embedded world 2015 Exhibition&Conference ... it´s a smarter world

Streaming speech and music using Bluetooth Low Energy

Marcel Meli, Olivier Rion
Zurich University of Applied Sciences
Institute of Embedded Systems
Winterthur, Switzerland
Marcel.Meli@zhaw.ch

Abstract— The new Bluetooth Low Energy standard has mostly been promoted and used for applications where low data rate is needed. For cases that require more bandwidth, Bluetooth Classic is still the preferred and marketed solution. A typical example is the streaming of music. In view of the popularity of Ble and its physical layer data rate, it is legitimate to investigate its practical throughput limits. To this purpose, we looked at different ways of using Ble transceivers to transmit data as fast as possible. We built a demonstrator that allows us to emulate several frame variations in order to improve the data rate for more demanding systems. The unconventional use of MIC positions for data allows us to be more efficient in transferring data. In this paper we summarize the first results of an on-going work.

Keywords—Blutooth Classic; wireless systems; Bluetooth Smart; Audio; codec;

I. INTRODUCTION

The Bluetooth Low Energy (Ble) standard introduced a few years ago has seen a rapid acceptance by the market. There are several applications that allow sensors to be connected with smartphones and other consumer devices. Bluetooth low energy (now marketed as Bluetooth Smart) complements Bluetooth Classic by making available to the user a wireless standard that requires less energy. The reduction in energy needs is made possible by a certain number of compromises, including a smaller frame size. The potential of Ble has already led to its adoption by several IC manufacturers. This will probably result in an improvement of performance and a price war that can only benefit the end user. It is likely that Ble ICs will be used in markets and products they were not foreseen for, thanks to cost and availability. For some applications, it could mean taking Ble to its limits, also as far as data throughput is concerned. It could also mean using the transceivers in proprietary modes, when compatibility is not required. A proprietary mode can allow the reduction of the overhead needed by Ble and thus increase the useful data rate.

Although Ble is not foreseen for high data rates, it is legitimate to ask questions about its limitations.

How fast can one transfer data between applications that communicate with Ble device?

- If one remains within the standard
- If one makes compromises on certain layers
- Are there other ways to increase the throughput?

In December 2014, the BT SIG announced the release of version 4.2 [2].That version allows larger data frames to be sent, which improves the communication efficiency of the system. It is likely that the new version will progressively take over. However, for the coming years, version 4.0 and version 4.1 will still be used in products. In this work we considered only version 4.0 and version 4.1. THROUGHPUT LIMITS

How much “useful data” can be sent using a Ble transceiver? Since there is a bit rate imposed by the physical layer (1 Mb/s) the answer depends on other parameters.

- The level (layer) at which the data is taken out and used by the application
- The ability of the embedded system to deal with the amount of data (process and react fast enough to avoid delays leading to loss of bandwidth)
- The number of retransmissions due to packet errors during the communication.

The resource problem is related to the devices that are used and to the efficiency of the Ble stack that is implemented. We will not deal with that and assume that appropriate devices are

used. Similarly, we will assume that there are no errors during the communication between physical layers. Consequently, we will only look at the effect of the stack layers on the throughput.

Fig. 1 shows the different layers and the useful data / overhead at each stage. In references [6,7,8] information can be found about the way throughput is calculated.

A. *Fully compatible with Ble*

In a fully Ble-compatible system, data will be transmitted between 2 parties with acknowledge frames. The communication is done in slots with clearly defined timings. The Inter Frame Spacing (IFS) is at least 150µs. Assuming that the connection intervals are optimally filled with frames from one of the communication parties, we arrive at the following:

1) *Unidirectional communication*

Source transmits with maximum data size and sink receives. Sink transmits only minimum acknowledge frames (10 bytes). The timing overhead is 300µs [4,5,6,7,8].

- Using notifications: The maximum useful data size is 20 bytes and the overhead is 17 bytes. This leads to a maximal throughput of 236.7 kb/s (29.6 kB/s)
- Using read blob: The maximum useful data size is 22 bytes and the overhead is 15 bytes. This leads to a maximal throughput of 260.3 kb/s (32.5 kB/s). This value must however be reduced, because of the necessity of sending extra commands.

2) *Bidirectional*

Both parties transmit using maximal data size. The acknowledgement is included in the data frames.

- Using notifications: The useful data is 20 bytes for each device. The overhead in frame is 17 bytes for each device. The maximum throughput in each direction is:179.3 kb/s (22.4kB/s)
- Using read blob: The useful data is 22 bytes for each device. The overhead in frame is 15 bytes for each device. The maximum throughput in each direction is:197.3 kb/s (24.6kB/s). As above, this values must be reduced due to extra commands.

B. *Partially compatible frames*

It is possible to use reserved values in some of the Ble stack layers parameters in order to increase the throughput. This can introduce some incompatibilities now or lead to possible incompatibilities in the future. The parameters can be so chosen as to minimize the compatibility problems. There are too many combinations possible. We only show a few possibilities here.

- Proprietary CID. By defining a proprietary CID, we spare 1 extra byte (opcode) that can be used for data.

  Use of the MIC bytes. By using the MIC positions for data and assuming that the application will not use (or will refuse) authentication and encryption, we make room for 4 extra bytes.

  Using those 5 bytes leads to useful packets of up to 27 bytes with an overhead of 14 bytes. This results in 305.1 kb/s (38.1 kB/s) in one direction.

- Changes related to the compatibility can be done in other layers, leading to an even greater departure from the standard. The patent application filled by Apple Inc [8] makes the difference in the header of the data PDU. With a proprietary header, 27 bytes are available for useful data. In that case, 10 bytes are needed for overhead. This results in an application data rate of 319.5 kb/s (39.9kB/s) in one direction.

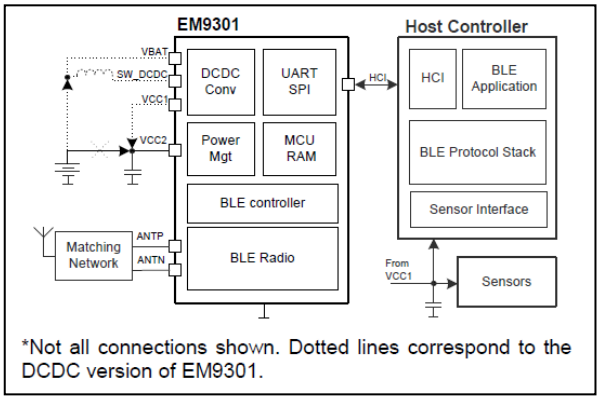
C. *Modifications at LL level*

With access to the link layer, one could define the size and timing of the frames to further increase the amount of useful information sent. A full frame can be sent with an IFS of 150µs. With and overhead of 2 bytes and 37 bytes of useful data, it is possible to achieve 562kb/s (70kB/s). The modifications are however so important that one could as well use a fully proprietary radio to obtain even better results.

The new version of Ble (4.2) allows larger frames to be sent. One can therefore expect a better efficiency, leading to better throughput values than those achievable with 4.0 and 4.1

III. A DESIGN FOR VERIFICATION

We designed a flexible system which allowed us to make verifications and measurements between parties transferring data using Ble. Fig. 2 shows the system. It includes Ble controllers, host microcontrollers, and an Android smartphone. Data generated by the Smartphone can be sampled and sent



over Ble to another device.

As Ble controller, we used the EM9301 [1]. That device communicates with the external host using the SPI/UART interface link. The host we used has the needed resources to generate and process the data that we want to send/receive (A/D, D/A). The host resources also allow an efficient processing of the communication frames. The EM9301 integrates the Physical and Link layers. The use of the HCI layer to communicate with the host gives the needed flexibility to handle the frames. The host uses a low-power Ble stack developed at ZHAW-InES. Our knowledge of the stack

allowed us to tune it and optimize it for different purposes. The communications parties are on one hand a smartphone sourcing data (for example audio data) and on the other hand a receiver that uses the data (for example sending it to a speaker).

We chose an android phone with Android Accessory functionality for several reasons. A very important one is that it gives us a better control since the stack runs externally, on the accessory board. This control is required to implement different communication use cases. The Ble controller is a peripheral on the accessory board that is connected to the smartphone using USB.

IV. POSSIBLE USES

The possible data rate calculated above show that streaming at more than 200kb/s is possible, providing that the communication parties have the needed resources to process the data. There are several applications that can make use of this amount of data. One of them is the streaming of audio. The needed data rate depends on the quality that is expected. By using the appropriate codecs, the amount of data to transfer can be reduced. However, resources must be made available on the host for the signal processing required by the codec. More information about the codecs and their use for low data rate can be found in the references [9,10].

Requirements for speech are not high; they can be met by Ble, with or without codec. For music, the sustained bit rate is higher and depends on the quality that is expected. There are different possibilities.

The SBC codec that is used in Bluetooth Classic can work with different bit rates. It will support up to 345kb/s. Some variations with lower data rate will clearly fit the data rate that is possible with Ble transceivers. The same can be said for MP3.

V. TESTS AND RESULTS

We tested several configurations to verify the data rate. We also configured the system to sample music from the smartphone and sent it to the sink device. The configurations and some of the results can be seen in the figures at the end of this document.

Using the setup described earlier, we generated and transmitted data from a source to a sink. We observed that at times, the data rate approached the expected values. But there are also “holes” where nothing is sent, which reduces the average data rate. The discrepancies that can be observed are often linked to the difficulties in filling up connection events fully. Some other discrepancies could be traced to possible software bugs.

A test for the audio was also made. A music sample was read from the smartphone, played and transmitted (28kB/s).

Several other applications were successfully implemented using the Android phone.

VI. FUTURE WORK

ZHAW-InES is currently investigating and testing a special configuration that is fully compatible with Ble (4.1) and that should allow more than 400kb/s of useful data. We are also investigating the tradeoffs between energy needed for compression and energy required for transmission/reception.

VII. CONCLUSIONS

With some of the current Ble controllers implementing (4.0 or 4.1) it is possible to reach data transfer throughput of more than 200kb/s, providing that the systems have enough computing resources to deal with the data. With some proprietary modifications at higher layers, one can approach 300 kb/s. Using an appropriate sniffer, we were able to verify such data rates on a setup using appropriate microcontrollers. This data rate is enough to allow the transfer of speech or the streaming of music compressed with some variations of codecs such as SBC or MP3.

REFERENCES

[1] Link to EM9301 datasheet. http://www.emmicroelectronic.com/sites/default/files/public/products/datasheets/9301_ds_0.pdf

[2] Bluetooth specifications https://www.bluetooth.org/

[3] M. Gysin, A. Müller, A. Rüst; ”Achievable Data Rates on Simultaneously Connected Bluetooth Low Energy Devices”, Proceedings of Wireless Congress 2014, Munich.

[4] Deepti Gill ; Neeti Panchal & Leena, “Low Energy Bluetooth”, International Journal of Research (IJR) Vol-1, Issue-10 November 2014

[5] A. Rüst, M. Gysin, A. Müller, M. Würms, " Simultaneously connecting devices through Bluetooth Smart”, Embedded World Conference, February 2014"

[6] C. Gomez, I. Demirkol, J. Paradells, “Modeling the Maximum Throughput of Bluetooth Low Energy in an Error-Prone Link”, IEEE Communications Letters, VOL. 15, NO. 11, NOVEMBER 2011

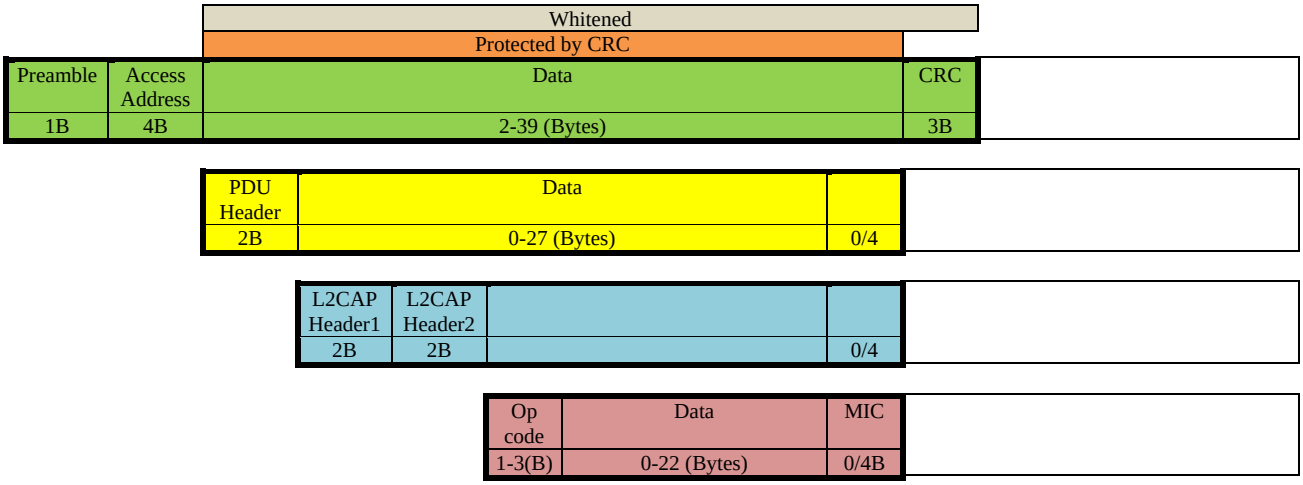
[7] C. Gomez, J. Oller, J. Paradells, “Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology”, Sensors 2012, 12, 11734-11753

[8] J. Linde, B. J. Tucker,; Apple Inc “Audio transfer using the bluetooth low energy standard. Patent” Patent application US20130045684 A1

[9] M. Meli, M. Gysel, Marc Sommerhalder “Using IEEE 802.15.4 / ZigBee in audio applications”, Embedded World, Nuremberg February 2006

[10] Comparison of different Audio Codecs. http://en.wikipedia.org/wiki/Comparison_of_audio_coding_formats

a) General format of frames at different stack layers



b) Format of empty response. Size is 10 bytes

Preamble	Access Address	PDU Header	No Data	MIC	CRC
1B	4B	2B	0B	0B	3B

c) Format of notification without MIC information. Max size is 17 + 20 = 37 bytes

Preamble	Access Address	PDU Header	L2CAP Header1	L2CAP Header2	ATT Opcode	Data	MIC	CRC
1B	4B	2B	2B	2B	3B	20B	0B	3B

d) Format of read blob without MIC information. Max size is 15 + 22 = 37 bytes

Preamble	Access Address	PDU Header	L2CAP Header1	L2CAP Header2	Op code	Data	MIC	CRC
1B	4B	2B	2B	2B	1B	22B	0B	3B

e) Format of CID without MIC information. Max size is 14 + 23 = 37 bytes

Preamble	Access Address	PDU Header	L2CAP Header1	L2CAP Header2	Data	MIC	CRC
1B	4B	2B	2B	2B	23B	0B	3B

f) Proprietary CID. MIC used for data. Max size is 14 + 27 = 41 bytes

Preamble	Access Address	PDU Header	L2CAP Header1	L2CAP Header2	Data	CRC
1B	4B	2B	2B	2B	27B	3B

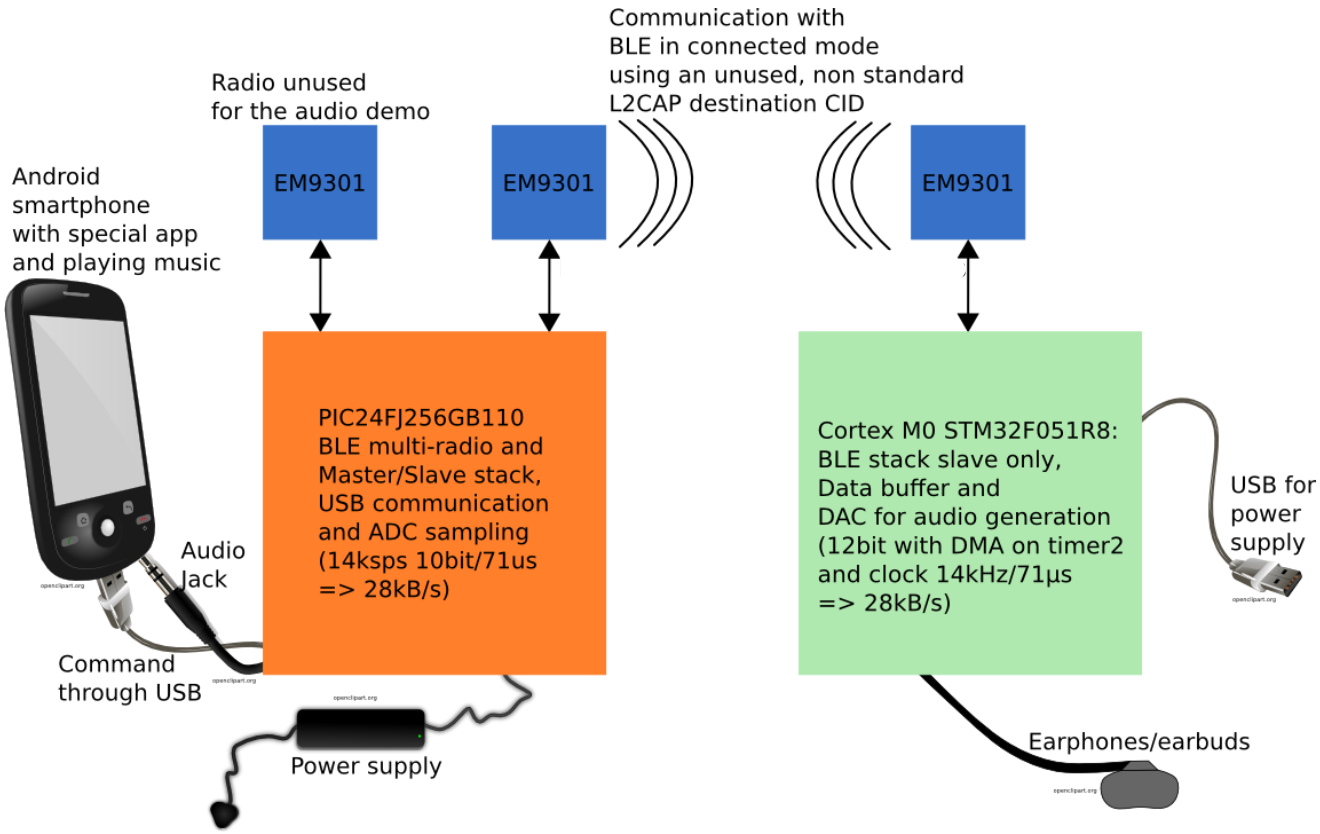
g) Proprietary. Apple Inc patent ?

Preamble	Access Address	PDU Header	Data	MIC	CRC
1B	4B	2B	33B	0/4B	3B

h) Proprietary value in header of ADV. Max size is 10 + 37 = 47 bytes

Preamble	Access Address	Header	Data	CRC
1B	4B	2B	37B	3B

Fig. 1 Frame structure in different configurations



The demonstrator uses the Ble connected mode to transmit audio with speech quality.
The useful data rate is 28kB/s
The ADC samples at 14 ksample/s, using 10 bits packed in 2 bytes.
A music file is read from the smartphone, played, sampled and transmitted. A user interface on the smartphone allows the setup.
A 12-bit DAC is used on the receiving side. The received data is converted to an analog signal that is sent to a speaker.

Fig. 2 The setup can be used for different tests. Streaming of audio, multi-channel operations, chat,…etc.

Fig.3 Setup to stream data and measure throughput rate

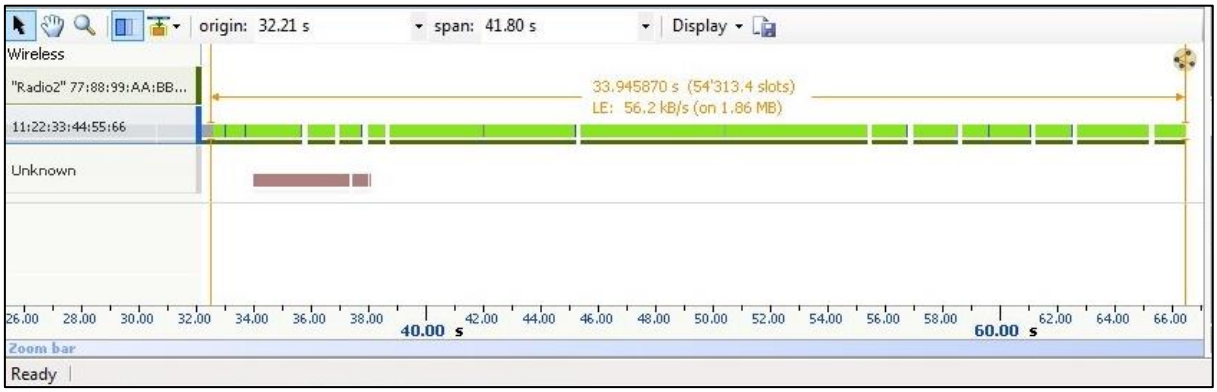


Fig.4a) Bidirectional communication.
Equal data size from each party. An average of 56.2 kB/s for both directions is reached (here over 30 seconds). One can clearly see that there are holes in the transmission; they represent times when the connection intervals are not fully filled with data. This happens for different reasons. Such holes reduce the maximum possible throughput.
The form of the frame in each direction is as follows:

Preamble	Access Address	PDU Header	L2CAP Header1	L2CAP Header2	Data	MIC	CRC	There is no encryption. 1 MIC position is used for data effectively giving 24B for application data.
1B	4B	2B	2B	2B	23B	1B	3B	

Important. The sniffer calculation is based on 28B (PDU data). On the section below it is [18 00 FF FF …..5E 5F]
Considering the 28B of the PDU data. If all the holes are filled, the max throughput for the application will be (28B data, 10B overhead, 150µs for 1 IFS). That is 493.3kb/s (61.7 kB/s)

There are 4 additional bytes (in each direction) that cannot be used as application data.
The effective application data will be 24B. In the figure below it means [48 49 4A ….. 5E 5F]
If all the holes are filled, the max throughput for the application will be (24B useful data, 14B overhead, 150µs for 1 IFS) 422.9kb/s for both directions. That is 52.8kB/s.

The information from the sniffer can be corrected for 24B of effective data. It gives 48.1kB/s for both directions.
Therefore: max possible = 52.8kB/s Measured here = 48.1 kB/s

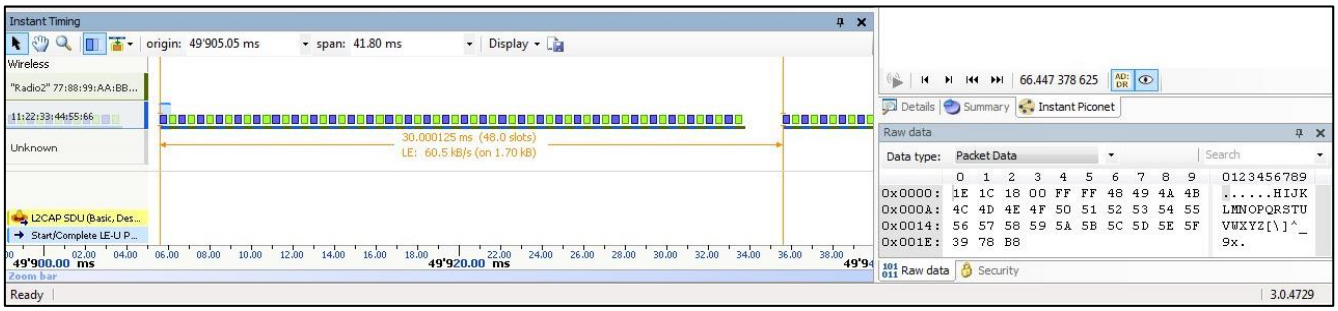


Fig.4b) Bidirectional communication with equal data size. Zoom on a smaller part
Blue and green represent the transmitted frames of the different devices.

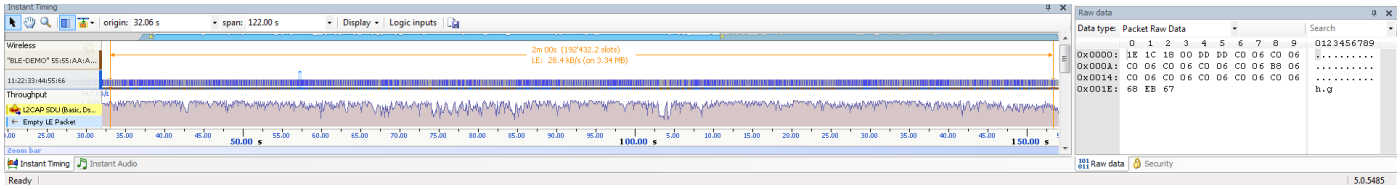


Fig. 5 Streaming audio. Data analog data is sampled and sent over Ble

Unidirectional transfer using custom CID (DD DD) and MIC bytes for data.
27 bytes can be sent as application data.

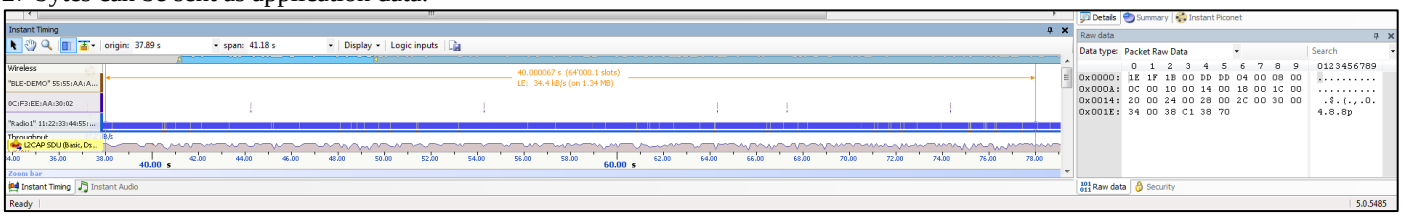


Fig. 6a) The transfer was made here on a long time period. (average of 34.4kB/s over 40 seconds).

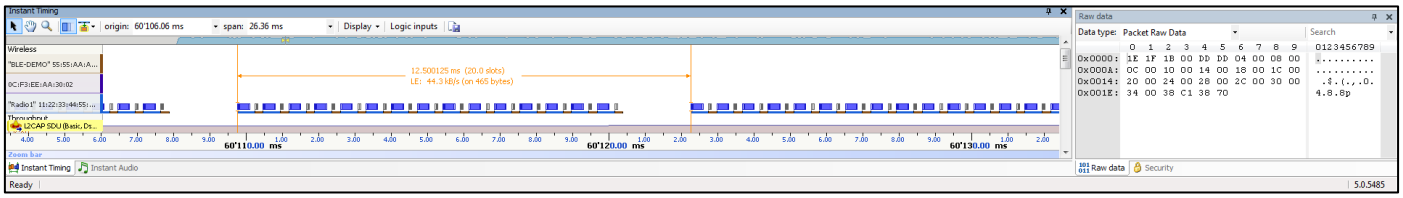


Fig 6b) a zoom on one slot shows holes that could be filled with additional packets to increase throughput

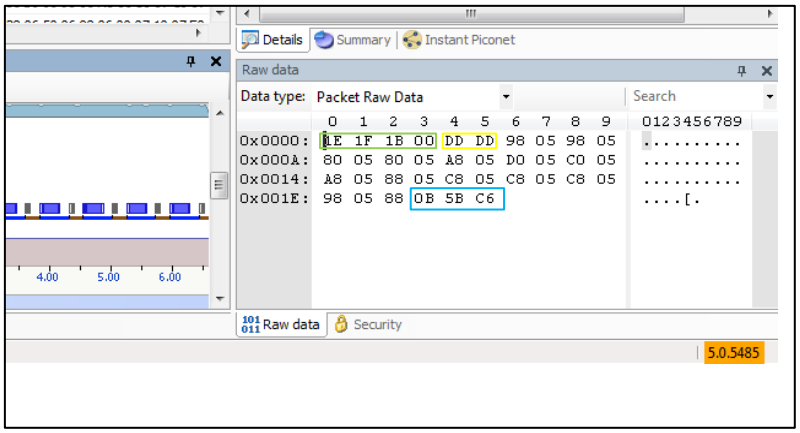


Fig. 6c) An example of PDU data seen by the sniffer

Preamble	Access Address	PDU Header	L2CAP Header1	L2CAP Header2	Data	MIC	CRC	There is no encryption. All 4 MIC positions are used for data effectively giving 27B for application data.
1B	4B	2B	2B	2B	23B	4B	3B	