User manual

DA14580 BLE data streaming application

UM-B-030

Abstract

This document describes a maximum throughput application capable of streaming data over BLE using the DA14580.

Contents

Figures

Tables

1 Terms and definitions

BLE	Bluetooth Low Energy
GTL	Generic Transport Layer

2 References

1. UM-B-017, GTL interface in integrated processor application, Dialog Semiconductor

3 Introduction

Bluetooth Low Energy (BLE) aims for low power consumption and infrequent short data transmissions. Trying to send large amount of data, although possible, may not be as efficient and the maximum throughput cannot be guaranteed. Still, real life applications occasionally require the exchange of large amount of data at highest possible rate to guarantee a perfect user experience. Such cases may be the exchange of large files (firmware updates, large parameters, etc.) or the streaming of related data such as short bursts of audio.

In this document a data streaming application is presented, that demonstrates the maximum throughput capabilities of the DA14580 using the smallest ATT_MTU size of 23 bytes. All experiments described have been done using this ATT_MTU size and further investigation is required to understand the effect of a larger ATT_MTU size. Different results may be obtained when a larger ATT_MTU size is used and the fragmentation and assembly capabilities of the BLE stack are used to their full extent.

4 Data transfer type

There are multiple ways to transfer data over BLE, each being best fitted for a specific use case and none intended for data streaming. Some transfers require a request from the central device in order to be initiated and other transfers require an acknowledgement from the central device to terminate. To achieve the maximum possible throughput, we selected the simplest types of data transfer that require no handshaking messages:

- Notification of peripheral to central device communication
- Write No Response for central to peripheral device communication

The maximum payload size in both cases is 20 bytes (ATT_MTU-3) per transfer.

There is an upper limit of the number of BLE packets that can be exchanged within one connection interval. Table 1 summarises the maximum number of transmits only (single direction) per connection interval for the DA14580-01.

Table 1: Maximum number of packets transmitted/received (simplex) per connection interval

Connection interval duration (ms)	Maximum number of packets transmitted/received
7.5	7
8.75	9
10	11
11.25	13
12.5	15
15	17 (peripheral device) or 18 (central device)

Also, there is an absolute maximum of 18 packets for the DA14580, determined by the number of buffers available in the exchange table.

5 Streaming simplex data

We have developed a simple algorithm that allows the use of the DA14580 at streaming rates up to 150 kbit/s.

1. Fill the buffers up to maximum number of packets allowed for this connection interval.
2. Use the time during the transmission of data to fill all remaining buffers up to 17. When an error occurs, stop sending data and resume sending in the next connection event.
3. Upon the connection end event, add only as many packets as needed to reach the target number of packets.
4. Go back to step 2.

The target number of buffers that can be sent per connection event is hardcoded in an array and set every time a new connection occurs or when the connection parameters change. This number is not always equal to the maximum number described in Table 1.

When we are trying to send more data within a connection interval, the end of the connection event and the beginning of the next transmission become closer in time. Since we use more buffers to send data, fewer buffers are available to prepare for the next connection interval. Thus more packets will have to be prepared and processed at the connection end event, before the start of the next period.

The buffers that are not used during a connection event should be filled while the BLE is transmitting. The CPU is usually idle at that point.

In Table 2 we have accumulated the results of the sustained rate achieved using the aforementioned algorithm, for simplex communication (single direction). The maximum of 17 buffers is reached for a connection interval of 21.25 ms. For other connection interval durations (>21.25 ms), we can easily derive the expected rate using the following formula:

rate (kbit/s) = 17 (packets) * 20 (bytes/packet) * 8 (bits/byte) / connection interval (ms)

Table 2: Simplex sustained rates of DA14580-01

Connection interval duration (ms)	Simplex sustained rate (kbit/s)
7.5	149.3
8.75	146.29
10	144
11.25	142.2
12.5	128
13.75	139.6
15	138.6
16.25	128
17.5	137.14
18.75	128
20	128
21.25	128

6 Full duplex communication

The reception is less demanding compared to the transmission. Every 4 packets or upon a complete event, the received data are pushed to the upper layers and the H/W buffers are freed. So there is no critical time period during which the buffers should be emptied before the beginning of the next event.

In the full duplex case, more time is spent during a connection interval to receive data. Consequently, fewer packets will be transmitted per connection interval compared to the simplex case.

Surprisingly, the overall bandwidth (receive plus transmit) is larger than in the simplex case. The reason is related to the way the packets are acknowledged in the simplex case: a complete packet with no payload is sent to acknowledge good reception. In the case of full duplex, the same packet carries the payload of the remote party making more efficient use of the existing bandwidth.

An analytic table with the results per connection interval is not provided in this document, but we have measured rates of 256 kbit/s (128 kbit/s per direction) for a connection interval of 10 ms.

7 BLE data streaming application evaluation

The BLE data streaming application is built around the following blocks:

- The streaming logic block
- Modifications to BLE interrupt handlers to get accurate event flags
- A block to collect the statistics (metrics)
- Two applications (central and peripheral device) controlled via GTL
- Other utilities

7.1 Streaming logic

The streaming logic accepts transmit requests from the application, stores them in a FIFO and forwards them to L2CC at a regulated rate. It should be called periodically by the main loop at least two times within a connection interval. It will check the pending requests, the state of the transmit queue and decide how many requests should be serviced.

7.1.1 Initialisation points

- `stream_fifo_init ( )`

  This function is called to clear the FIFO contents and initialise the FIFO state variables.

7.1.2 Entry points

- `stream_queue_more_data ( )`

  This function is called periodically from the main loop. It will check the state within the connection event (beginning or end) and service any pending FIFO requests accordingly.

- `stream_fifo_add ( )`

  This function accepts requests from the application to transmit data. The same function is used for Notification or Write No Response messages.

- `stream_fifo_check ( )`

  This function can be used instead of the callback mechanism to probe whether a specific FIFO item is free or in use.

7.1.3 Other functions

- `stream_queue_data_until ( )`
  This function fills the transmit queue up to a given point.
- `stream_queue_more_data_during_tx ( )`
  This function services pending FIFO requests at the beginning or during a connection event.
- `stream_queue_more_data_end_of_event ()`
  This function services pending FIFO requests at the end of a connection event.
- `send_pkt_to_l2cc ( )`
  This function creates the message to the L2CC layer.

7.2 BLE interrupt handlers

To accurately probe the state of the connection interval and the number of errors that occur during the BLE transmission, minor modifications have been applied to the BLE interrupt handling routines. No functions are required to be managed or called from the application.

7.2.1 Other functions

- `BLE_RF_DIAG_Handler( )`
  Code has been added to this function to capture the start of a connection event.
- `BLE_EVENT_Handler( )`
  Code has been added to this function to capture the end of a connection event and measure failed receptions.
- `BLE_RX_Handler( )`
  Code has been added to this function to capture failed receptions.

7.3 Metrics block

This block provides a point to collect information related to the transmission and reception of data and periodically generate a report.

7.3.1 Initialisation points

- `init_metrics ( )`
  This function is called to initialise state variables.

7.3.2 Entry points

- `add_err_packet()`
  This function is called to add a receive error
- `add_packets_tx ( )`
  This function is called to increment the counter of packets transmitted.
- `add_packets_rx ( )`
  This function is called to increment the counter of packets received.

7.3.3 Other functions

- `calc_update_avail ( )`

  This function is called to update the number of available buffers in the transmit queue.

- `calc_update ( )`

  This function is periodically called at the end of a connection event. It will copy the statistics collected to a global structure periodically and restart the measurement.

- `measure_errors_received( )`

  This function is called within the BLE interrupts to parse the received packets an to detect and report errors.

7.4 Application controlled over GTL

Two different applications have been developed: one for the peripheral device and one for the central device. Both are controlled via GTL. SmartSnippets can perform the following actions through GTL:

- Set the central device into Scanning mode (central device only)
- Connect/disconnect to/from a specific device (central device only)
- Start/stop the transmission of data. (central and peripheral device)
- Collect the statistics (central and peripheral device)

More information on GTL and GTL based applications can be found in Ref. [1].

7.5 Other utilities

Some more blocks are required for the demo application.

7.5.1 Streaming profile

A dedicated streaming profile is exposed by the peripheral to the central device. The only use of the profile is to expose 10 parameters to the client and their respective handles. The handle is required to properly construct and address a Notification or a Write no Response message. Any 20-byte maximum length handle of any active profile could be used instead.

7.5.2 Test packet generator

A simple test packet generator is used to feed data into the FIFO when SmartSnippets sends the command to do so.

7.5.2.1 Initialisation points

- `test_pkt_init ( )`

  This function is called to initialise test packet generator state variables.

7.5.2.2 Entry points

- `test_pkt_gen ( )`

  This function is called periodically from the main loop. If the device is in streaming mode, it will try to completely fill the FIFO with data.

7.5.2.3 Other functions

- `test_callback ( )`

  This function is called every time a packet is pushed in the L2CC. The test packet generator uses this callback to increment an index number within the packet data. This index is used for testing and debugging purposes only.

8 SmartSnippets data rate monitor

The SmartSnippets tool kit is equipped with a Data Rate Monitor application enabling the user to explore the maximum data rate capabilities of the DA14580 in a real life environment.

8.1 Setup

Two DA14580 Development Kits are required for the test. The user should launch two instances of the SmartSnippets tool kit; one will be used to control the central device and the other one to control the peripheral device. The Data Rate Monitor tab is shown in Figure 1.

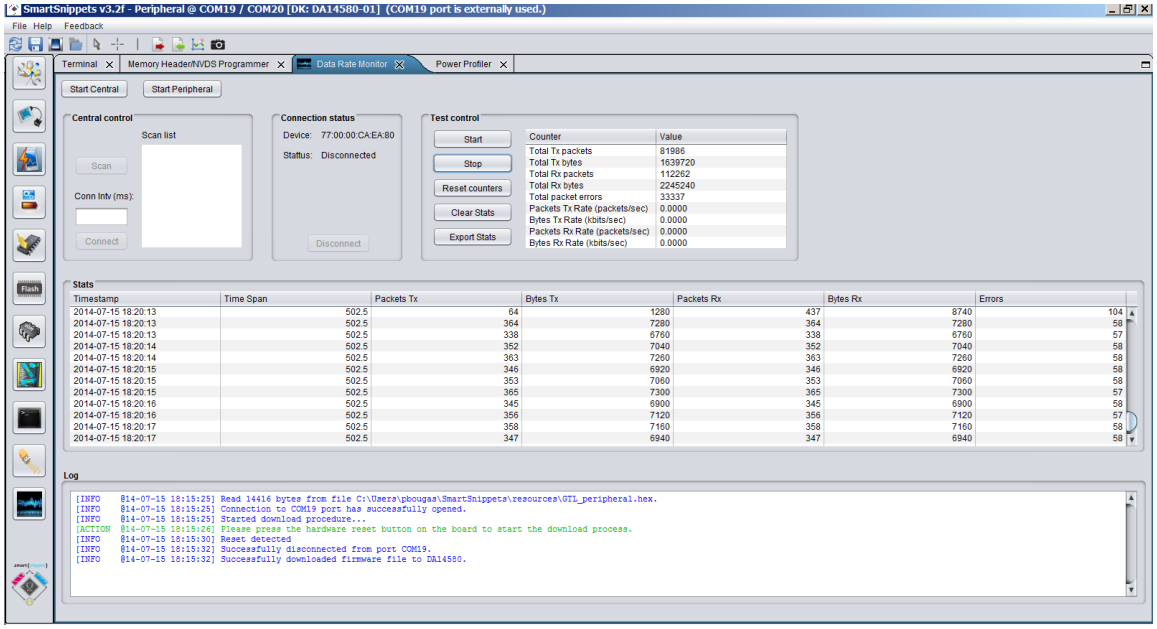

Figure 1: The SmartSnippets Data Rate Monitor tool

1. Initially the user should press the Start Central button to download the firmware to the Central device controlled by SmartSnippets.
2. It will then press the Start Peripheral button button to download the firmware to the Peripheral device controlled by SmartSnippets.
3. After the firmware has been successfully downloaded, the user will set the Central device into scanning mode pressing the Scan button.
4. The device will respond with all the MAC addresses of devices advertising in range populating the Scan list control.
5. The user will select the peripheral's device MAC, fill the desired Conn Interval in ms and press Connect.
6. Upon successful connection, the Stats window on both SmartSnippets (Central and Peripheral) will start being populated with reports from the devices on the number of packets, bytes and errors received and transmitted. Everything except the errors should appear as zero at this point.
7. The user can control if the Peripheral or Central device is transmitting data using the Start and Stop buttons. After pressing the Start button, the device controlled by the SmartSnippet instance will start sending data to the remote device connected to it and report the number of data sent over a specific Time Span. The remote party will automatically upload reports on the bytes and packets received.

SmartSnippets accumulates the total number of packets and number of bytes exchanged and calculates the mean rate during the latest reported time span. All of this information is displayed in a dedicated control. There are also buttons available to clear the accumulated counters and to clear all the reports from the device in order to restart a test without having to disconnect.

All the reports contained in the Stats control can be export to a .csv file for further analysis.

9 Revision history

Revision	Date	Description
1.0	18-Jul-2014	Initial version.

Status definitions

Status	Definition
DRAFT	The content of this document is under review and subject to formal approval, which may result in modifications or additions.
APPROVED or unmarked	The content of this document has been approved for publication.

Disclaimer

Information in this document is believed to be accurate and reliable. However, Dialog Semiconductor does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information. Dialog Semiconductor furthermore takes no responsibility whatsoever for the content in this document if provided by any information source outside of Dialog Semiconductor.

Dialog Semiconductor reserves the right to change without notice the information published in this document, including without limitation the specification and the design of the related semiconductor products, software and applications.

Applications, software, and semiconductor products described in this document are for illustrative purposes only. Dialog Semiconductor makes no representation or warranty that such applications, software and semiconductor products will be suitable for the specified use without further testing or modification. Unless otherwise agreed in writing, such testing or modification is the sole responsibility of the customer and Dialog Semiconductor excludes all liability in this respect.

Customer notes that nothing in this document may be construed as a license for customer to use the Dialog Semiconductor products, software and applications referred to in this document. Such license must be separately sought by customer with Dialog Semiconductor.

All use of Dialog Semiconductor products, software and applications referred to in this document are subject to Dialog Semiconductor's Standard Terms and Conditions of Sale, unless otherwise stated.

RoHS Compliance

Dialog Semiconductor complies to European Directive 2001/95/EC and from 2 January 2013 onwards to European Directive 2011/65/EU concerning Restriction of Hazardous Substances (RoHS/RoHS2).
Dialog Semiconductor's statement on RoHS can be found on the customer portal https://support.diasemi.com/. RoHS certificates from our suppliers are available on request.

Contacting Dialog Semiconductor

Germany Headquarters
Dialog Semiconductor GmbH
Phone: +49 7021 805-0

United Kingdom
Dialog Semiconductor (UK) Ltd
Phone: +44 1793 757700

The Netherlands
Dialog Semiconductor B.V.
Phone: +31 73 640 8822

Email:
enquiry@diasemi.com

North America
Dialog Semiconductor Inc.
Phone: +1 408 845 8500

Japan
Dialog Semiconductor K. K.
Phone: +81 3 5425 4567

Taiwan
Dialog Semiconductor Taiwan
Phone: +886 281 786 222

Web site:
www.dialog-semiconductor.com

Singapore
Dialog Semiconductor Singapore
Phone: +65 64 849929

China
Dialog Semiconductor China
Phone: +86 21 5178 2561

Korea
Dialog Semiconductor Korea
Phone: +82 2 3469 8291