PSoC 4 BLE – Throughput Maximization

Objective

This example demonstrates how you can maximize throughput in a BLE communication.

Overview

This example demonstrates maximum data transfer in a BLE communication from a device. This can happen in two ways, and hence there are four projects as part of this example –

1. Data transfer using GATT Notifications on a characteristic – Outgoing
2. Data transfer using GATT Notifications on a characteristic – Incoming
3. Data transfer using L2CAP Connection-Oriented channels – Outgoing
4. Data transfer using L2CAP Connection-Oriented channels – Incoming

The device initializes a buffer of 512 bytes and once connected, keeps sending data continuously. The amount of data sent by this device depends on the MTU of the receiving side.

The example also calculates the received data throughput in the incoming projects and shows it on the UART terminal.

For more details on the GATT Notifications, refer to the Bluetooth 4.1 Specification, Volume 3, Part G, Section 4.10.

For more details on the L2CAP Connection-Oriented channels, refer to the Bluetooth 4.1 Specification, Volume 3, Part A, Section 3.4.

Requirements

Design Tool: PSoC Creator 3.1 SP1, CySmart 1.0

Programming Language: C (GCC 4.8.4 – included with PSoC Creator)

Associated Devices: All PSoC 4 BLE devices

Required Hardware: CY8CKIT-042-BLE Bluetooth® Low Energy (BLE) Pioneer Kit

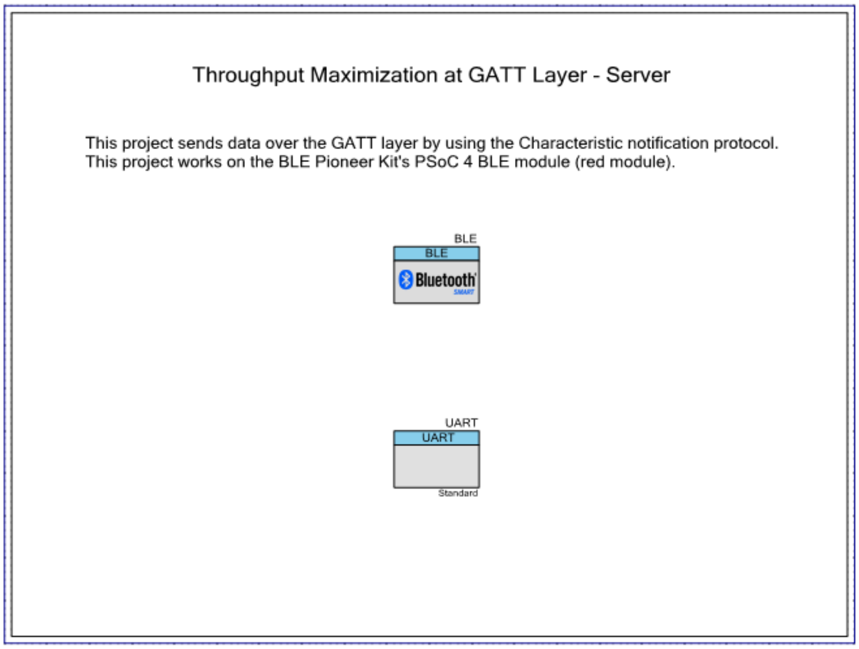
Hardware Setup

The BLE Pioneer Kit has all of the necessary hardware required for this lab. There is no special setup required.

Data Transfer via GATT Notifications

PSoC Creator Schematic

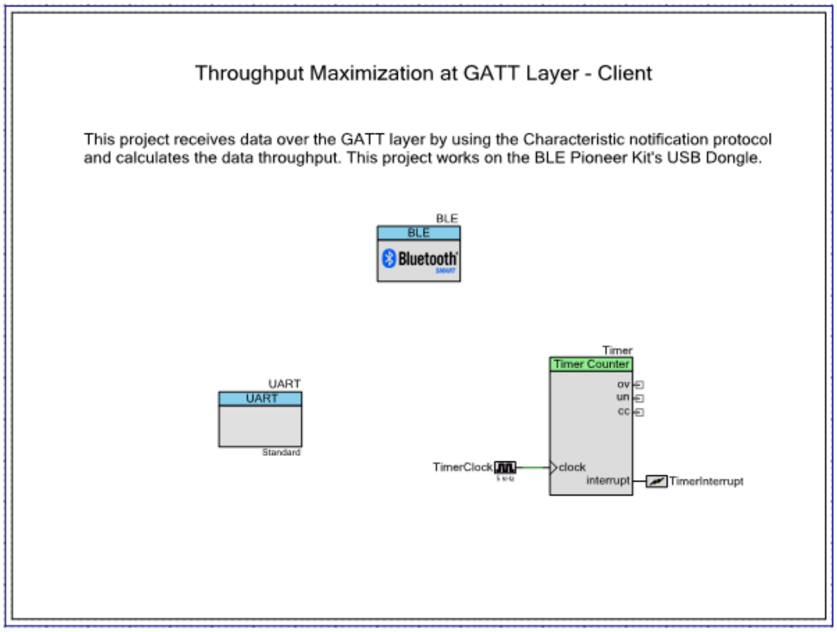
Figure 1. PSoC Creator Schematic - GATT Notifications - Outgoing



Throughput Maximization at GATT Layer - Server

This project sends data over the GATT layer by using the Characteristic notification protocol.
This project works on the BLE Pioneer Kit's PSoC 4 BLE module (red module).

Figure 2. PSoC Creator Schematic - GATT Notifications - Incoming



Throughput Maximization at GATT Layer - Client

This project receives data over the GATT layer by using the Characteristic notification protocol
and calculates the data throughput. This project works on the BLE Pioneer Kit's USB Dongle.

Project Description

There are two projects for GATT Notifications – Outgoing and Incoming. The Outgoing project sends data which can be received by the Incoming project. The Outgoing project can also work with the CySmart PC tool.

In the example workspace, program the project **GATT Notification – Data Incoming** onto the USB Dongle and **GATT Notification – Data Outgoing** onto the BLE Pioneer Kit baseboard with PSoC 4 BLE module.

The Outgoing project implements a GATT Server and GAP Peripheral role. It sends data over BLE using GATT notifications on a custom characteristic. The project works on a BLE Pioneer Kit with PSoC 4 BLE module.

Once the device is connected, the Client can send an MTU Exchange Request to let the Server know about its MTU size. Now when the Client enables notifications on the Server, the Server sends (MTU - 3) bytes of data continuously, whenever its BLE stack is free.

If no MTU Exchange is done, then a default MTU of 23 bytes is taken and thus 20 bytes of data is continuously sent. The maximum MTU size can be 512 bytes.

The Incoming project implements a GATT Client and GAP Central role. It receives the data from the Outgoing project and calculates the amount of data received in 10 seconds. This is then converted into a throughput value in kbps terms. The MTU size used is 512.

The Incoming project works with a UART based terminal. When scanning, the list of devices is shown on UART. On the terminal, pressing 'C' followed by a number results in a new connection to that device. Pressing 'S' refreshes the scan list. Pressing 'D' disconnects the device from the connection.

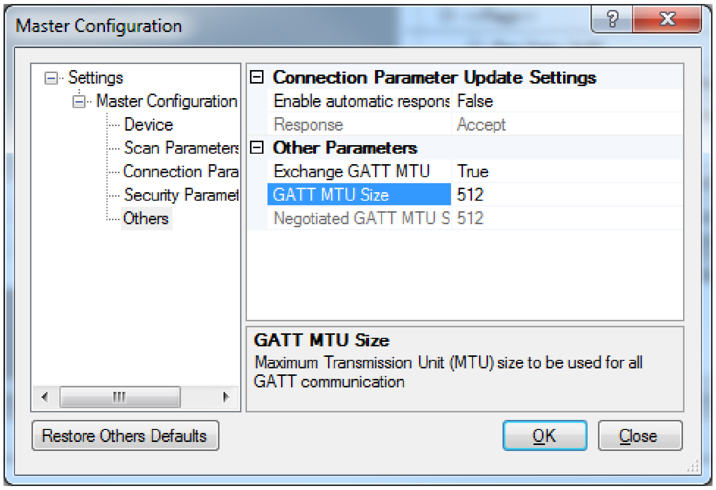
Testing Outgoing project with CySmart

Open CySmart tool and configure your MTU size to be 512 (this is the default as well). See Figure 3.

Once the MTU is set, connect to the device and enable notifications on the custom characteristic (click **Discover All Attributes** followed by **Enable All Notifications**). You will start getting data on the custom characteristic. See Figure 4.

Depending on the MTU size, the amount of data received will vary.

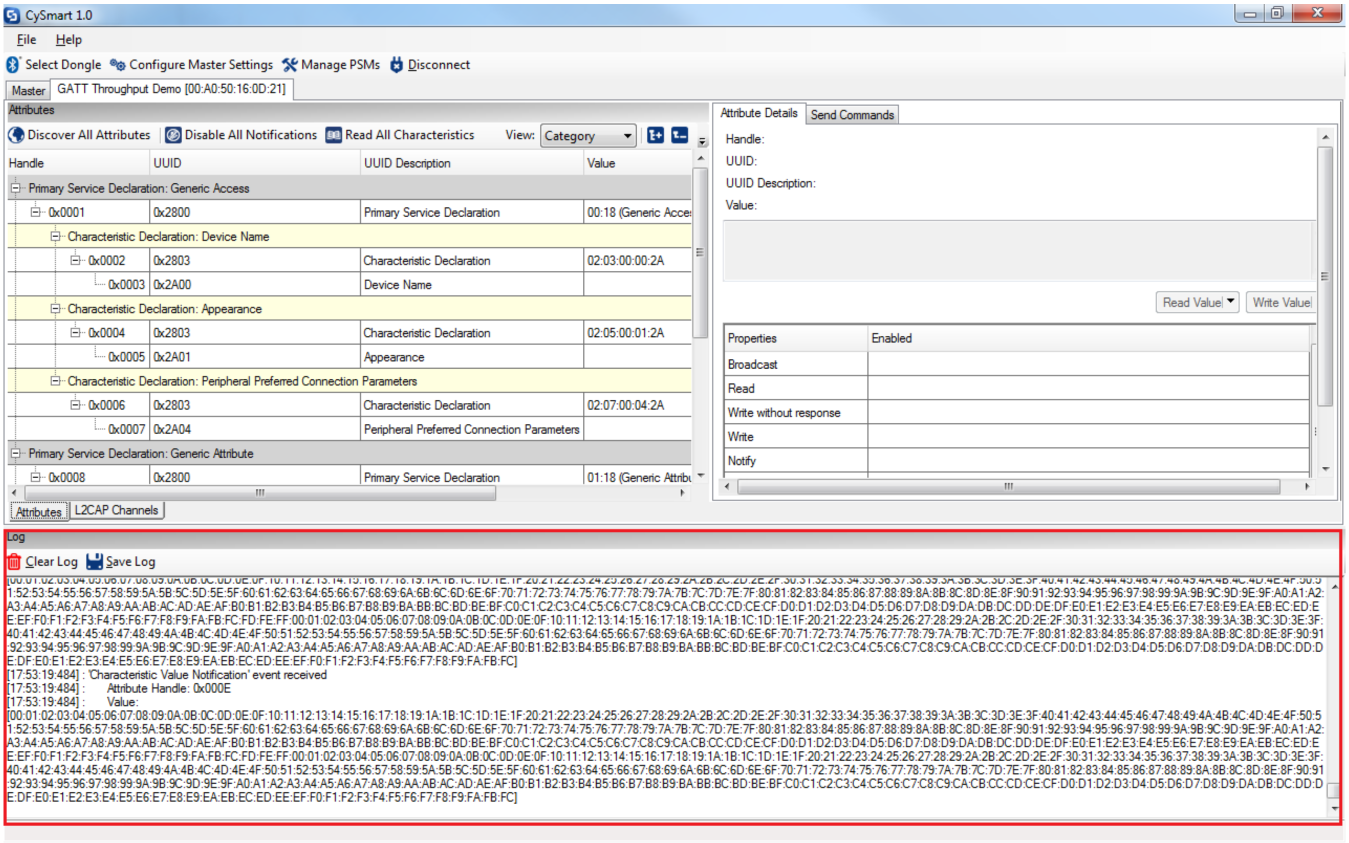
Figure 3. Configuring CySmart MTU Size for GATT Data



Connection Parameter Update Settings	
Enable automatic respons	False
Response	Accept
Other Parameters	
Exchange GATT MTU	True
GATT MTU Size	512
Negotiated GATT MTU S	512

GATT MTU Size
Maximum Transmission Unit (MTU) size to be used for all GATT communication

Figure 4. Data on CySmart



Handle	UUID	UUID Description	Value
Primary Service Declaration: Generic Access			
0x0001	0x2800	Primary Service Declaration	00:18 (Generic Acce:
Characteristic Declaration: Device Name			
0x0002	0x2803	Characteristic Declaration	02:03:00:00:2A
0x0003	0x2A00	Device Name	
Characteristic Declaration: Appearance			
0x0004	0x2803	Characteristic Declaration	02:05:00:01:2A
0x0005	0x2A01	Appearance	
Characteristic Declaration: Peripheral Preferred Connection Parameters			
0x0006	0x2803	Characteristic Declaration	02:07:00:04:2A
0x0007	0x2A04	Peripheral Preferred Connection Parameters	
Primary Service Declaration: Generic Attribute			
0x0008	0x2800	Primary Service Declaration	01:18 (Generic Attribu

Handle:
UUID:
UUID Description:
Value:

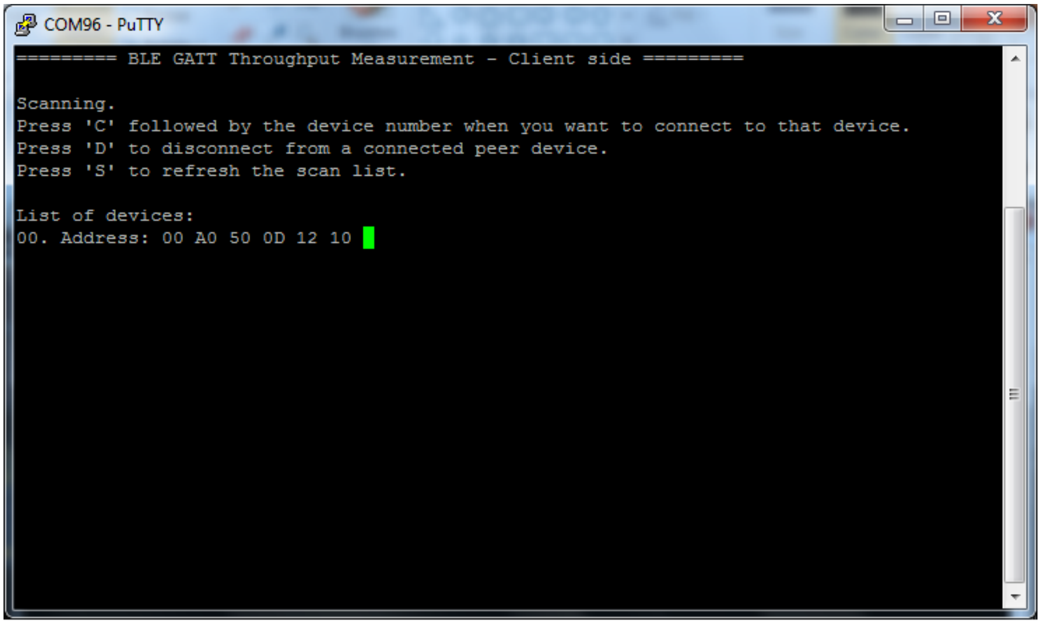
Properties	Enabled
Broadcast	
Read	
Write without response	
Write	
Notify	

```
[00:01:02:03:04:05:06:07:08:09:0A:0B:0C:0D:0E:0F:10:11:12:13:14:15:16:17:18:19:1A:1B:1C:1D:1E:1F:20:21:22:23:24:25:26:27:28:29:2A:2B:2C:2D:2E:2F:30:31:32:33:34:35:36:37:38:39:3A:3B:3C:3D:3E:3F:40:41:42:43:44:45:46:47:48:49:4A:4B:4C:4D:4E:4F:50:5
1:52:53:54:55:56:57:58:59:5A:5B:5C:5D:5E:5F:60:61:62:63:64:65:66:67:68:69:6A:6B:6C:6D:6E:6F:70:71:72:73:74:75:76:77:78:79:7A:7B:7C:7D:7E:7F:80:81:82:83:84:85:86:87:88:89:8A:8B:8C:8D:8E:8F:90:91:92:93:94:95:96:97:98:99:9A:9B:9C:9D:9E:9F:A0:A1:A2:
A3:A4:A5:A6:A7:A8:A9:AA:AB:AC:AD:AE:AF:B0:B1:B2:B3:B4:B5:B6:B7:B8:B9:BA:BB:BC:BD:BE:BF:C0:C1:C2:C3:C4:C5:C6:C7:C8:C9:CA:CB:CC:CD:CE:CF:D0:D1:D2:D3:D4:D5:D6:D7:D8:D9:DA:DB:DC:DD:DE:DF:E0:E1:E2:E3:E4:E5:E6:E7:E8:E9:EA:EB:EC:ED:E
E:EF:F0:F1:F2:F3:F4:F5:F6:F7:F8:F9:FA:FB:FC:FD:FE:FF:00:01:02:03:04:05:06:07:08:09:0A:0B:0C:0D:0E:0F:10:11:12:13:14:15:16:17:18:19:1A:1B:1C:1D:1E:1F:20:21:22:23:24:25:26:27:28:29:2A:2B:2C:2D:2E:2F:30:31:32:33:34:35:36:37:38:39:3A:3B:3C:3D:3E:3F:
40:41:42:43:44:45:46:47:48:49:4A:4B:4C:4D:4E:4F:50:51:52:53:54:55:56:57:58:59:5A:5B:5C:5D:5E:5F:60:61:62:63:64:65:66:67:68:69:6A:6B:6C:6D:6E:6F:70:71:72:73:74:75:76:77:78:79:7A:7B:7C:7D:7E:7F:80:81:82:83:84:85:86:87:88:89:8A:8B:8C:8D:8E:8F:90:91
:92:93:94:95:96:97:98:99:9A:9B:9C:9D:9E:9F:A0:A1:A2:A3:A4:A5:A6:A7:A8:A9:AA:AB:AC:AD:AE:AF:B0:B1:B2:B3:B4:B5:B6:B7:B8:B9:BA:BB:BC:BD:BE:BF:C0:C1:C2:C3:C4:C5:C6:C7:C8:C9:CA:CB:CC:CD:CE:CF:D0:D1:D2:D3:D4:D5:D6:D7:D8:D9:DA:DB:DC:DD:D
E:DF:E0:E1:E2:E3:E4:E5:E6:E7:E8:E9:EA:EB:EC:ED:EE:EF:F0:F1:F2:F3:F4:F5:F6:F7:F8:F9:FA:FB:FC]
[17:53:19:484] : 'Characteristic Value Notification' event received
[17:53:19:484] :       Attribute Handle: 0x000E
[17:53:19:484] :       Value:
[00:01:02:03:04:05:06:07:08:09:0A:0B:0C:0D:0E:0F:10:11:12:13:14:15:16:17:18:19:1A:1B:1C:1D:1E:1F:20:21:22:23:24:25:26:27:28:29:2A:2B:2C:2D:2E:2F:30:31:32:33:34:35:36:37:38:39:3A:3B:3C:3D:3E:3F:40:41:42:43:44:45:46:47:48:49:4A:4B:4C:4D:4E:4F:50:5
1:52:53:54:55:56:57:58:59:5A:5B:5C:5D:5E:5F:60:61:62:63:64:65:66:67:68:69:6A:6B:6C:6D:6E:6F:70:71:72:73:74:75:76:77:78:79:7A:7B:7C:7D:7E:7F:80:81:82:83:84:85:86:87:88:89:8A:8B:8C:8D:8E:8F:90:91:92:93:94:95:96:97:98:99:9A:9B:9C:9D:9E:9F:A0:A1:A2:
A3:A4:A5:A6:A7:A8:A9:AA:AB:AC:AD:AE:AF:B0:B1:B2:B3:B4:B5:B6:B7:B8:B9:BA:BB:BC:BD:BE:BF:C0:C1:C2:C3:C4:C5:C6:C7:C8:C9:CA:CB:CC:CD:CE:CF:D0:D1:D2:D3:D4:D5:D6:D7:D8:D9:DA:DB:DC:DD:DE:DF:E0:E1:E2:E3:E4:E5:E6:E7:E8:E9:EA:EB:EC:ED:E
E:EF:F0:F1:F2:F3:F4:F5:F6:F7:F8:F9:FA:FB:FC:FD:FE:FF:00:01:02:03:04:05:06:07:08:09:0A:0B:0C:0D:0E:0F:10:11:12:13:14:15:16:17:18:19:1A:1B:1C:1D:1E:1F:20:21:22:23:24:25:26:27:28:29:2A:2B:2C:2D:2E:2F:30:31:32:33:34:35:36:37:38:39:3A:3B:3C:3D:3E:3F:
40:41:42:43:44:45:46:47:48:49:4A:4B:4C:4D:4E:4F:50:51:52:53:54:55:56:57:58:59:5A:5B:5C:5D:5E:5F:60:61:62:63:64:65:66:67:68:69:6A:6B:6C:6D:6E:6F:70:71:72:73:74:75:76:77:78:79:7A:7B:7C:7D:7E:7F:80:81:82:83:84:85:86:87:88:89:8A:8B:8C:8D:8E:8F:90:91
:92:93:94:95:96:97:98:99:9A:9B:9C:9D:9E:9F:A0:A1:A2:A3:A4:A5:A6:A7:A8:A9:AA:AB:AC:AD:AE:AF:B0:B1:B2:B3:B4:B5:B6:B7:B8:B9:BA:BB:BC:BD:BE:BF:C0:C1:C2:C3:C4:C5:C6:C7:C8:C9:CA:CB:CC:CD:CE:CF:D0:D1:D2:D3:D4:D5:D6:D7:D8:D9:DA:DB:DC:DD:D
E:DF:E0:E1:E2:E3:E4:E5:E6:E7:E8:E9:EA:EB:EC:ED:EE:EF:F0:F1:F2:F3:F4:F5:F6:F7:F8:F9:FA:FB:FC]
```

Testing Outgoing project with Incoming project

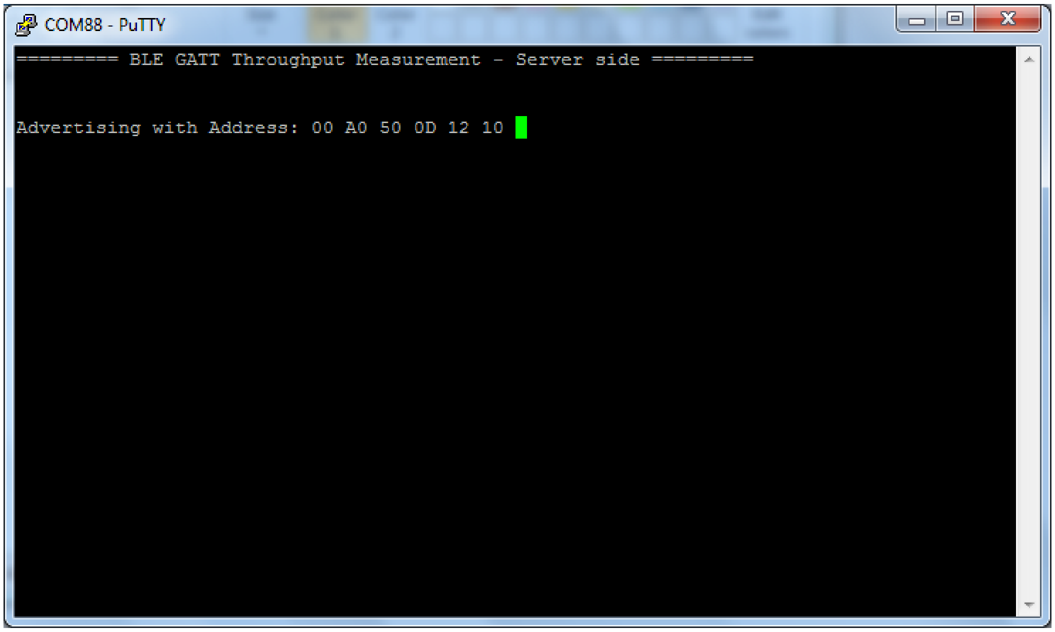
To actually calculate the throughput, the incoming project is needed. To use the same, connect a terminal emulator (such as Putty or Tera Term) each to the Kit and the Dongle. The COM settings are: Baud rate – 115200 bps, Data bits – 8, Stop bits – 1, Parity – None. The Outputs are shown in Figure 5 and Figure 6.

Figure 5. Terminal output - GATT Notification - Incoming Device



```
========= BLE GATT Throughput Measurement - Client side =========

Scanning.
Press 'C' followed by the device number when you want to connect to that device.
Press 'D' to disconnect from a connected peer device.
Press 'S' to refresh the scan list.

List of devices:
00. Address: 00 A0 50 0D 12 10
```

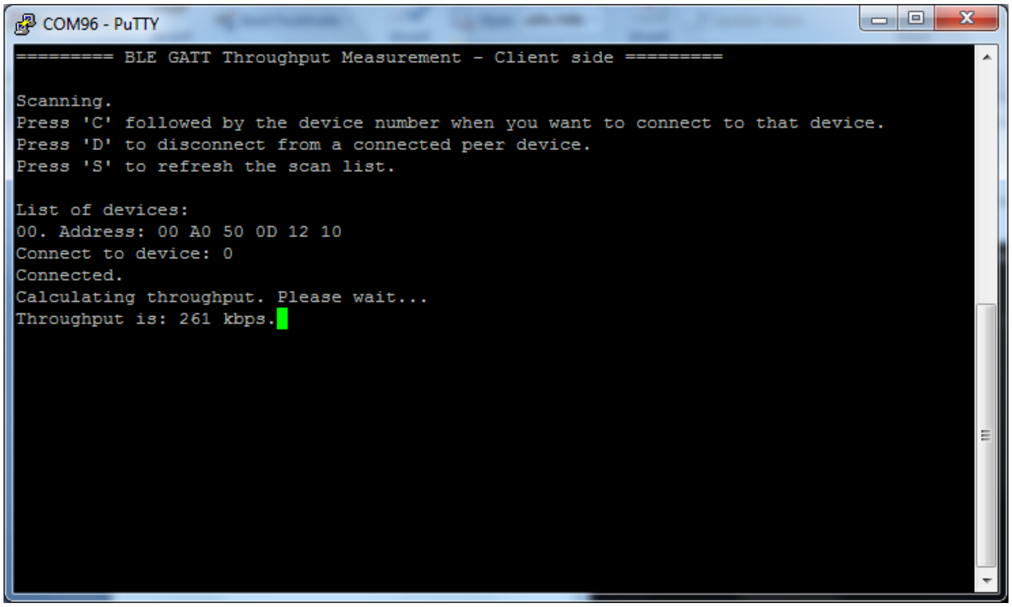
Figure 6. Terminal output - GATT Notification - Outgoing Device



```
========= BLE GATT Throughput Measurement - Server side =========


Advertising with Address: 00 A0 50 0D 12 10
```

Connect to the Outgoing project and wait for the throughput calculation to happen. Once done, the throughput is shown on the screen. This depends on the distance between the two devices, connection interval, MTU size, interference etc. See Figure 7.

Figure 7. Terminal output - GATT Notification - Throughput Calculation

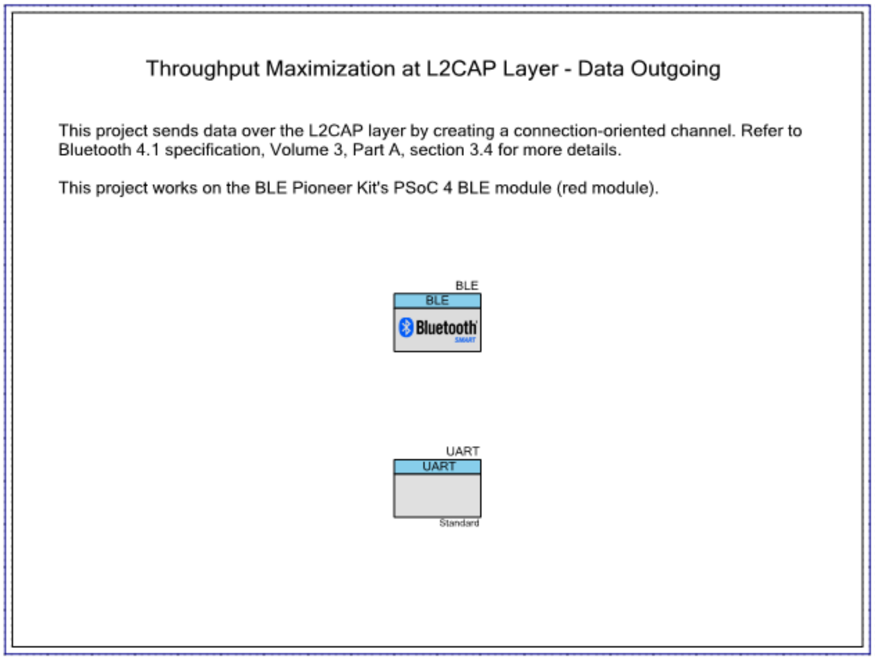


```
========= BLE GATT Throughput Measurement - Client side =========

Scanning.
Press 'C' followed by the device number when you want to connect to that device.
Press 'D' to disconnect from a connected peer device.
Press 'S' to refresh the scan list.

List of devices:
00. Address: 00 A0 50 0D 12 10
Connect to device: 0
Connected.
Calculating throughput. Please wait...
Throughput is: 261 kbps.
```

Data Transfer via L2CAP Connection-Oriented Channels

PSoC Creator Schematic

Figure 8. PSoC Creator Schematic - L2CAP Channel - Outgoing

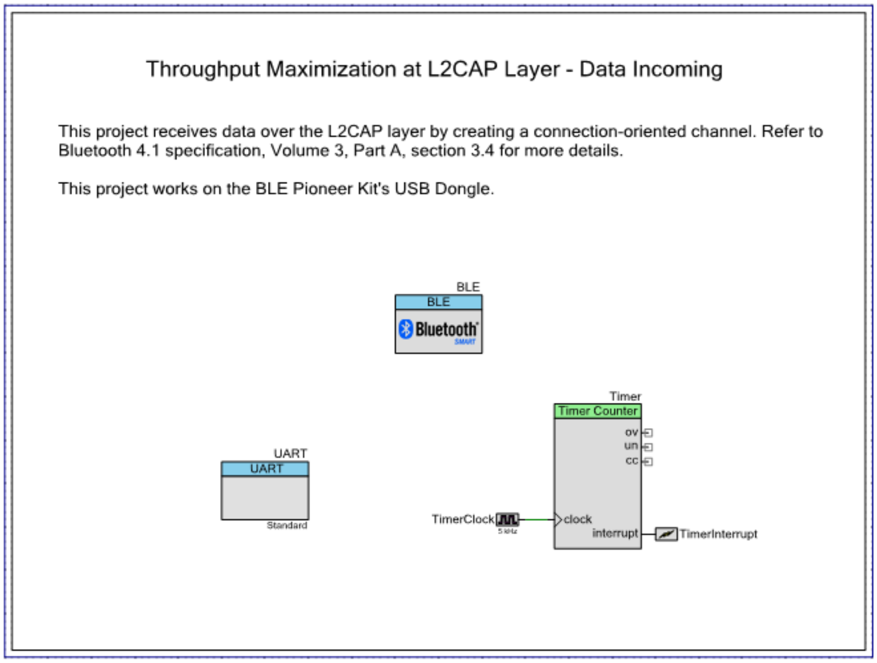


Throughput Maximization at L2CAP Layer - Data Outgoing

This project sends data over the L2CAP layer by creating a connection-oriented channel. Refer to Bluetooth 4.1 specification, Volume 3, Part A, section 3.4 for more details.

This project works on the BLE Pioneer Kit's PSoC 4 BLE module (red module).

Figure 9. PSoC Creator Schematic - L2CAP Channel - Incoming



Throughput Maximization at L2CAP Layer - Data Incoming

This project receives data over the L2CAP layer by creating a connection-oriented channel. Refer to Bluetooth 4.1 specification, Volume 3, Part A, section 3.4 for more details.

This project works on the BLE Pioneer Kit's USB Dongle.

Project Description

Similar to GATT Notifications, there are two projects for L2CAP Channel - Outgoing and Incoming. The Outgoing project sends data which can be received by the Incoming project. The Outgoing project can also work with the CySmart PC tool.

In the example workspace, program the project **L2CAP Channel - Data Incoming** onto the USB Dongle and **L2CAP Channel - Data Outgoing** onto the BLE Pioneer Kit baseboard with PSoC 4 BLE module.

The Outgoing project sends continuous data over the L2CAP layer by using a connection-oriented channel. The protocol of GATT Server and Client is not required here since L2CAP is a lower-layer implementation than the GATT layer. To know more about L2CAP channels, refer to Bluetooth 4.1 Specification, Volume 3, Part A, Section 3.4. This project works on a BLE Pioneer Kit with PSoC 4 BLE module.

Once connected to the peer device, this device sends a request to create a new connection-oriented channel and upon creation of that channel the device starts sending data. The device can send data until its credits exhaust. Using the concepts of credits, the other device can control the amount of data it wants to receive. Once the peer device sends additional credits to this device, more data can be sent by this device.

Each packet can contain (MTU - 2) bytes of data, where the MTU is defined by the peer device. Maximum size of MTU can be 512 bytes.

The Incoming project receives the data from the Outgoing project and calculates the amount of data received in 10 seconds. This is then converted into a throughput value in kbps terms. The MTU size used is 512.

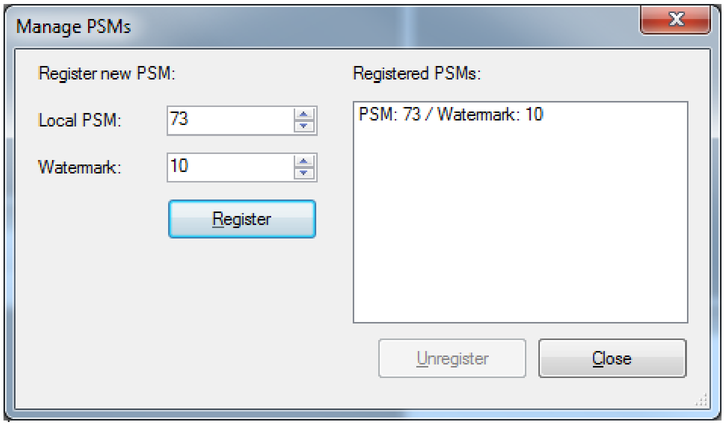
The Incoming project works with a UART based terminal. When scanning, the list of devices is shown on UART. On the terminal, pressing ‘C’ followed by a number results in a new connection to that device. Pressing ‘S’ refreshes the scan list. Pressing ‘D’ disconnects the device from the connection.

Note: The BLE component in the **L2CAP Channel - Data Outgoing** project is v1.10. Do not update the BLE component to v1.20 when prompted by PSoC Creator, as the newer version needs a patch for L2CAP functionality.

Testing Outgoing project with CySmart

In the CySmart tool, click on **Manage PSMs** and register a new PSM of value 73 and Watermark 10. See Figure 10.

Figure 10. Registering a PSM with CySmart



Local PSM: 73

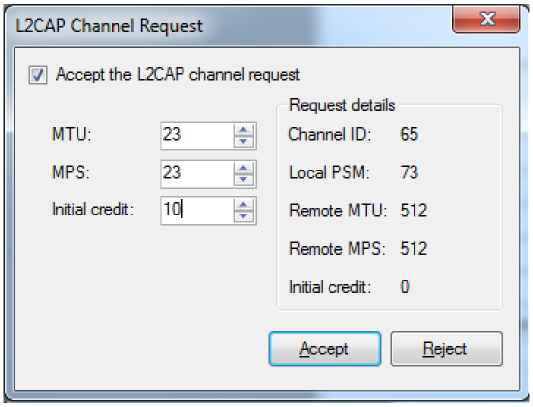
Watermark: 10

PSM: 73 / Watermark: 10

Once the PSM is registered, connect to the device, and you will get a popup for a new connection-oriented channel creation at L2CAP. See Figure 11.

Accept this request with an MTU = 23, MPS = 23, and Initial Credits = 10.

Figure 11. Connection-Oriented channel request on CySmart



L2CAP Channel Request

☑ Accept the L2CAP channel request

MTU: 23

MPS: 23

Initial credit: 10

Request details

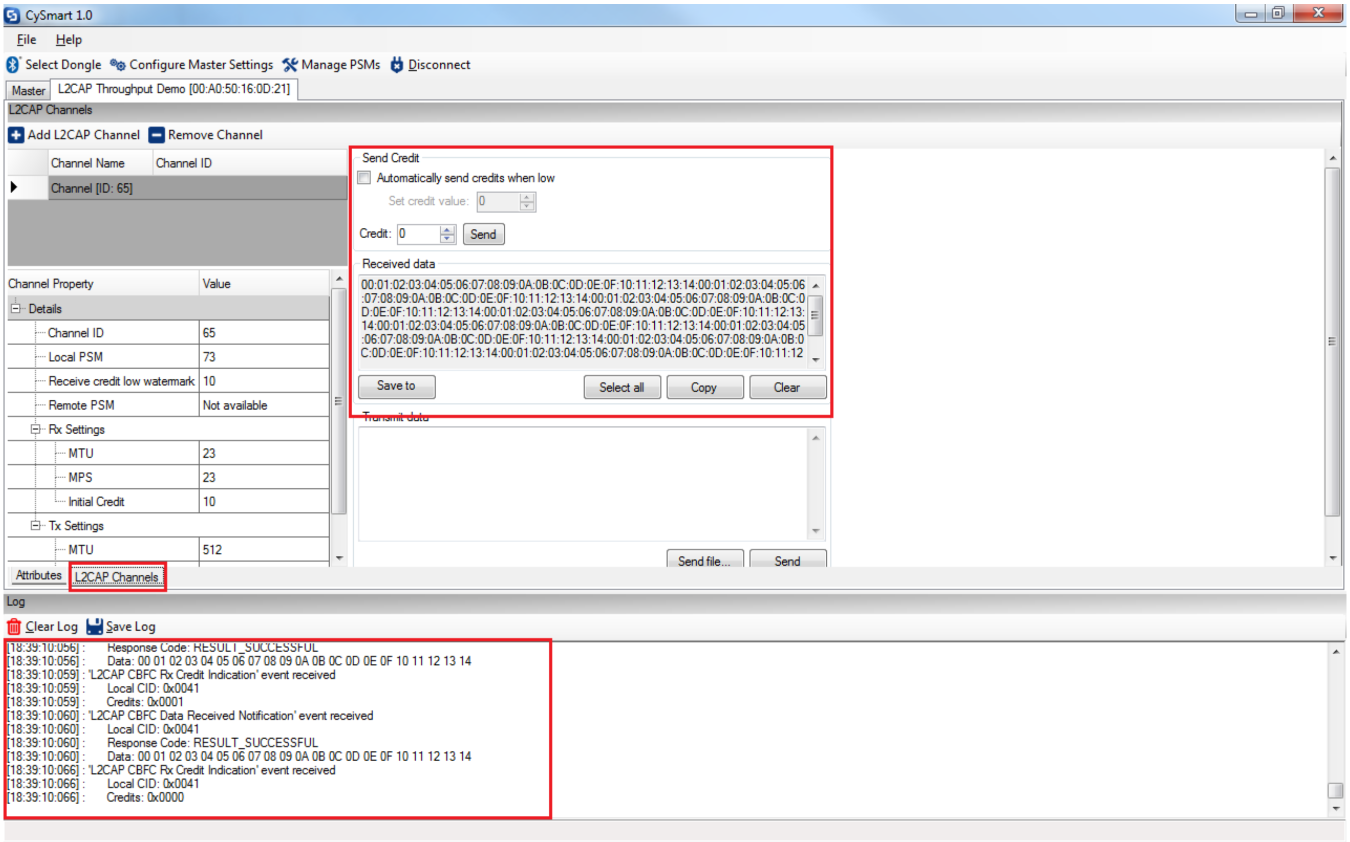
Channel ID:	65
Local PSM:	73
Remote MTU:	512
Remote MPS:	512
Initial credit:	0

Once you accept this request, you will get data packets (numbering as much as the Initial credit number) on the L2CAP layer. You can switch to **L2CAP Channels** tab in CySmart to see this data. See Figure 12.

Once the credits exhaust, you can send more credits to the device from the tool. You can also automate the credit flow by sending a particular number of credits whenever credits are low.

To stop the data flow you can either click on **Remove Channel** or you can **Disconnect** entirely.

Figure 12. Getting data on L2CAP channels in CySmart



Channel Name	Channel ID
Channel [ID: 65]	

Channel Property	Value
Details	
Channel ID	65
Local PSM	73
Receive credit low watermark	10
Remote PSM	Not available
Rx Settings	
MTU	23
MPS	23
Initial Credit	10
Tx Settings	
MTU	512

☐ Automatically send credits when low

Set credit value: 0

Credit: 0

Received data

```
00:01:02:03:04:05:06:07:08:09:0A:0B:0C:0D:0E:0F:10:11:12:13:14:00:01:02:03:04:05:06
:07:08:09:0A:0B:0C:0D:0E:0F:10:11:12:13:14:00:01:02:03:04:05:06:07:08:09:0A:0B:0C:0
D:0E:0F:10:11:12:13:14:00:01:02:03:04:05:06:07:08:09:0A:0B:0C:0D:0E:0F:10:11:12:13:
14:00:01:02:03:04:05:06:07:08:09:0A:0B:0C:0D:0E:0F:10:11:12:13:14:00:01:02:03:04:05
:06:07:08:09:0A:0B:0C:0D:0E:0F:10:11:12:13:14:00:01:02:03:04:05:06:07:08:09:0A:0B:0
C:0D:0E:0F:10:11:12:13:14:00:01:02:03:04:05:06:07:08:09:0A:0B:0C:0D:0E:0F:10:11:12
```

```
[18:39:10:056] :      Response Code: RESULT_SUCCESSFUL
[18:39:10:056] :      Data: 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14
[18:39:10:059] : 'L2CAP CBFC Rx Credit Indication' event received
[18:39:10:059] :      Local CID: 0x0041
[18:39:10:059] :      Credits: 0x0001
[18:39:10:060] : 'L2CAP CBFC Data Received Notification' event received
[18:39:10:060] :      Local CID: 0x0041
[18:39:10:060] :      Response Code: RESULT_SUCCESSFUL
[18:39:10:060] :      Data: 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14
[18:39:10:066] : 'L2CAP CBFC Rx Credit Indication' event received
[18:39:10:066] :      Local CID: 0x0041
[18:39:10:066] :      Credits: 0x0000
```

Testing Outgoing project with Incoming project

Similar to the GATT Notification example, the L2CAP outgoing project can be tested along with the incoming project to get similar numbers on the actual data throughput.

Reverting the Dongle Functionality back to Original

To revert to the original functionality of the USB Dongle, locate the Dongle hex file in the Kit installation directory. A typical installation path is:

C:\Program Files\Cypress\CY8CKIT-042-BLE Kit\1.0\Firmware\BLE Dongle\Hex Files\BLE_Dongle_CySmart.hex

Open PSoC Programmer and load this file in the tool. Then connect to the KitProg of the USB Dongle in PSoC Programmer and program the hex file.

Related Documents

Table 1 lists all relevant application notes, code examples, knowledge base articles, device datasheets, and Component / user module datasheets.

Table 1. Related Documents

Document	Title	Comment
AN91267	Getting Started with PSoC 4 BLE	Provides an introduction to PSoC 4 BLE device that integrates a Bluetooth Low Energy radio system along with programmable analog and digital resources.
AN91445	Antenna Design Guide	Provides guidelines on how to design an antenna for BLE applications.