Fast, Simple Debugging for Bluetooth Low Energy Applications

Frontline Test Equipment
Bluetooth Special Interest Group

This webinar will start momentarily. Your microphones will be muted during this session

Muhammad Ulislam – Technical Marketing Manager at Bluetooth SIG

David Bean – President of Frontline Test Equipment (FTE)

Roger Feeley – Senior Software Engineer at FTE

Panelist Introductions

Fast, Simple Debugging for Bluetooth Low Energy Applications

Frontline Test Equipment
Bluetooth Special Interest Group

Agenda

- Introduction to Frontline Air Interface Sniffer: BPA 500
- Introduction to Hello Bluetooth Profile
- Bluetooth Low Energy Air Interface packet structure
- Analyzing trace for “Hello Bluetooth” custom profile
- DEMO: Hello Bluetooth and DecoderScript
- Q&A

Development tools

- The following tools were used in order to create the demos
  - Texas Instrument CC2540 mini DK
  - MAC OS Lion + PTS dual mode dongle to run on a simulator
  - Frontline Air Interface Sniffer
- What will you walk away with this session
  - A captured trace using Frontline sniffer
  - Hello Bluetooth XML files
  - Hello Bluetooth profile source code on TI CC2540 and hex files
  - Hello Bluetooth Profile source code on iOS

Introduction to Frontline Air Interface Sniffer: BPA 500

Introduction to Hello Bluetooth Profile

Bluetooth Low Energy Air Interface packet structure

Analyzing trace for “Hello Bluetooth” custom profile

DEMO: Hello Bluetooth and DecoderScript

Q&A

Introduction to Air Sniffers – BPA 500

- Passive Listening
- Non-Intrusive – invisible to the device under test
- Decrypt communication streams
- Decode commands to make them easy to read
- Analyze results and data

Air Sniffers – ComProbe BPA 500

ComProbe BPA 500 Dual Mode *Bluetooth* Protocol Analyzer (*Bluetooth* v4.0 + HS)

- "Classic" (BR/EDR)
- low energy
- 802.11 - High Speed (when combined with the **ComProbe 802.11 Analyzer**)

Air Sniffers - find the problem *fast!*

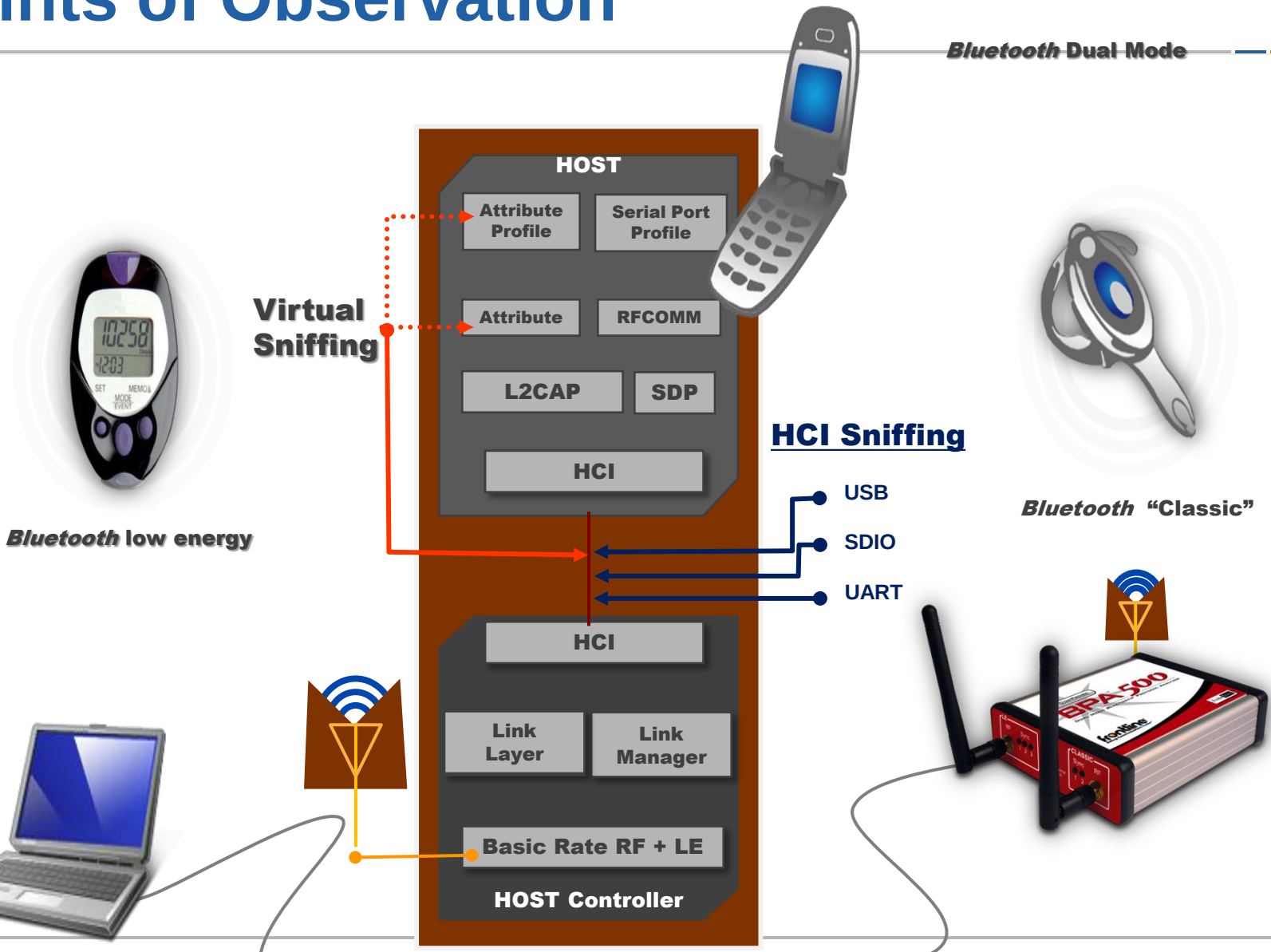
The ComProbe® BPA® 500 Dual Mode *Bluetooth* Protocol Analyzer is to a developer what an X-ray machine is to a doctor.

Multiple Points of Observation

BPA 500 can sniff:

- Air Traffic
  - "Classic" (BR/EDR)
  - low energy
  - Dual mode – "classic" AND low energy
- HCI Traffic with BPA 500 add-ons
- Virtual Sniffing (software sniffing)
- BTSnoop (Free file format for logging data readable in Frontline viewers)

Points of Observation



Sniffs Air – Dual Mode

Sniffs low energy and “Classic” *Bluetooth* devices

Dual mode *Bluetooth* device

Bluetooth device

low energy *Bluetooth* device

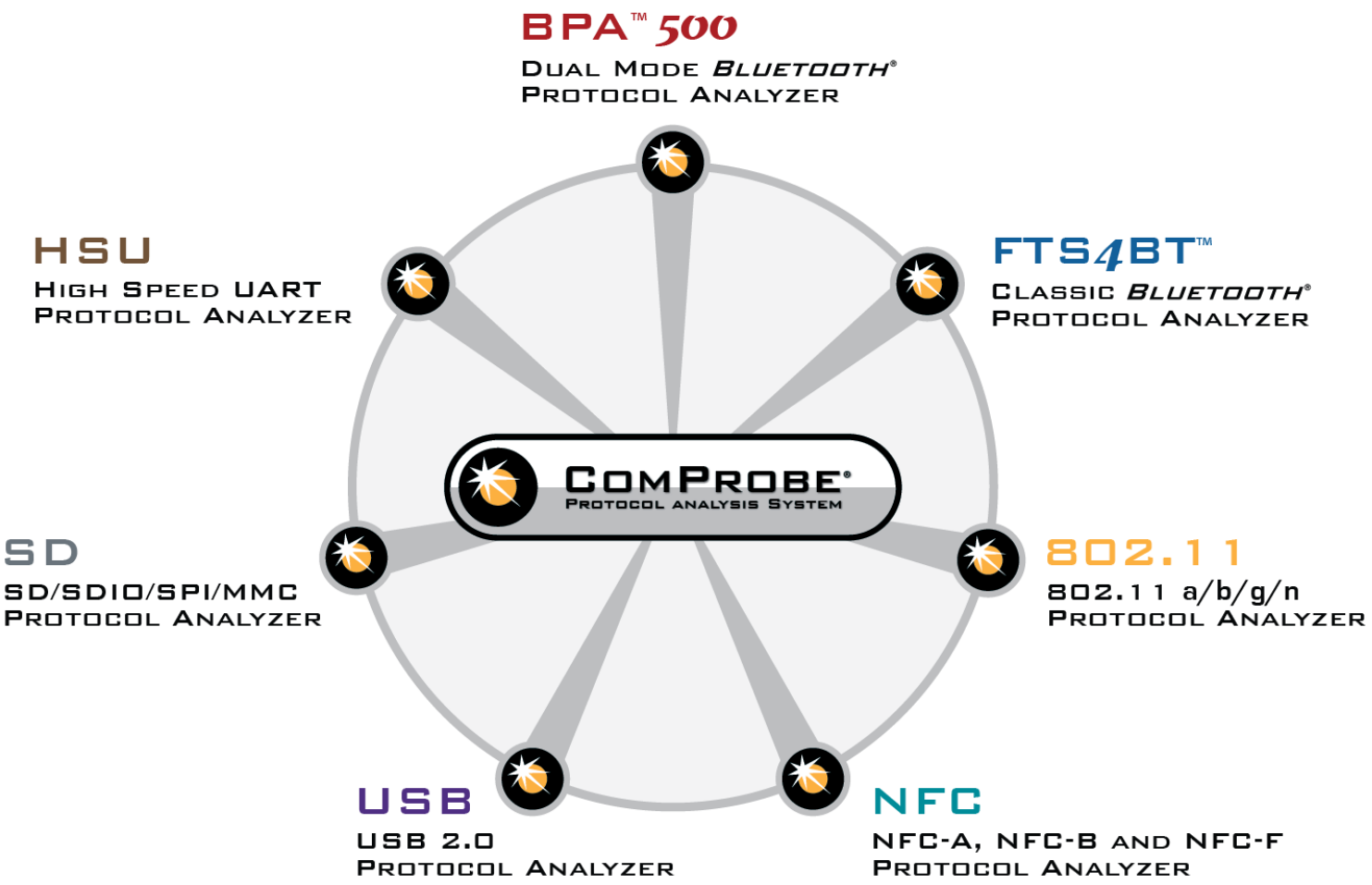
Displays all packets into a single view

Frontline Supported Profiles & Protocols

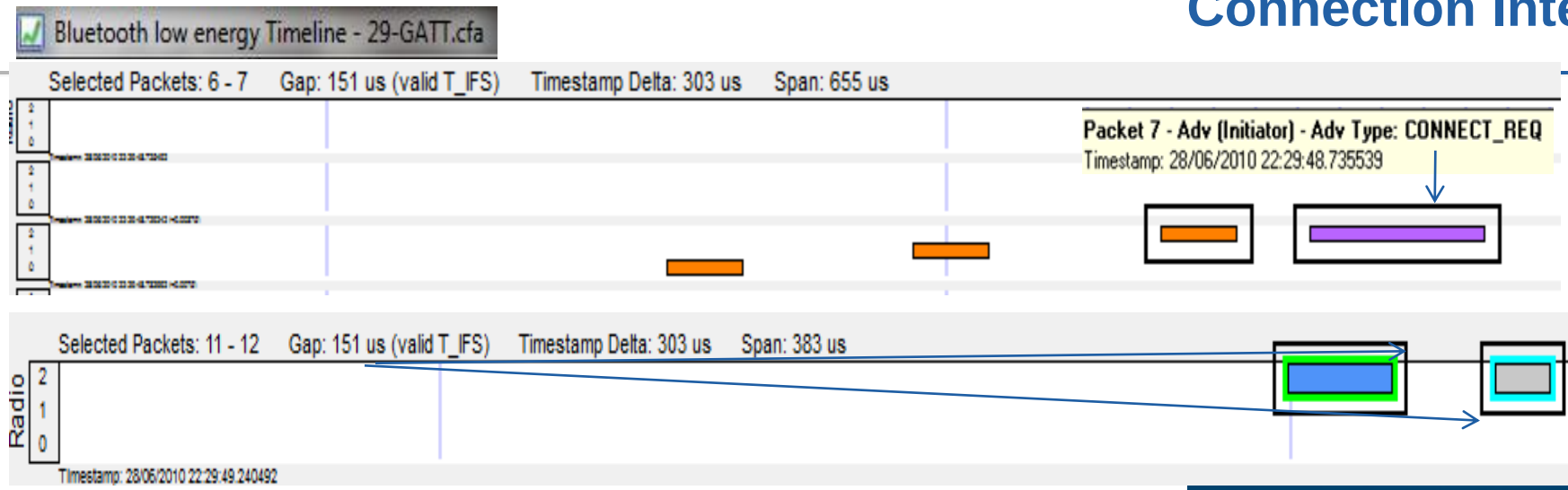
- All *Bluetooth* Specification supported
- All *Bluetooth* Profiles and Protocols Supported

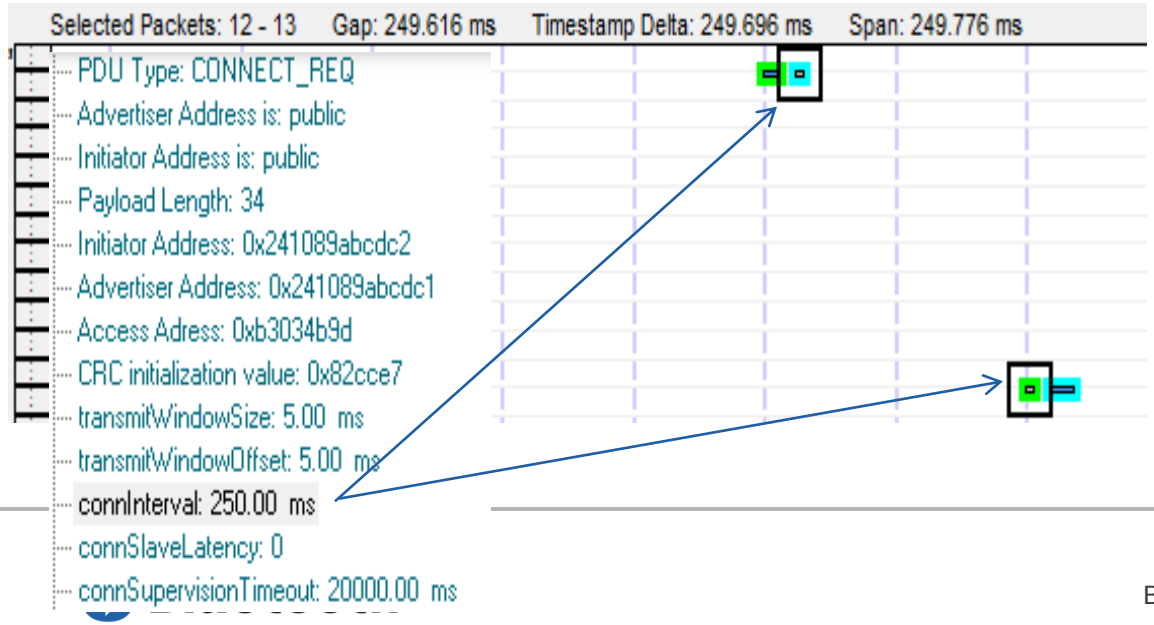
- Visit the **developer.bluetooth.org** site for the latest additions and releases to protocols and profiles related to *Bluetooth* low energy.

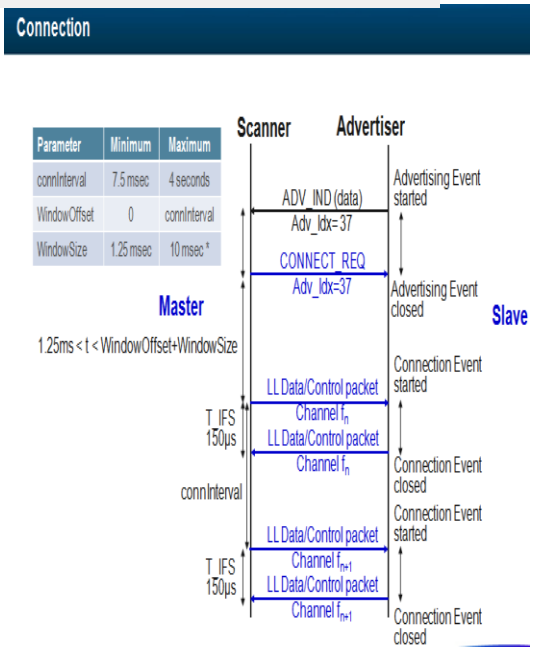
Frontline ComProbe® Family of Analyzers



Connection Interval







Introduction to Frontline Air Interface Sniffer: BPA 500

Introduction to Hello Bluetooth Profile

Bluetooth Low Energy Air Interface packet structure

Analyzing trace for “Hello Bluetooth” custom profile

DEMO: Hello Bluetooth and DecoderScript

Q&A

Hello Bluetooth Profile

- Education Profile – Demonstrate creating custom profiles
- Creating a Custom Profile Process
  - Step 1: Articulating Use Case
  - Step 2: Identifying Characteristics
  - Step 3: Defining Services
  - Step 4: Defining Profile
  - Step 5: Generating Attribute Table
- Hello Bluetooth XML representation

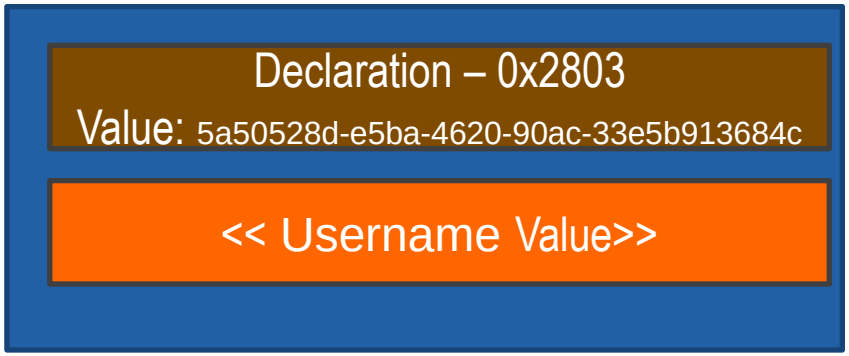
STEP 1: Articulating a Use Case

- Bluetooth Enabled Business Cards
- Bluetooth Enabled Registration desk
- Use Case
  - Person walks to the registration desk.
  - Based on proximity (measured by RSSI), business card establishes a connection
  - Exchange information (Name)
  - “Welcome to <Confernece> 2012, Name”

18

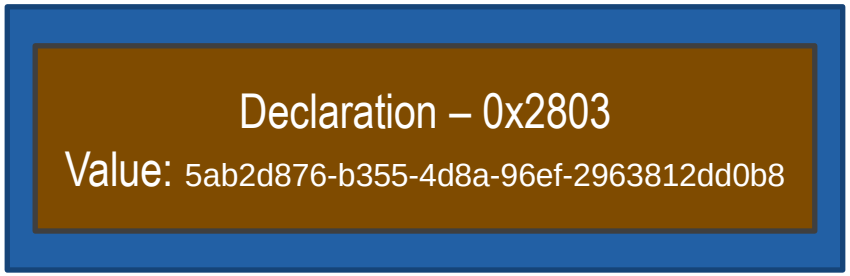
Step 2: Identifying Characteristics

- Characteristics are defined attribute types that contain a single logical value.
- Characteristic: <<Username>>.
- UUID Generator: 128 bit characteristics UUID - 5a50528d-e5ba-4620-90ac-33e5b913684c
- Permissions: Read
- Size: utf-8 string



Step 3: Defining Service

- Services are collections of characteristics and relationships to other services that encapsulate the behavior of part of a device.
- Service: <<Hello Service>>
- UUID Generator: 128 bit characteristics UUID - 5ab2d876-b355-4d8a-96ef-2963812dd0b8
- Characteristic: <<Username>> - mandatory



Step 4: Profile

- Profiles are high level definitions that define how services can be used to enable an application or use case.
- Roles
  - Hello Server
  - Hello Client
- Connection Parameters
  - Connection Interval: 80 msec
  - Slave Latency: 0
  - Supervision Timeout: 2 seconds
- Hello Client Behaviors
  - If RSSI value > threshold, read <<name>> , display “Welcome to AHM, <<name>>
  - If RSSI value < threshold, clear the display

Attribute Table Example – Hello Server

Handle	Attribute Type	Value	Permissions
0x00030	«Primary Service Declaration» 0x2800	«Hello Service» 5ab2d876-b355-4d8a-96ef-2963812dd0b8	R
0x00031	«Characteristic Declaration» 0x2803	{r, 0x0003, «User Name»}	R
0x00032	«User Name» 5a50528d-e5ba-4620-90ac-33e5b913684c	“Muhammad”	R

Hello Bluetooth XML representation

- GATT schema
  - http://schemas.bluetooth.org
  - http://schemas.bluetooth.org/Documents/profile.xsd
  - http://schemas.bluetooth.org/Documents/service.xsd
  - http://schemas.bluetooth.org/Documents/characteristic.xsd
- GATT based profiles are represented in xml files which follow the rules specified by the xsd files
  - HelloBluetoothProfile.xml
  - HelloService.xml
  - Username.xml

Introduction to Frontline Air Interface Sniffer: BPA 500

Introduction to Hello Bluetooth Profile

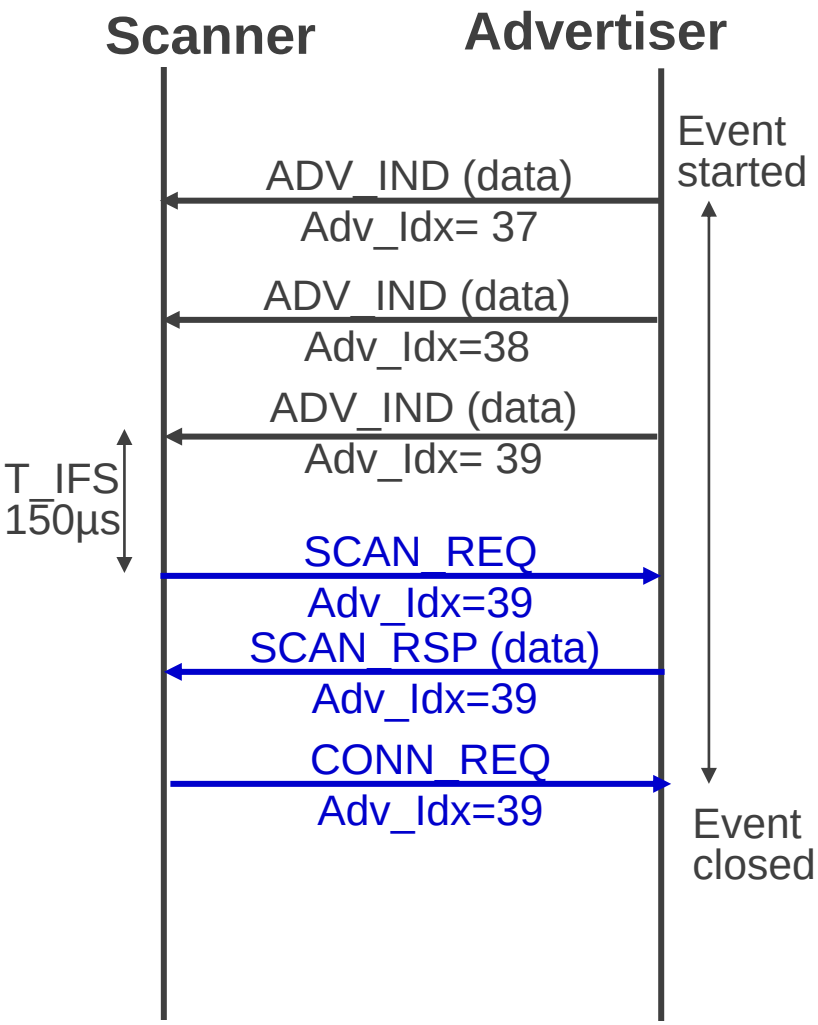
Bluetooth Low Energy Air Interface packet structure

Analyzing trace for “Hello Bluetooth” custom profile

DEMO: Hello Bluetooth and DecoderScript

Q&A

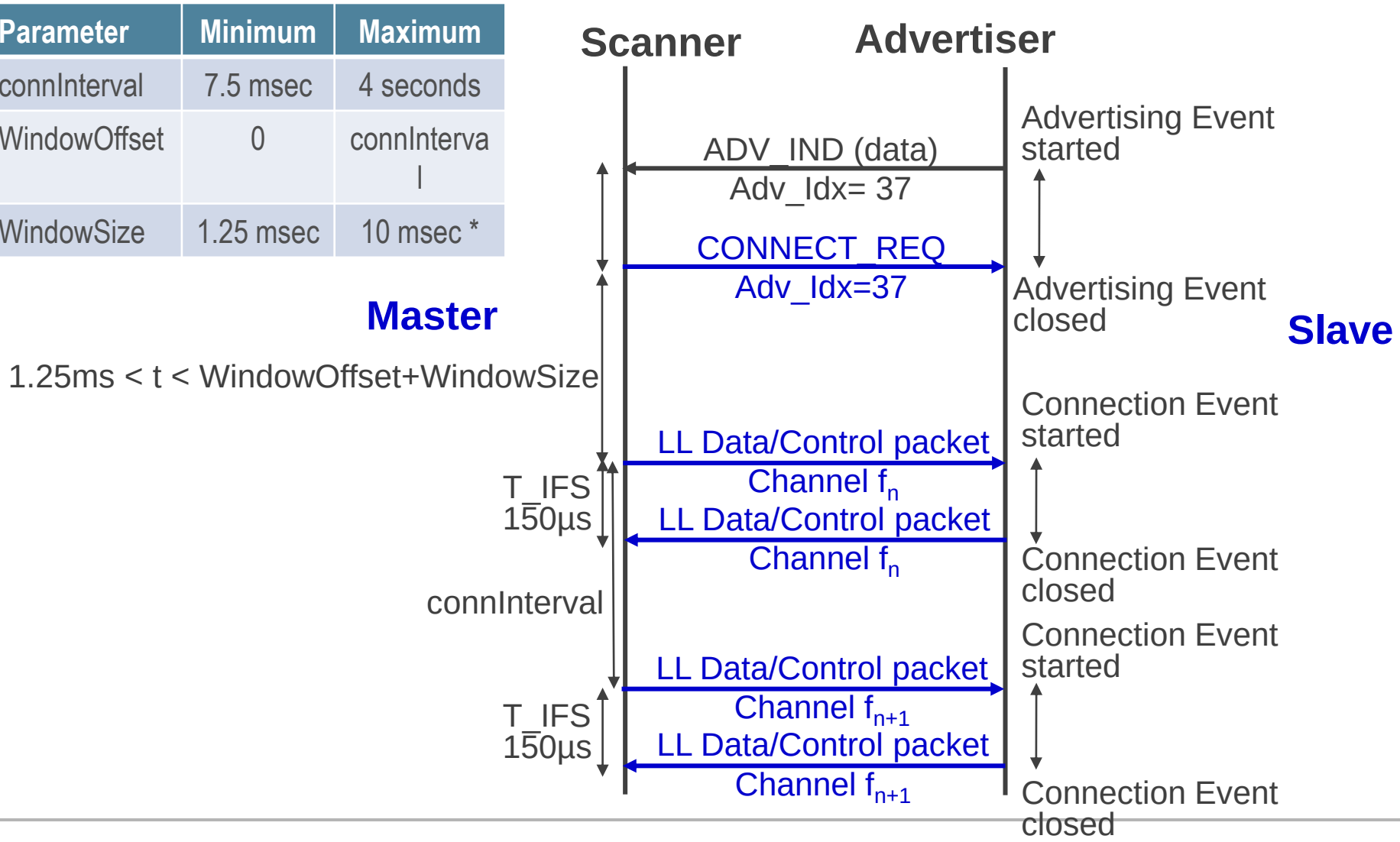
Device Discovery & Connection



- Advertiser sends ADV_IND on all three channels (37,38,39)
- Scanner is only listening on channel 39
- Device Discovery consist of
  - SCAN_REQ
  - SCAN_RSP
- CONN_REQ specifies
  - Connection interval
  - Slave Latency
  - Supervision Timeout
  - Channel Map
  - Hopping sequence

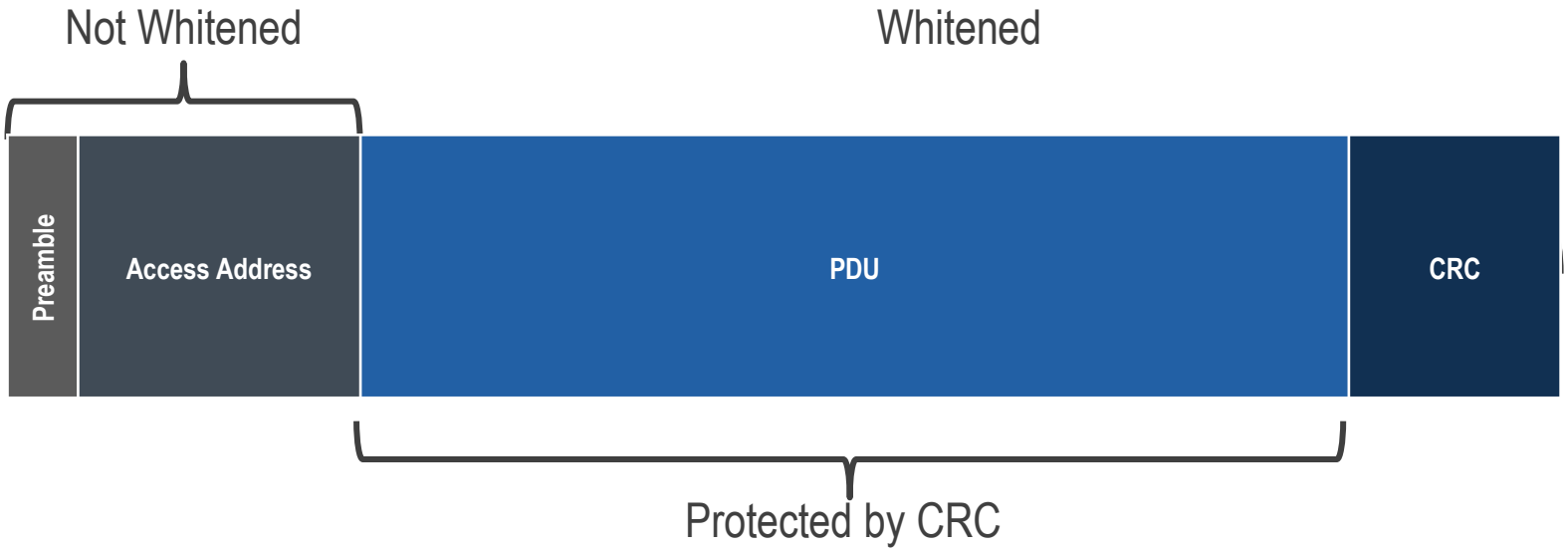
Data Transfer

Parameter	Minimum	Maximum
connInterval	7.5 msec	4 seconds
WindowOffset	0	connInterval
WindowSize	1.25 msec	10 msec *



One Packet Format

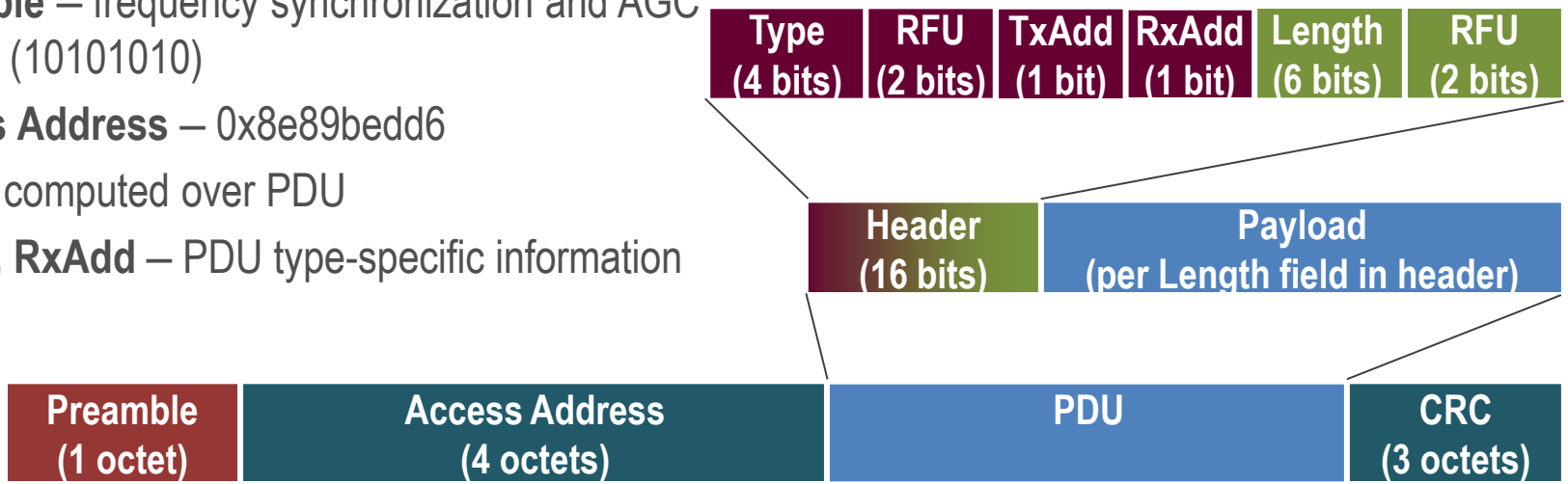
- Used for Advertising and Data Channel Packets
- Preamble (0x55, 0xAA)
  - Frequency synchronization, symbol timing estimation, AGC training
- Access Address
  - Advertising packets – always 0x8e89bed6
  - Data packets – different for each link layer connection
- Packet Data Unit
  - Defined based upon packet types



Air Interface Packets – Advertising Packets

Type	Packet	Usage
0000	ADV_IND	Connectable undirected advertising event
0001	ADV_DIRECT_IND	Connectable directed advertising event
0010	ADV_NONCONN_IND	Non-connectable undirected advertising event
0011	SCAN_REQ	Scan request for further information from advertiser
0100	SCAN_RSP	Response to scan request from scanner
0101	CONNECT_REQ	Connect request by Initiator
0110	ADV_DISCOVER_IND	Discoverable undirected advertising event

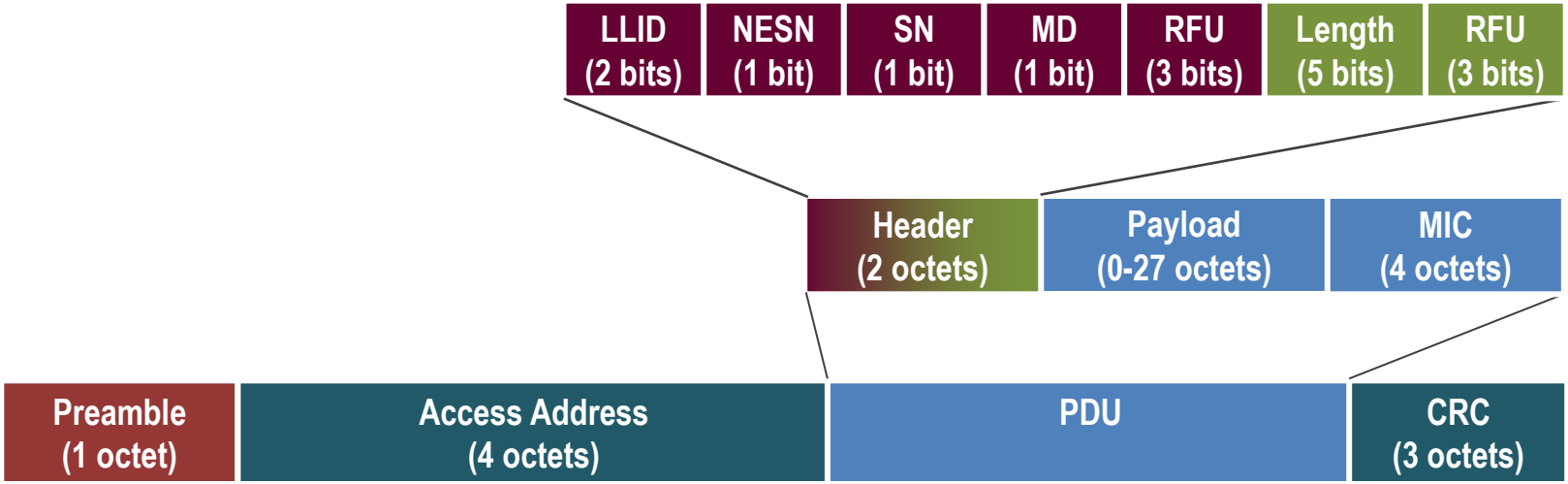
- **Preamble** – frequency synchronization and AGC training (10101010)
- **Access Address** – 0x8e89bedd6
- **CRC** – computed over PDU
- **TxAdd, RxAdd** – PDU type-specific information



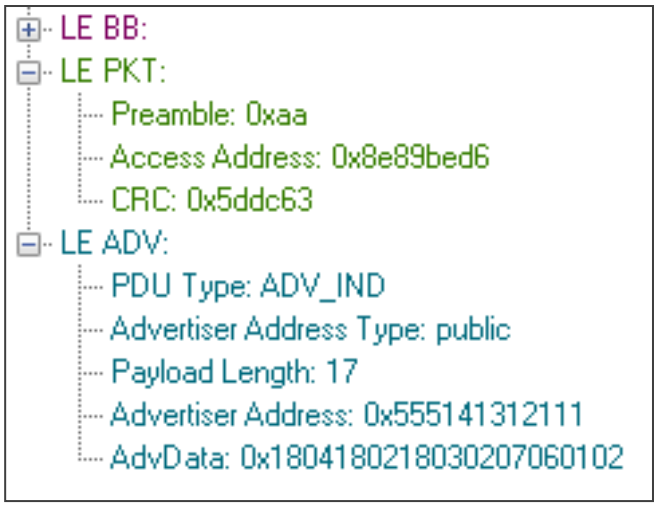
Air Interface Packets – LL Data Channel

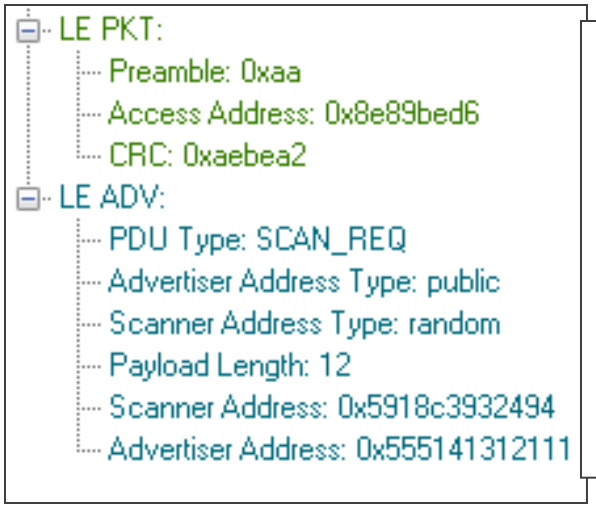
Field	Purpose and Encoding
LLID	**0x01** = Continuation/empty L2CAP packet **0x02** = Start of an L2CAP packet **0x03** = LL Control packet
NESN	Next Expected Sequence Number
SN	Sequence Number
MD	More data

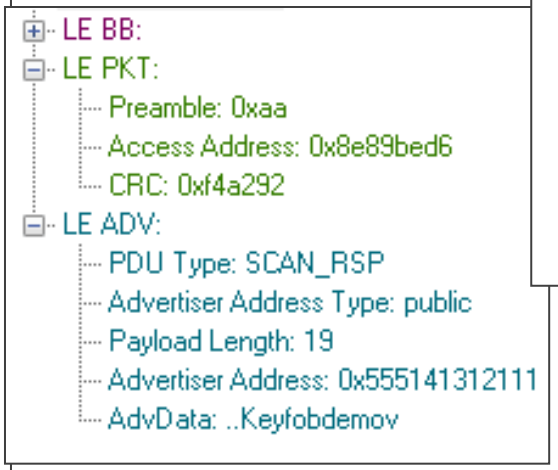
- **Preamble** – frequency synchronization and AGC training (`01010101`) or (`10101010`)
- **Access Address**– 32 bit link layer connection access address
- **CRC** – computed over PDU
- **MIC** – Message Integrity Code, for use with encrypted links

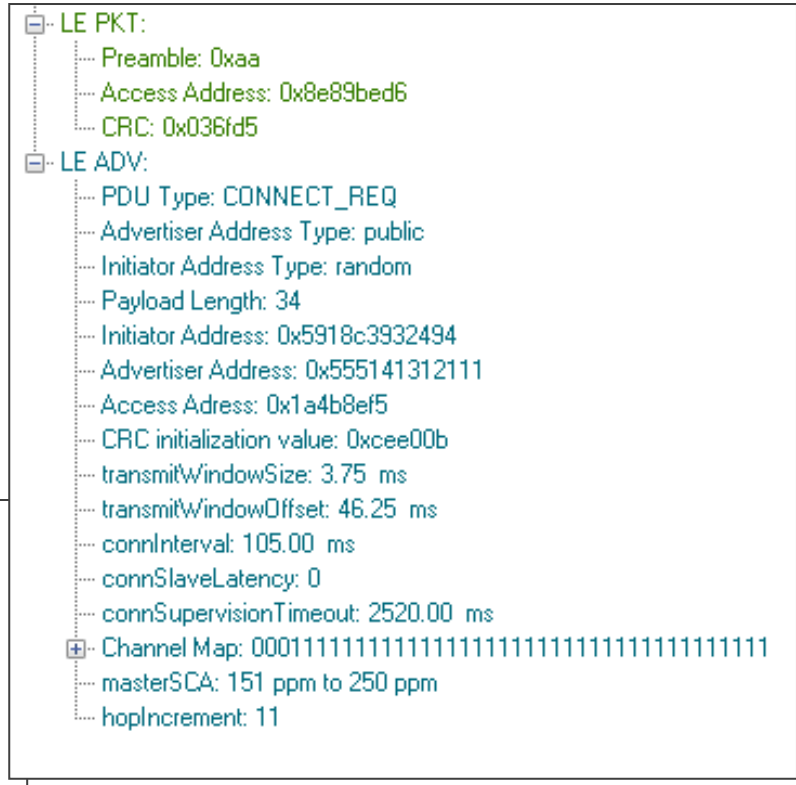


Example – Advertisement Packet

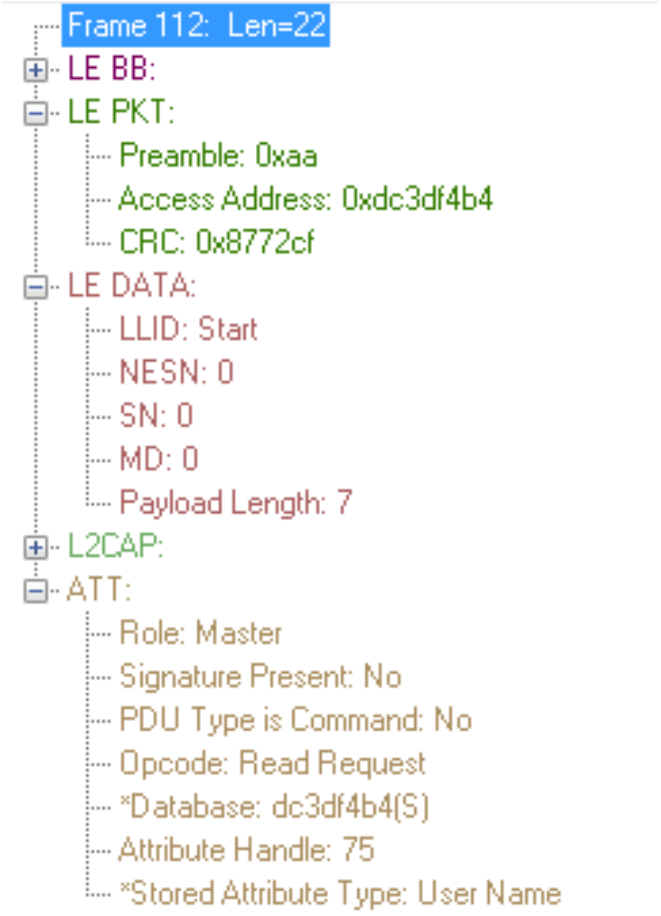


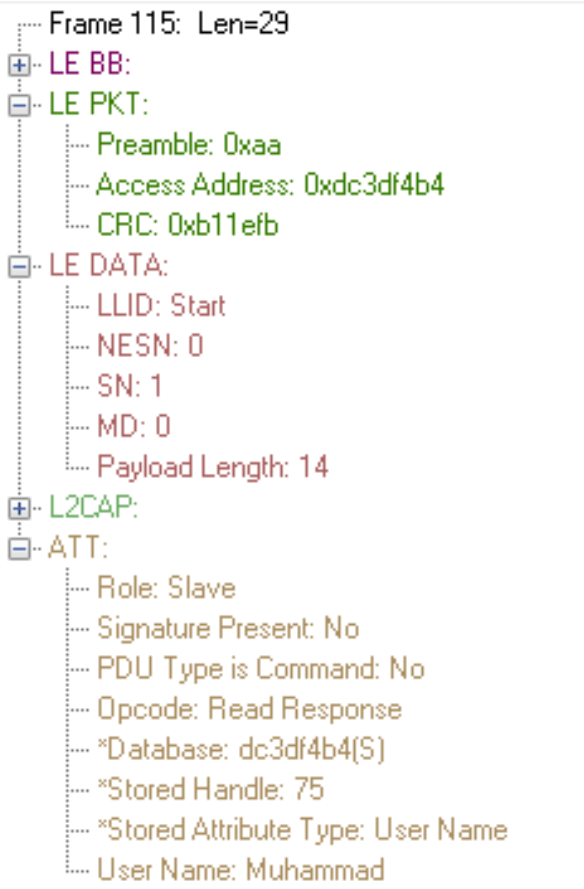






Example – Data packets





Introduction to Frontline Air Interface Sniffer: BPA 500

Introduction to Hello Bluetooth Profile

Bluetooth Low Energy Air Interface packet structure

Analyzing trace for “Hello Bluetooth” custom profile

DEMO: Hello Bluetooth and DecoderScript

Q&A

GATT vs. ATT

- GATT is a profile
- ATT is a protocol

GATT Heirarchy

- GATT is a collection of services. Everything is a service.
  - Services may contain other services.
  - Services contain characteristics.
    - Characteristics are chunks of data that can be written and read.
    - Characteristics may have descriptors. Descriptors define or modify the behavior of characteristics.

GATT Database

- A database is a list of GATT Services.
- Every device has a database.
- During discovery, each device will ask a series of questions
- Each device builds a copy of the other devices database.

Handles vs. UUIDs

- A UUID is simply an abstraction of a data type. Some examples:
  - 0x1808 = Glucose (service)
  - 0x2A24 = Model Number (characteristic)
  - 0x27A1 = Parsec (unit of measure)
- A handle is an abstraction of the location of a piece of data.

Handles are arbitrary

- There are a few rules about which services use which handles or how they use them.
  - All characteristics must be within the handle range reserved by the service.
  - Descriptors must follow the characteristics they are describing
- Services need not be contiguous
- Characteristics need not be contiguous
- It’s entirely possible for both parties in a conversation to have the same service running at different handles.
- It’s entirely possible for parties in the conversation to have different services running at the same handle.

Where do I find the UUIDs

- http://www.Developer.bluetooth.org is an excellent site managed by the Bluetooth SIG. It contains not only the UUIDs for all services, characteristics, and descriptors but the latest layouts. It should be considered the reference during development.

- Frontline is current with this website and often just a little ahead of it.
- If you find a disagreement between the developer portal and the Frontline decode:
  - Check the decoder version by floating your cursor over the ATT tab. The version will be displayed as a date related to the developer portal. You might need an upgrade.
  - If you are up to date, please report it as a bug.

Discovery

- Four kinds of discovery:
  1. Primary Service
  2. Relationship (Included Service)
  3. Characteristic
  4. Descriptor
- We will cover Primary Service and Characteristic Discovery.

Real Work

- Three kinds:
  - Read/Write — Just like it sounds. You can read and write to a value
  - Indication — A characteristic can be instructed to push data to you
    - Each Indication must be acknowledged
  - Notification — A characteristic can be instructed to push data to you
    - No Acknowledgment

- A variation of read/write is Control. You are reading and writing to a ‘Control Point’ to start, stop or modify some sort of action or status.
- Example: You might use a control point to turn on an indicator light.

A note about sniffing

- The sniffer must capture the discovery process.

- Otherwise there is no way to map a handle back to a UUID and decode the data.

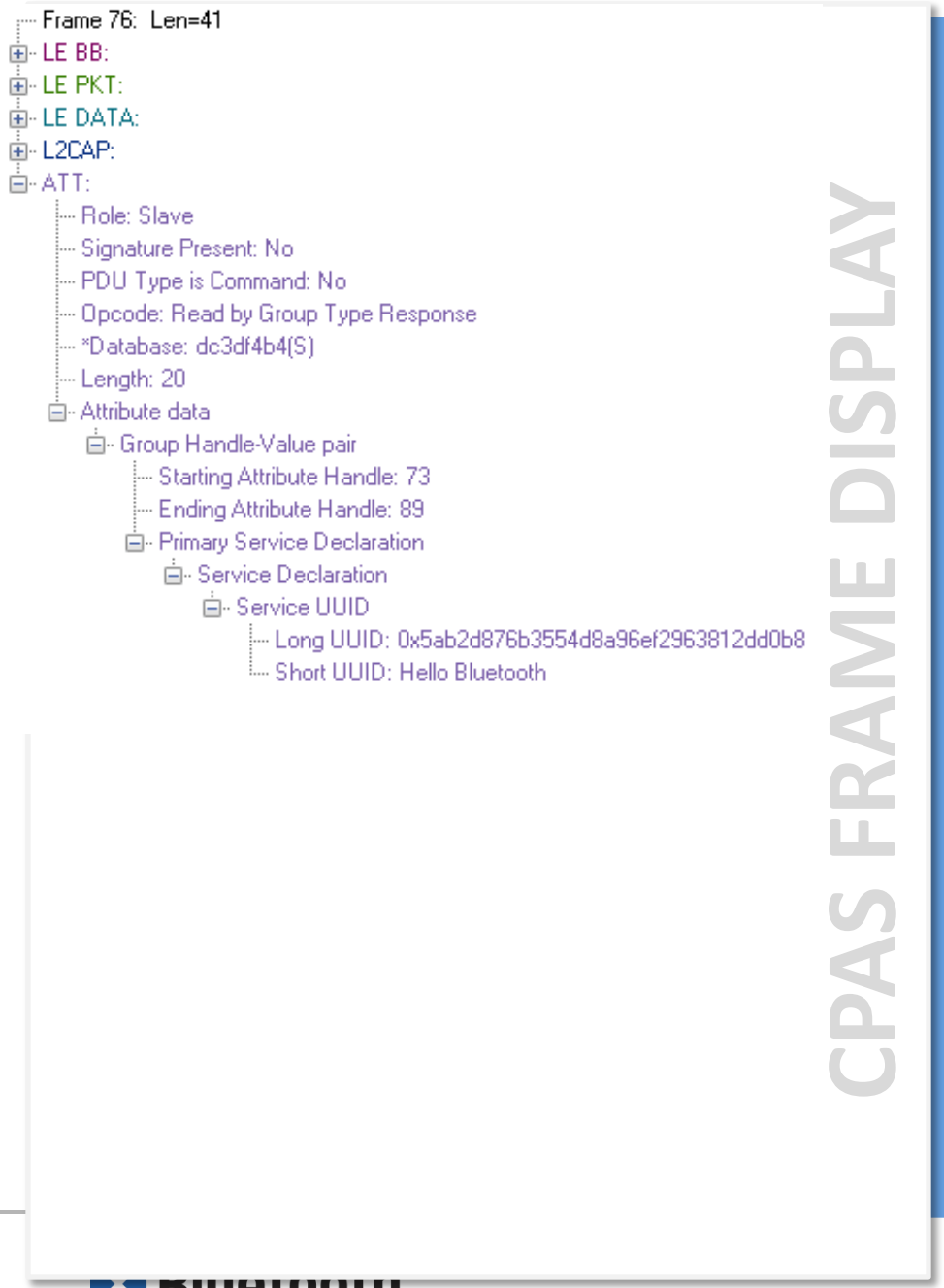
Discover Primary Services

- Core Document 4.0
- Volume 3
- Part G
- Section 4.4

- Frame 71: Len=26
- LE BB:
- LE PKT:
- LE DATA:
- L2CAP:
- ATT:
  - Role: Master
  - Signature Present: No
  - PDU Type is Command: No
  - Opcode: Read by Group Type Request
  - *Database: dc3df4b4(S)
  - Starting Attribute Handle: 45
  - Ending Attribute Handle: 65535
  - Attribute Group Type: Primary Service

CPAS FRAME DISPLAY

- **Read by Group Type Request: "Give me a list of every Primary Service you have".**

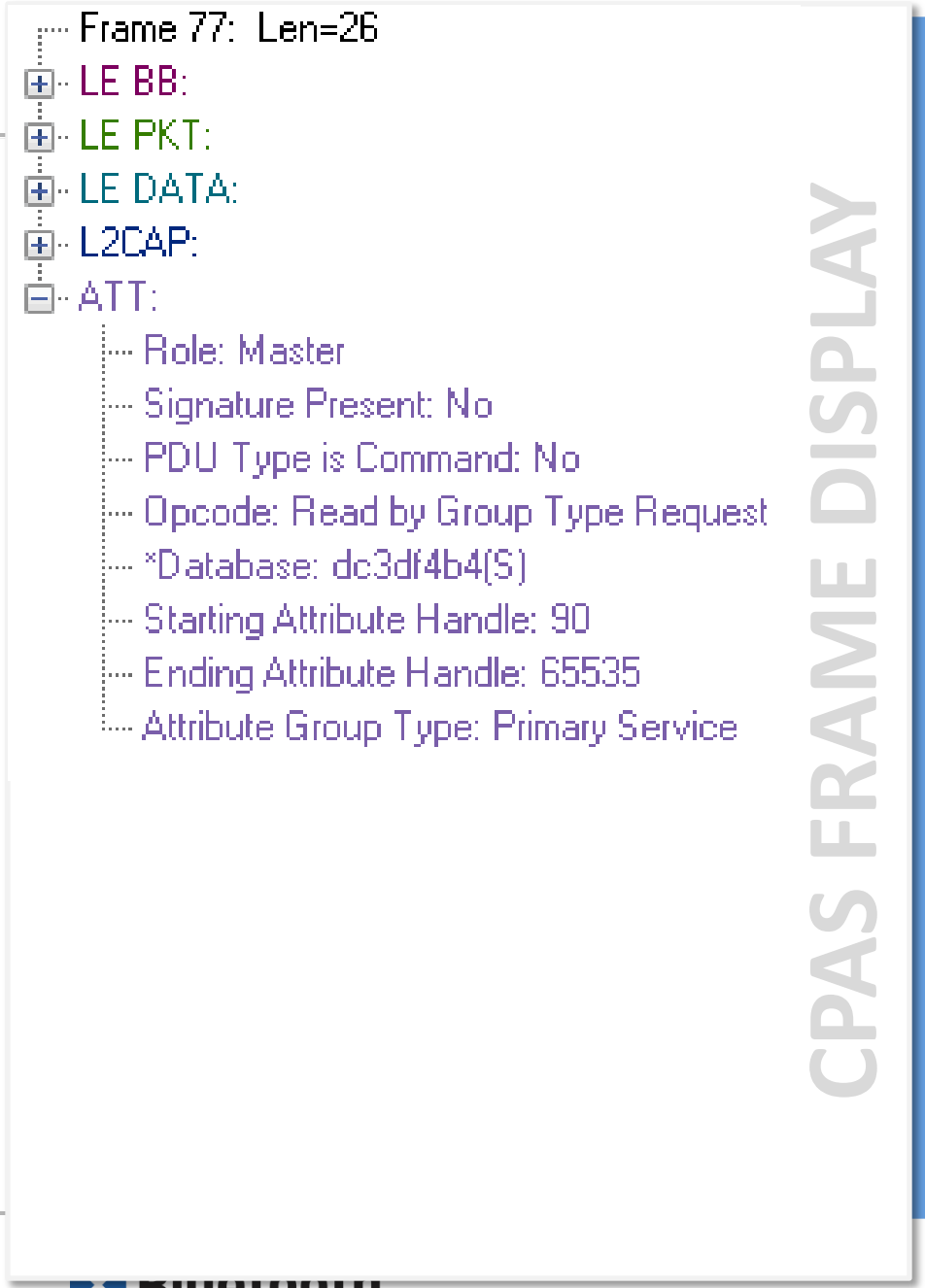


CPAS FRAME DISPLAY

- ***Read by Group Type Response: "I have a Service and here it is with its starting and ending handle".***

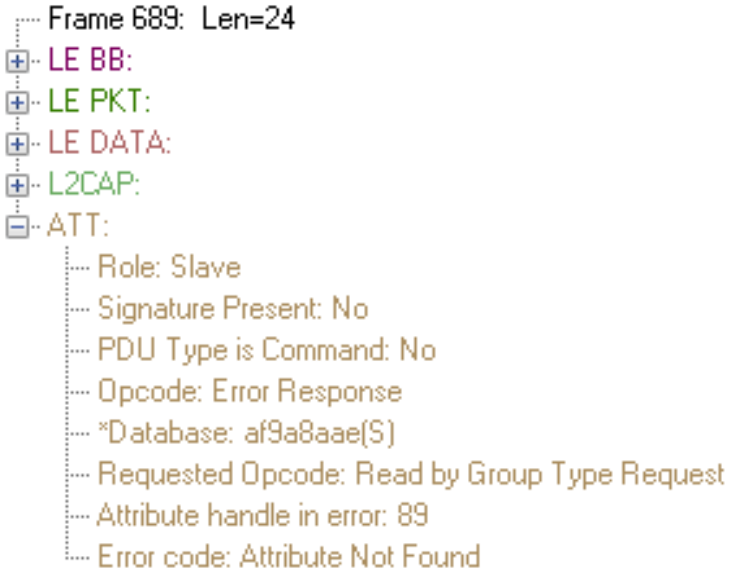
73	Hello Bluetooth
.	
.	
.	
.	
.	
89	

- ***Handles don't have to be contiguous***
- ***Services can be in any order.***
- ***These may not be all of the services***



- **Read by Group Type Request: "Give me a list of every Primary Service you have starting at handle 90".**
- **Start is set to end of last service**

73	Hello Bluetooth
.	
.	
.	
.	
.	
89	



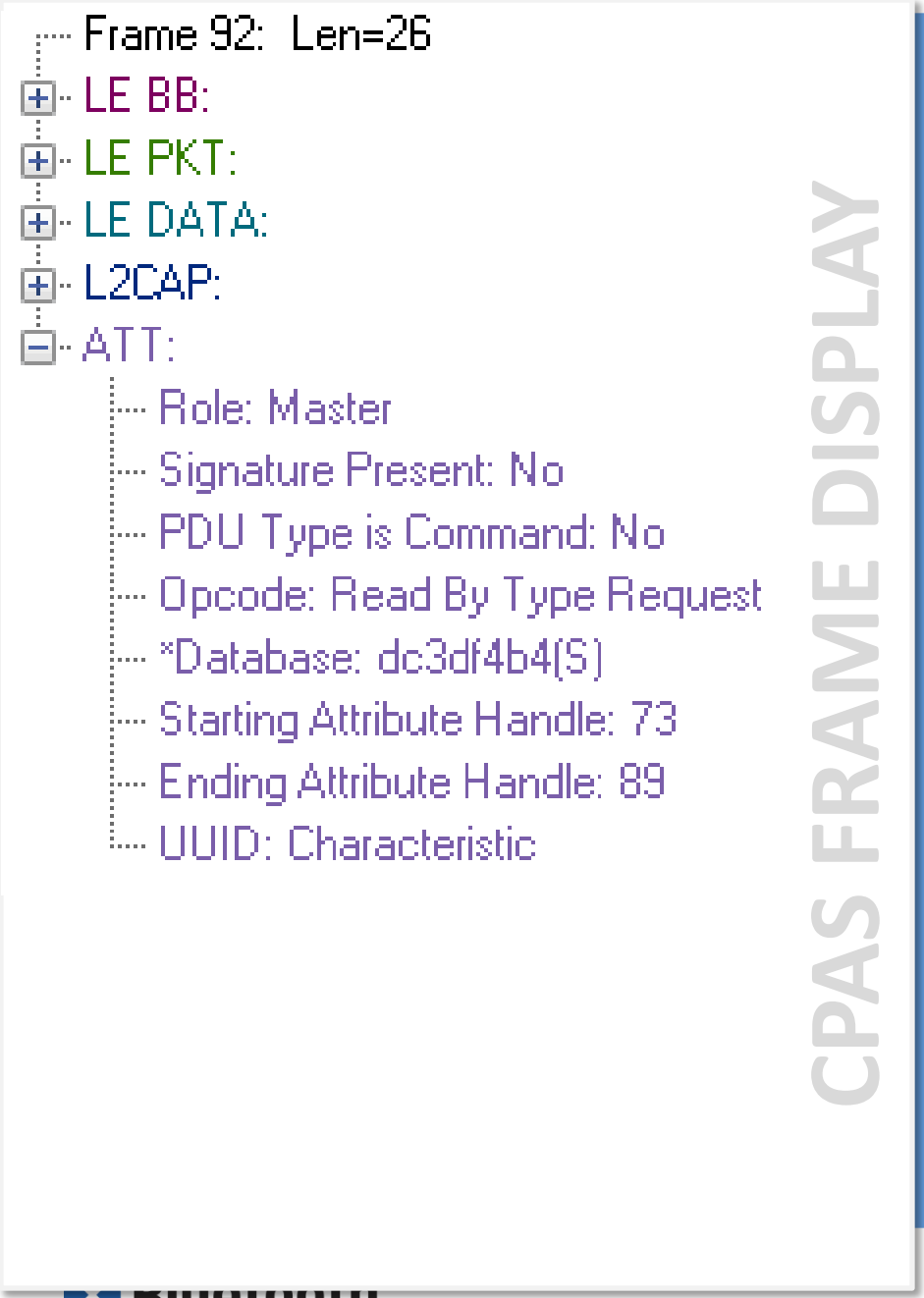
CPAS FRAME DISPLAY

- ***Read by Group Type Response: "I have no more Primary Services".***
- **Lack of Context. Response does not restate the question.**

73	Hello Bluetooth
.	
.	
.	
.	
.	
89	

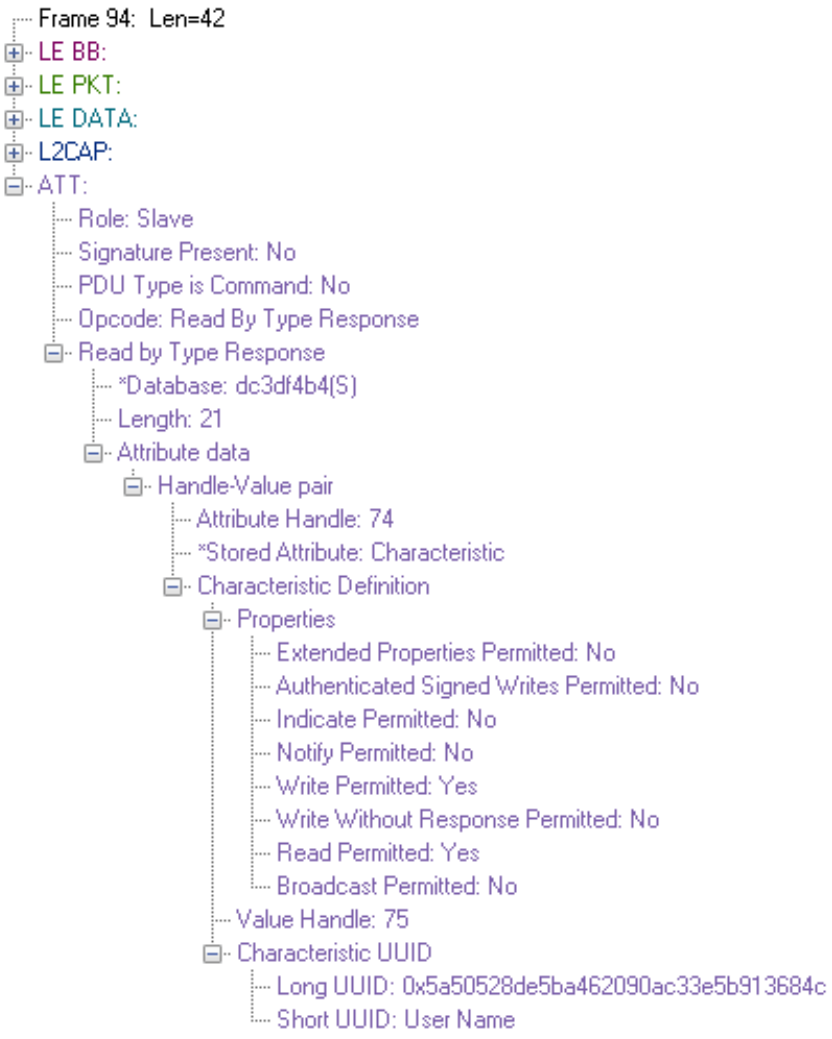
Discover Characteristics

Core Document 4.0
Volume 3
Part G
Section 4.6



- **Read by Type Request: "Give me a list of all characteristics between handles 73 and 89 inclusive".**
- **We do this process service by service.**

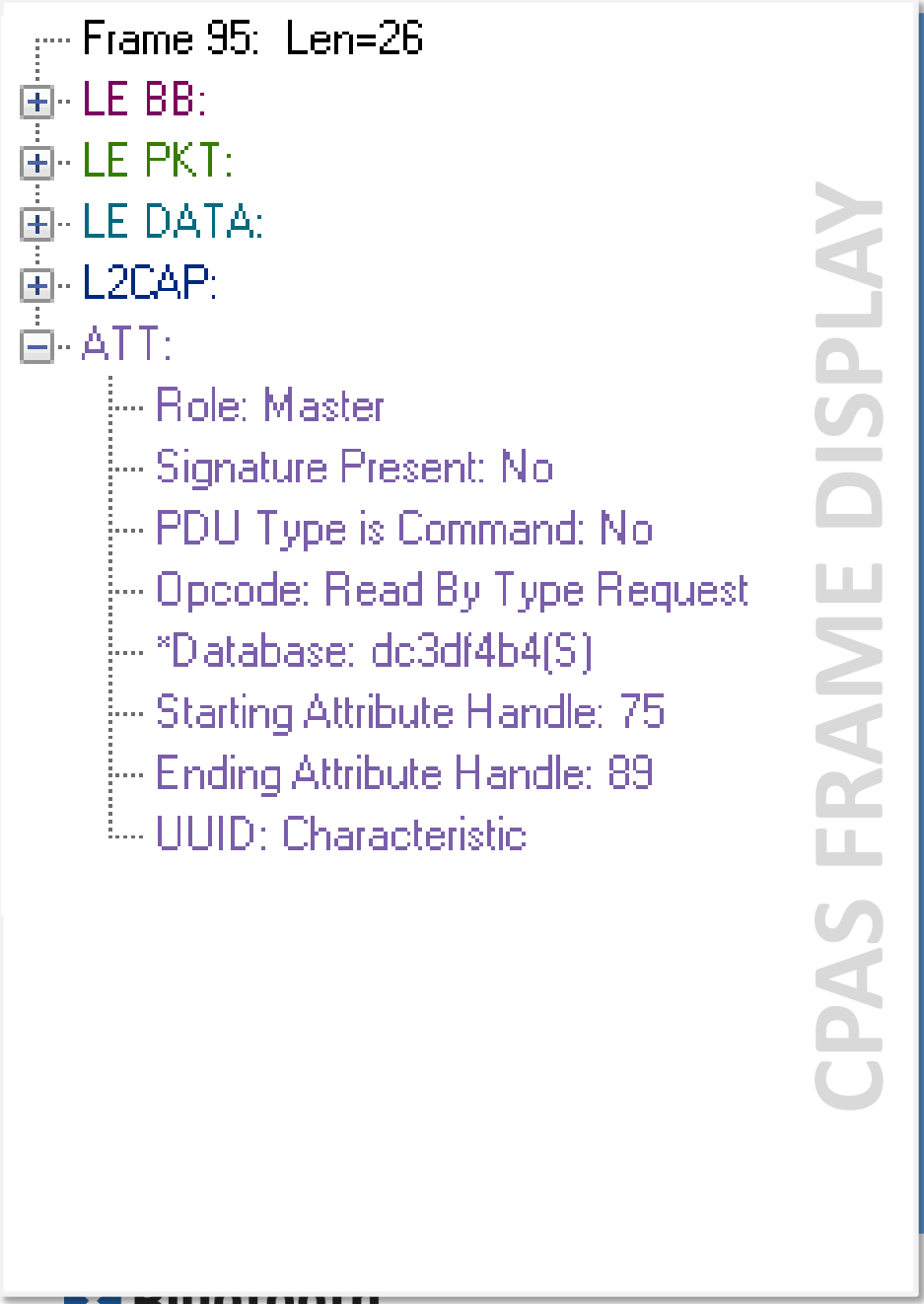
73	Hello Bluetooth
.	
.	
.	
.	
.	
89	



CPAS FRAME DISPLAY

- **Read by Type Response: "I have a characteristic at handle 74 that contains a value at handle 75."**

73	Hello Bluetooth
74	Characteristic
75	User Name
89	



CPAS FRAME DISPLAY

- Read by Type Request: “Give me a list of all characteristics between handles 75 and 89 inclusive”.
- Why is the starting handle not 74? Spec says that the start should be the last attribute handle + 1.
- The last characteristic was at 74 so we add 1 to that.

73	Hello Bluetooth
74	Characteristic
75	User Name
89	

- Frame 103: Len=24
- LE BB:
- LE PKT:
- LE DATA:
- L2CAP:
- ATT:
  - Role: Slave
  - Signature Present: No
  - PDU Type is Command: No
  - Opcode: Error Response
  - *Database: dc3df4b4(S)
  - Requested Opcode: Read By Type Request
  - Attribute handle in error: 75
  - Error code: Attribute Not Found

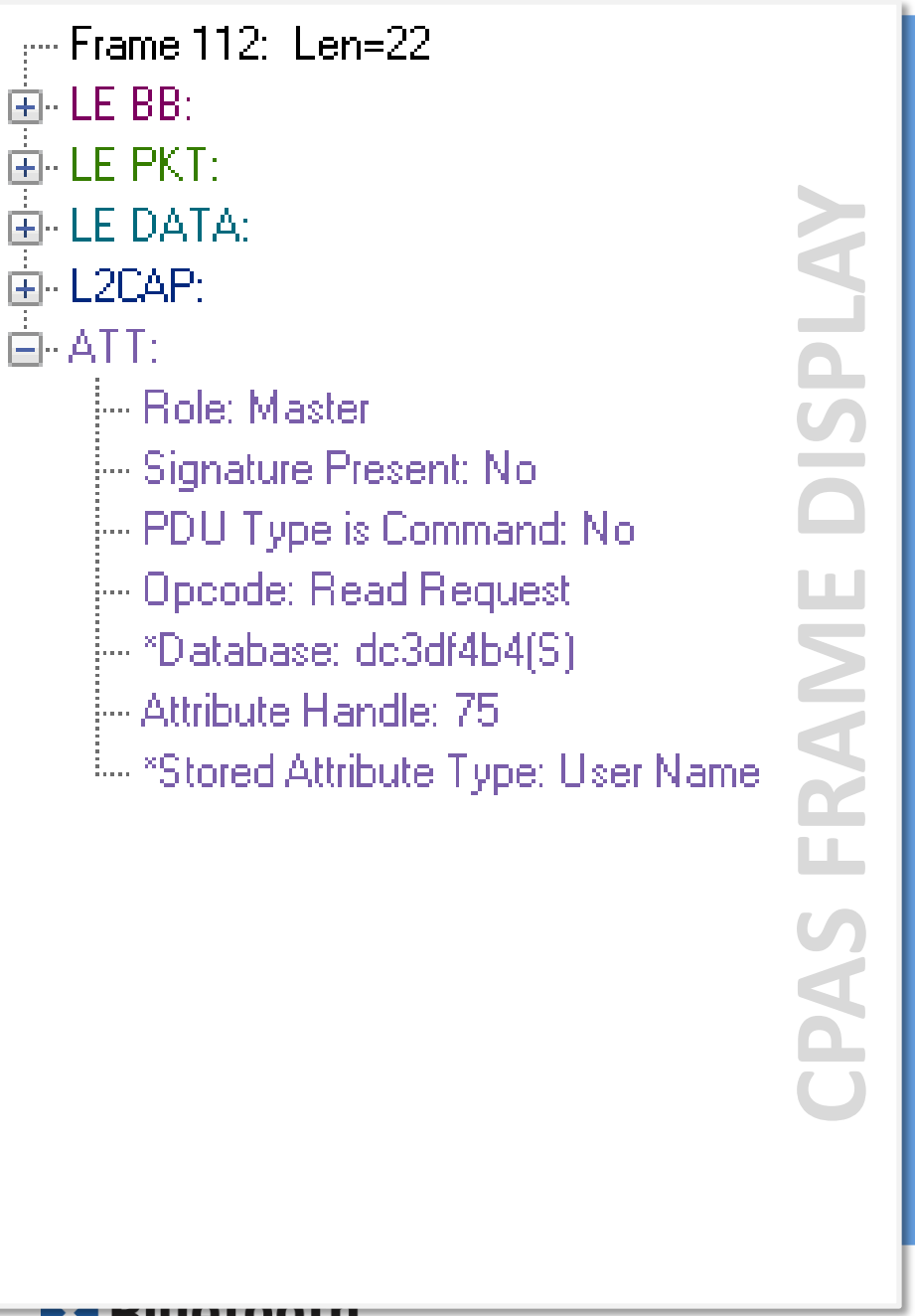
CPAS FRAME DISPLAY

- Read by Type Response: "No more"

73	Hello Bluetooth
74	Characteristic
75	User Name

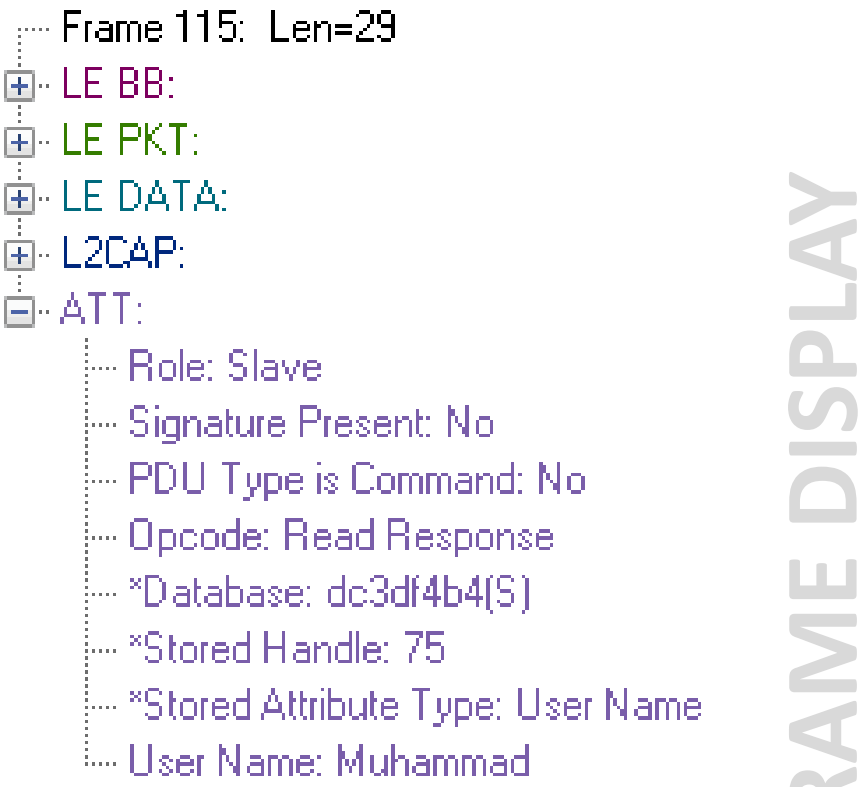
Doing Real Work

Read Data



- **Read Request: Show me the contents of handle 75.**
- **We don't use UUIDs, just handles.**
- **The items with the asterisk are bits of data that Frontline remembers from the discovery process.**

73	Hello Bluetooth
74	Characteristic
75	User Name



CPAS FRAME DISPLAY

- **Read Response: The contents is…**
- **The response does NOT contain the reference to handle 75. You must remember that.**
- **All it returns is contents.**
- **The Frontline product remembers it and presents it for your convenience.**
- **The Frontline product remembers not only the requested handle but maps the descriptor back to the characteristic it is describing.**

73	Hello Bluetooth
74	Characteristic
75	User Name

We’ve covered the basics

There are more commands in ATT

Introduction to Frontline Air Interface Sniffer: BPA 500

Introduction to Hello Bluetooth Profile

Bluetooth Low Energy Air Interface packet structure

Analyzing trace for “Hello Bluetooth” custom profile

DEMO: Hello Bluetooth and DecoderScript

Q&A

Decoding Custom Profiles – DecoderScript

- Powerful and easy to use – a few lines of code and you're up an running
- Installed with Frontline ComProbe software
  - C:\Program Files (x86)\Frontline Test System II\Frontline ComProbe Protocol Analysis System xx.xx.xx.xx\Development Tools\DecoderScript Manual.pdf
- We'll be at the Shanghai UPF. Feel free to drop by and discuss your custom decoder needs.

Introduction to Frontline Air Interface Sniffer: BPA 500

Introduction to Hello Bluetooth Profile

Bluetooth Low Energy Air Interface packet structure

Analyzing trace for “Hello Bluetooth” custom profile

DEMO: Hello Bluetooth and DecoderScript

Q&A

Get Started / More Information

- Learn more about Frontline tools for ***Bluetooth*** **low energy** (and more!) – www.fte.com/bluetooth
- Interested in creating your own custom profiles? Install our ComProbe software (free to download) and use the included **Decoder Script** tools to make your own custom profiles!
- If you don't already have it download the **ComProbe Protocol Analysis System** software from the Frontline website – www.fte.com/getbpa500
- Coming to UPF 42? Come to **Frontline's "hands on" training session** where you can see first hand our tools in action.


Bluetooth®
SPECIAL INTEREST GROUP