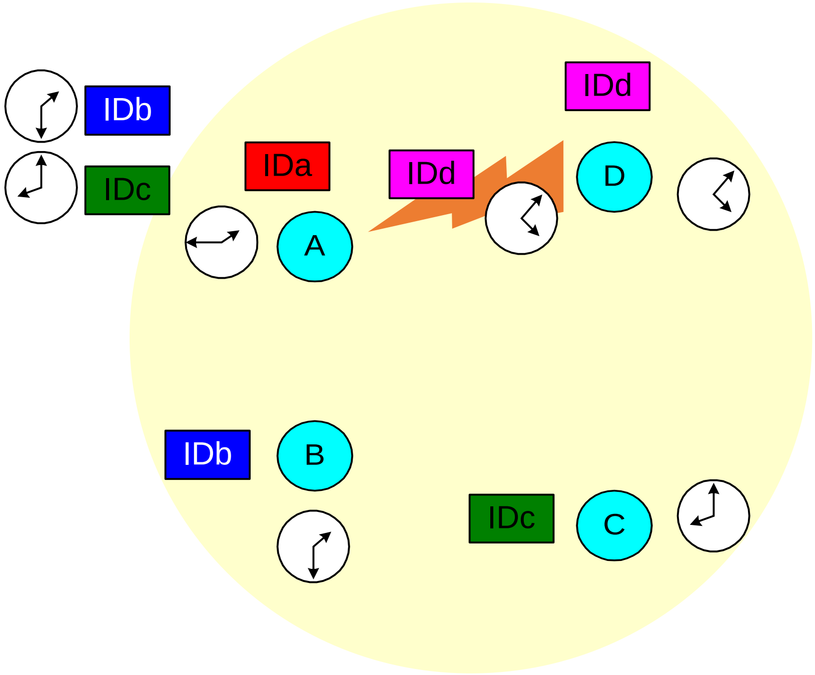
- A wants to search for stations
- …
- A does an inquire again
- C answers with FHS packet

# Scanning units



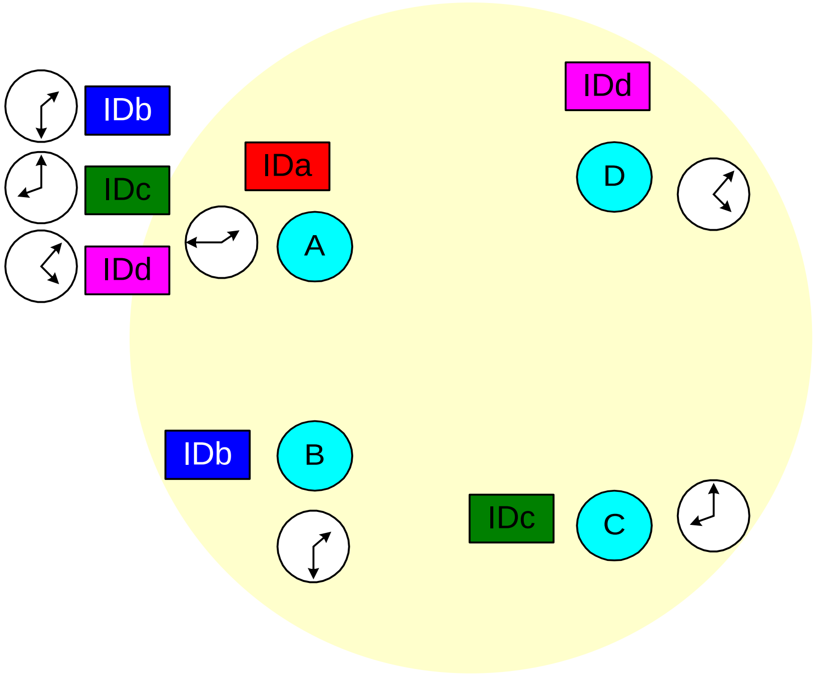
- A wants to search for stations
- A does an inquire again

# Scanning units



- A wants to search for stations
- …
- A does an inquire again
- D answers with FHS packet

# Scanning units



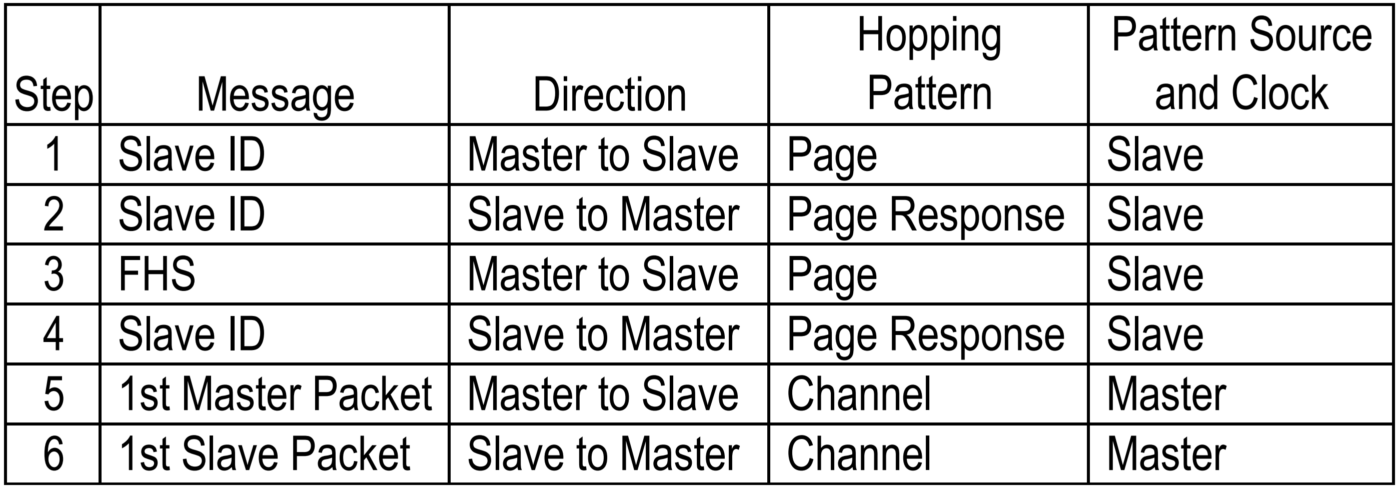
- A has all the information it needs about the units in the cell

# Inquiry scanning: summary

- Inquiry scanning has a common address
  - And a common frequency pattern (from 32 frequencies)
- All devices can page this address (and become masters)
- All machines hearing an inquiry will answer the inquiry request
- There is a detector (correlator hit) in the slaves, that detects inquiries, before answering with a FHS providing:
  - Device ID and Clock
- A machine in low power waits a random time before answering again to a scan
- If there is a collision on answering to a scan, they also wait a random period before answering again

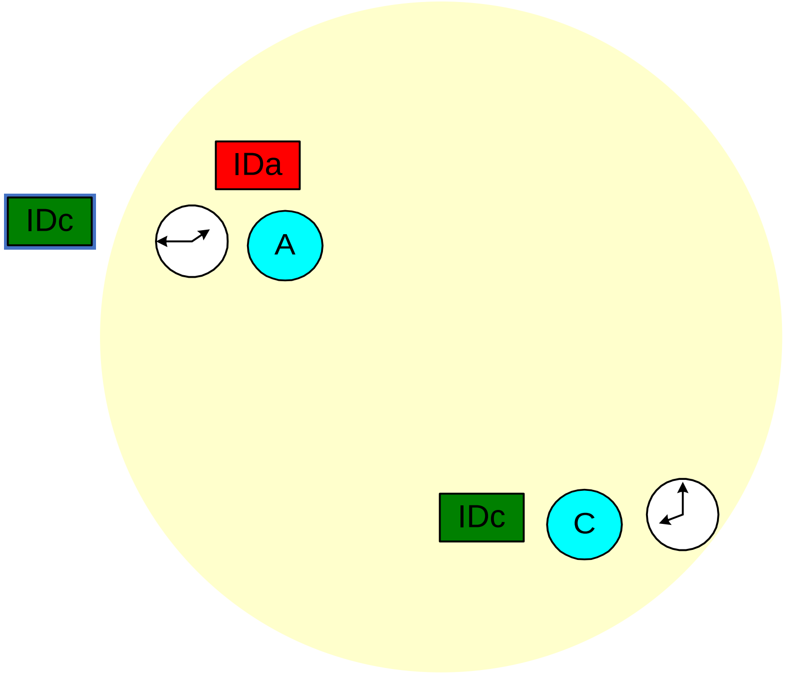
# Paging: Will you connect to me?

- Very similar to inquire
- Still have not synchronized clocks or frequencies
- Establishes actual Piconet connection with a device that it knows about
- Connection process involves a 6 steps of communication between the master and the slave



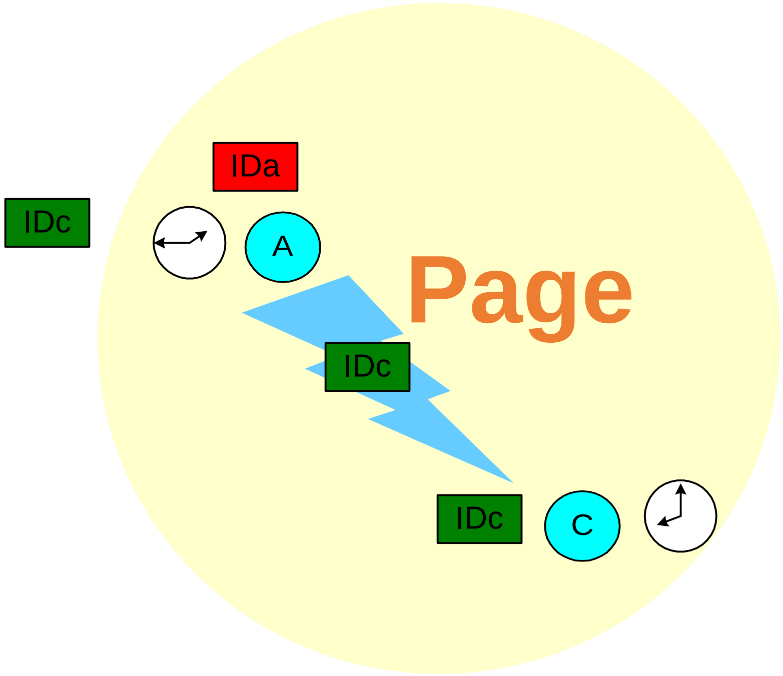
| Step | Message | Direction | Hopping Pattern | Pattern Source and Clock |
|---|---|---|---|---|
| 1 | Slave ID | Master to Slave | Page | Slave |
| 2 | Slave ID | Slave to Master | Page Response | Slave |
| 3 | FHS | Master to Slave | Page | Slave |
| 4 | Slave ID | Slave to Master | Page Response | Slave |
| 5 | 1st Master Packet | Master to Slave | Channel | Master |
| 6 | 1st Slave Packet | Slave to Master | Channel | Master |

# Master Paging Slave



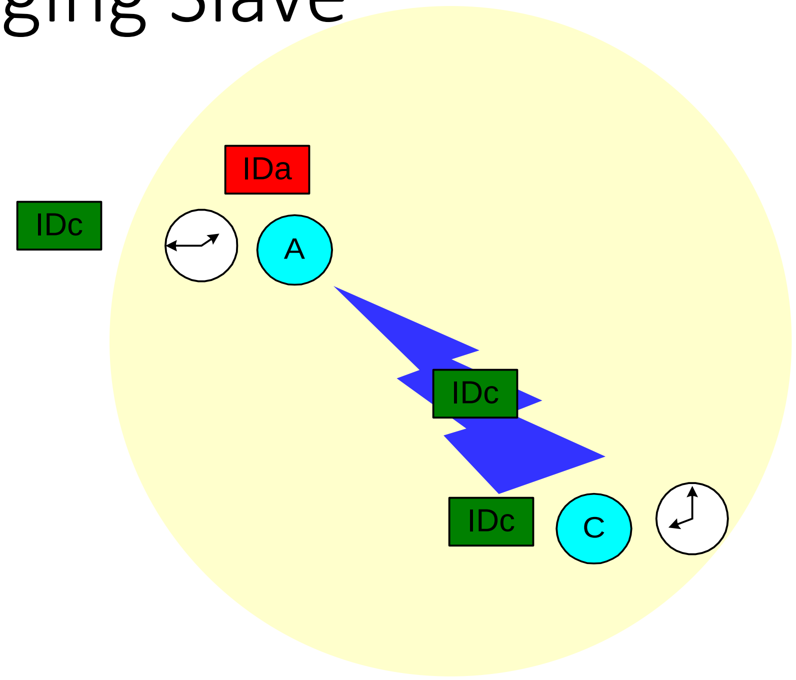
- Paging:
  - Assumes the master has C *deviceID* and *Clock*

# Master Paging Slave



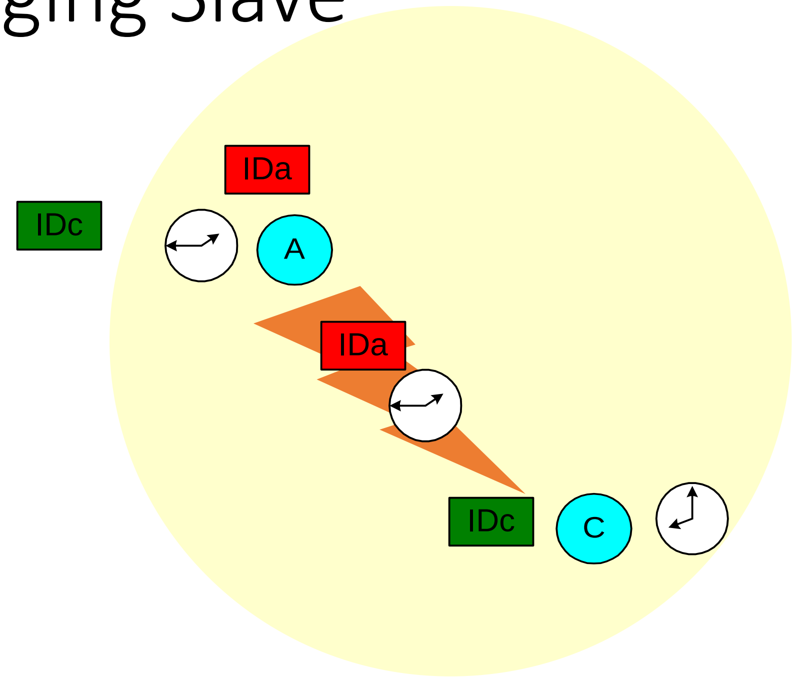
- Paging:
  - Assumes the master has C deviceID and Clock
    - A pages C with the deviceID of C

# Master Paging Slave



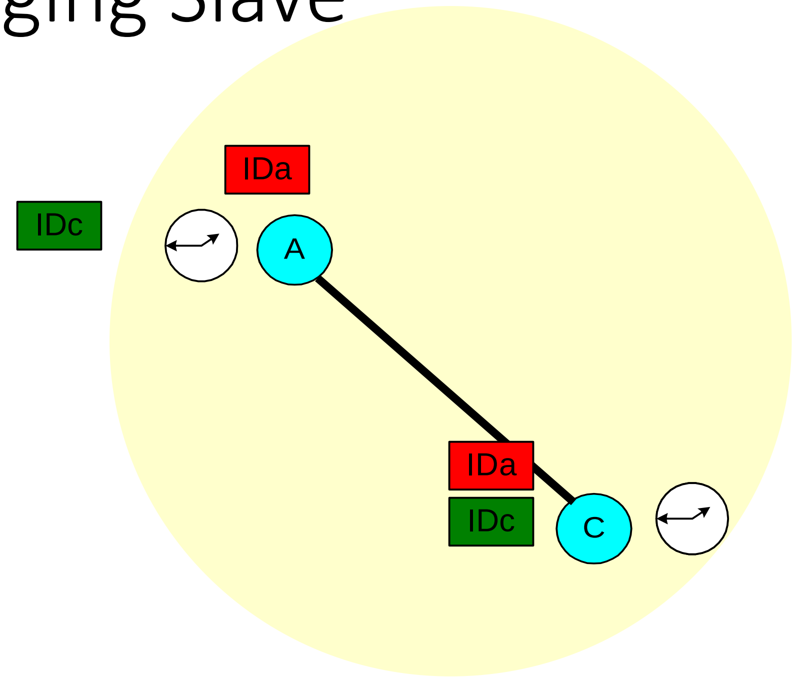
- Paging: master has the Device ID  and Clock
  - A pages C with the deviceID of C
  - C answers A with his deviceID

# Master Paging Slave



- Paging: master has the Device ID  and Clock
  - A pages C with the deviceID of C
  - C answers A with his deviceID
  - A sends C his deviceID and Clock (FHS packet)

# Master Paging Slave

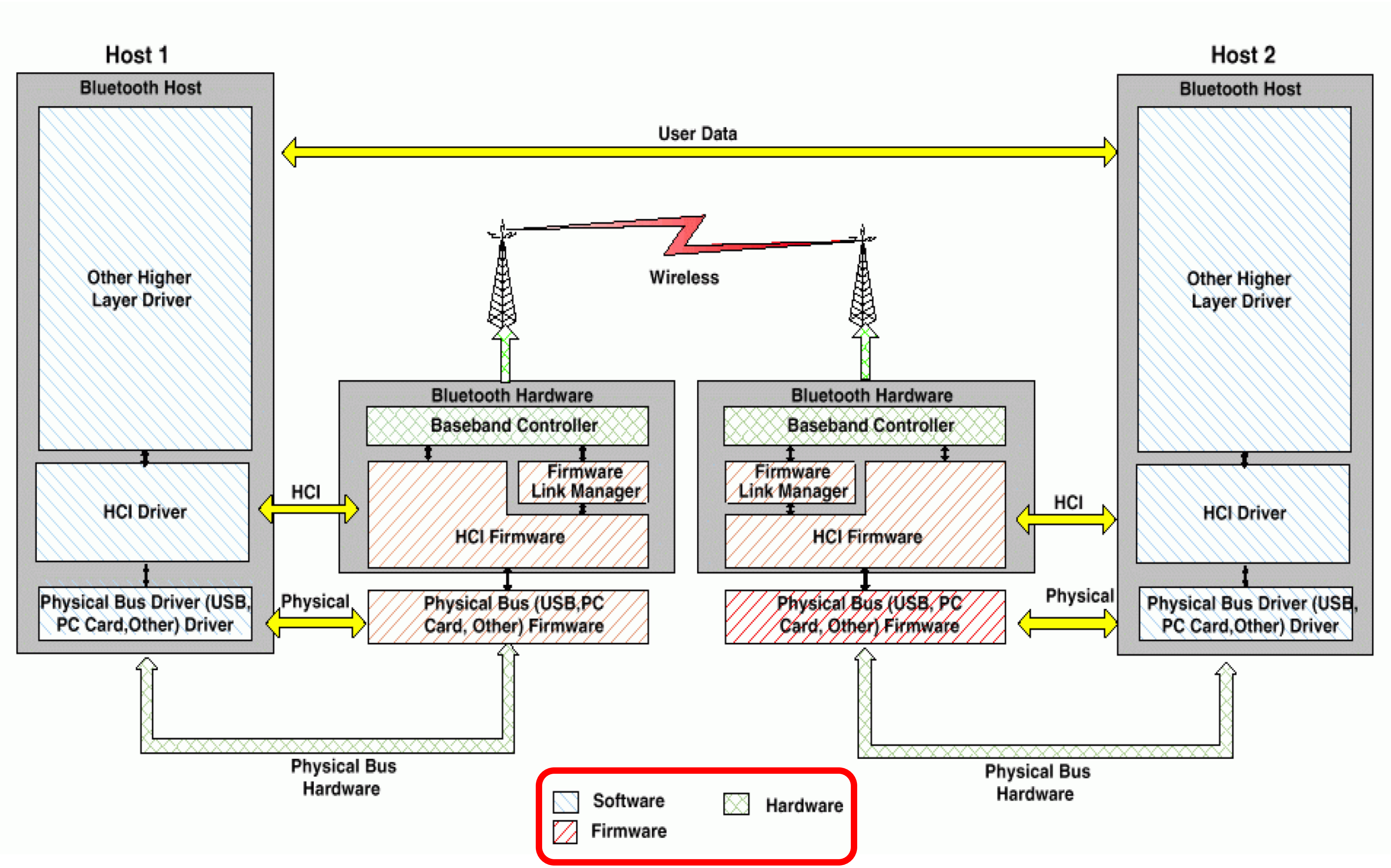


- Paging: master has the Device ID  and Clock
  - A pages C with the deviceID of C
  - C answers A with his deviceID
  - A send C his deviceID and Clock (FHS packet)
  - A becomes master of C

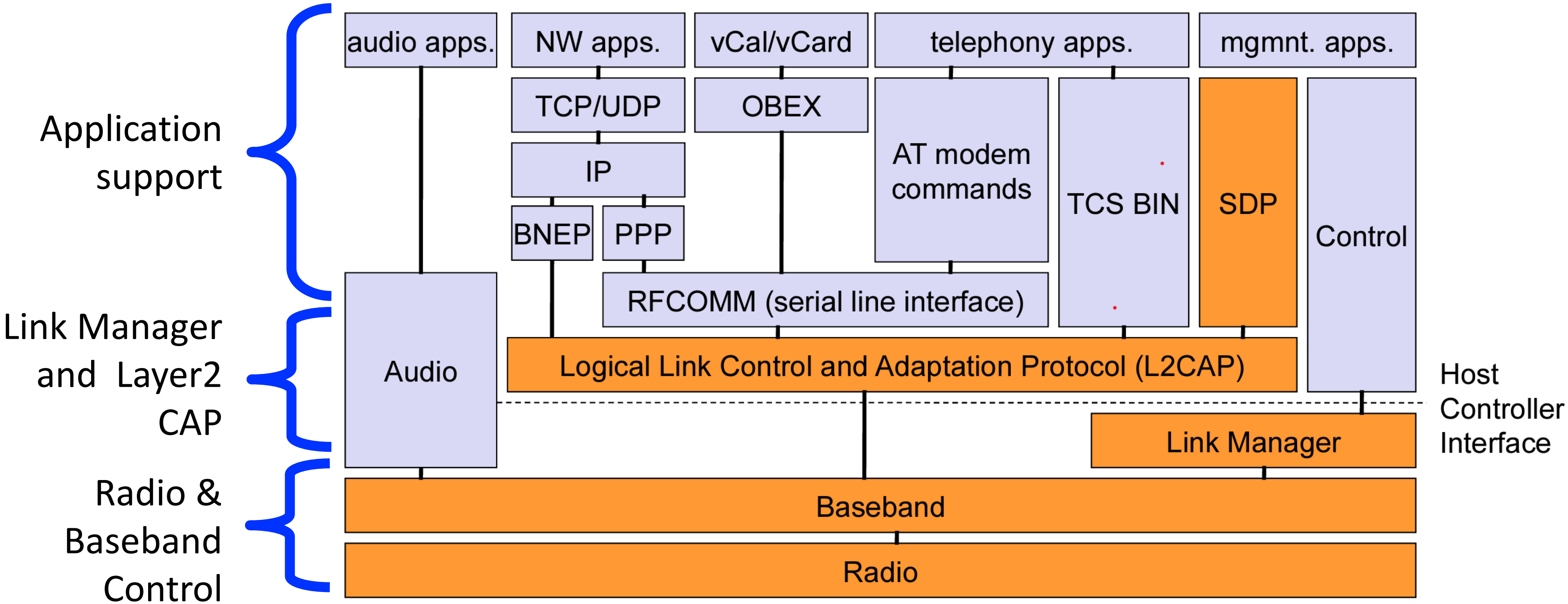
# Outline

- Bluetooth networks
- Piconet operation
  - Inquiry
  - Paging
- Bluetooth stack
- Profiles and security
- BT 4.0 BLE

# Communication between two BT devices

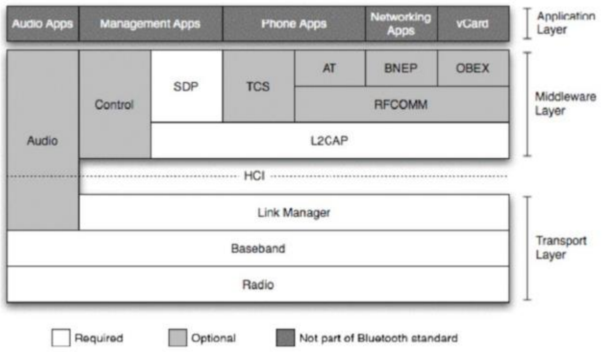


# Stack Bluetooth

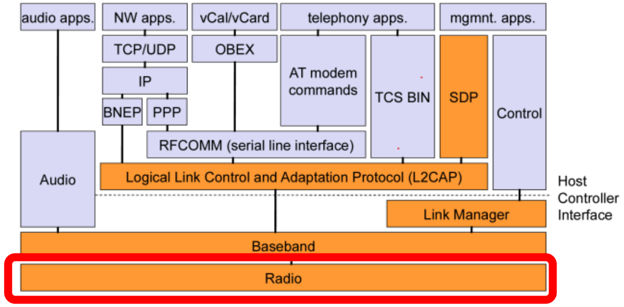


Bluetooth includes:

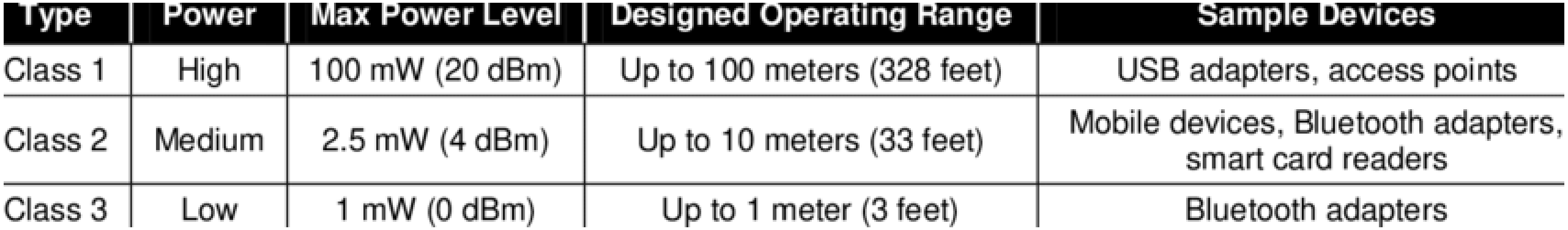
- A HW description
- An environment for applications



# Bluetooth Protocol



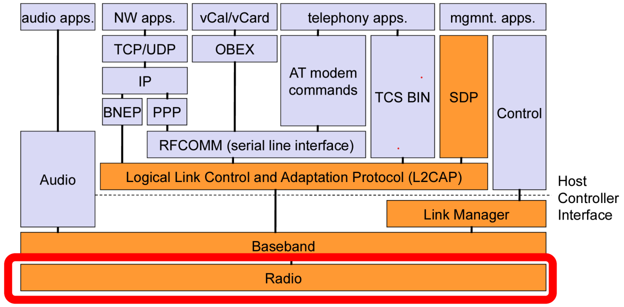
- Radio layer
  - Defines requirements for a Bluetooth radio transceiver
  - Handles conformity to 2.4GHz (ISM) band
  - Establishes specifications for using *Spread-Spectrum Frequency Hopping* (FHSS)
  - Classifies device into one of three power classes:
    - Long range; Class 1 - 100mW, 100m
    - Normal/standard range; Class 2 - 2.5mW, 10m
    - Short range; Class 3 - 1 mW, 1m

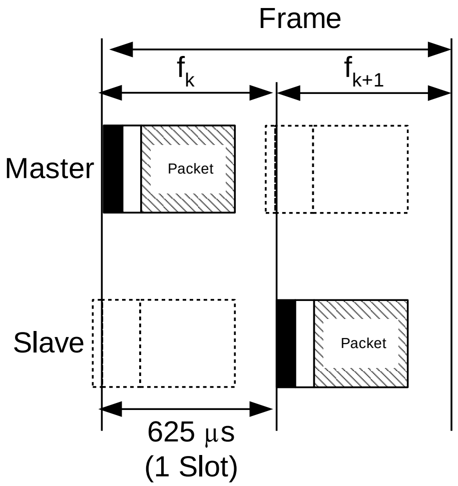


| Type | Power | Max Power Level | Designed Operating Range | Sample Devices |
|---|---|---|---|---|
| Class 1 | High | 100 mW (20 dBm) | Up to 100 meters (328 feet) | USB adapters, access points |
| Class 2 | Medium | 2.5 mW (4 dBm) | Up to 10 meters (33 feet) | Mobile devices, Bluetooth adapters, smart card readers |
| Class 3 | Low | 1 mW (0 dBm) | Up to 1 meter (3 feet) | Bluetooth adapters |

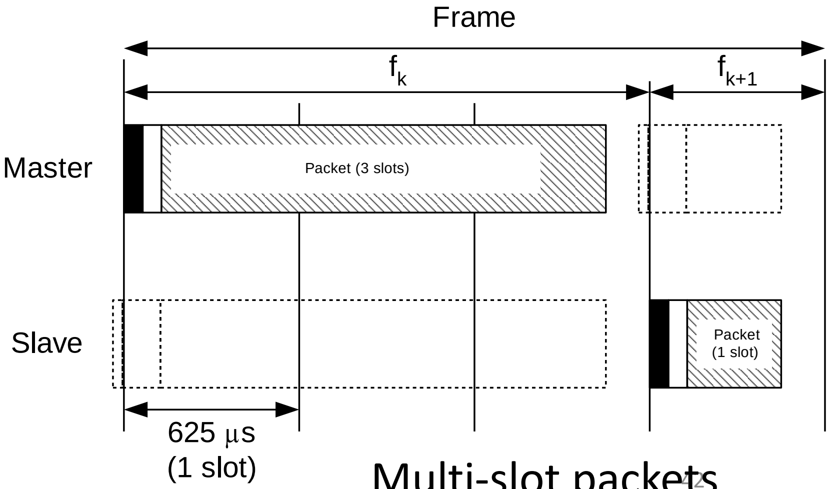
# Radio Layer

- Radio: FH SS
  - 79 channels of 1 Mb/s
  - Hoping: per slot
    - Packets have 1, 3, or 5 slots of 625 uS
    - Hoping (nominal) 1600 times per second
  - Frame includes two packets
    - Transmission followed by reception
  - Radio designed to low cost and universal usage
    - noise, synchronous action technology 2.4GHz, etc...,



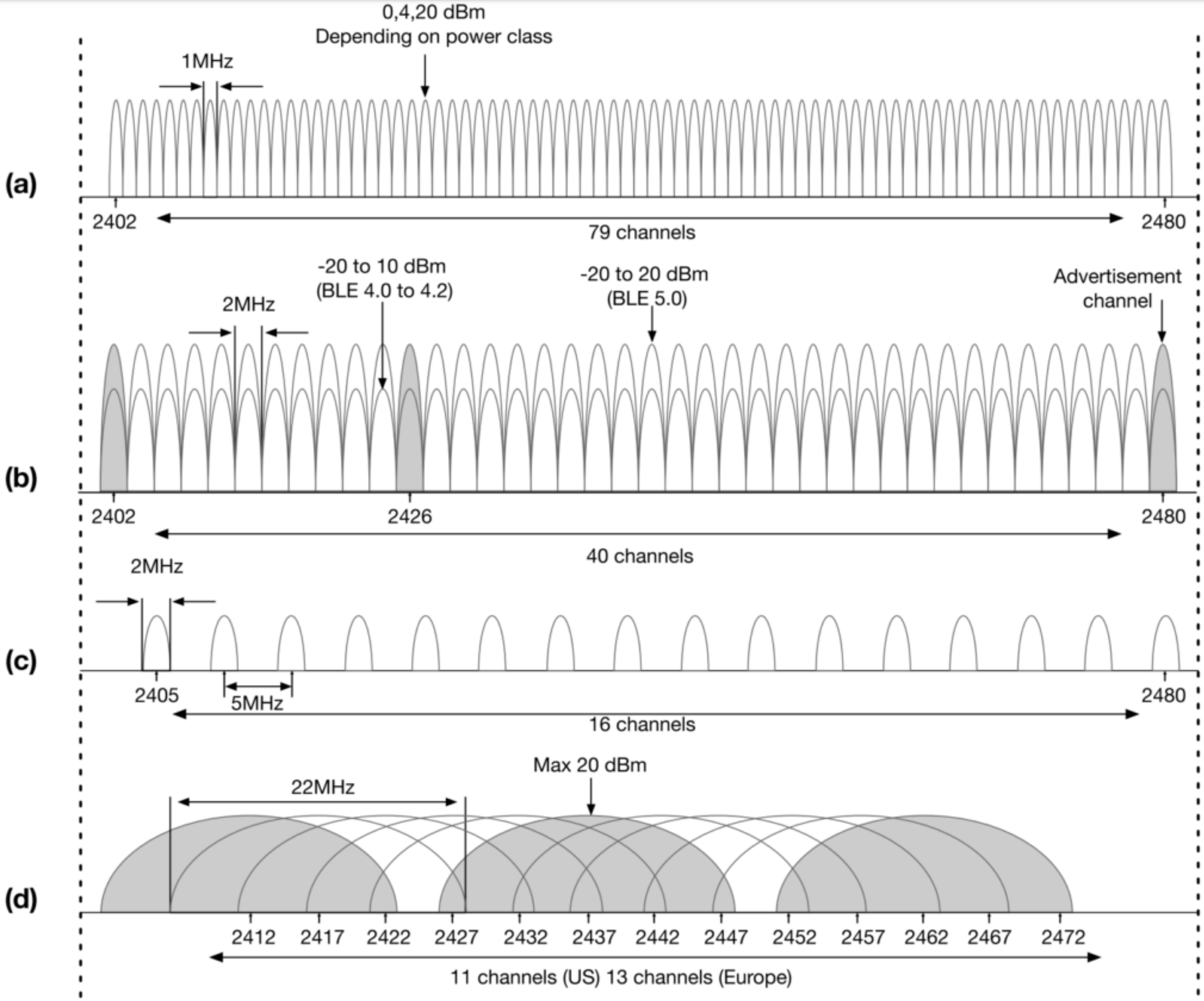


TDD



Multi-slot packets

# Bluetooth spectrum (comparison)



(a) Traditional Bluetooth; 79 channels with 1MHz width

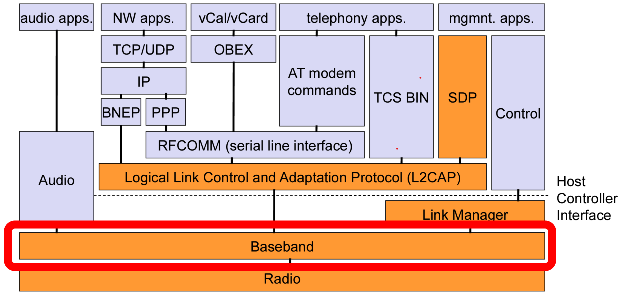
(b) BLE (4.0-4.2 and 5.0); 40 channels 2MHz wide; 3 ‘advertisement channels’

(c) 16 channels used by IEEE 802.15.4 based networks (e.g. ZigBee)

(d) IEEE 802.11b™ DSS channels; 22MHz wide channels

Image from IEEE Access: “Low-Power Wireless for the Internet of Things: Standards and Applications.
https://www.researchgate.net/publication/328843842_Low-Power_Wireless_for_the_Internet_of_Things_Standards_and_Applications

# Baseband in Bluetooth

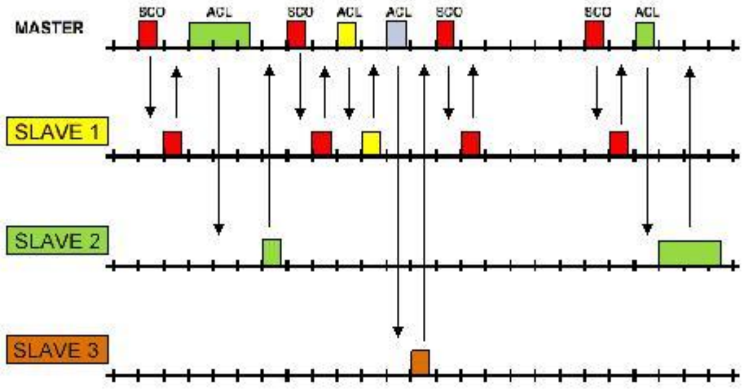


- Manages physical channels and logical lines
  - Controls device addressing, channel control, power-saving operations, and flow control and synchronization among devices
  - Implements TDD aspects: master and slave switch in communications

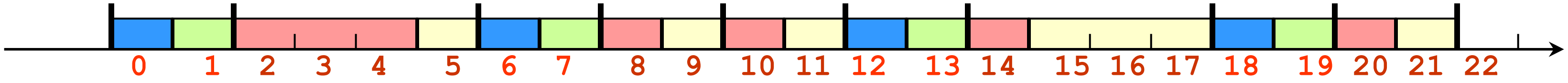
- Works closely with Link controller:
  - Manages link (a)synchronism
  - Controls paging and inquiries
  - Controls power save modes

# Baseband link types

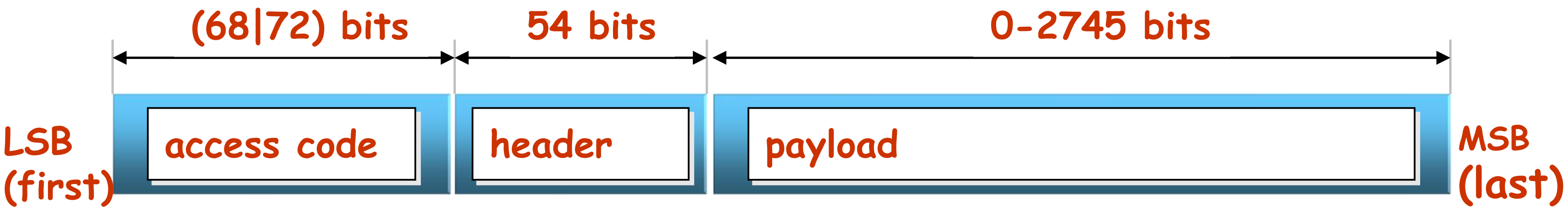
- Polling-based (TDD) frame transmissions
  - 1 slot: 0.625 uS (max 1600 slots/sec)
  - Master/Slave slots (even-/odd-numbered slots)
  - Polling: master always "polls" slaves
- Synchronous Connection-Oriented (SCO) link
  - "Circuit-switched"
    - Periodic single-slot frame assignment
  - Symmetric 64Kbps full-duplex
- Asynchronous Connection-Less (ACL) link
  - Frame switching
  - Asymmetric bandwidth
    - Variable frame size (1-5 slots)
      - max. 721 kbps (57.6 kbps return channel)
      - 108.8 - 432.6 kbps (symmetric)



| | SCO | ACL |
|---|---|---|
| Master | 🟦 | 🟥 |
| Slave | 🟩 | 🟨 |

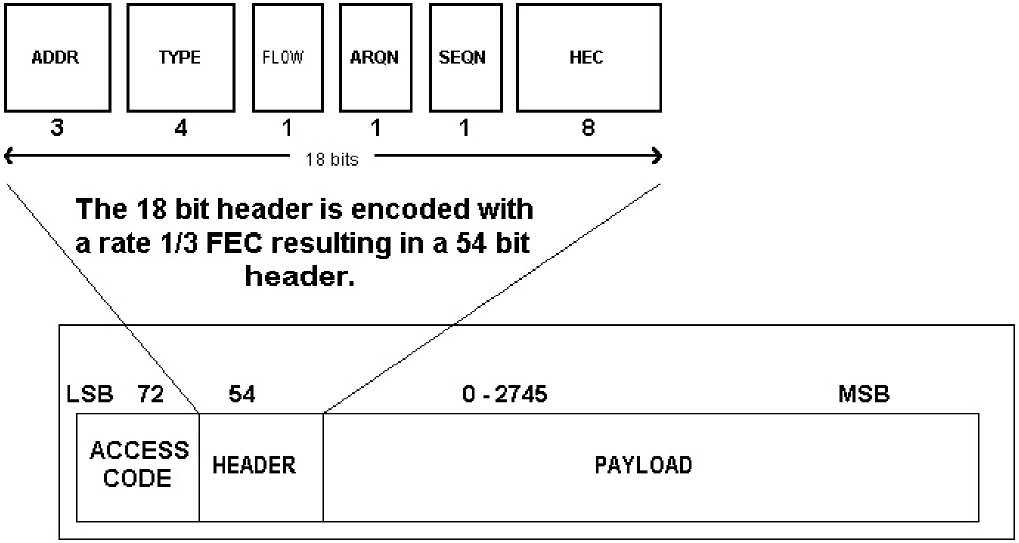


# Baseband Frame



- **Access Code**: time synchronization, offset, paging, inquiry
  - 3 types:
    - Channel Access Code (CAC), piconet identification, synchronization, DC offset
    - Device Access Code (DAC), paging and replies
    - Inquiry Access Code (IAC), inquiries (GIAC, general; DIAC, dedicated)
- **Header**: packet acknowledgement and numbering, flow control, slave address, error checking
- **Payload**: voice, data or both (DV packets)
  - When data, the payload has an additional internal header

# Baseband Packet


ADDR
TYPE
FLOW
ARQN
SEQN
HEC
3
4
1
1
1
8
18 bits
The 18 bit header is encoded with a rate 1/3 FEC resulting in a 54 bit header.
LSB
72
54
0 - 2745
MSB
ACCESS CODE
HEADER
PAYLOAD

| | |
|---|---|
| ADDR | 000 is for broadcasting |
| TYPE | 16 types<br>Also specifies the length of the packet<br>Dependent on the type of connection, i.e., ACL or SCO |
| FLOW | If the buffer in the recipient is full, a STOP (0) is sent<br>A GO (1) is sent for indicating that more data packets can be received |
| ARQN | ACK (1) is sent if the data is successfully received<br>A NAK (0) is sent if data was not received or contains errors |
| SEQN | Determines the sequence of received packet |
| HEC | Value to check for the integrity of the header information |

# Packets: Common

| TYPE | NAME | # | DESCRIPTION |
|---|---|---|---|
| Common | ID | 1 | Carries device access code (DAC) or inquiry access code (IAC). |
| | NULL | 1 | NULL packet has no payload. Used to get link information and flow control. Not acknowledged. |
| | POLL | 1 | No payload. Acknowledged. Used by master to poll the slaves to know whether they are up or not. |
| | FHS | 1 | A special control packet for revealing Bluetooth device address and the clock of the sender. Used in page master response, inquiry response and frequency hop synchronization. 2/3 FEC encoded. |
| | DM1 | 1 | To support control messages in any link type. can also carry regular user data. Occupies one slot. |

# Packets: Synchronous Connection-Oriented (SCO)

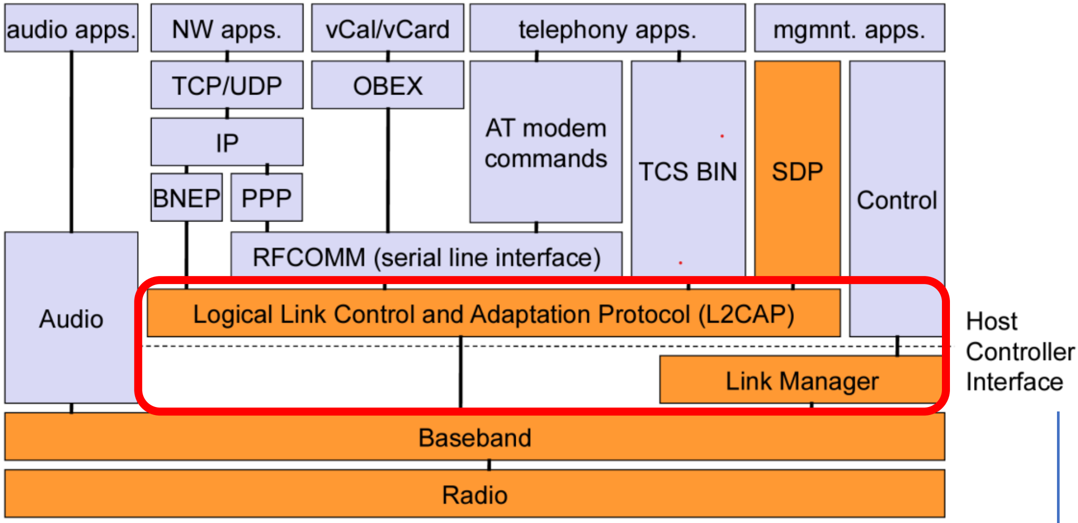
| | | | |
|---|---|---|---|
| SCO | HV1 | 1 | Carries 10 information bytes. Typically used for voice transmission. 1/3 FEC encoded. |
| | HV2 | 1 | Carries 20 information bytes. Typically used for voice transmission. 2/3 FEC encoded. |
| | HV3 | 1 | Carries 30 information bytes. Typically used for voice transmission. Not FEC encoded. |
| | DV | 1 | Combined data-voice packet. Voice field not protected by FEC. Data field 2/3 FEC encoded. Voice field is never retransmitted but data field can be. |

# Packets : Assynchronous Connection-Less (ACL)

| | | | |
|---|---|---|---|
| ACL | DM1 | 1 | Carries 18 information bytes. 2/3 FEC encoded. |
| | DH1 | 1 | Carries 28 information bytes. Not FEC encoded. |
| | DM3 | 3 | Carries 123 information bytes. 2/3 FEC encoded. |
| | DH3 | 3 | Carries 185 information bytes. Not FEC encoded. |
| | DM5 | 5 | Carries 226 information bytes. 2/3 FEC encoded. |
| | DH5 | 5 | Carries 341 information bytes. Not FEC encoded. |
| | AUX1 | 1 | Carries 30 information bytes. Resembles DH1 but no CRC code. |

# Adaptation protocols

- Link Manager
  - Carries out link setup above baseband, with authentication, link configuration and other protocols
    - Support protocol multiplexing
      - BT may support other protocols besides IP
    - Segmenting and reassembly
- Link Layer Control & Adaptation (L2CAP)
  - Link control protocol, provides connection-oriented and connectionless data services to upper layer protocols
    - Handles ACL and SCO connections
    - Handle QoS specifications per connection (logical channel)
    - Manages concepts as "group of connections"



- Host Controller Interface (HCI)
  - Allows command line access to the baseband layer and LM for control and status information
    - Current interfaces: USB; UART; RS-232
  - Made up of three parts:
    - HCI firmware, HCI driver, Host Controller Transport Layer

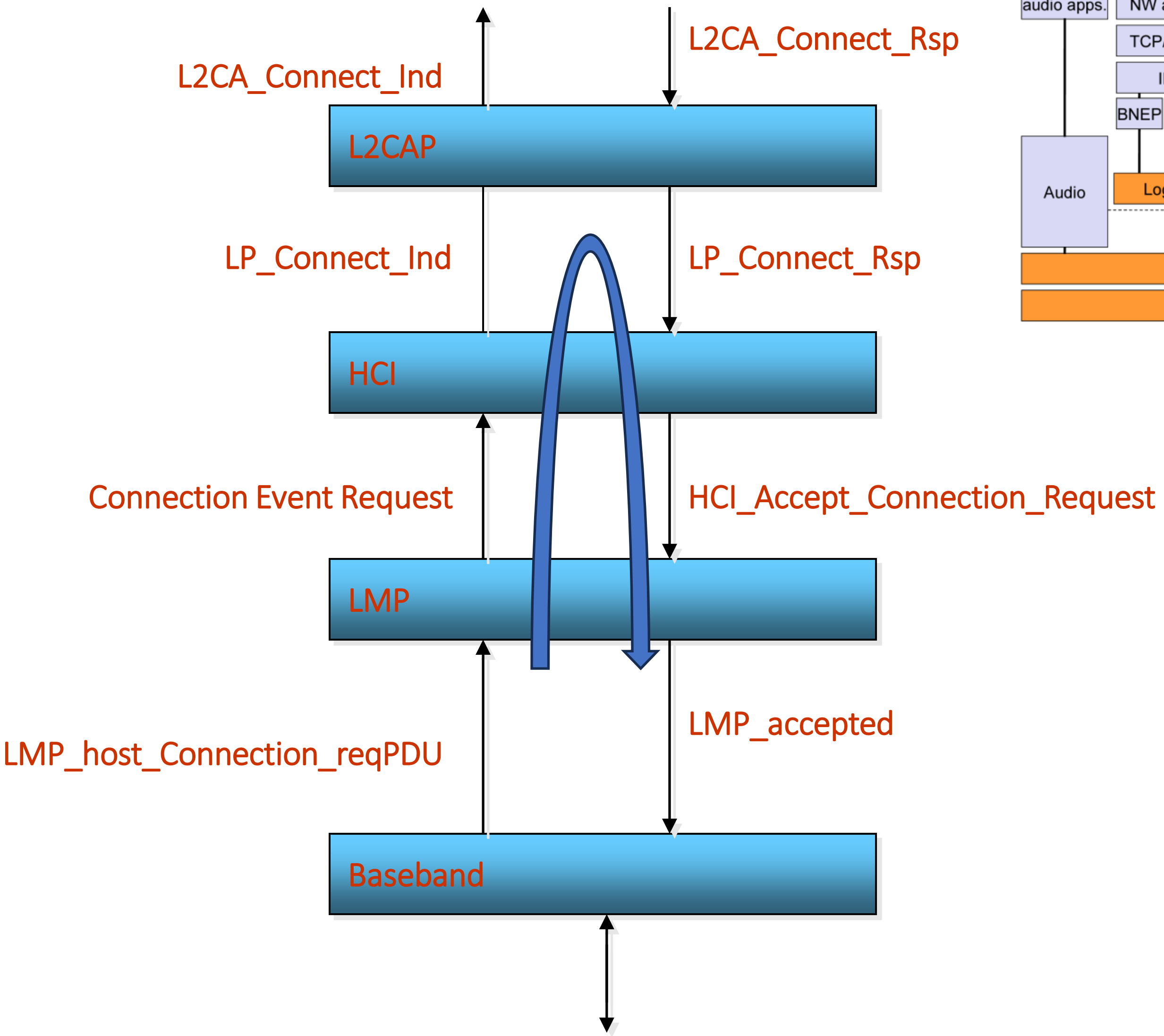
# Host-Controller Interface (HCI)

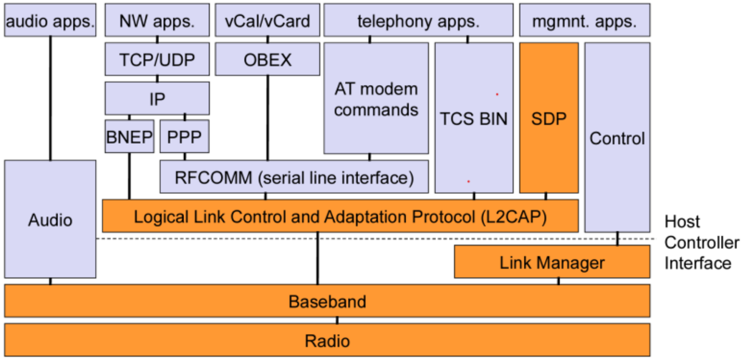
- Specifies all interactions between a host and a Bluetooth radio controller
- Defines how commands, events, asynchronous and synchronous data packets are exchanged
- HCI Packet Types
  - Command (0x01)
    - Each command is assigned a 2 byte Opcode which it's divided into two fields, called the OpCode Group Field (OGF) and OpCode Command Field (OCF)
  - Asynchronous Data (0x02)
  - Synchronous Data (0x03)
  - Events (0x04)

See Bluetooth Lab guide Annexes for packet formats

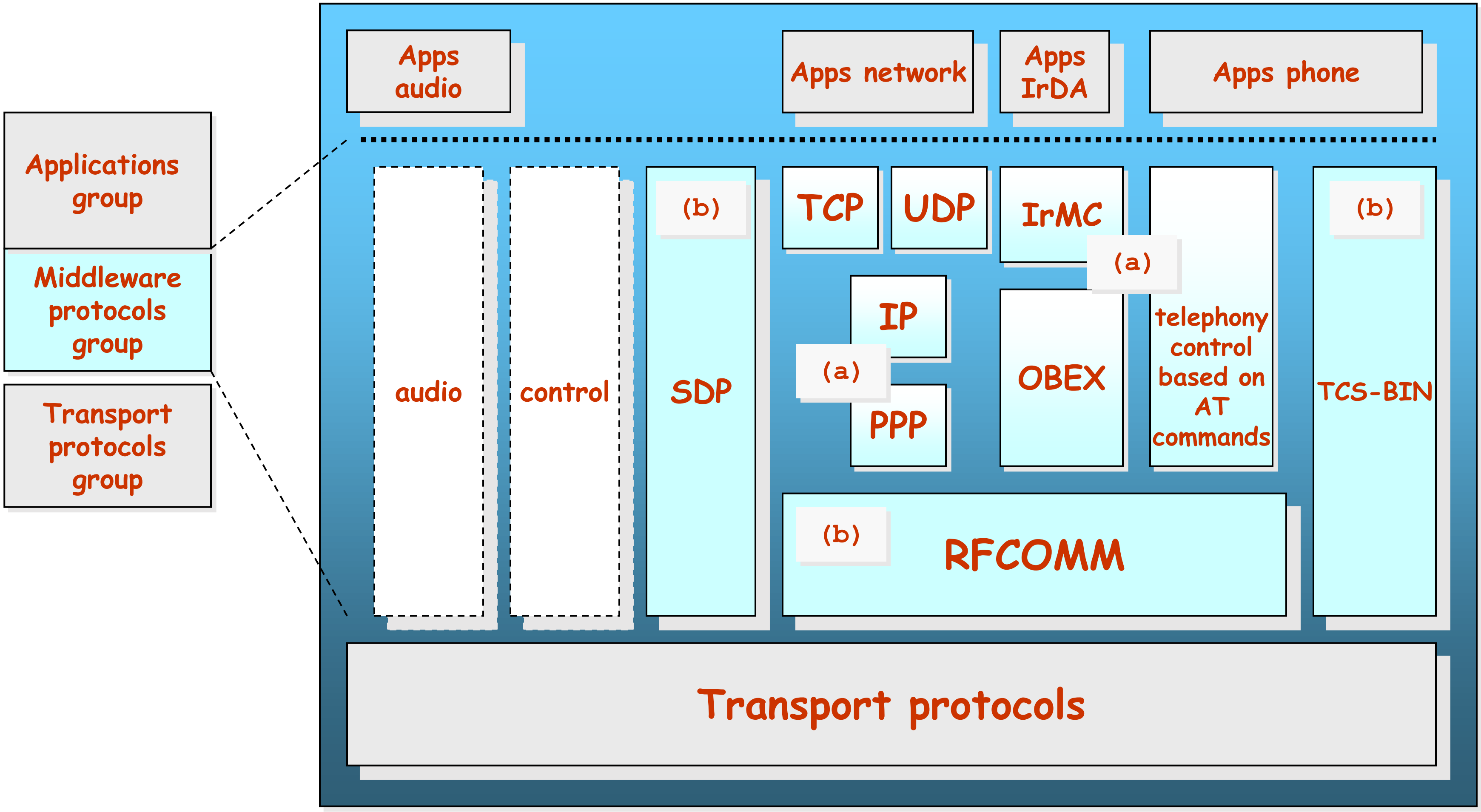
Complete list of HCI Commands, Events and Error Codes:
https://lisha.ufsc.br/teaching/shi/ine5346-2003-1/work/bluetooth/hci_commands.html

# Interlayer communication





# Protocols (middleware)



Protocol reusage

BT aims to reuse older protocols (e.g. WAP, OBEX–IrDA)
Interaction with applications and phones, as commonly done before

```
a: common protocol
b: Bluetooth dedicated protocol
```

SDP: Service Discovery Protocol
OBEX: Facilitates binary transfers between BT devices
TCP-BIN: Telephony-control protocol binary (call control)

# Middleware

- **Service Discovery Protocol** (SDP)
  - Provides a way for applications to detect which services are available and their characteristics
  - Protocol question ➤➤ answer
    - Search and browsing of services
  - Defines a format for service registry
    - Information provided by the service *attributes,* a name (ID) + value
    - IDs can be universal (UUID)

# Middleware

- **RFCOMM** (Serial Port Emulation Protocol)
  - Based on GSM TS07.10
  - Emulates a serial port, supporting all traditional applications that were able to use a serial port
  - Supports multiple ports over a single physical channel between two devices

- **Telephony Control Protocol Spec** (TCS)
  - Handles call control (setup, release)
  - Group management for gateways, serving multiple devices
    - Audioconference, e.g.

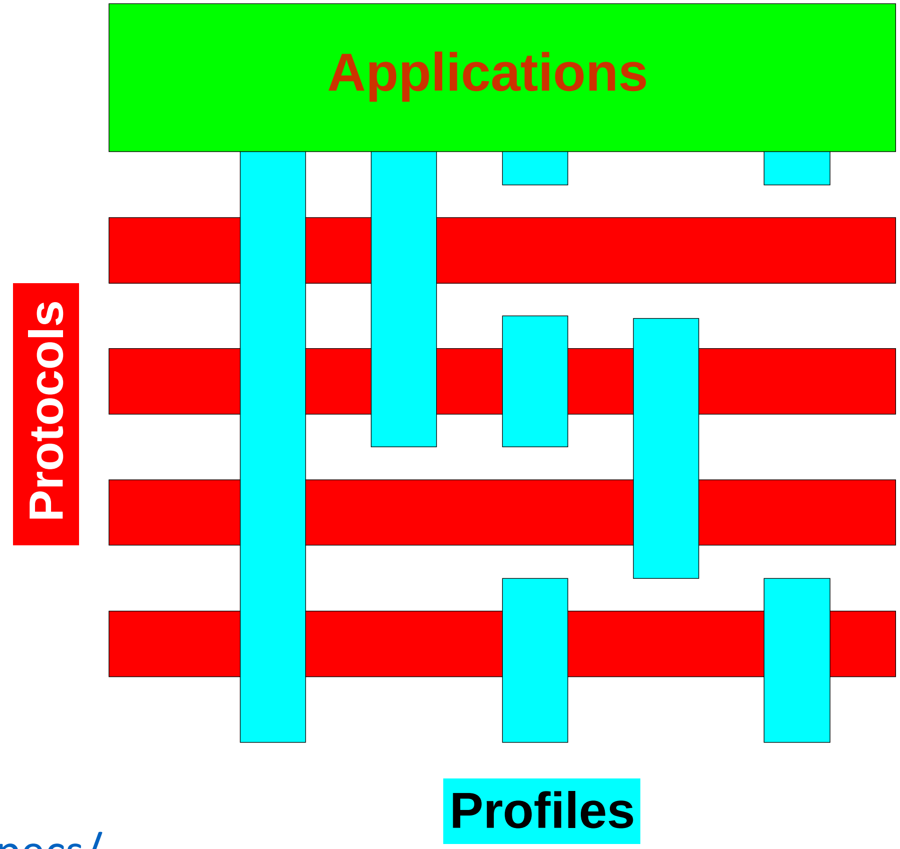
# Outline

- Bluetooth networks
- Piconet operation
  - Inquiry
  - Paging
- Bluetooth stack
- Profiles and security
- BT 4.0 BLE

# Interoperability: Profiles

- Profile: base for BT interoperability (BT too much flexible!)
- “vertical cut” in Bluetooth stack
- A given usage model (typical solution)
- Each BT device supports one or more profiles

https://www.bluetooth.com/specifications/specs/



# Profiles (v.1)

- Generic Access
  - Profile SDA (*Service Discovery Application*)

  - Profiles for serial port, including:
    - Profile Dial-up
    - Profile Fax
    - Profile Headset
    - LAN Access (uses PPP)
    - Profile for generic object exchange (OBEX)
      - File transfer
      - Data synchronization
      - Push-pull

- Profile of cordless phone (TCS-BIN)
  - Profile interphone
  - Profile Cordless Telephony

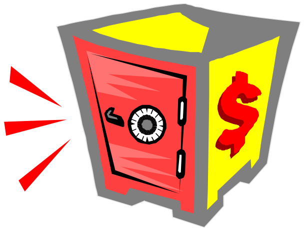
# Profiles (v.2)

- **Advanced Audio Distribution Profile** (A2DP)
  - Dual-channel audio stream through a stereo headset
  - Can also be used to make calls, and users can switch between music and calls at the touch of a button

- **Audio/Video Remote Control Profile** (AVRCP)
  - Provides a standard interface to control TVs, hi-fi equipment, and so forth
  - A single remote control (or other device) to control all the AV equipment to which a user has access
  - Defines how to control characteristics of streaming media (pausing, stopping, and starting playback and volume control)

- **Hands-Free Profile (HFP)**
  - Use a gateway device to place and receive calls for a hand-free device
  - Example: vehicle using a mobile phone as a gateway device. Car’s audio system and an installed microphone are used instead of the phone's audio

# Bluetooth: security

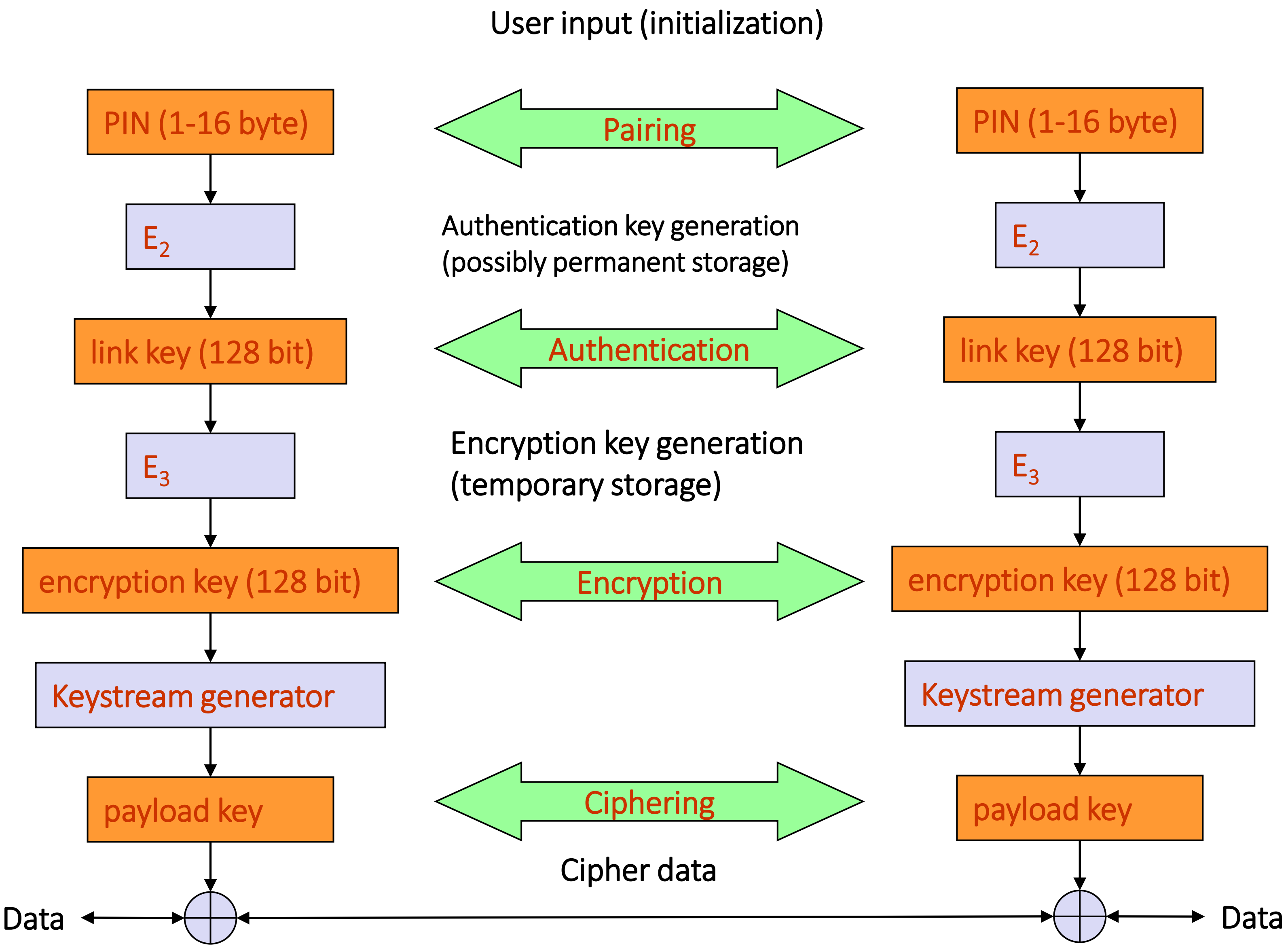


- Devices can be:
  - “Trusted”
  - “Untrusted”
    - Also “unknown” devices
- Services security types:
  - Open services – cypher only
  - Authentication only – machine ID
  - Authentication and authorization (ID+explicit service grant)
- Levels of security:
  - Mode 1
    - No security
  - Mode 2
    - Security guaranteed at service level
  - Mode 3
    - Security guaranteed at link level

# Bluetooth: security features

- Mechanisms used in BT for security
  - Fast frequency hopping
  - Low range
  - Authentication
    - Two way challenge/response mechanism
  - Cypher (to ensure privacy)
    - Data between two devices can be encrypted
    - Keys used
      - Cypher size configurable (0-16bytes) by the devices, but there are security constrains (goverment)
      - Keys using standard well-known algorithms
  - Security initialization – device pairing
    - PIN (user input)
    - Shared key

# Security


User input (initialization)
PIN (1-16 byte)
Pairing
PIN (1-16 byte)
$E_2$
Authentication key generation
(possibly permanent storage)
$E_2$
link key (128 bit)
Authentication
link key (128 bit)
$E_3$
Encryption key generation
(temporary storage)
$E_3$
encryption key (128 bit)
Encryption
encryption key (128 bit)
Keystream generator
Keystream generator
payload key
Ciphering
payload key
Cipher data
Data
Data

# Outline

- Bluetooth networks
- Piconet operation
  - Inquiry
  - Paging
- Bluetooth stack
- Profiles and security
- BT 4.0 BLE

# Bluetooth 4.0: Low Energy

# Short range wireless application areas

| | Voice | Data | Audio | Video | **State** |
|---|---|---|---|---|---|
| **Bluetooth ACL/HS** | | Y | Y | | |
| **Bluetooth SCO/eSCO** | Y | | | | |
| **Bluetooth low energy (BLE)** | | | | | Y |
| **Wi-Fi** | (VoIP) | Y | Y | Y | |
| **Wi-Fi Direct** | Y | Y | Y | | |
| **ZigBee** | | | | | Y |

**State = low bandwidth, average/low latency data**

Low Power

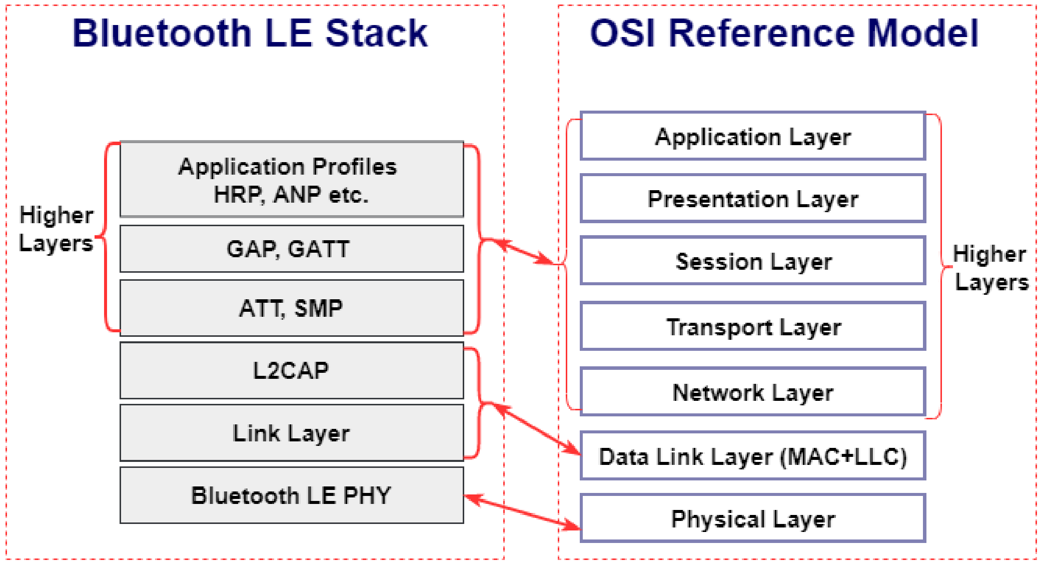
# What is Bluetooth Low Energy (BLE)?

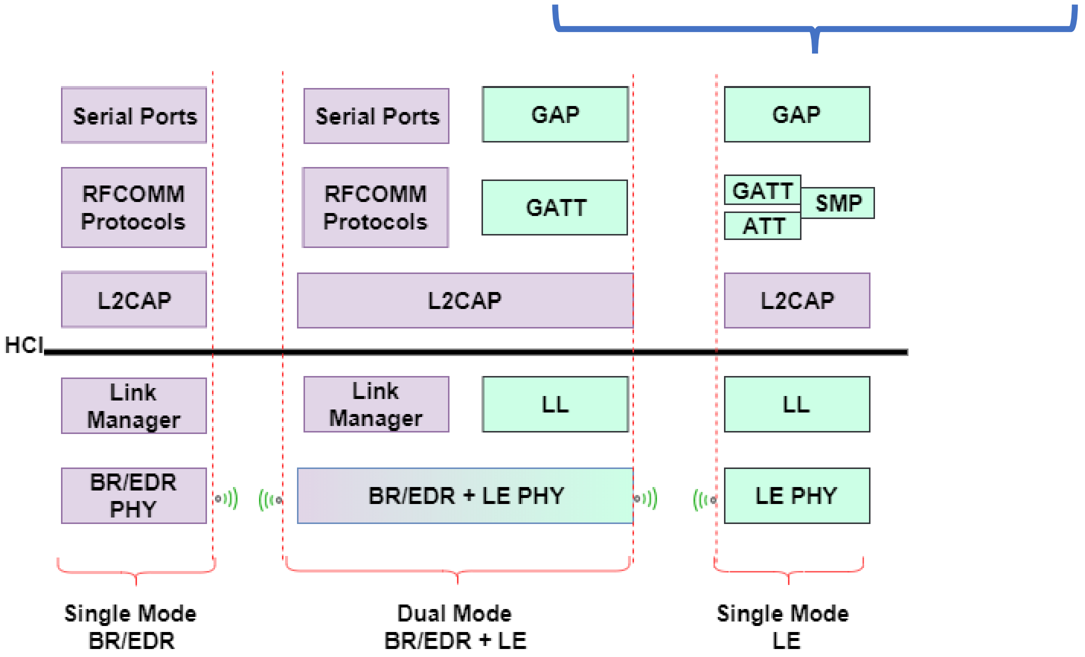
- Bluetooth Low Energy is an open, short range radio technology
  - Blank sheet of paper design
  - Different to Bluetooth classic (BR/EDR)
  - Optimized for ultra low power
  - Enable coin cell battery use cases
    - < 20mA peak current
    - < 5 uA average current

# Basic concepts of BLE

- Everything is optimized for lowest power consumption
  - Short packets reduce TX peak current
  - Short packets reduce RX time
  - Less RF channels to improve discovery and connection time
  - Simple state machine
  - Single protocol
  - Needs a gateway for Internet access
  - Etc.

# BLE Protocol Stack





# Bluetooth Low Energy factsheet

| | |
|---|---|
| Range: | **~ 150 meters open field** |
| Output Power: | **~ 10 mW (10dBm)** |
| Max Current: | **~ 15 mA** |
| Latency: | **3 ms** |
| Topology: | **Star** |
| Connections: | **> 2 billion** |
| Modulation: | **GFSK @ 2.4 GHz** |
| Robustness: | **Adaptive Frequency Hopping, 24 bit CRC** |
| Security: | **128bit AES CCM** |
| Sleep current: | **~ 1µA** |
| Modes: | **Broadcast, Connection, Event Data Models, Reads, Writes** |