# Bluetooth Integration

## Sterling-LWB

*Application Note*

## INTRODUCTION

The goal of this document includes the following:

- Define how to patch the Sterling-LWB for Bluetooth integration into your host device
- Define how to integrate either the Bluedroid or BlueZ Blutooth stack into your host device
- Explain basic commands for scanning and connecting with Bluetooth

## PATCHRAM AND FIRMWARE DOWNLOAD

To patch the Sterling-LWB, we recommend that you use Broadcom's patchram_plus utility, which is also used to patch the SSD40NBT/MSD40NBT radio on Linux/Android. The source code for this utility is GPL. You can download it from the Ubuntu repository at the following link: https://launchpad.net/ubuntu/+source/brcm-patchram-plus/0.1.1.

You must use the cross-compiler in your platform to build this application.

To patch the Sterling-LWB, perform the following steps:

1. Make sure that the BT_REG_ON pin is pulled high. This pin should be connected to a GPIO on your platform and must be asserted before Bluetooth can be used.
2. Execute **brcm_patchram_plus** with the following parameters:

```
"brcm_patchram_plus --patchram {path to .hcd file}  --enable_hci --
no2bytes --tosleep 1000 {path to /dev/ UART interface} &"
```

   Where ***{path to .hcd file}*** is the Bluetooth firmware file which is provided with the LWB release package.

   Once this patching is complete, the next step is Bluetooth stack integration.

| | |
|---|---|
| **Note:** | Assuming that brcm_patchram_plus is allowed to run in the background (with the '&') there is no need to execute hciattach; the patchram utility handles this while it is active. |





Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

## BlueZ Stack Integration

To run BlueZ in your host platform, complete the following steps:

1. Check if the following APT (Advanced Packaging Tool) library has been installed properly on the host side by entering the following into the Linux/Android shell terminal:

```
sudo apt-get update
sudo apt-get install libusb-dev libdbus-1-dev libglib2.0-dev libudev-
dev libical-dev libreadline-dev
```

2. Choose the BlueZ version you prefer to run and untar the file in your system by entering the following commands into the Linux/Android shell terminal.

```
wget bluez-5.41.tar.xz
tar xvf bluez-5.41.tar.xz
```

3. Run the following commands to install the BlueZ stack to your system.

```
cd bluez-5.41
sudo ./configure --disable-systemd --enable-tools
sudo make
sudo make install
sudo mv /usr/local/sbin/hciconfig /usr/sbin
sudo mv /usr/local/bin/gatttool /usr/sbin
```

4. After HCI interface up, you can use the command “hcitool dev” to display local devices that are available.

**Note:** The command “hcitool scan” will begin inquiry of remote Bluetooth devices. For each discovered device, its device name will be printed.

More on hcitool is available at http://linux.die.net/man/1/hcitool.

## BlueDroid Stack Integration

Since Android 4.2, BlueDroid has been the native Bluetooth stack, which is contributed by Broadcom. If your host system is not running Android v4.2 or newer, you may download BlueDroid from the following link:

https://android.googlesource.com/platform/external/bluetooth/bluedroid/

BlueDroid is divided into two layers:

1. Bluetooth Embedded System (BTE) for implementing core Bluetooth functionality
2. Bluetooth Application Layer (BTA), for communication with Android applications

Figure 1 explains the architecture of Bluetooth on Android and is reproduced from the following source: https://source.android.com/devices/bluetooth.html.





Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

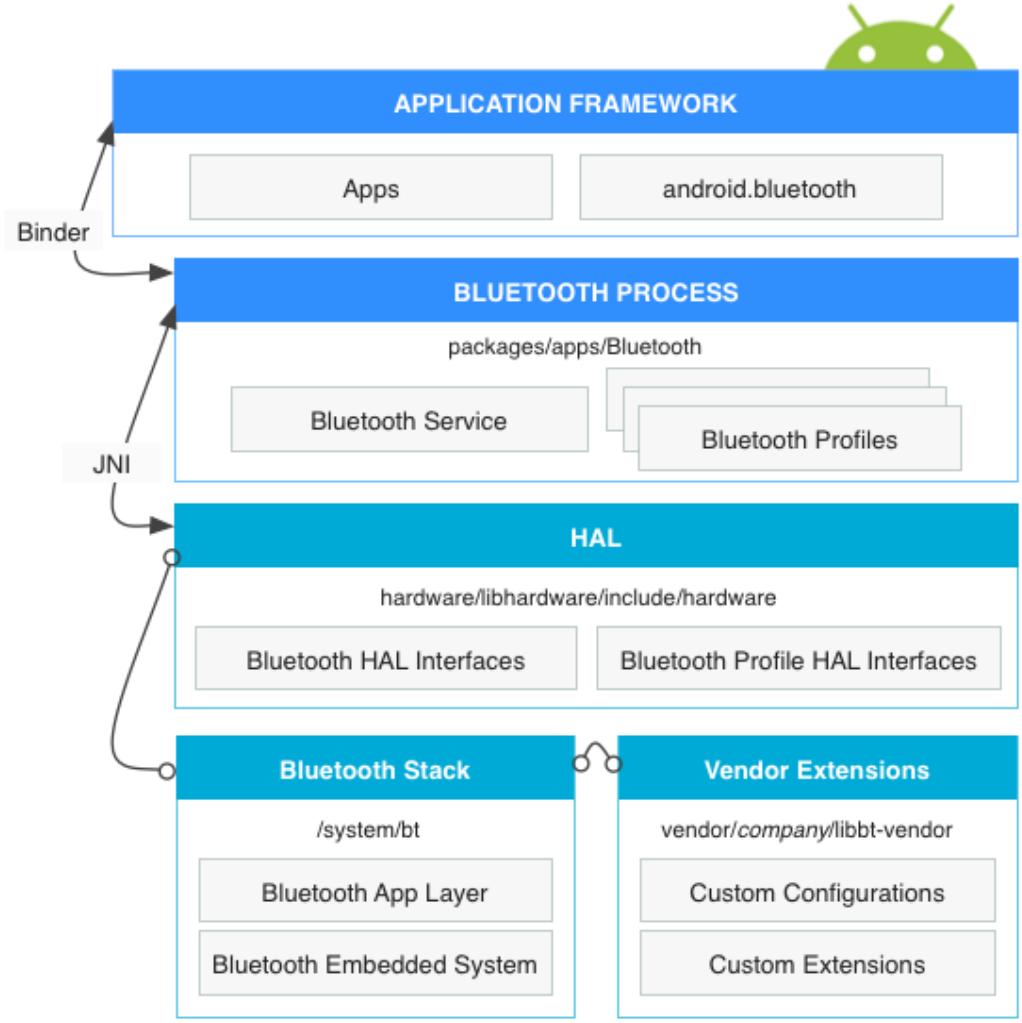


***Figure 1: Bluetooth Architecture on Android***

HCI (Host Control Interface) is the communication protocol between the host stack and control stack. Unlike BlueZ, the HCI layer is included in BlueDroid already.

From the architecture of BlueDroid, only the following two processes are left to drive the Bluetooth subsystem up by using the BlueDroid stack:

- Bluetooth APP process
- System framework process

BlueDroid excludes any IPC (Inter-process Communication) mechanism, which means it is no longer running in an independent daemon process (unlike BlueZ); it must be embedded into your system.

The API for both legacy BT and BLE profiles is described in greater detail here:
https://developer.android.com/reference/android/bluetooth/package-summary.html





Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

To start up Bluetooth and begin scanning for Bluetooth devices, the following may be required.

- **BluetoothAdaptor()** – This API is required for any and all Bluetooth activity. The return of this API is the device's own Bluetooth adapter. Pick up the Bluetooth adapter that you want to set; your application can interact with it using this object.
- **isEnable()** – This API checks whether Bluetooth is currently enabled. If you need to enable Bluetooth, call **startActivityForResult()** with REQUEST_ENABLE_BT constant.
- **startDiscovery()** – Use this API to scan for legacy devices (for example, the remote device discovery process).
  The discovery process usually involves an inquiry scan of about 12 seconds followed by a page scan of each new device to retrieve its Bluetooth name. Once you have found a device to which you want to connect, always stop discovery with **cancelDiscovery()** before attempting a connection.
- **startLeScan()** – The purpose of this API is to find BLE devices.

## More Information

For more information on the Sterling-LWB including software, datasheets, and further documentation, visit the Sterling-LWB product page.

## Revision History

| Version | Date | Notes | Approver |
|---|---|---|---|
| 1.0 | 09 Aug 2016 | Initial Release | Miles Chung |
| 1.1 | 12 Sept 2018 | Fixed syntax issue with patchram parameter | John Nosky |





Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600