STMicroelectronics SensorTile Tutorial: Introduction to Bluetooth Low Energy (BLE) Interfaces

Table of Contents

1. Introduction to This Tutorial

SensorTile has a Bluetooth module (BLUENRG-MS) to support the BLE communication. In the tutorial, we will introduce the BLE_SampleApp project in the starter firmware package to explore the BLE communication on SensorTile.

The Tutorial steps provide:

1. An introduction to installation of Bluetooth Low Energy (BLE) starter firmware.
2. An introduction to BLE connection to SensorTile via BlueZ on Linux system.

For more information regarding the SensorTile board, please open the following link.
www.st.com/sensortile

1.1. List of Required Equipment and Materials

1) 1x STMicroelectronics SensorTile kit.
2) 1x STMicroelectronics Nucleo Board.
3) 1x Personal Computer with two USB type-A inputs OR you must have a powered USB hub.
4) 1x USB 2.0 A-Male to Micro-B Cable (micro USB cable).
5) 1x USB 2.0 A-Male to Mini-B Cable (mini USB cable).
6) Network access to the Internet.
7) A Linux machine (Beaglebone, Intel Edison, Rpi, and etc)

1.2. Prerequisite Tutorials

It is recommended that users have completed and are familiar with the contents of the following tutorials before proceeding.

1. All previous tutorials.

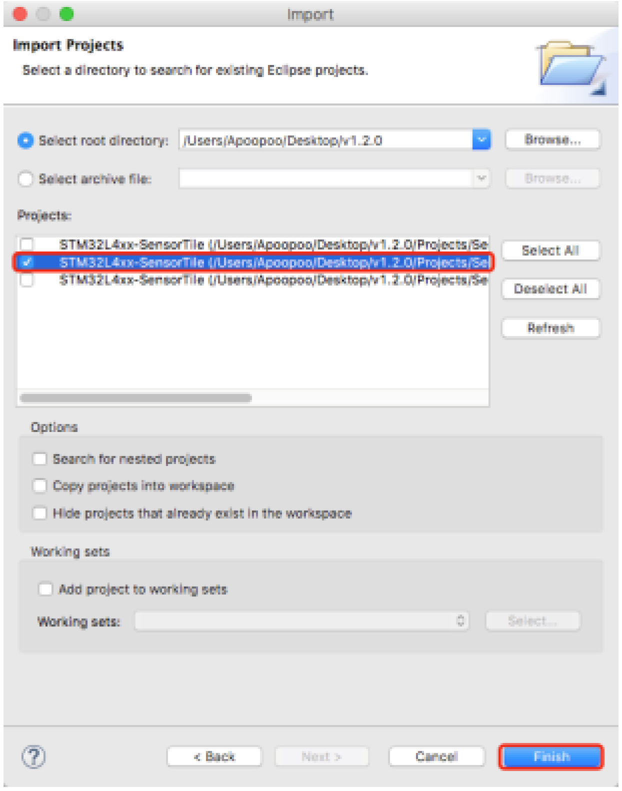
Your instructor will ensure you have the required background regarding BLE fundamentals including BLE architecture and BLE protocol.

2. BLE Firmware Installation

In the previous documents, we explored the DataLog application and AudioLoop application in the SensorTile starter firmware package. We will now dig into BLE firmware and examine how to communicate with SensorTile using BLE via Bluez (Bluetooth communication utility for Linux platform) on Linux platform.

1. Open the IDE (Eclipse or System WorkBench) on your personal computer as instructed in the Tutorial 1**.** Select the same workspace as in Tutorial 1.

2. Once the IDE is open, first remove your current project in your IDE and import the BLE_SampleApp project from corresponding directories as instructed in the document labeled ***STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example***. In order to properly import the BLE project, you need to make sure you *uncheck* DataLog and AudioLoop. See Figure 1.



Import

Import Projects

Select a directory to search for existing Eclipse projects.

● Select root directory: /Users/Apoopoo/Desktop/v1.2.0 [Browse...]

○ Select archive file: [Browse...]

Projects:

☐ STM32L4xx-SensorTile (/Users/Apoopoo/Desktop/v1.2.0/Projects/Se

☑ STM32L4xx-SensorTile (/Users/Apoopoo/Desktop/v1.2.0/Projects/Se

☐ STM32L4xx-SensorTile (/Users/Apoopoo/Desktop/v1.2.0/Projects/Se

[Select All] [Deselect All] [Refresh]

Options

☐ Search for nested projects

☐ Copy projects into workspace

☐ Hide projects that already exist in the workspace

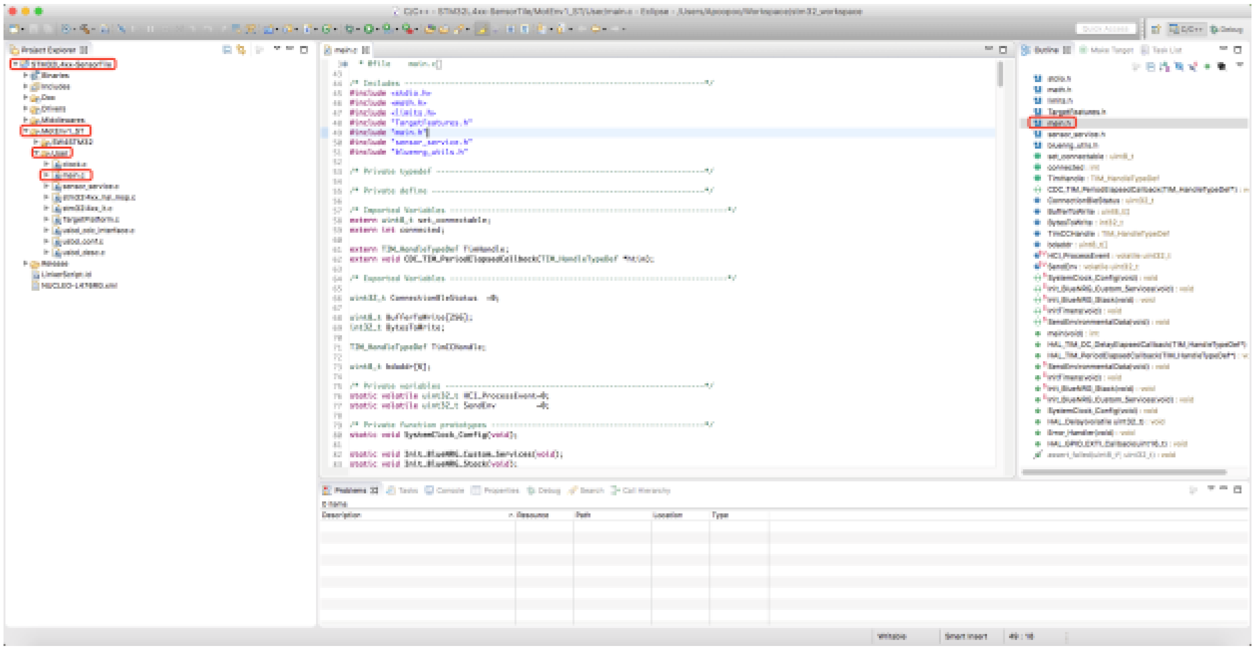
Working sets

☐ Add project to working sets

Working sets: [Select...]

Figure 1: Import BLE_SampleApp project from starter firmware package.

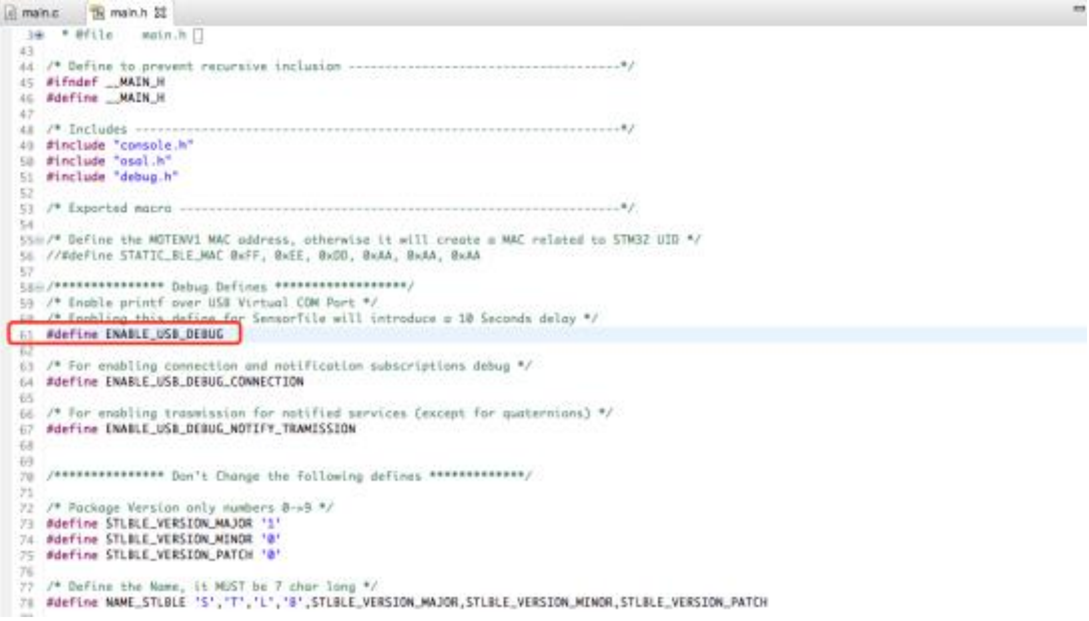
3. Once you successfully import the BLE project. Open main.c and main.h. See Figure 2.



```
 * @file    main.c
/* Includes ------------------------------------------------------------------*/
#include <stdio.h>
#include <math.h>
#include <limits.h>
#include "TargetFeatures.h"
#include "main.h"
#include "sensor_service.h"
#include "bluenrg_utils.h"

/* Private typedef -----------------------------------------------------------*/

/* Private define ------------------------------------------------------------*/

/* Imported Variables -------------------------------------------------------------*/
extern uint8_t set_connectable;
extern int connected;

extern TIM_HandleTypeDef TimHandle;
extern void CDC_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim);

/* Exported Variables -------------------------------------------------------------*/

uint32_t ConnectionBleStatus  =0;

uint8_t BufferToWrite[256];
int32_t BytesToWrite;

TIM_HandleTypeDef TimCCHandle;

uint8_t bdaddr[6];

/* Private variables ---------------------------------------------------------*/
static volatile uint32_t HCI_ProcessEvent=0;
static volatile uint32_t SendEnv         =0;

/* Private function prototypes -----------------------------------------------*/
static void SystemClock_Config(void);

static void Init_BlueNRG_Custom_Services(void);
static void Init_BlueNRG_Stack(void);
```

Figure 2: Open main.c and main.h.

4. Modify the define parameter at line 61 in main.h such that it matches Figure 3. This will enable the USB debug interface for the BLE firmware.



```
 3    * @file    main.h
43
44 /* Define to prevent recursive inclusion -------------------------------------*/
45 #ifndef __MAIN_H
46 #define __MAIN_H
47
48 /* Includes ------------------------------------------------------------------*/
49 #include "console.h"
50 #include "osal.h"
51 #include "debug.h"
52
53 /* Exported macro ------------------------------------------------------------*/
54
55 /* Define the MOTENV1 MAC address, otherwise it will create a MAC related to STM32 UID */
56 //#define STATIC_BLE_MAC 0xFF, 0xEE, 0xDD, 0xAA, 0xAA, 0xAA
57
58 /*************** Debug Defines ******************/
59 /* Enable printf over USB Virtual COM Port */
60 /* Enabling this define for SensorTile will introduce a 10 Seconds delay */
61 #define ENABLE_USB_DEBUG
62
63 /* For enabling connection and notification subscriptions debug */
64 #define ENABLE_USB_DEBUG_CONNECTION
65
66 /* For enabling trasmission for notified services (except for quaternions) */
67 #define ENABLE_USB_DEBUG_NOTIFY_TRAMISSION
68
69
70 /*************** Don't Change the following defines *************/
71
72 /* Package Version only numbers 0->9 */
73 #define STLBLE_VERSION_MAJOR '1'
74 #define STLBLE_VERSION_MINOR '0'
75 #define STLBLE_VERSION_PATCH '0'
76
77 /* Define the Name, it MUST be 7 char long */
78 #define NAME_STLBLE 'S','T','L','B',STLBLE_VERSION_MAJOR,STLBLE_VERSION_MINOR,STLBLE_VERSION_PATCH
```

Figure 3: Modify the define parameter and enable USB debug interface for BLE.

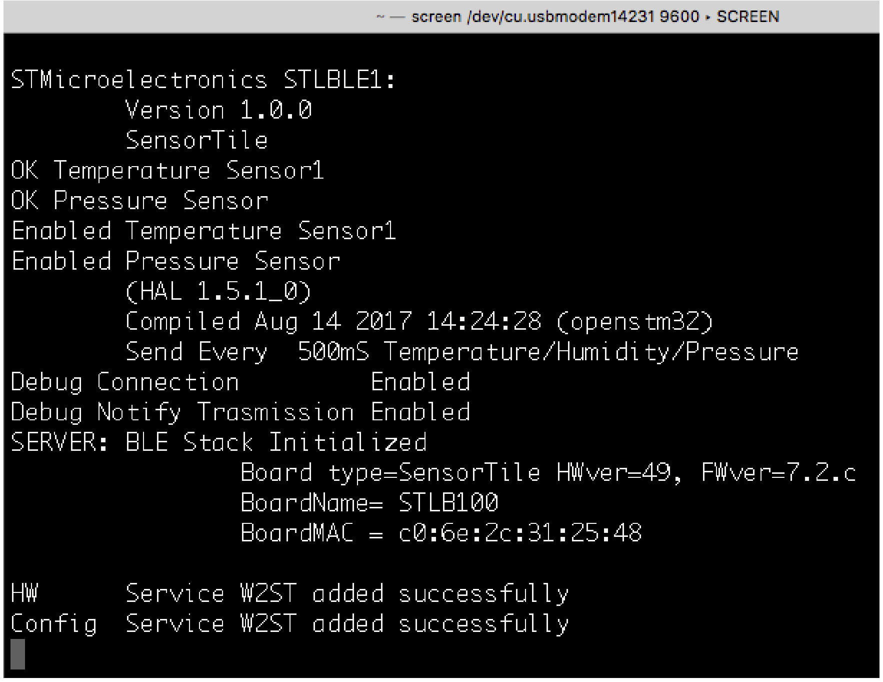
5. Save the modification.

6. Terminate and remove all previous applications from the SensorTile board.

7. Compile and run the BLE_SampleApp application on the SensorTile board in debug mode.

8. Examine the LED on SensorTile. Your SensorTile should start to blink once every second, which indicates that SensorTile is running BLE firmware properly.

9. Now you can check the BLE debug interface through USB streaming. You can use terminal screen command (Mac user) or Putty serial connection (Windows user) to check the BLE operations on SensorTile as you check the DataLog data streaming, which is illustrated in ***STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example*** -> ***Example Data Logging Project -> Debug***.
   The BLE debug interface is indicated as Figure 4. Make sure you do not close the debug interface because you will use it in the following session.



```
STMicroelectronics STLBLE1:
        Version 1.0.0
        SensorTile
OK Temperature Sensor1
OK Pressure Sensor
Enabled Temperature Sensor1
Enabled Pressure Sensor
        (HAL 1.5.1_0)
        Compiled Aug 14 2017 14:24:28 (openstm32)
        Send Every  500mS Temperature/Humidity/Pressure
Debug Connection         Enabled
Debug Notify Trasmission Enabled
SERVER: BLE Stack Initialized
                Board type=SensorTile HWver=49, FWver=7.2.c
                BoardName= STLB100
                BoardMAC = c0:6e:2c:31:25:48

HW      Service W2ST added successfully
Config  Service W2ST added successfully
```

Figure 4: BLE debug interface via USB streaming.

3. BLE Communication via BlueZ

SensorTile is able to communicate with mobile devices like smartphones and embedded Linux devices such as BeagleBone and Rpi through Bluetooth. BlueZ is the tool we will use on embedded Linux to deal with the BLE communication. In this session, we will use BeagleBone Wireless Green to introduce BlueZ.

3.1. BlueZ Introduction

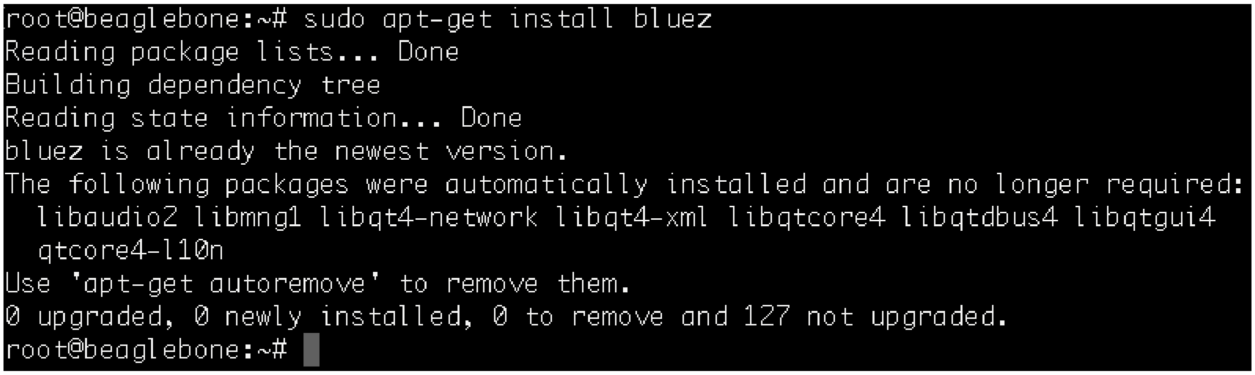
BlueZ is the official Bluetooth stack for Linux kernel-based family of operating systems. Its goal is to program an implementation of the Bluetooth wireless standards specifications for Linux. BlueZ also provides support for the core Bluetooth layers and protocols. It is flexible, efficient and uses a modular implementation. For more information, you can refer to BlueZ website http://www.bluez.org/.

3.2. BlueZ Installation and Update

1. Log into your BeagleBone and make sure your BeagleNone is successfully connected with WiFi.

2. Update your apt-get by using command ***$ sudo apt-get update*** in BeagleBone.

3. Type the command ***$ sudo apt-get install bluez*** to install BlueZ or check your current BlueZ version if it has already been installed. See Figure 5.

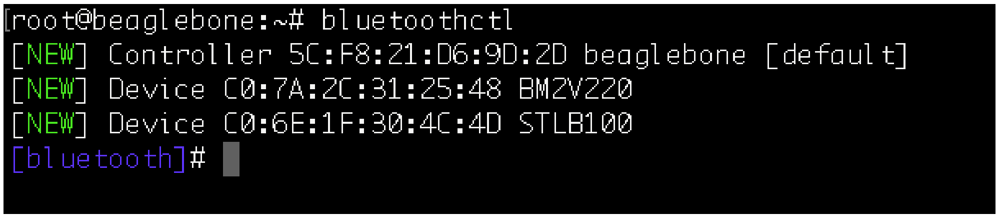


```
root@beaglebone:~# sudo apt-get install bluez
Reading package lists... Done
Building dependency tree
Reading state information... Done
bluez is already the newest version.
The following packages were automatically installed and are no longer required:
  libaudio2 libmng1 libqt4-network libqt4-xml libqtcore4 libqtdbus4 libqtgui4
  qtcore4-l10n
Use 'apt-get autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 127 not upgraded.
root@beaglebone:~#
```

Figure 5: Use apt-get to check installation of BlueZ.

3.3. Bluetoothctl Utilities in BlueZ

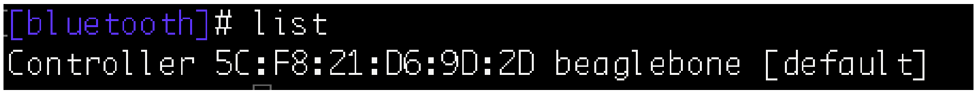
1. Once you have confirmed that you have the BlueZ installed on your BeagleBone, you can use the command ***$ bluetoothctl*** to use the bluetoothctl utilities interactively in BeagleBone. See Figure 6.



```
root@beaglebone:~# bluetoothctl
[NEW] Controller 5C:F8:21:D6:9D:2D beaglebone [default]
[NEW] Device C0:7A:2C:31:25:48 BM2V220
[NEW] Device C0:6E:1F:30:4C:4D STLB100
[bluetooth]# 
```

Figure 6: Bluetoothctl utilities in BlueZ.

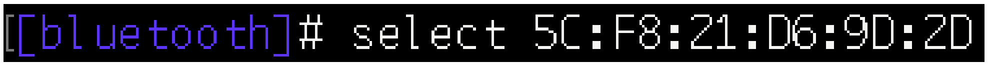
2. Type ***list*** in bluetoothctl and this will show your device Bluetooth controller MAC address. See Figure 7.



```
[bluetooth]# list
Controller 5C:F8:21:D6:9D:2D beaglebone [default]
```

Figure 7: Bluetoothctl Bluetooth controller MAC address.

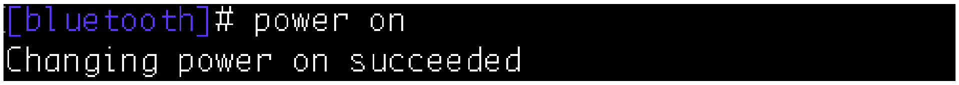
3. Type ***select MAC_ADDRESS*** as your default Bluetooth device controller as Figure 8.



```
[bluetooth]# select 5C:F8:21:D6:9D:2D
```

Figure 8: Select the default controller.

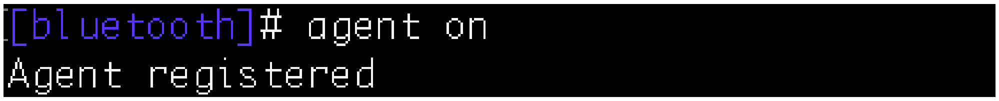
4. Type ***power on*** to turn on the power as Figure 9.



```
[bluetooth]# power on
Changing power on succeeded
```

Figure 9: Turn on controller power.

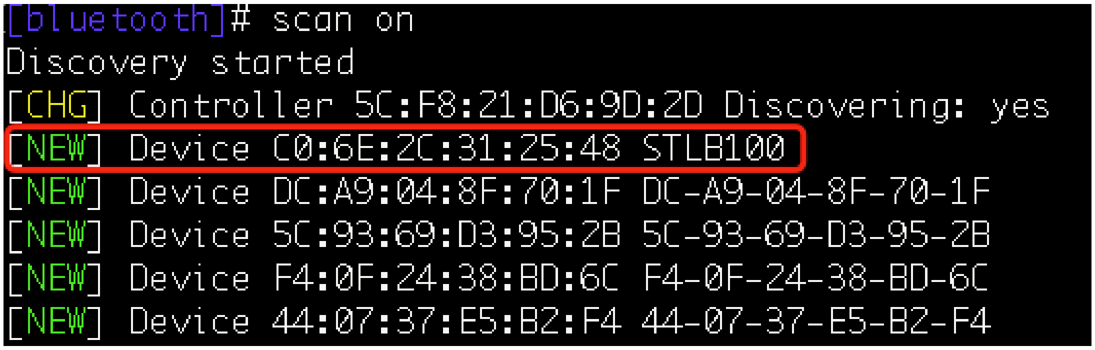
5. Type ***agent on*** to turn on agent. See Figure 10.



```
[bluetooth]# agent on
Agent registered
```

Figure 2: Turn on agent.

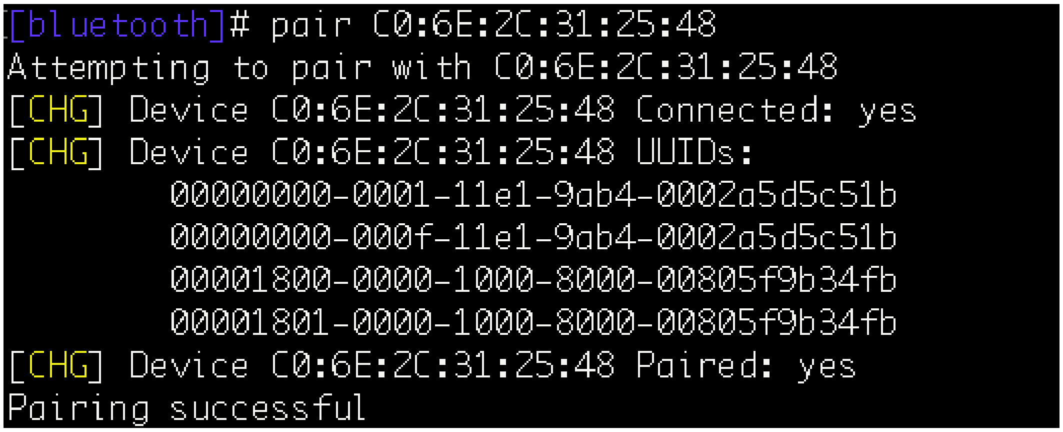
6. Type ***scan on*** to scan the Bluetooth devices and the device with name STLB100 is your SensorTile device. See Figure 11.



```
[bluetooth]# scan on
Discovery started
[CHG] Controller 5C:F8:21:D6:9D:2D Discovering: yes
[NEW] Device C0:6E:2C:31:25:48 STLB100
[NEW] Device DC:A9:04:8F:70:1F DC-A9-04-8F-70-1F
[NEW] Device 5C:93:69:D3:95:2B 5C-93-69-D3-95-2B
[NEW] Device F4:0F:24:38:BD:6C F4-0F-24-38-BD-6C
[NEW] Device 44:07:37:E5:B2:F4 44-07-37-E5-B2-F4
```

Figure 11: Scan Bluetooth devices.

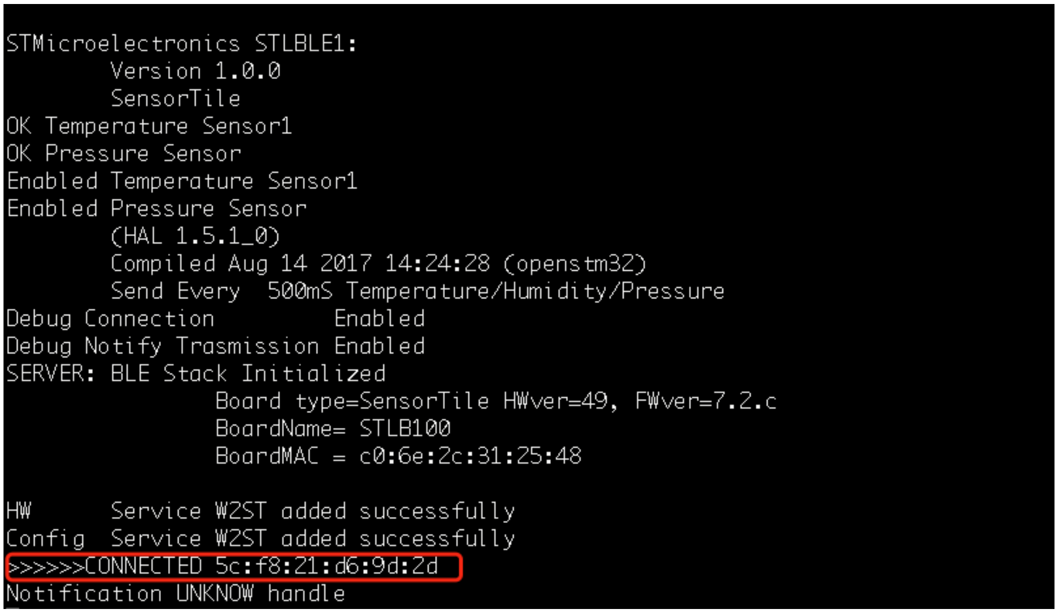
7. Type ***pair MAC_ADDRESS_SensorTile*** to make your BeagleBone pair with SensorTile. See Figure 12. The Beaglebone automatically connects with the SensorTile after pairing.



```
[bluetooth]# pair C0:6E:2C:31:25:48
Attempting to pair with C0:6E:2C:31:25:48
[CHG] Device C0:6E:2C:31:25:48 Connected: yes
[CHG] Device C0:6E:2C:31:25:48 UUIDs:
        00000000-0001-11e1-9ab4-0002a5d5c51b
        00000000-000f-11e1-9ab4-0002a5d5c51b
        00001800-0000-1000-8000-00805f9b34fb
        00001801-0000-1000-8000-00805f9b34fb
[CHG] Device C0:6E:2C:31:25:48 Paired: yes
Pairing successful
```

Figure 12: Pair devices.

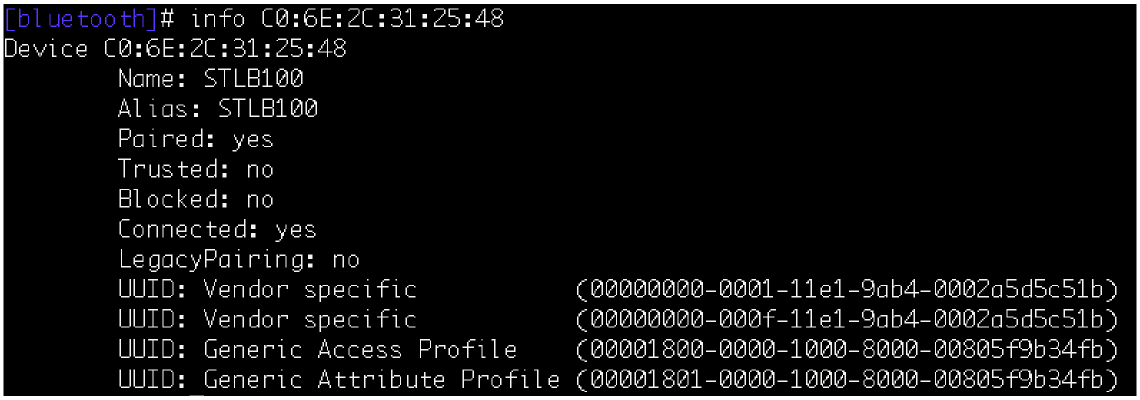
8. Now, you can switch back to the BLE debug interface, you can find that you have already connected with your Beaglebone. See Figure 13. You can also check the LED on SensorTile, which ***stops*** blinking.



```
STMicroelectronics STLBLE1:
        Version 1.0.0
        SensorTile
OK Temperature Sensor1
OK Pressure Sensor
Enabled Temperature Sensor1
Enabled Pressure Sensor
        (HAL 1.5.1_0)
        Compiled Aug 14 2017 14:24:28 (openstm32)
        Send Every  500mS Temperature/Humidity/Pressure
Debug Connection         Enabled
Debug Notify Trasmission Enabled
SERVER: BLE Stack Initialized
                Board type=SensorTile HWver=49, FWver=7.2.c
                BoardName= STLB100
                BoardMAC = c0:6e:2c:31:25:48

HW      Service W2ST added successfully
Config  Service W2ST added successfully
>>>>>>CONNECTED 5c:f8:21:d6:9d:2d
Notification UNKNOW handle
```

Figure 13: BLE debug interface return message.

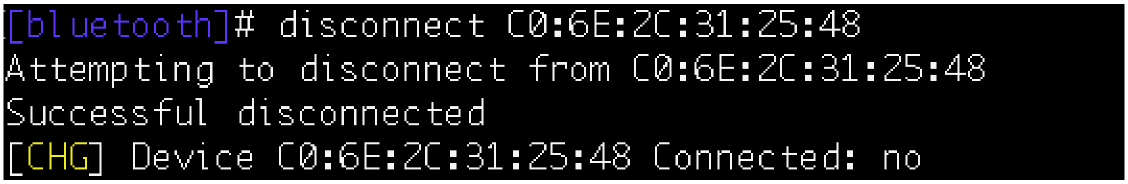
9. Switch back to BeagleBone and type ***info MAC_ADDRESS_SensorTile*** in bluetoothctl utilities. You can see more information related to your SensorTile BLE information. See Figure 14.



```
[bluetooth]# info C0:6E:2C:31:25:48
Device C0:6E:2C:31:25:48
        Name: STLB100
        Alias: STLB100
        Paired: yes
        Trusted: no
        Blocked: no
        Connected: yes
        LegacyPairing: no
        UUID: Vendor specific           (00000000-0001-11e1-9ab4-0002a5d5c51b)
        UUID: Vendor specific           (00000000-000f-11e1-9ab4-0002a5d5c51b)
        UUID: Generic Access Profile    (00001800-0000-1000-8000-00805f9b34fb)
        UUID: Generic Attribute Profile (00001801-0000-1000-8000-00805f9b34fb)
```

Figure 14: Information of SensorTile BLE.

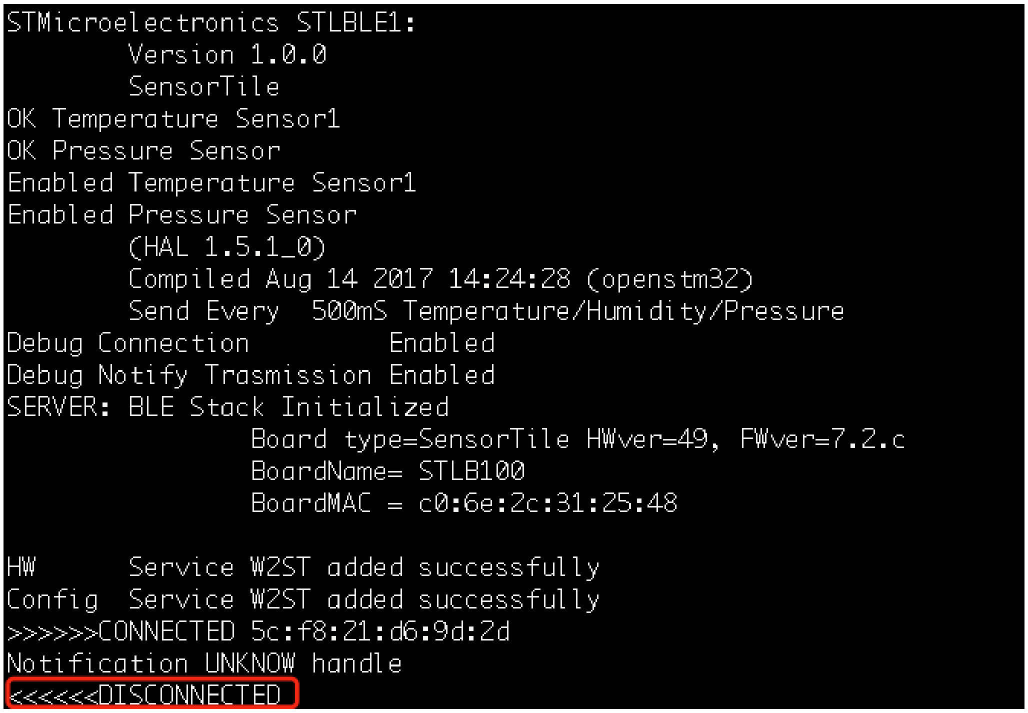
10. You can disconnect the BLE connection with the SensorTile by typing ***disconnect MAC_ADDRESS_SensorTile*** in bluetoothctl utilities. See Figure 15.



```
[bluetooth]# disconnect C0:6E:2C:31:25:48
Attempting to disconnect from C0:6E:2C:31:25:48
Successful disconnected
[CHG] Device C0:6E:2C:31:25:48 Connected: no
```

Figure 15: BLE disconnection with SensorTile.

11. You can switch back to the BLE debug interface to check the feedback information. See Figure 16. You can also check the LED on SensorTile. It starts to blink again because SensorTile is advertising itself when there is no BLE connection with it.



```
STMicroelectronics STLBLE1:
        Version 1.0.0
        SensorTile
OK Temperature Sensor1
OK Pressure Sensor
Enabled Temperature Sensor1
Enabled Pressure Sensor
        (HAL 1.5.1_0)
        Compiled Aug 14 2017 14:24:28 (openstm32)
        Send Every  500mS Temperature/Humidity/Pressure
Debug Connection         Enabled
Debug Notify Trasmission Enabled
SERVER: BLE Stack Initialized
                Board type=SensorTile HWver=49, FWver=7.2.c
                BoardName= STLB100
                BoardMAC = c0:6e:2c:31:25:48

HW      Service W2ST added successfully
Config  Service W2ST added successfully
>>>>>>CONNECTED 5c:f8:21:d6:9d:2d
Notification UNKNOW handle
<<<<<<DISCONNECTED
```

Figure 16: SensorTile debug interface.

12. Now you can switch back to BeagleBone and type ***connect MAC_ADDRESS_SensorTile*** in bluetoothctl utilities to re-connect with SensorTile. You can also type ***help*** to explore all the utilities in bluetoothclt. When there is an interaction with SensorTile, you can always check the SensorTile BLE debug interface for more information.

13. Type ***exit*** to quit bluetoothclt utility.