WL18xx TI Bluetooth Stack GATM Demo App

This page is under construction

Return to WL18xx AM335x TI Bluetooth Stack

Contents

Demo Overview

Note: **: This instructions can be used to run this demo on the AM335x Platform.**

Generic Attribute Profile (GATT) is built on top of the Attribute Protocol (ATT) and establishes common operations and a framework for the data transported and stored by the Attribute Protocol. GATT defines two roles: Server and Client. The GATT roles are not necessarily tied to specific GAP roles and may be specified by higher layer profiles. GATT and ATT are not transport specific and can be used in both BR/EDR and LE. However, GATT and ATT are mandatory to implement in LE since it is used for discovering services.

This application allows the user to use a console to use Bluetooth Low Energy (BLE) to establish connection between two BLE devices, add and delete services , read, write, notify and indicate characteristics between the server and client.

It is recommended that the user visits the kit setup Getting Started Guide for AM335x (http://processors.wiki.ti.com/index.php/WL18xx_AM335x_TI_Bluetooth_Stack) pages before trying the application described on this page.

Running the Bluetooth Code

Once the am335x is initialized on the terminal, follow the instructions;

Note: When using a CC256x controller you will need to change the init script in order to enable BLE. You can use the following command in the AM335x device terminal

```
cp -f /lib/firmware/TIInit_6.7.16_ble_on.bts /lib/firmware/TIInit_6.7.16.bts (For CC256xB)
OR
cp -f /lib/firmware/TIInit_11.8.32_ble_on.bts /lib/firmware/TIInit_11.8.32.bts (For CC256xC)
```

Enter to the Sample Application folder, for this example “cd BluetopiaPM/bin”

```
root@am335x-evm:~# cd BluetopiaPM/bin/
root@am335x-evm:~/BluetopiaPM/bin#
```

Demo Application

This section provides a description of how to use the demo application to connect two configured boards and communicate over Bluetooth. GATT is a simple Client-Server connection process. We will setup one of the boards as a server and the other board as a client. We will then initiate a connection from the client to the server. Once connected, we can transmit data between the two devices over Bluetooth.

GENERIC ATTRIBUTE PROFILE (GATT) - Server role

> To verify the GATM Server role, first run the "LinuxGATM_SRV" executable, initialize, and power on the local device if necessary, and register for GATT callback events.

Start the Server and the Demo application by typing “./SS1BTPM & ./LinuxGATM_SRV”

```
root@am335x-evm:~/BluetopiaPM/bin# ./SS1BTPM & ./LinuxGATM_SRV
[1] 1789
```

Now the help screen is displayed

```
******************************************************************
* Command Options: 1) Initialize                                 *
*                  2) Cleanup                                    *
*                  3) QueryDebugZoneMask                         *
*                  4) SetDebugZoneMask                           *
*                  5) SetDebugZoneMaskPID                        *
*                  6) ShutdownService                            *
*                  7) RegisterEventCallback,                     *
*                  8) UnRegisterEventCallback,                   *
*                  9) QueryDevicePower                           *
*                  10)SetDevicePower                             *
*                  11)QueryLocalDeviceProperties                 *
*                  12)SetLocalDeviceName                         *
*                  13)SetLocalClassOfDevice                      *
*                  14)SetDiscoverable                            *
*                  15)SetConnectable                             *
*                  16)SetPairable                                *
*                  17)StartDeviceDiscovery                       *
*                  18)StopDeviceDiscovery                        *
*                  19)QueryRemoteDeviceList                      *
*                  20)QueryRemoteDeviceProperties                *
*                  21)AddRemoteDevice                            *
*                  22)DeleteRemoteDevice                         *
*                  23)UpdateRemoteDeviceAppData                  *
*                  24)DeleteRemoteDevices                        *
*                  25)PairWithRemoteDevice                       *
*                  26)CancelPairWithRemoteDevice                 *
*                  27)UnPairRemoteDevice                         *
*                  28)QueryRemoteDeviceServices                  *
*                  29)QueryRemoteDeviceServiceSupported          *
*                  30)QueryRemoteDevicesForService               *
*                  31)QueryRemoteDeviceServiceClasses            *
*                  32)AuthenticateRemoteDevice                   *
*                  33)EncryptRemoteDevice                        *
*                  34)ConnectWithRemoteDevice                    *
*                  35)DisconnectRemoteDevice                     *
*                  36)SetRemoteDeviceLinkActive                  *
*                  37)CreateSDPRecord                            *
*                  38)DeleteSDPRecord                            *
*                  39)AddSDPAttribute                            *
*                  40)DeleteSDPAttribute                         *
*                  41)EnableBluetoothDebug                       *
*                  42)RegisterAuthentication                     *
*                  43)UnRegisterAuthentication                   *
*                  44)PINCodeResponse                            *
*                  45)PassKeyResponse                            *
*                  46)UserConfirmationResponse                   *
*                  47)ChangeSimplePairingParameters              *
*                  48)RegisterGATTCallback                       *
*                  49)UnRegisterGATTCallback                     *
*                  50)QueryGATTConnections                       *
*                  51)SetLocalDeviceAppearance                   *
*                  52)StartAdvertising                           *
*                  53)StopAdvertising                            *
*                  54)RegisterService                            *
*                  55)UnRegisterService                          *
*                  56)IndicateCharacteristic                     *
*                  57)NotifyCharacteristic                       *
*                  58)ListCharacteristics                        *
*                  59)ListDescriptors                            *
*                  60)QueryPublishedServices                     *
*                  61)SetAdvertisingInterval                     *
*                  62)SetAuthenticatedPayloadTimeout             *
*                  63)QueryAuthenticatedPayloadTimeout           *
*                  64)EnableSCOnly                               *
*                  65)RegenerateP256LocalKeys                    *
*                  66)OOBGenerateParameters                      *
*                  67)ChangeLEPairingParameters                  *
*                  Help, Quit.                                   *
******************************************************************
```

a) Initialize the profile by typing “1 1” or “Initialize 1”.
b) Powering up the Bluetooth device by typing “10 1” or "SetDevicePower 1”, if the device is already powered up you’ll receive an error that the device is already powered up

```
GATM>Initialize 1

BTPM_Initialize() Success: 0.
DEVM_RegisterEventCallback() Success: 5.
GATM>SetDevicePower 1

echo 44 > /sys/class/gpio/export
sh: line 0: echo: write error: Device or resource busy
echo out > /sys/class/gpio/gpio44/direction
echo 0 > /sys/class/gpio/gpio44/value
echo 1 > /sys/class/gpio/gpio44/value
BT COMM PORT (/dev/ttyS4): 1
Changing HCI baud rate to 3000000
Status: Executing BTS Script /lib/firmware/TIInit_6.12.26.bts.
Status: BTS Script successfully executed.
DEVM_PowerOnDevice() Success: 0.
GATM>
Device Powered On.
GATM>
```

c) Now register Authentication.

```
GATM>RegisterAuthentication

DEVM_RegisterAuthentication() Success: 5.
GATM>
```

d) Now register for GATT callback events.

```
GATM>RegisterGATTCallback

GATM_RegisterEventCallback() Success: 1.
GATM>
```

e) Start advertising the server

Note: For this example we will use low duty cycle advertising to a Peer device that uses random address type while the own address type is also random address.

```
Usage: StartAdvertising [Flags] [Duration] [BD ADDR].
where Flags is a bitmask in the following table
Bitmask Number - [Bit is set]           [Bit is off]
    0x00000001 - [Local Public Address] [Local Random Address]
    0x00000002 - [Discoverable]         [Non Discoverable]
    0x00000004 - [Connectable]          [Non Connectable]
    0x00000010 - [Advertise Name]       [Advertise Name off]
    0x00000020 - [Advertise Tx Power]   [Advertise Tx Power off]
    0x00000040 - [Advertise Appearance] [Advertise Appearance off]
    0x00000100 - [Peer Public Address]  [Peer Random Address]

When Connectable bit (0x0004) is set:
    0x00000200 - [Direct Connectable]   [Undirect Connectable]

When Direct Connectable bit (0x0200) is set:
    0x00000400 - [Low Duty Cycle]       [High Duty Cycle]

Note: BD ADDR is needed only when Direct Connectable flags is in use.
```

```
GATM>StartAdvertising 0x676 100 62C9608E4F9B


Local Device Properties Changed.
Device Flags: 0x80020000
LE Scan Mode:    FALSE, 0x00000000
LE Adv Mode:     TRUE, 0x00000064
GATM>
LE Advertising Started.
GATM>DEVM_StartAdvertising() Success: Duration 100 seconds.
GATM>
```

f) Query device properties in order to discover the BD Address

```
GATM>QueryLocalDeviceProperties

DEVM_QueryLocalDeviceProperties() Success: 0.
BD_ADDR:      88C255CBEE4F
HCI Ver:      0x0007
HCI Rev:      0x0000
LMP Ver:      0x0007
LMP Sub Ver:  0x9A0A
Device Man:   0x000D (Texas Instruments Inc.)
Device Flags: 0x80020000
BLE Address Type: Resolvable Random
BLE BD_ADDR:      417129032882
COD:          0x1C0424
Device Name:  WL18xx Device
Disc. Mode:   TRUE , 0x00000000
Conn. Mode:   TRUE , 0x00000000
Pair. Mode:   TRUE , 0x00000000
LE Scan Mode:    FALSE, 0x00000000
LE Adv Mode:     TRUE, 0x00000064
GATM>
```

g) Initiate connection from the client. (Section f) in the client section).

```
Local Device Properties Changed.
Device Flags: 0x80000000
LE Scan Mode:    FALSE, 0x00000000
LE Adv Mode:     FALSE, 0x00000000
GATM>
LE Advertising Stopped.
GATM>
GATT Connection Event
    Connection Type: LE
    Remote Address:  62C9608E4F9B
    MTU:             23


GATM>
Remote Device Found.
BD_ADDR:             62C9608E4F9B
Address Type:        Resolvable Random Address.
Device Name:
LE Type:             Single Mode
Device Flags:        0x40008000
LE RSSI:             0
LE Paired State :    FALSE
LE Connect State:    TRUE
LE Encrypt State:    FALSE
GATT Services Known: FALSE
GATM>
```

```
Remote Device Properties Changed.
BD_ADDR:              62C9608E4F9B
Address Type:         Resolvable Random Address.
LE Type:              Single Mode
Device Flags:         0x40008000
LE Connect State:     TRUE
GATM>
```

Note: if needed pairing procedure can be done, description will be in the following steps, if pairing is not needed, jump to step XXX

h) change the pairing parameters of you choice

```
Usage: ChangeLEPairingParameters:
       [I/O Capability (0 = Display Only, 1 = Display Yes/No, 2 = Keyboard Only,
                        3 = No Input/Output, 4 = Keyboard Display )]
       [Bonding Type (0 = No Bonding, 1 = Bonding)]
       [MITM Requirement (0 = No, 1 = Yes)]
       [SC Enable (0 = No, 1 = Yes)]
       [P256DebugMode (0 = No, 1 = Yes)]
       [OOBSupport (0 = No - Default, 1 = Yes)]
       [Keypress (0 = No - Default, 1 = Yes)].
```

Note: For this example we will use, Display Yes/No, Bonding, MITM, SC Enable and no P256Debug Mode, the OOBSupport and Keypress will be set to their defaults

```
GATM>ChangeLEPairingParameters 1 1 1 1 0

Current I/O Capabilities 1: Display Yes No.
Bonding Mode 1: Bonding.
MITM Protection: TRUE
SC: TRUE
P256DebugMode FALSE
OOBSupport FALSE
Keypress FALSE.
GATM>
```

i) Initiate pairing from the client. (Section h) in the client section).

```
Authentication Request received for 62C9608E4F9B.
I/O Capability Request: LE.
DEVM_AuthenticationResponse() Success.
GATM>
Authentication Request received for 62C9608E4F9B.
Numeric Comparison Request LE.
User Confirmation: 670107

Respond with the command: UserConfirmationResponse
```

Note: Each pairing procedure may ask for different response according to the pairing parameters, in this example we will need to answer with "UserConfirmationResponse" with the parameter "1" in order to accept the pairing request when the User Confirmation value is the same in both sides.

```
GATM>UserConfirmationResponse 1

DEVM_AuthenticationResponse(), User Confirmation Response Success.
GATM>
Remote Device Properties Changed.
BD_ADDR:              62C9608E4F9B
Address Type:         Resolvable Random Address.
LE Type:              Single Mode
Device Flags:         0x40018000
LE Encrypt State:     TRUE
GATM>
Remote Device Address Changed.
Remote Device Address:          84DD2093482D.
Previous Remote Device Address: 62C9608E4F9B.
GATM>
Remote Device Properties Changed.
BD_ADDR:              84DD2093482D
Address Type:         Public
LE Type:              Single Mode
Resolv. BD_ADDR:      62C9608E4F9B

GATM>
LE Remote Device Pairing Status: 84DD2093482D, SUCCESS (0x80000000)
GATM>
Remote Device Properties Changed.
BD_ADDR:              84DD2093482D
Address Type:         Public
LE Type:              Single Mode
Device Flags:         0x4003C001
LE Paired State :     TRUE
GATM>
Remote Device 84DD2093482D Services Status: LE, SUCCESS.
GATM>
Remote Device Properties Changed.
BD_ADDR:              84DD2093482D
Address Type:         Public
LE Type:              Single Mode
Device Flags:         0x4003D001
GATT Services Known: TRUE
GATM>
Remote Device Properties Changed.
BD_ADDR:              84DD2093482D
Address Type:         Public
LE Type:              Single Mode
Device Flags:         0x4003F001
Device Appearance:    0
GATM>
Remote Device Properties Changed.
BD_ADDR:              84DD2093482D
Address Type:         Public
```

```
Device Name:         AM44UserConfirmationResponse
LE Type:             Single Mode
Device Flags:        0x4003F001
GATM>
```

j) When pairing procedure is complete and both controllers support the LE ping from Bluetooth specification 4.1, you will be able to change the authenticated payload timeout.

```
Usage: QueryAuthenticatedPayloadTimeout [BD_ADDR].
```

```
GATM>QueryAuthenticatedPayloadTimeout 84DD2093482D

Authenticated Payload Timeout: 30000 ms.
GATM>
```

```
Usage: SetAuthenticatedPayloadTimeout [BD_ADDR] [Authenticated Payload Timout (In ms)].
```

```
GATM>SetAuthenticatedPayloadTimeout 84DD2093482D5000

DEVM_SetAuthenticatedPayloadTimeout() Success.
GATM>
```

k) If needed, answer to user confirmation response

```
GATM>UserConfirmationResponse 1
DEVM_AuthenticationResponse(), User Confirmation Response Success.
GATM>
Remote Device Properties Changed.
BD_ADDR:             84DD209C58DD
Address Type:        Public
LE Type:             Dual Mode
Device Flags:        0xC0018005
LE Encrypt State:    TRUE
GATM>
LE Remote Device Pairing Status: 84DD209C58DD, SUCCESS (0x80000000)
GATM>
Remote Device Properties Changed.
BD_ADDR:             84DD209C58DD
Address Type:        Public
LE Type:             Dual Mode
Device Flags:        0xC001C005
LE Paired State :    TRUE
GATM>
Remote Device 84DD209C58DD Services Status: LE, SUCCESS.
GATM>
Remote Device Properties Changed.
BD_ADDR:             84DD209C58DD
Address Type:        Public
LE Type:             Dual Mode
Device Flags:        0xC001D005
GATT Services Known: TRUE
GATM>
Remote Device Properties Changed.
BD_ADDR:             84DD209C58DD
Address Type:        Public
LE Type:             Dual Mode
Device Flags:        0xC001F005
Device Appearance:   0
GATM>
```

l) After the connection was established, Register services in order to work with them

```
Usage: RegisterService [Service Index (0 - 5)].
```

```
GATM>RegisterService 0

 Registering Service, UUID: BC65B80102957A8B1410526C8154FE6A
 Registered Service, Service ID: 1.
 Registered Characteristic, UUID: BC65B80102957A8B1410526C414DFE6A
 Registered Characteristic, UUID: BC65B80102957A8B1410526C9651FE6A
 Registered Characteristic, UUID: BC65B80102957A8B1410526C8B52FE6A
 Service Registered.
*****************************************************************
   Service Start Handle: 0x4000
   Service End Handle:   0x4006

 Characteristic: : BC65B80102957A8B1410526C414DFE6A
Characteristic Declaration @ Handle: 0x4001
Characteristic Value       @ Handle: 0x4002
 Characteristic: : BC65B80102957A8B1410526C9651FE6A
Characteristic Declaration @ Handle: 0x4003
Characteristic Value       @ Handle: 0x4004
 Characteristic: : BC65B80102957A8B1410526C8B52FE6A
Characteristic Declaration @ Handle: 0x4005
Characteristic Value       @ Handle: 0x4006
*****************************************************************
```

```
GATM>RegisterService 1

Registered Persistent UID: 0x02030212.
Start Handle:              0x0203.
End Handle:                0x0212.
 Registering Service, UUID: BC65B80102957A8B1410526C6568FE6A
 Registered Service, Service ID: 2.
 Registered Include to Service ID 1.
 Registered Characteristic, UUID: BC65B80102957A8B1410526CA355FE6A
```

```
 Registered Characteristic, UUID: BC65B80102957A8B1410526C8956FE6A
 Registered Characteristic, UUID: BC65B80102957A8B1410526C405CFE6A
 Registered Descriptor, UUID: BC65B80102957A8B1410526C2A5DFE6A
 Registered Characteristic, UUID: BC65B80102957A8B1410526C325EFE6A
 Registered Characteristic, UUID: BC65B80102957A8B1410526C0163FE6A
 Registered Characteristic, UUID: BC65B80102957A8B1410526CCA66FE6A
 Registered Descriptor, UUID: BC65B80102957A8B1410526C9A67FE6A
 Service Registered.
*****************************************************************
   Service Start Handle: 0x0203
   Service End Handle:   0x0212

Include Attribute @ Handle: 0x0204
 Characteristic: : BC65B80102957A8B1410526CA355FE6A
Characteristic Declaration @ Handle: 0x0205
Characteristic Value       @ Handle: 0x0206
 Characteristic: : BC65B80102957A8B1410526C8956FE6A
Characteristic Declaration @ Handle: 0x0207
Characteristic Value       @ Handle: 0x0208
 Characteristic: : BC65B80102957A8B1410526C405CFE6A
Characteristic Declaration @ Handle: 0x0209
Characteristic Value       @ Handle: 0x020A
 Characteristic Descriptor: : BC65B80102957A8B1410526C2A5DFE6A
Characteristic Descriptor @ Handle: 0x020B
 Characteristic: : BC65B80102957A8B1410526C325EFE6A
Characteristic Declaration @ Handle: 0x020C
Characteristic Value       @ Handle: 0x020D
 Characteristic: : BC65B80102957A8B1410526C0163FE6A
Characteristic Declaration @ Handle: 0x020E
Characteristic Value       @ Handle: 0x020F
 Characteristic: : BC65B80102957A8B1410526CCA66FE6A
Characteristic Declaration @ Handle: 0x0210
Characteristic Value       @ Handle: 0x0211
 Characteristic Descriptor: : BC65B80102957A8B1410526C9A67FE6A
Characteristic Descriptor @ Handle: 0x0212
*****************************************************************
GATM>
```

m) Now that services were created, check the properties of the handles.

```
GATM>ListCharacteristics

Key:
x (xx), where X is the operation and XX are any security flags
ENC = Encryption required, MITM = Man in the Middle protection required

W  = Writable
WO = Write without Response
R  = Readable
N  = Notifiable
I  = Indicatable
S  = Signed Writes allowed

Characteristic Handle:     0x4002 (16386)
Characteristic Properties: R, W (ENC,MITM), I.
Characteristic Handle:     0x4004 (16388)
Characteristic Properties: R (ENC), W (ENC), S (MITM), N.
Characteristic Handle:     0x4006 (16390)
Characteristic Properties: W.
Characteristic Handle:     0x0206 (518)
Characteristic Properties: R (ENC,MITM), W (ENC), WO (ENC).
Characteristic Handle:     0x0208 (520)
Characteristic Properties: R, WO.
Characteristic Handle:     0x020A (522)
Characteristic Properties: R (ENC), W (ENC).
Characteristic Handle:     0x020D (525)
Characteristic Properties: R (ENC), W (ENC).
Characteristic Handle:     0x020F (527)
Characteristic Properties: R (ENC), WO (ENC,MITM).
Characteristic Handle:     0x0211 (529)
Characteristic Properties: W (ENC).
GATM>
```

n) Now check the properties of the Characteristics

```
GATM>ListDescriptors

Key:
x (xx), where X is the operation and XX are any security flags
ENC = Encryption required, MITM = Man in the Middle protection required

W  = Writable
R  = Readable

Descriptor Handle:     0x020B (523)
Descriptor Properties: R (ENC).
Descriptor Handle:     0x0212 (530)
Descriptor Properties: R.
GATM>
```

o) When the client tries to write value to handle with write properties you'll see (Section j) in the client section)

```
GATT Write Request Event
    Connection Type:  LE
    Remote Address:   84DD209C58DD
    Service ID:       2
    Request ID:       10
    Attribute Offset: 6
    Data Length:      1
    Value:
0F
GATM_WriteResponse() success.
```

p) Now try to notify characteristic from the server to the client

```
Usage: NotifyCharacteristic [Service Index (0 - 5)] [Attribute Offset] [BD_ADDR].
Valid usages:
   NotifyCharacteristic 0 3 [BD_ADDR] (Attribute Handle is 0x4004 (16388)
```

```
GATM>NotifyCharacteristic 0 3 84DD209C58DD

Notification sent successfully.
GATM>
```

q) Now try to indicate characteristic from the server to the client

```
Usage: IndicateCharacteristic [Service Index (0 - 5)] [Attribute Offset] [BD_ADDR].
Valid usages:
   IndicateCharacteristic 0 1 [BD_ADDR] (Attribute Handle is 0x4002 (16386)
```

```
GATM>IndicateCharacteristic 0 1 84DD209C58DD

Sent Indication, TransactionID 1.
GATM>
GATT Handle-Value Confirmation Event
    Connection Type:  LE
    Remote Address:   84DD209C58DD
    Service ID:       1
    Request ID:       1
    Attribute Offset: 2
    Status:           0
GATM>
```

GENERIC ATTRIBUTE PROFILE (GATT) - Client role

To verify the GATM Server role, first run the "LinuxGATM_CLT" executable, initialize, and power on the local device if necessary, and register for GATT callback events.

Start the Server and the Demo application by typing “./SS1BTPM & ./LinuxGATM_CLT”

```
root@am335x-evm:~/BluetopiaPM/bin# ./SS1BTPM & ./LinuxGATM_CLT
[1] 1789
```

Now the help screen is displayed

```
****************************************************************
* Command Options: 1) Initialize                               *
*                  2) Cleanup                                  *
*                  3) QueryDebugZoneMask                       *
*                  4) SetDebugZoneMask                         *
*                  5) SetDebugZoneMaskPID                      *
*                  6) ShutdownService                          *
*                  7) RegisterEventCallback,                   *
*                  8) UnRegisterEventCallback,                 *
*                  9) QueryDevicePower                         *
*                  10)SetDevicePower                           *
*                  11)QueryLocalDeviceProperties               *
*                  12)SetLocalDeviceName                       *
*                  13)SetLocalClassOfDevice                    *
*                  14)SetDiscoverable                          *
*                  15)SetConnectable                           *
*                  16)SetPairable                              *
*                  17)StartDeviceDiscovery                     *
*                  18)StopDeviceDiscovery                      *
*                  19)QueryRemoteDeviceList                    *
*                  20)QueryRemoteDeviceProperties              *
*                  21)AddRemoteDevice                          *
*                  22)DeleteRemoteDevice                       *
*                  23)UpdateRemoteDeviceAppData                *
*                  24)DeleteRemoteDevices                      *
*                  25)PairWithRemoteDevice                     *
*                  26)CancelPairWithRemoteDevice               *
*                  27)UnPairRemoteDevice                       *
*                  28)QueryRemoteDeviceServices                *
*                  29)QueryRemoteDeviceServiceSupported        *
*                  30)QueryRemoteDevicesForService             *
*                  31)QueryRemoteDeviceServiceClasses          *
*                  32)AuthenticateRemoteDevice                 *
*                  33)EncryptRemoteDevice                      *
*                  34)ConnectWithRemoteDevice                  *
*                  35)DisconnectRemoteDevice                   *
*                  36)SetRemoteDeviceLinkActive                *
*                  37)CreateSDPRecord                          *
*                  38)DeleteSDPRecord                          *
*                  39)AddSDPAttribute                          *
*                  40)DeleteSDPAttribute                       *
*                  41)EnableBluetoothDebug                     *
*                  42)RegisterAuthentication                   *
*                  43)UnRegisterAuthentication                 *
*                  44)PINCodeResponse                          *
*                  45)PassKeyResponse                          *
*                  46)UserConfirmationResponse                 *
*                  47)ChangeSimplePairingParameters            *
*                  48)RegisterGATTCallback                     *
*                  49)UnRegisterGATTCallback                   *
```

```
*                 50)QueryGATTConnections                    *
*                 51)ReadValue                               *
*                 52)WriteValue                              *
*                 53)WriteValueCommand                       *
*                 54)SetLocalDeviceAppearance                *
*                 55)StartAdvertising                        *
*                 56)StopAdvertising                         *
*                 57)SetAdvertisingInterval                  *
*                 58)SetAndUpdateConnectionParameters        *
*                 59)SetAuthenticatedPayloadTimeout          *
*                 60)QueryAuthenticatedPayloadTimeout        *
*                 61)EnableSCOnly                            *
*                 62)RegenerateP256LocalKeys                 *
*                 63)OOBGenerateParameters                   *
*                 64)ChangeLEPairingParameters               *
*                 Help, Quit.                                *
*************************************************************
```

a) Initialize the profile by typing “1 1” or “Initialize 1”.
b) Powering up the Bluetooth device by typing “10 1” or "SetDevicePower 1”, if the device is already powered up you’ll receive an error that the device is already powered up

```
GATM>Initialize 1

BTPM_Initialize() Success: 0.
DEVM_RegisterEventCallback() Success: 5.
GATM>SetDevicePower 1

echo 44 > /sys/class/gpio/export
sh: line 0: echo: write error: Device or resource busy
echo out > /sys/class/gpio/gpio44/direction
echo 0 > /sys/class/gpio/gpio44/value
echo 1 > /sys/class/gpio/gpio44/value
BT COMM PORT (/dev/ttyS4): 1
Changing HCI baud rate to 3000000
Status: Executing BTS Script /lib/firmware/TIInit_6.12.26.bts.
Status: BTS Script successfully executed.
DEVM_PowerOnDevice() Success: 0.
GATM>
Device Powered On.
GATM>
```

c) Now register Authentication.

```
GATM>RegisterAuthentication

DEVM_RegisterAuthentication() Success: 5.
GATM>
```

d) Now register for GATT callbock events.

```
GATM>RegisterGATTCallback

GATM_RegisterEventCallback() Success: 1.
GATM>
```

e) Query device properties in order to discover the BD Address

```
GATM>QueryLocalDeviceProperties

DEVM_QueryLocalDeviceProperties() Success: 0.
BD_ADDR:      84DD2093482D
HCI Ver:      0x0007
HCI Rev:      0x0000
LMP Ver:      0x0007
LMP Sub Ver:  0xAC0A
Device Man:   0x000D (Texas Instruments Inc.)
Device Flags: 0x80000000
BLE Address Type: Resolvable Random
BLE BD_ADDR:      62C9608E4F9B
COD:          0x1C0424
Device Name:  AM44UserConfirmationResponse
Disc. Mode:   TRUE , 0x00000000
Conn. Mode:   TRUE , 0x00000000
Pair. Mode:   TRUE , 0x00000000
LE Scan Mode:    FALSE, 0x00000000
LE Adv Mode:     FALSE, 0x00000000
GATM>
```

f) Change the connection and scan parameters

Note: You can change the default connection and scan parameters before initiating the connection, or changing also the current connection parameters when the connection exists.

Note: This function can change the Connection Scan parameters and/or the Scan parameters if needed.

```
Usage: SetAndUpdateConnectionAndScanBLEParameters
       [Remote BD ADDR] [Address Type] (0 - Public, 1 - Static, 2 - Private Resolvable,
                                        3 - Private NonResolvable)
       [Connection_Interval_Min] (Range: 10..4000 in ms) [Connection_Interval_Max] (Range: 20..4000 in ms)
       [Slave_Latency] (Range: 0..500 in ms)[Supervision_Timeout] (Range: 100..32000 in ms)
       [ConnectionScanInterval] [ConnectionScanWindow] (Range: 3..10240 in ms)
       [ScanInterval] [ScanWindow] (Range: 3..10240 in ms)
SetAndUpdateConnectionAndScanBLEParameters default parameters: 20 40 0 5000 100 100 20 60
```

```
GATM>SetAndUpdateConnectionAndScanBLEParameters 0 0 100 200 0 7500 200 200 10 10
```

```
Only Default parameters stored, Bluetooth Device doesn't exists Error: -10041.
```

Note: When using this function before initiating the connection, it will change the default connection parameters and also will return a numeric error that will tell that the device doesn't exists.

g) Initiate connection to the server

```
Usage: ConnectWithRemoteDevice [BD_ADDR] [Connect LE (1 = LE, 0 = BR/EDR)] [ConnectFlags (Optional)].
where ConnectFlags is a bitmask of
    0x00000001 = Authenticate
    0x00000002 = Encrypt
    0x00000004 = Use Local Random Address
    0x00000008 = Use Peer Random Address
```

```
GATM>ConnectWithRemoteDevice 4EDCA28AB166 1 0x0c

Attempting to Connect With (LE) Remote Device: 4EDCA28AB166 (Flags = 0x8000000C).

Remote Device Found.
BD_ADDR:              4EDCA28AB166
Address Type:         Public
Device Name:
LE Type:              Single Mode
Device Flags:         0x40000000
LE RSSI:              0
LE Paired State :     FALSE
LE Connect State:     FALSE
LE Encrypt State:     FALSE
GATT Services Known: FALSE
GATM>DEVM_ConnectWithRemoteDevice() Success: 0.
GATM>
Remote Device Connection Status: 4EDCA28AB166, 0 (SUCCESS)
GATM>
Remote Device Properties Changed.
BD_ADDR:              4EDCA28AB166
Address Type:         Static
LE Type:              Single Mode
Device Flags:         0x40048000
LE Connect State:     TRUE
GATM>
GATT Connection
    Connection Type: LE
    Remote Address:  4EDCA28AB166
    MTU:             23


GATM>
```

Note: if needed pairing procedure can be done, description will be in the following steps, if pairing is not needed, jump to step j

h) change the pairing parameters of you choice

```
Usage: ChangeLEPairingParameters:
       [I/O Capability (0 = Display Only, 1 = Display Yes/No, 2 = Keyboard Only,
                        3 = No Input/Output, 4 = Keyboard Display )]
       [Bonding Type (0 = No Bonding, 1 = Bonding)]
       [MITM Requirement (0 = No, 1 = Yes)]
       [SC Enable (0 = No, 1 = Yes)]
       [P256DebugMode (0 = No, 1 = Yes)]
       [OOBSupport (0 = No - Default, 1 = Yes)]
       [Keypress (0 = No - Default, 1 = Yes)].
```

Note: For this example we will use, Display Yes/No, Bonding, MITM, SC Enable and no P256Debug Mode, the OOBSupport and Keypress will be set to their defaults

```
GATM>ChangeLEPairingParameters 1 1 1 1 0

Current I/O Capabilities 1: Display Yes No.
Bonding Mode 1: Bonding.
MITM Protection: TRUE
SC: TRUE
P256DebugMode FALSE
OOBSupport FALSE
Keypress FALSE.
GATM>
```

i) After changing the pairing parameters initiate the pairing procedure

```
Usage: PairWithRemoteDevice [BD_ADDR] [Type (1 = LE, 0 = BR/EDR)] [Pair Flags (optional)].
```

```
GATM>PairWithRemoteDevice 417129032882 1

Attempting to Pair With Remote LE Device: 417129032882.

Authentication Request received for 417129032882.
I/O Capability Request: LE.
DEVM_AuthenticationResponse() Success.
GATM>DEVM_PairWithRemoteDevice() Success: 0.
GATM>
Authentication Request received for 417129032882.
Numeric Comparison Request LE.
User Confirmation: 670107

Respond with the command: UserConfirmationResponse
GATM>
```

Note: Each pairing procedure may ask for different response according to the pairing parameters, in this example we will need to answer with "UserConfirmationResponse" with the parameter "1" in order to accept the pairing request when the User Confirmation value is the same in both sides.

```
GATM>UserConfirmationResponse 1

DEVM_AuthenticationResponse(), User Confirmation Response Success.
GATM>
Remote Device Properties Changed.
BD_ADDR:              417129032882
Address Type:         Static
LE Type:              Single Mode
Device Flags:         0x40058000
LE Encrypt State:     TRUE
GATM>
Remote Device Address Changed.
Remote Device Address:          88C255CBEE4F.
Previous Remote Device Address: 417129032882.
GATM>
Remote Device Properties Changed.
BD_ADDR:              88C255CBEE4F
Address Type:         Public
LE Type:              Single Mode
Resolv. BD_ADDR:      417129032882

GATM>
LE Remote Device Pairing Status: 88C255CBEE4F, SUCCESS (0x80000000)
GATM>
Remote Device Properties Changed.
BD_ADDR:              88C255CBEE4F
Address Type:         Public
LE Type:              Single Mode
Device Flags:         0x4007C001
LE Paired State :     TRUE
GATM>
Remote Device 88C255CBEE4F Services Status: LE, SUCCESS.
GATM>
Remote Device Properties Changed.
BD_ADDR:              88C255CBEE4F
Address Type:         Public
LE Type:              Single Mode
Device Flags:         0x4007D001
GATT Services Known: TRUE
GATM>
Remote Device Properties Changed.
BD_ADDR:              88C255CBEE4F
Address Type:         Public
LE Type:              Single Mode
Device Flags:         0x4007F001
Device Appearance:    0
GATM>
Remote Device Properties Changed.
BD_ADDR:              88C255CBEE4F
Address Type:         Public
Device Name:          WL18xx Device
LE Type:              Single Mode
Device Flags:         0x4007F001
GATM>
```

j) When pairing procedure is complete and both controllers support the LE ping from Bluetooth specification 4.1, you will be able to change the authenticated payload timeout.

```
Usage: QueryAuthenticatedPayloadTimeout [BD_ADDR].
```

```
SetAuthenticatedPayloadTimeout 88C255CBEE4F 2000
```

```
Usage: SetAuthenticatedPayloadTimeout [BD_ADDR] [Authenticated Payload Timout (In ms)].
```

```
QueryAuthenticatedPayloadTimeout 88C255CBEE4F

Authenticated Payload Timeout: 2000 ms.
GATM>
```

k) Now search for services in the Server side. (This search will show the default services, before we add additional services to the server).

```
Usage: QueryRemoteDeviceServices [BD_ADDR] [Type (1 = LE, 0 = BR/EDR)] [Force Update] [Bytes to Query (specified if Force is 0)].
```

```
GATM>QueryRemoteDeviceServices 84DD2069DEBC 1 1 0

Attempting Query Remote Device 84DD2069DEBC For GATT Services.

Remote Device Properties Changed.
BD_ADDR:              84DD2069DEBC
Address Type:         Public
LE Type:              Dual Mode
GATT Services Known: FALSE
GATM>DEVM_QueryRemoteDeviceServices() Success: 0, Total Number Service Bytes: 0.
GATM>
Remote Device 84DD2069DEBC Services Status: LE, SUCCESS.
GATM>
Remote Device Properties Changed.
BD_ADDR:              84DD2069DEBC
Address Type:         Public
LE Type:              Dual Mode
Device Flags:         0xC005F005
GATT Services Known: TRUE
GATM>
Remote Device Properties Changed.
BD_ADDR:              84DD2069DEBC
```

```
Address Type:        Public
LE Type:             Dual Mode
Device Flags:        0xC005F005
Device Appearance:   0
GATM>
```

```
GATM>QueryRemoteDeviceServices 84DD2069DEBC 1 0 2000
Attempting Query Remote Device 84DD2069DEBC For GATT Services.
DEVM_QueryRemoteDeviceServices() Success: 320, Total Number Service Bytes: 320.
Returned Service Data (320 Bytes):...

Number of Services: 4
 Service UUID: 1801
    Start Handle: 0x0001 (1)
    End Handle:   0x0001 (1)
 Service UUID: 1800
    Start Handle: 0x0040 (64)
    End Handle:   0x0044 (68)
       Characteristic UUID: 2A00
       Handle:     0x0042 (66)
       Properties: 0x02
       Characteristic UUID: 2A01
       Handle:     0x0044 (68)
       Properties: 0x02
 Service UUID: 180A
    Start Handle: 0x0100 (256)
    End Handle:   0x0102 (258)
       Characteristic UUID: 2A50
       Handle:     0x0102 (258)
       Properties: 0x02
 Service UUID: 1802
    Start Handle: 0x0200 (512)
    End Handle:   0x0202 (514)
       Characteristic UUID: 2A06
       Handle:     0x0202 (514)
       Properties: 0x04
GATM>
```

l) After Registering the services in the server query the services again. (Section l) in the server section)

```
GATM>QueryRemoteDeviceServices 84DD2069DEBC 1 1 0

Attempting Query Remote Device 84DD2069DEBC For GATT Services.

Remote Device Properties Changed.
BD_ADDR:             84DD2069DEBC
Address Type:        Public
LE Type:             Dual Mode
GATT Services Known: FALSE
GATM>DEVM_QueryRemoteDeviceServices() Success: 0, Total Number Service Bytes: 0.
GATM>
GATM>
Remote Device 84DD2069DEBC Services Status: LE, SUCCESS.
GATM>
Remote Device Properties Changed.
BD_ADDR:             84DD2069DEBC
Address Type:        Public
LE Type:             Dual Mode
Device Flags:        0xC005F005
GATT Services Known: TRUE
GATM>
Remote Device Properties Changed.
BD_ADDR:             84DD2069DEBC
Address Type:        Public
LE Type:             Dual Mode
Device Flags:        0xC005F005
Device Appearance:   0
GATM>
```

```
GATM>QueryRemoteDeviceServices 84DD2069DEBC 1 0 2000

Attempting Query Remote Device 84DD2069DEBC For GATT Services.
DEVM_QueryRemoteDeviceServices() Success: 768, Total Number Service Bytes: 768.
Returned Service Data (768 Bytes):...

Number of Services: 6
 Service UUID: 1801
    Start Handle: 0x0001 (1)
    End Handle:   0x0001 (1)
 Service UUID: 1800
    Start Handle: 0x0040 (64)
    End Handle:   0x0044 (68)
       Characteristic UUID: 2A00
       Handle:     0x0042 (66)
       Properties: 0x02
       Characteristic UUID: 2A01
       Handle:     0x0044 (68)
       Properties: 0x02
 Service UUID: 180A
    Start Handle: 0x0100 (256)
    End Handle:   0x0102 (258)
       Characteristic UUID: 2A50
       Handle:     0x0102 (258)
       Properties: 0x02
 Service UUID: 1802
    Start Handle: 0x0200 (512)
    End Handle:   0x0202 (514)
       Characteristic UUID: 2A06
       Handle:     0x0202 (514)
       Properties: 0x04
 Service UUID: BC65B80102957A8B1410526C6568FE6A
    Start Handle: 0x0203 (515)
    End Handle:   0x0212 (530)
       Included Service UUID: BC65B80102957A8B1410526C8154FE6A
       Start Handle: 0x4000 (16384)
       End Handle:   0x4006 (16390)
```

```
      Characteristic UUID: BC65B80102957A8B1410526CA355FE6A
      Handle:     0x0206 (518)
      Properties: 0x0E
      Characteristic UUID: BC65B80102957A8B1410526C8956FE6A
      Handle:     0x0208 (520)
      Properties: 0x06
      Characteristic UUID: BC65B80102957A8B1410526C405CFE6A
      Handle:     0x020A (522)
      Properties: 0x0A

         Descriptor UUID: BC65B80102957A8B1410526C2A5DFE6A
         Handle:     0x020B (523)

      Characteristic UUID: BC65B80102957A8B1410526C325EFE6A
      Handle:     0x020D (525)
      Properties: 0x0A
      Characteristic UUID: BC65B80102957A8B1410526C0163FE6A
      Handle:     0x020F (527)
      Properties: 0x06
      Characteristic UUID: BC65B80102957A8B1410526CCA66FE6A
      Handle:     0x0211 (529)
      Properties: 0x08

         Descriptor UUID: BC65B80102957A8B1410526C9A67FE6A
         Handle:     0x0212 (530)

 Service UUID: BC65B80102957A8B1410526C8154FE6A
    Start Handle: 0x4000 (16384)
    End Handle:   0x4006 (16390)
       Characteristic UUID: BC65B80102957A8B1410526C414DFE6A
       Handle:     0x4002 (16386)
       Properties: 0x2A
       Characteristic UUID: BC65B80102957A8B1410526C9651FE6A
       Handle:     0x4004 (16388)
       Properties: 0x5A
       Characteristic UUID: BC65B80102957A8B1410526C8B52FE6A
       Handle:     0x4006 (16390)
       Properties: 0x08
GATM>
```

Note: You can see that now we have 6 services. earlier we had 4.

m) After knowing the services and their properties, try to read a value.

```
Usage: ReadValue [BD_ADDR] [AttributeHandle] [Offset] [ReadAll (Boolean)] [DisplayAsString (Optional)].
```

```
GATM>ReadValue 84DD2069DEBC 0x0208 0 1

GATM_ReadValue() Success: Transaction ID = 5.
GATM>
GATT Read Response Data
    Connection Type:  LE
    Remote Address:   84DD2069DEBC
    Transaction ID:   5
    Attribute Handle: 0x0208 (520)
    Final?:           YES
    Value Length:     36
    Value:
72 6C 67 7A 7A 52 7A 74 54 43 71 6E 43 42 42 72 73 6D 7E 21 25 5E 26 2A 28 29 2D 5F 2B 3D 7B 7D 5B 5D 7C 00

GATM>
```

n) Writing a value will be as following

```
Usage: WriteValue [BD_ADDR] [AttributeHandle] [Format (0 = ASCII String, 1 = Hex String)] [Value].
```

```
GATM>WriteValue 84DD2069DEBC 0x020A 1 0x0f

GATM_WriteValue() Success: Transaction ID = 9.
GATM>
GATT Write Response Data
    Connection Type:  LE
    Remote Address:   84DD2069DEBC
    Transaction ID:   9
    Attribute Handle: 0x020A (522)

GATM>
```

o) check if the write was successful

```
GATM>ReadValue 84DD2069DEBC 0x020A 0 1

GATM_ReadValue() Success: Transaction ID = 10.
GATM>
GATT Read Response Data
    Connection Type:  LE
    Remote Address:   84DD2069DEBC
    Transaction ID:   10
    Attribute Handle: 0x020A (522)
    Final?:           YES
    Value Length:     1
    Value:
0F

GATM>
```

p) When the server notifies a value the client will receive (Section p) on the server section)

```
GATT Handle Value Data
    Connection Type:  LE
    Remote Address:   84DD2069DEBC
    Type:             Notification
    Attribute Handle: 0x4004 (16388)
    Value Length:     20
    Value:
48 65 6C 6C 6F 20 57 6F 72 6C 64 2C 20 46 72 6F 6D 20 47 41

GATM>
```

m) When the server indicates a value the client will receive (Section q) on the server section)

```
GATT Handle Value Data
    Connection Type:  LE
    Remote Address:   84DD2069DEBC
    Type:             Indication
    Attribute Handle: 0x4002 (16386)
    Value Length:     20
    Value:
56 55 48 68 66 43 51 44 4C 76 42 7E 21 25 5E 26 2A 28 29 2D

GATM>
```

{{

1. switchcategory:MultiCore=

- For technical support on MultiCore devices, please post your questions in the C6000 MultiCore Forum
- For questions related to the BIOS MultiCore SDK (MCSDK), please use the BIOS Forum

Please post only comments related to the article **WL18xx TI Bluetooth Stack GATM Demo App** here.

Keystone=

- For technical support on MultiCore devices, please post your questions in the C6000 MultiCore Forum
- For questions related to the BIOS MultiCore SDK (MCSDK), please use the BIOS Forum

Please post only comments related to the article **WL18xx TI Bluetooth Stack GATM Demo App** here.

C2000=*For technical support on the C2000 please post your questions on The C2000 Forum. Please post only comments about the article **WL18xx TI Bluetooth Stack GATM Demo App** here.*

DaVinci=*For technical support on DaVincoplease post your questions on The DaVinci Forum. Please post only comments about the article **WL18xx TI Bluetooth Stack GATM Demo App** here.*

MSP430=*For technical support on MSP430 please post your questions on The MSP430 Forum. Please post only comments about the article **WL18xx TI Bluetooth Stack GATM Demo App** here.*

OMAP35x=*For technical support on OMAP please post your questions on The OMAP Forum. Please post only comments about the article **WL18xx TI Bluetooth Stack GATM Demo App** here.*

OMAPL1=*For technical support on OMAP please post your questions on The OMAP Forum. Please post only comments about the article **WL18xx TI Bluetooth Stack GATM Demo App** here.*

MAVRK=*For technical support on MAVRK please post your questions on The MAVRK Toolbox Forum. Please post only comments about the article **WL18xx TI Bluetooth Stack GATM Demo App** here.*

*For technical su please post you questions at http://e2e.ti.com Please post onl comments abou article **WL18xx Bluetooth Stac GATM Demo A** here.*

}}

Links

Amplifiers & Linear
Audio
Broadband RF/IF & Digital Radio
Clocks & Timers
Data Converters
DLP & MEMS
High-Reliability
Interface
Logic
Power Management
Processors

- ARM Processors
- Digital Signal Processors (DSP)
- Microcontrollers (MCU)
- OMAP Applications Processors

Switches & Multiplexers
Temperature Sensors & Control ICs
Wireless Connectivity