RN-HID-UM
Version 0.05

Bluetooth HID profile

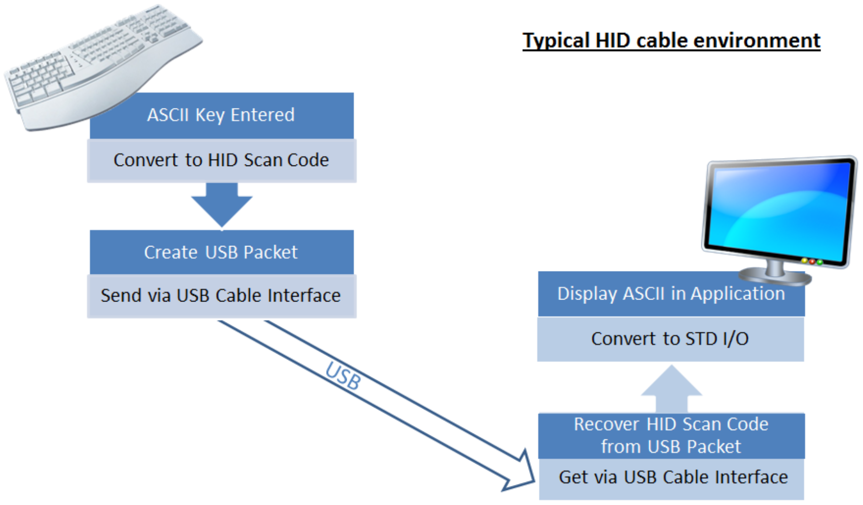
RN-HID-UM Version 0.05 09/12/11

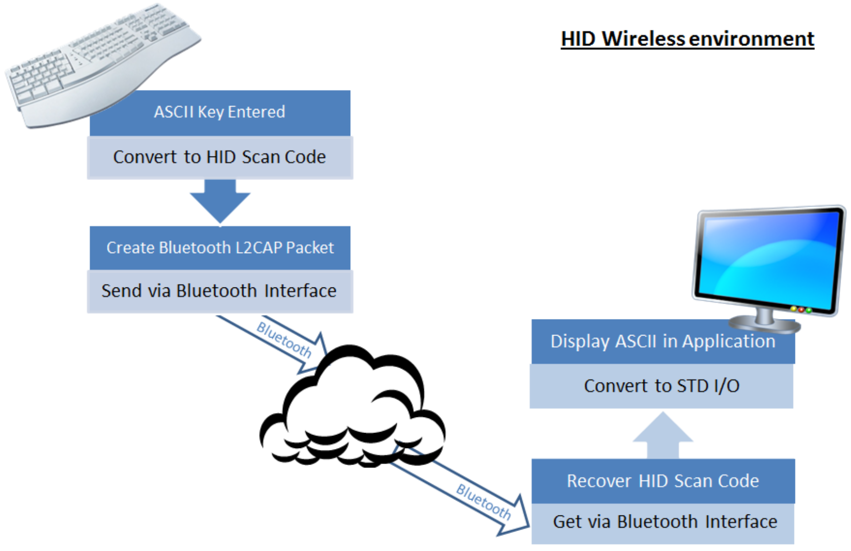
USER MANUAL

1.0. OVERVIEW

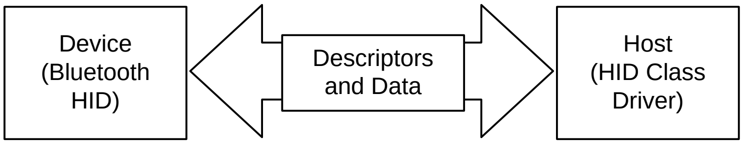
Roving Networks supports the Bluetooth Human Interface Device (HID) profile. This enables customers to develop products such as Bluetooth keyboard, mouse and pointing devices and other HID devices.

HID is not specific to USB or any other type of physical data transport. It is the intention of the Bluetooth HID profile to describe how to use the HID protocol over the Bluetooth wireless communications system, allowing manufacturers of input devices to produce high performance, interoperable, and secure wireless input devices.

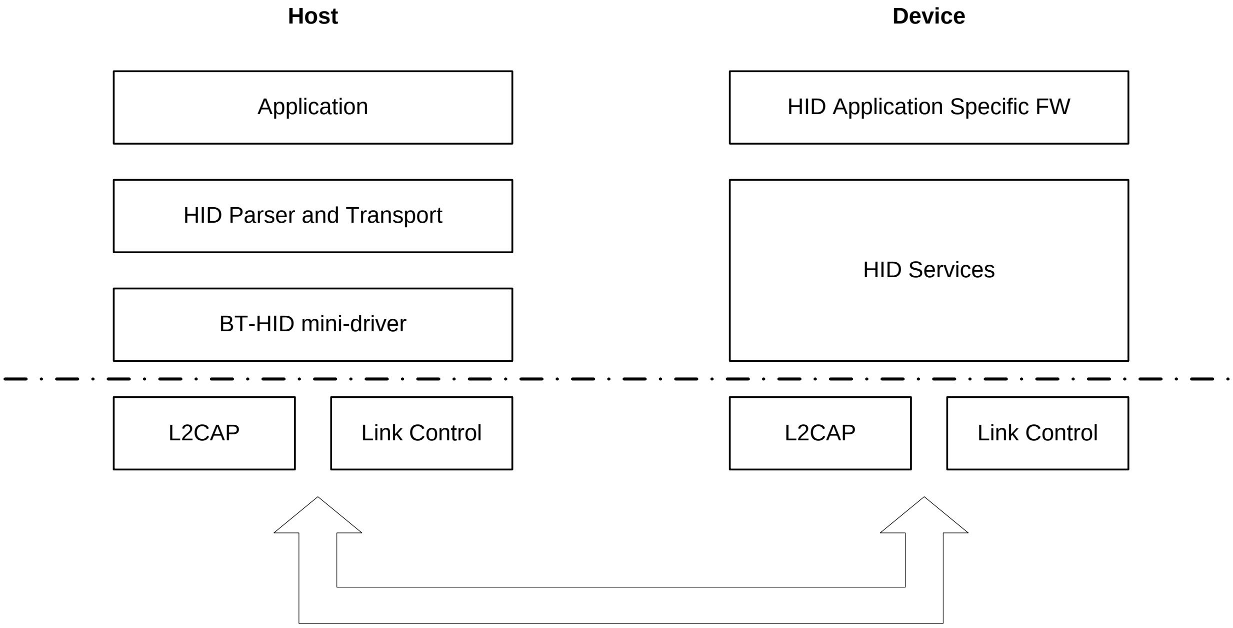

Typical HID cable environment
ASCII Key Entered
Convert to HID Scan Code
Create USB Packet
Send via USB Cable Interface
USB
Display ASCII in Application
Convert to STD I/O
Recover HID Scan Code from USB Packet
Get via USB Cable Interface



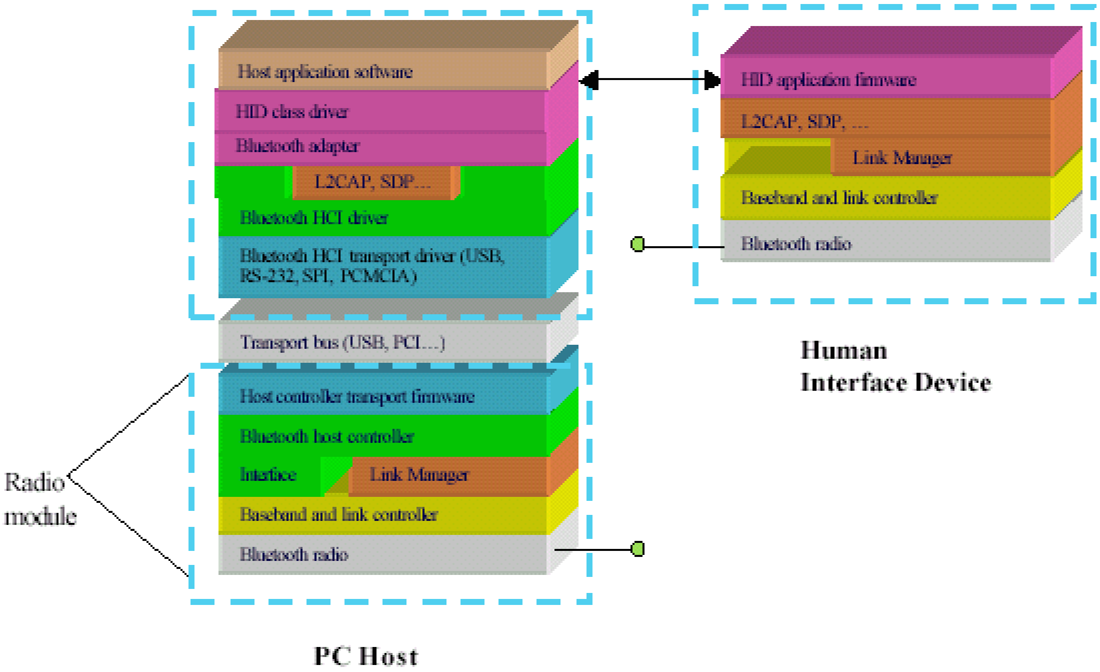
Information about a HID device is stored in segments of its ROM (read-only memory). These segments are called descriptors. An interface descriptor can identify a device as belonging to one of a finite number of classes. The HID class is the primary focus of this document. Other types of device classes described by USB specifications include display, audio, communication, and data storage. A HID class device uses a corresponding HID class driver to retrieve and route all data. The routing and retrieval of data is accomplished by examining the descriptors of the device and the data it provides.



The HID class device descriptor identifies which other HID class descriptors are present and indicates their sizes; for example, Report and Physical Descriptors. A Report descriptor describes each piece of data that the device generates and what the data is actually measuring. For example, a Report descriptor defines items that describe a position or button state. Physical descriptor sets are optional descriptors that provide information about the part or parts of the human body used to activate the controls on a device.



1.1 Profile Stack


Host application software
HID class driver
Bluetooth adapter
L2CAP, SDP...
Bluetooth HCI driver
Bluetooth HCI transport driver (USB, RS-232, SPI, PCMCIA)
Transport bus (USB, PCI...)
Host controller transport firmware
Bluetooth host controller
Interface
Link Manager
Baseband and link controller
Bluetooth radio
Radio module
PC Host
HID application firmware
L2CAP, SDP, ...
Link Manager
Baseband and link controller
Bluetooth radio
Human Interface Device

Above is an illustration of the software layers that reside in both the host and the human interface device for an example implementation. In this example, the host is a personal computer and has the upper layers of the Bluetooth software running on its native processor and is connected to a Bluetooth radio module via a transport bus such as USB. The HID in this example has its firmware embedded with the radio firmware, running on the same CPU, for the lowest possible cost implementation. Other implementations on the HID side are possible and equally valid.

1.2 Configurations/Roles

The following roles are defined for devices that comply with this profile:

- The *HID* (Human Interface Device) is the device providing the service of human data input and output to and from the host. Because the USB definition of HID includes all devices that report data in a similar fashion to HIDs, other devices such as remote sensors are included which may not interface directly with a human. Examples of HID devices are mice, joysticks, gamepads, keyboards, and also voltmeters and temperature sensors. The HID device is normally the slave in the Bluetooth piconet structure.
- The *host* is the device using or requesting the services of a Human Interface Device. Examples would be a personal computer, handheld computer, gaming console, industrial machine, or data -recording device. The host is normally the master in the Bluetooth piconet structure.

1.3 HID profile usage scenarios

There are 4 scenarios that are covered in this profile:

- **Desktop Computing Scenario:** In the traditional desktop computer scenario, use of Bluetooth wireless computer peripherals will free the desktop from multiple cables and allow input devices to be operated further from the desktop and in positions that are more comfortable. Users will be able to switch between multiple input devices without plugging and unplugging cables. Users will also be able to control different computers or host devices with at different times with a single Bluetooth HID device without concern for connecting cables.
- **Living Room Scenario**: HID devices with Bluetooth wireless technology will enable the ease of multiplayer gaming. The users are no longer tethered to the gaming machine and can be seated casually within a standard sized living room. Bluetooth devices will not require line of sight alignment with the receiver and the two-way capability allows remote displays and user feedback devices.
- **Conference Room Scenario**: A pointing device enabled with Bluetooth wireless technology will allow the presenter in a conference room to control the presentation on a video screen from up to 10 meters away, without the need to be near the host or the projection device. The device need not be designed for operation on a flat surface.
- **Remote Monitoring Scenario**: Battery powered monitoring devices with Bluetooth wireless technology will provide many benefits to the end user. Some examples of these devices include temperature sensors, remote thermostats, security devices, pressure sensors, voltmeters, etc. By using Bluetooth wireless technology and the HID standard, monitoring systems can be installed quickly without running new wires to each of the installed sensors. The low power modes provided by Bluetooth wireless technology will provide long battery life for the remote transmitters.

Potential HID devices enabled with Bluetooth wireless technology include:

- Computer keyboards and keypads
- Trackballs, mice, and other pointing devices
- Game controllers (gamepads, joysticks, steering wheels, etc.)
- Battery operated sensors (temperature, pressure, security, etc.)
- Simple alphanumeric remote displays
- Universal remote controls
- Bar code scanners

1.4 Profile requirements:

This is a brief outline of Bluetooth requirements detailed by this specification:

- **Master/Slave roles**. Although there are no mandated master/slave roles, it is recommended that Bluetooth Human Interface Devices normally be a slave device, in order to avoid having the host radio multiplex between piconets.
- **Discoverability**. It is recommended that all Human Interface Devices use limited discoverable mode only, since devices are normally used in a 1:1 relationship with a host. Once the device address is known by the main host, there is no need to allow further discovery of its address unless the user specifically allows it for a brief period.
- **Authentication/bonding/encryption**. This profile requires support for authentication and encryption for keyboards, keypads, and other HIDs that transmit sensitive information. This is due to the extraordinary sensitivity of the information that commonly travels from these devices (usernames, passwords, emails, passkeys). It is optional for other types of HIDs.

- **Configuration**. Bluetooth HIDs should be easy for consumers to configure out of the box. This profile will give examples of systematic configuration of Bluetooth peripherals.
- **Performance.** Bluetooth HIDs should have responsiveness (latency) similar to wired USB input devices, and provide superior performance to most other types of proprietary wireless input devices.
- **Battery life**. Bluetooth HIDs should fully utilize the power saving mechanisms provided by the Bluetooth Specification, such as Park, Hold, and Sniff modes, to achieve battery life comparable with existing wireless human input devices.
- **Host software**. The host will typically implement the HID profile by writing an interface driver (sometimes called a *miniport* driver on a PC host) between a standard HID class driver and the Bluetooth L2CAP and Link Manager layers.

1.5 Roving Networks' firmware overview

Roving Networks' Bluetooth modules and adapters run the firmware onboard which includes the Bluetooth protocol stack and profiles. This onboard firmware supports Bluetooth HID profile and Serial Port Profile (SPP). Switching between the profiles is done using software commands.

1.6 Profile configuration

Switching between the HID and SPP profile is done using the following commands:

```
S~,0                    //enables SPP protocol
R,1                     // reboot using SPP
```

To switch back to HID protocol, use the following command

```
S~,6                    // enables HID
R,1                     // reboot using HID
```

1.7 Device discovery and pairing

By default devices made using Roving Networks' HID profile will be discoverable as keyboard. The device type can be configured by using the HID flag register in firmware.

This is a bit mapped register and can be accessed in command mode on the module. The command to set the register is SH, <hex word value>

GH returns the current value of the register.

Default factory setting is 0000 which corresponds to keyboard.

The value is entered as a 4 char HEX word. At present time only the lower 9 bits are defined as follows:

Bit Position	9	8	7-4 (COD)	3	2-0
Definition	FORCE HID mode if PIO11 is high on power-up.	Toggle Virtual Keyboard on iOS when first connected	0000 = Keyboard 0001 = Game Pad 0010 = Mouse 0011 = COMBO 0100 = JOYSTICK 0101 = DIGITIZER 0110 = SENSOR 0111 = USE CFG 1XXX = reserved	Send Output reports over UART	Number of Paired devices to reconnect to

BITS 7-4 control two settings:

1. The COD that is advertized by the module
2. The HID report descriptor and hence the available reports that can be sent.

BIT 8 is an OVER RIDE ENABLE. If this bit is SET, the firmware will check the level of PIO11 on power up and if it is HIGH, it will enable HID mode. The default profile can then be set to SPP mode using "S~,0" to allow the SPP profile to boot by default (allowing remote configuration for example from BT clients with SPP) and PIO11 used to over ride SPP and run in HID mode.

2.0 HID RAW MODE

The basic HID mode allows the transmission of most common keys on a keyboard. Special keys (like multimedia keys) need thebe transmitted using the raw mode. The raw mode enables sending of raw HID reports, one report at a time.

By sending raw reports customers can build HID devices such as mouse, combo keyboard/mouse, joystick and gamepad.

The HID raw report is sent using 0xFD code. The raw report format is

Start (0xFD)	Length	Type	Raw report

Keyboard report:

0xFD	9 bytes	0x1	Modifier	0x00	Key code 1	Key code 2	Key code 3	Key code 4	Key code 5	Key code 6

Mouse raw report:

0xFD	5 bytes	0x2	Buttons	X-stop	Y-stop	Wheel

Consumer report in keyboard or combo mode:

0xFD	3 bytes	0x3	Byte 1	Byte 2

Joystick mode:

0xFD	6 bytes	Not used	Buttons	X1	Y1	X2	Y2

3.0 IMPLEMENTING KEYBOARD USING HID PROFILE

A full HID keyboard can be implemented using the HID profile by sending 0xFE code. The format for the keyboard report is as follows:

start	Length = 1 modifier + #keys	Keys from 1 to 6
0xFE	0, 2,3,4,5,6,7	1 to 6 bytes (not sent if null report)

Any key combination can be sent using this report and custom HID keyboards can be created.

Example: to send the '1' key the format would be: 0xFE 0x02 0x00 0x1E

A special case to send a "all keys down" report this is sent with: 0xFE 0x00

(This returns all keys to OFF or NOT PRESSED state)

3.1 Virtual Keyboard toggle using PIO9

When a HID connection is made to an iOS device, the Virtual Keyboard is hidden by default. It is often useful or required to pop the keyboard back up for data entry on the touch screen of the iOS device. Raising this PIO line will toggle the state of the Virtual Keyboard. The line needs to go from LOW to HIGH for at least 200ms for the toggle to occur.

Note that the virtual keyboard toggle must be enabled in the HID flag register for this feature to work.

3.2 Key-Mapper register

This register is used to allow any ASCII code to replace another ASCII code. It is useful in cases for example where one might want to toggle the iOS device on screen Keyboard which is code 0x7F, but the device cannot generate a 0x7F.

If the register is non-zero, the upper byte is the key to replace, and the lower is the replacement

The command to set the register is S=, <hex word value>

G= will return the current value of the register. The value also shows up in the Advanced settings using the "E" command.

Default factory setting is 0000 (not enabled)

The value must be entered as a 4 char HEX word.

Example, when the tilda, '~' , code is sent, toggle the keyboard.

To do this you would enter S=,7e7f

3.3 Disconnect key

A special key value of hex 0x00 (zero) has been created which if sent will cause a Bluetooth disconnect to occur. This can be useful to control the connection by just sending a single key.

In combination with the Key-Mapper above, any key can be used as a disconnect key.

Example: setup the "Z" Capital Z key to be the disconnect key.

Capital Z is a HEX 5A, so you would use

S=,5A00 to map this key to be the 0x00 disconnect key

3.4 Output reports

The connected device may send a report back to the module. If the Output Report Flag is set in the HIDFLAGS register, these will be output on the UART with the following format:

start	Num bytes	The report
0xFE	1-8	data

For example: HID keyboard Output reports the Keyboard LED status:

0xFE 0x2 0x1 <the LED status byte >

3.5 Key Lock Status

A special report code of 0xFF is reserved to return over the UART the current status of the keys.

function		status byte returned over the UART
Num Lock		1 (bit 0)
Caps Lock		2 (bit 1)
Scroll Lock		4 (bit 2)

4.0 CONSUMER REPORT

a HID raw report can be used to send some additional keys below, using 0xFD code.

Format is 0xFD 0x3 0x3 <low byte > <high byte >

Consumer key function	Report Bit
AC Home	0x1
AL Email Reader	0x2
AC Search	0x4
AL Keyboard Layout (virtual apple keyboard toggle)	0x8
Volume UP	0x10
Volume Down	0x20
Mute	0x40
Play/Pause	0x80
Scan Next Track	0x100
Scan Previous Track	0x200
Stop	0x400
Eject	0x800
Fast Forward	0x1000
Rewind	0x2000
Stop/Eject	0x4000
AL Internet Browser	0x8000

Example: to send a volume UP, you would send

0xFD 0x03 0x03 0x10 0x00

to actuate the key and then of course to release it

0xFD 0x03 0x03 0x00 0x00

5.0 APPENDIX: ASCII CODES-HID REPORT TABLES

6.0 REFERENCES

[1] Bluetooth SG, Human interface Profile overview
URL: https://www.bluetooth.org/Building/HowTechnologyWorks/ProfilesAndProtocols/HID.htm

[2] USB.org, HID usage tables
URL: http://www.usb.org/developers/devclass_docs/Hut1_12v2.pdf

[3] USB.org, HID technology
URL: http://www.usb.org/developers/hidpage/

Roving Networks, Inc.
102 Cooper Court
Los Gatos, CA 95032
+1 (408) 395-5300
sales@rovingnetworks.com
www.rovingnetworks.com

Roving Networks reserves the right to make corrections, modifications, and other changes to its products, documentation and services at any time. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

Roving Networks assumes no liability for applications assistance or customer's product design. Customers are responsible for their products and applications which use Roving Networks components. To minimize customer product risks, customers should provide adequate design and operating safeguards.

Roving Networks products are not authorized for use in safety-critical applications (such as life support) where a failure of the Roving Networks product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use.