



tech **t▶lks**

WELCOME

Simplify Your Bluetooth[®]
Design using Python Scripts

Kris Young



Agenda

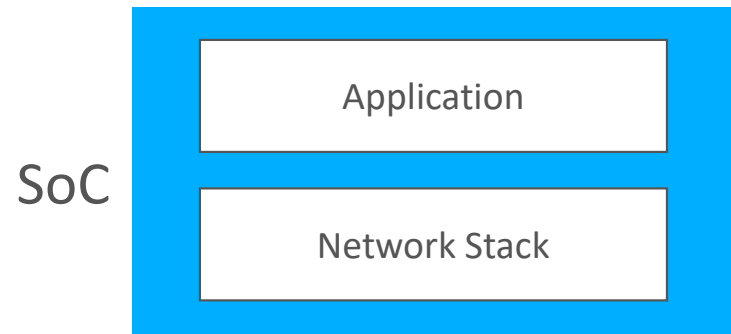
- Brief overview of Silicon Labs Bluetooth® Network Co-Processors
- Why use Python with Bluetooth®?
- How to implement Silicon Labs Bluetooth® with Python
- Demo: Client for health thermometer service



Silicon Labs Bluetooth® Application Modes

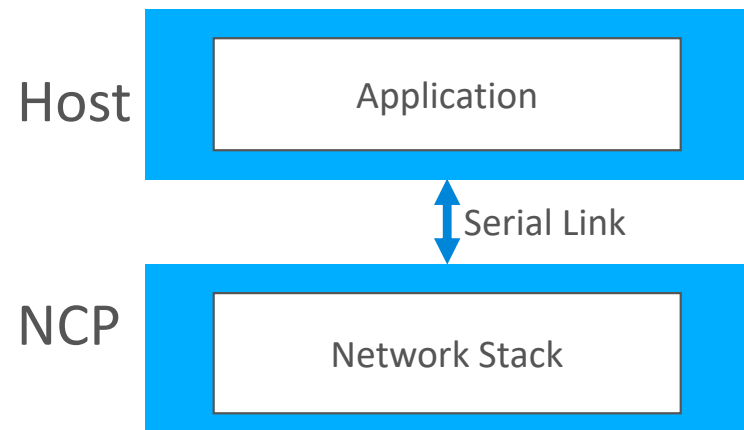
System on Chip (SoC)

- Application and Network Stack are both resident in the same device



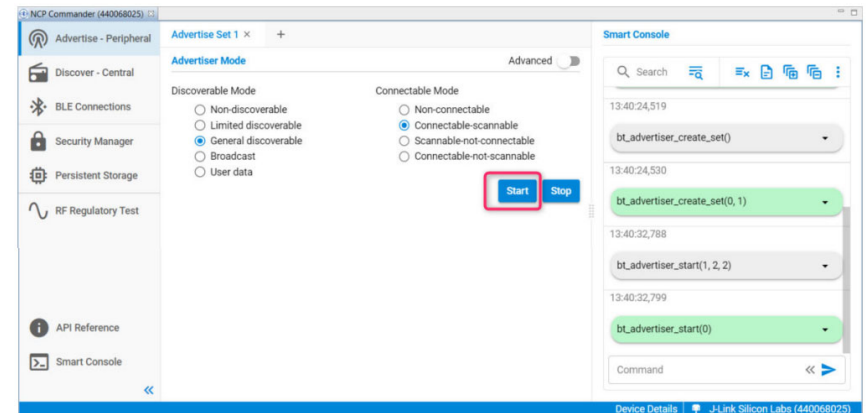
Network Co-Processor (NCP)

- Network Stack is on the NCP device
- Application is on a separate host processor
- A serial link is used to communicate between the host processor and the NCP



NCP Host Programming

- Bluetooth NCP Commander
 - Easy-to-use GUI tool to issue BGAPI commands one-by-one
 - Portable: Works on most desktop platforms (Windows, Mac, Ubuntu Linux)
 - Not programmable
- C Host Sample apps
 - Complicated code (even though most parts are not to be touched)
 - Complicated to build on Windows platform (needs Cygwin/MSYS2 environment)
 - Portable for Posix environments
 - Programmable
- Python BGAPI
 - Easy-to-use: commands can be issues one-by-one or in a script
 - Portable: Works on most host platforms (desktop and embedded)
 - Programmable



Python BGAPI – Use Cases

Produce easily portable host applications

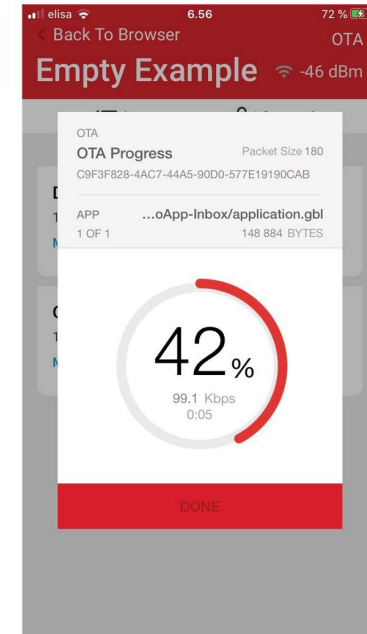
- Can run the same application without modification on any platform that supports Python 3 (Mac, PC, Raspberry Pi, etc.)

Mobile app emulation

- Use python to quickly implement a client with similar functionality to an intended mobile app
- Allows decoupling of device firmware development from mobile app software development

Automated hardware/software testing

- Python can be used to create a test application (either automated or interactive) to connect to your device and exercise all of its features

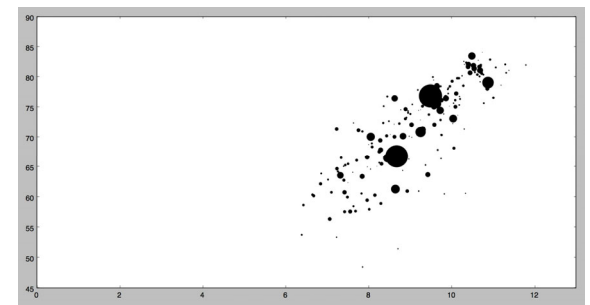
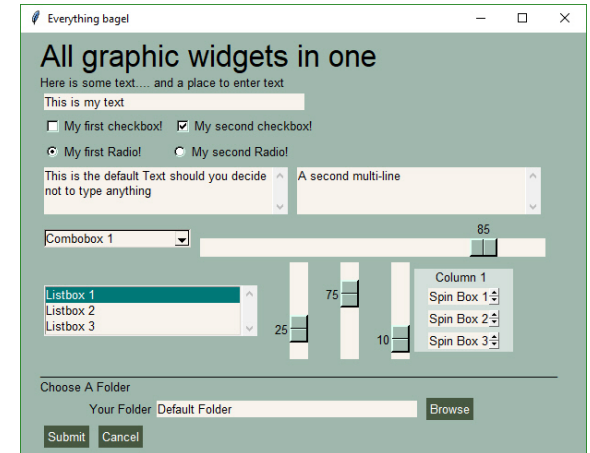


Python BGAPI – Use Cases

Building applications with higher level functionality is much easier in Python than C

- GUI functionality (Tkinter, PyQt)
- TCP for MQTT (paho), HTTP (SimpleHttpServer), etc.
- File I/O (JSON, CSV, etc.)
- Data visualization and graphing (matplotlib, ggplot2, etc.)
- Cryptography (PyCrypto)
- Specialty hardware interfacing (test equipment, etc.)
- Machine learning

For anything you want to implement, there's probably a python library available to make implementation easier!



Using Python BGAPI

- Available at <https://pypi.org/project/pybgapi/>

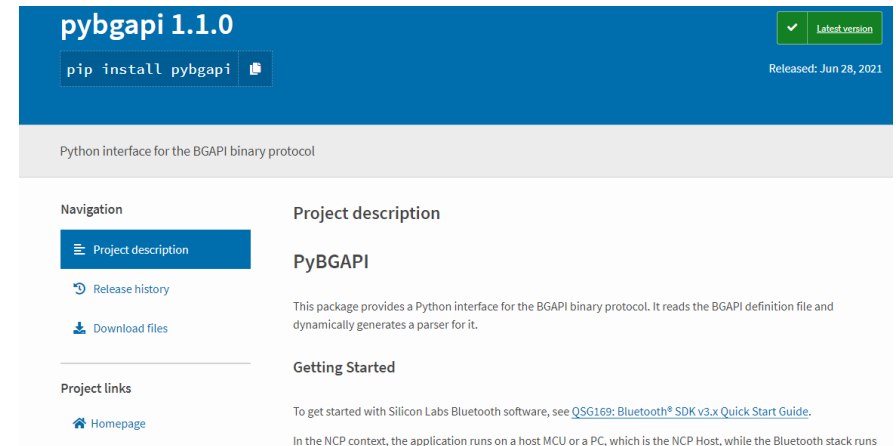
```
pip install pybgapi
```

- Basic documentation provided at that URL

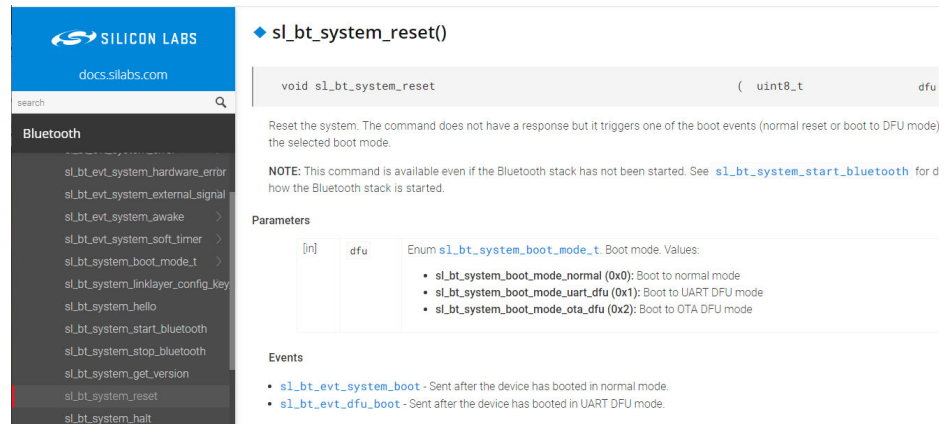
- Requires Python ≥ 3.6

- License: zlib/libpng

- API documentation available at <http://docs.silabs.com/bluetooth/latest>



The screenshot shows the PyPI page for the `pybgapi` package, version 1.1.0. The page includes a navigation menu with options for 'Project description', 'Release history', and 'Download files'. The 'Project description' section states that the package provides a Python interface for the BGAPI binary protocol. The 'Getting Started' section provides instructions on how to use the package with Silicon Labs Bluetooth software.



The screenshot shows the Silicon Labs Bluetooth API documentation for the `sl_bt_system_reset()` function. The function signature is `void sl_bt_system_reset (uint8_t dfu)`. The documentation includes a description of the function, a note about its availability, and a list of parameters. The parameters section shows a table with two columns: `[in]` and `dfu`. The `dfu` parameter is an enum `sl_bt_system_boot_mode_t` with three values: `sl_bt_system_boot_mode_normal (0x0)`, `sl_bt_system_boot_mode_uart_dfu (0x1)`, and `sl_bt_system_boot_mode_ota_dfu (0x2)`. The events section lists `sl_bt_evt_system_boot` and `sl_bt_evt_dfu_boot`.

Using Python BGAPI

Once installed, simply import:

```
>>> import bgapi
```

Establish the connector with the NCP. This can be done using a serial port:

```
>>> connection = bgapi.SerialConnector('COM3')
```

Alternatively, a TCP socket can be used. The Silicon Labs Wireless Starter kit (WSTK) connected via ethernet exposes VCOM on TCP port 4901:

```
>>> connection =  
bgapi.SocketConnector(('192.168.1.149', 4901))
```



Using Python BGAPI

Next initialize bgapi with the connector and the xapi SDK API definition file, for example in Gecko SDK Suite 3.2:

```
C:\SiliconLabs\SimplicityStudio\v5\developer\sdk\gecko_sdk_suite\v3.2\protocol\bluetooth\api\sl_bt.xapi
```

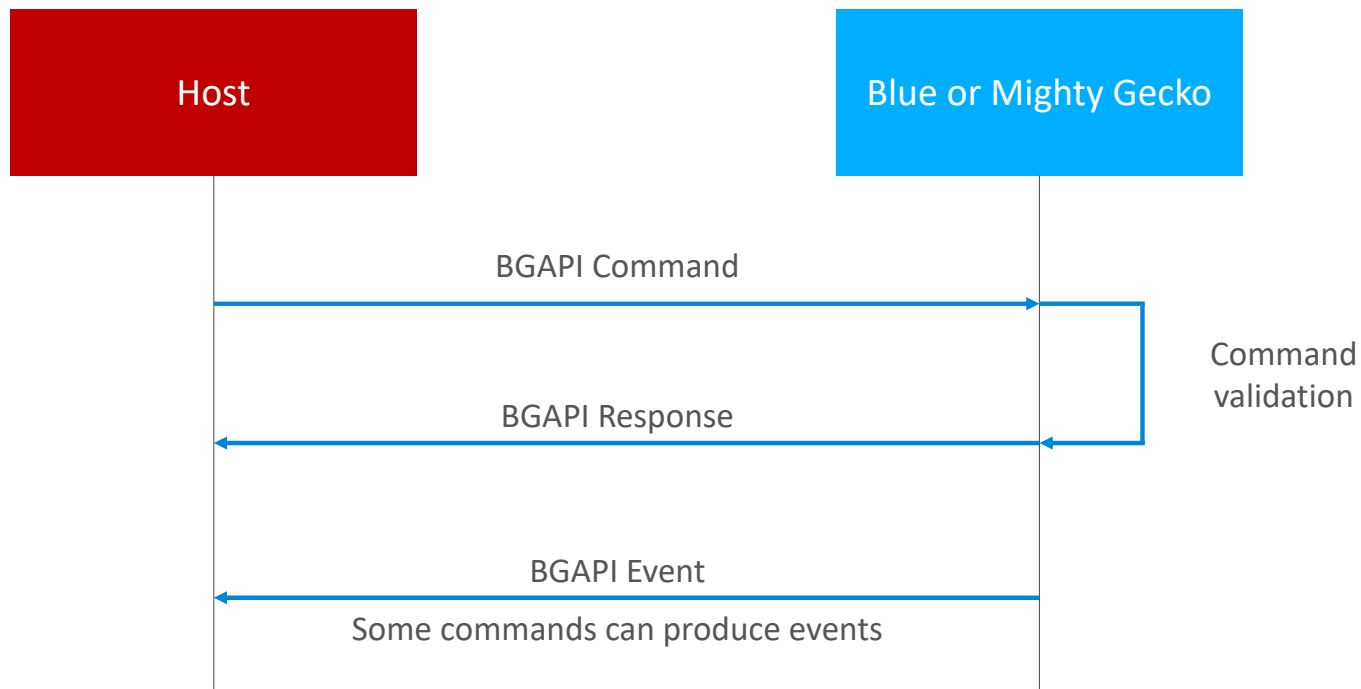
```
>>> node = bgapi.BGLib(connection,  
    'sl_bt.xapi')
```

```
>>> node.open()
```

```
>>> node.is_open()
```

```
True
```

BGAPI Serial Protocol Message Exchange



Using Python BGAPI – Example Command with Response

In the Python interactive console:

```
>>> response = node.bt.system.hello()
>>> response
bt_rsp_system_hello(result=0)
>>> response.result
0
```

NOTE: result=0 is SL_STATUS_OK – so the command executed successfully!

◆ sl_bt_system_hello()

```
sl_status_t sl_bt_system_hello ( )
```

Verify whether the communication between the host and the device is functional.

NOTE: This command is available even if the Bluetooth stack has not been started. See [sl_bt_system_start_bluetooth](#) for description of how the Bluetooth stack is started.

Returns

SL_STATUS_OK if successful. Error code otherwise.

Using Python BGAPI – Example Command with Event

In the python interactive console:

```
>>> node.bt.system.reset(0)
>>> events = node.get_events()
>>> events
[bt_evt_system_boot(major=3, minor=1, patch=2,
build=256, bootloader=17563648, hw=1,
hash=3896915237)]
>>> events[0].major
3
>>> events[0].minor
1
>>> hex(events[0].hash)
'0xe8463525'
>>> hex(events[0].bootloader)
'0x10c0000'
```

◆ sl_bt_system_reset()

```
void sl_bt_system_reset ( uint8_t dfu )
```

Reset the system. The command does not have a response but it triggers one of the boot events (normal reset or boot to DFU mode) depending on the selected boot mode.

NOTE: This command is available even if the Bluetooth stack has not been started. See [sl_bt_system_start_bluetooth](#) for description of how the Bluetooth stack is started.

Parameters

[in]	dfu	Enum sl_bt_system_boot_mode_t . Boot mode. Values:
		<ul style="list-style-type: none">• sl_bt_system_boot_mode_normal (0x0): Boot to normal mode• sl_bt_system_boot_mode_uart_dfu (0x1): Boot to UART DFU mode• sl_bt_system_boot_mode_ota_dfu (0x2): Boot to OTA DFU mode

Events

- [sl_bt_evt_system_boot](#) - Sent after the device has booted in normal mode.
- [sl_bt_evt_dfu_boot](#) - Sent after the device has booted in UART DFU mode.

Using Python BGAPI – Example Event Handler

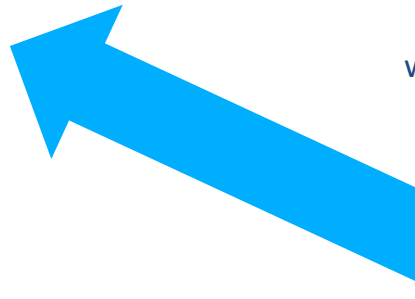
Event Handler

```
def sl_bt_on_event(evt):
    global node
    global state
    global service
    global characteristic

    if evt == 'bt_evt_system_boot':
        print("Boot event received! Major version: {}, minor
version:{}".format(
            evt.major, evt.minor))
    elif evt == 'bt_evt_connection_opened':
        print("connection opened")
    #elif evt == 'bt_evt_other_event':
    # Handle other events here
    else:
        print("Unhandled event: {}".format(evt))
```

Main Loop

```
while True:
    try:
        evt = node.get_events(max_events = 1)
        if evt:
            # Call event handler
            sl_bt_on_event(evt[0])
    except (KeyboardInterrupt, SystemExit) as e:
        if node.is_open():
            node.close()
        print("Exiting...")
        sys.exit(1)
```



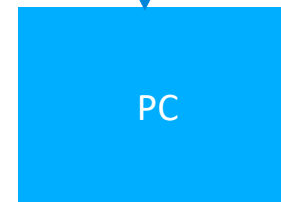
Demo: Health Thermometer Client

- Use BG22 Thunderboard as our NCP device (NCP-empty demo)
- Use another BG22 Thunderboard as our health thermometer server (soc-thermometer demo)
- Use a Python script on the host to:
 - Scan for a device advertising the Health Thermometer Service
 - Connect to the device
 - Discover the services
 - Discover the characteristics
 - Subscribe to indications on the temperature measurement characteristic within the Health Thermometer service
 - Print temperature measurement values (received via indication) to the console, along with the RSSI of the link

BG22 Thunderboard (ncp-empty)



BG22 Thunderboard (soc-thermometer)



Flashing Firmware

Pre-built example NCP-empty and soc-thermometer firmware images are available within Simplicity Studio for most Silicon Labs boards

After plugging your board into the USB port...

1. Go to the Simplicity Studio "Launcher" perspective
2. Select target board under "Debug Adapters"
3. Select the target SDK
4. Select "Example Projects and Demos"
5. Enable filtering on "Demos"
6. Click "Run" to flash the desired demo image



Thunderboard Sense 2



BG22 Thunderboard

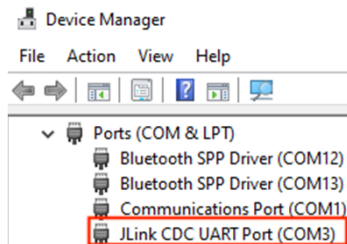


Wireless Starter Kit (WSTK) with Radio Board

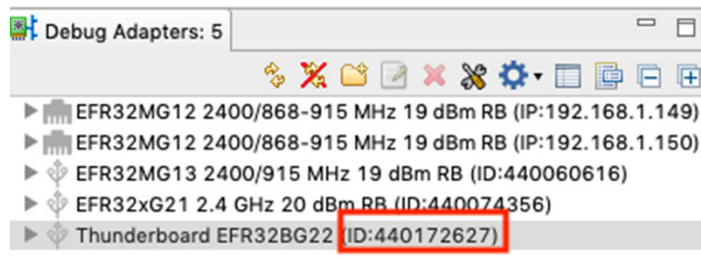
Demo: Health Thermometer Client

Determine the serial port of your NCP for Python BGAPI init

- In Windows, Thunderboard/WSTK UART ports show up in the Device Manager as “Jlink CDC UART Port”. Note, it’s easier to identify if only one device is plugged in.



- In Mac, Thunderboard/WSTK UART ports show up as `/dev/tty.usbmodem#####` where ##### is derived from the J-Link adapter serial number



```
$ ls /dev/tty.usbmodem*  
/dev/tty.usbmodem0004400606161  
/dev/tty.usbmodem0004401726271  
/dev/tty.usbmodem0004400743561
```