DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

Gamepad

From Free60 Project

Table of contents

General information

The gamepads have 11 buttons, 2 triggers, 2 sticks and 1 D-Pad. The wired gamepad has a regular USB connector, the wireless uses the RF Module in the Xbox360. Both talk the same USB protocol.

The Play and Charge Kit for the wireless controller only provides power and a trickle charge. It does not change the wireless controller to a wired controller. The USB data lines are not active on the play and charge kit. It will not charge AA rechargeable batteries. The trickle charge is only available at the four prong jack at the bottom of the battery compartment. The play and charge can be plugged into any USB port, it does not have to be one on the 360.

The gamepad HID device

The gamepad is a regular USB HID device, but it has been crippled in a slight way:

- The device uses the `0xff` DeviceClass ('Vendor Specific') while normal HID devices use `0x03`. Therefore normal HID drivers won't attach to it automatically.
- The device has no USB Report Descriptor, making the operating system unable to determine its device layout.

Both problems are not hard to overcome; some operating systems (the BSDs for example) already override the USB Report Descriptors for some devices because they were shipped with broken ones.

A replacement report descriptor is available from the Free60 CVS repository (*http://cvs.sourceforge.net/viewcvs.py/*checkout*/free60/hid-desc/uxb360gp_rdesc.h*). The layout of this descriptor is the same as the Windows driver, except that the big X button has been mapped to button 11. On Windows, it's unmapped.

Input report

Once in a while, a USB HID device sends back a so-called input report which contains all information about its current state. The length of the input report is the same as the original Xbox gamepad; 20 bytes.

Its button/trigger/pad/stick alignment is as listed below:

Offset	Length (bits)	Description	Windows driver
0x00.0	8	Message type	
0x01.0	8	Packet size (20 bytes = 0x14)	
0x02.0	1	D-Pad up	D-Pad up
0x02.1	1	D-Pad down	D-Pad down
0x02.2	1	D-Pad left	D-Pad left
0x02.3	1	D-pad right	D-Pad right
0x02.4	1	Start button	Button 8

0x02.5	1	Back button	Button 7
0x02.6	1	Left stick press	Button 9
0x02.7	1	Right stick press	Button 10
0x03.0	1	Button LB	Button 5
0x03.1	1	Button RB	Button 6
0x03.2	1	Xbox logo button	
0x03.3	1	Unused	
0x03.4	1	Button A	Button 1
0x03.5	1	Button B	Button 2
0x03.6	1	Button X	Button 3
0x03.7	1	Button Y	Button 4
0x04.0	8	Left trigger	Z-axis down
0x05.0	8	Right trigger	Z-axis up
0x06.0	16	Left stick X-axis	X-axis
0x08.0	16	Left stick Y-axis	Y-axis
0x0a.0	16	Right stick X-axis	X-turn
0x0c.0	16	Right stick Y-axis	Y-turn
0x0e.0	48	Unused	

All eight-bit values are unsigned. The 16-bit values are signed little-endian. The first byte (Message type) will be 0x01 for a LED status message and 0x00 for a normal input report message.

Output report

LED Control

Some control over the LEDs surrounding the XBox button is provided, corresponding to the markings 1, 2, 3 and 4. This is controlled using message type 0x01.

To select a new pattern for the LEDs, send a 3-byte packet of the following form:

0103XX

0x01 is the message type, 0x03 is the message length, and 0xXX is the desired pattern:

Pattern	Description
0x00	All off
0x01	All blinking
0x02	1 flashes, then on
0x03	2 flashes, then on
0x04	3 flashes, then on
0x05	4 flashes, then on
0x06	1 on
0x07	2 on
0x08	3 on
0x09	4 on
0x0A	Rotating (e.g. 1-2-4-3)
0x0B	Blinking*
0x0C	Slow blinking*
0x0D	Alternating (e.g. 1+4-2+3), then back to previous*

The previous setting will be used for any itmes with * (all blinking, or 1, 2, 3 or 4 on).

Rumbler Control

Rumbling is also similar to on the original controller. Rumble commands take the following 8-byte form:

000800bbll000000

Where b is the speed to set the motor with the big weight, and l is the speed to set the small weight (0x00 to 0xFF in both cases).

The headset-port

- Headset Port

Missing image
Headset_port_pinout.jpg
Pinout for the headset port on the wired and wireless Xbox 360 controller

Baud Rate: Unknown, Data 1: RX or TX, Data 2: RX or TX

A mini-keyboard for text entry can be plugged into this port.

The headset data protocol

FreeBSD ships with a driver called ugen(4) which is just a fallback driver for USB devices that do not have a matching driver. It allows you to read and write to the descriptors of the device. Descriptor 3 is used for the microphone. Descriptor 4 is the earpiece.

At this moment there isn't a lot of information available about the transfer protocol. The protocol for the microphone and the earpiece are the same, but the latter one uses half the sample rate of the first one. The following test shows this:

```
$ cat /dev/ugen0.3 > myvoice
# tell a funny joke to the microphone and press ^C
$ cat myvoice > /dev/ugen0.4
```

Playback will take twice as long.

The microphone emits 8000 bytes per second of 4 bits signed PCM, thus it's 16 KHz. The earpiece only consumes 4000 bytes, so it can only emit 8 KHz PCM (4 KHz sound at best).

lsusb output

Linux's `lsusb` utility tells us the following about the gamepad.

```
Bus 002 Device 003: ID 045e:028e Microsoft Corp.
Device Descriptor:
  bLength                18
  bDescriptorType         1
  bcdUSB               2.00
  bDeviceClass          255 Vendor Specific Class
  bDeviceSubClass       255 Vendor Specific Subclass
  bDeviceProtocol       255 Vendor Specific Protocol
  bMaxPacketSize0         8
  idVendor           0x045e Microsoft Corp.
  idProduct          0x028e
  bcdDevice            1.10
  iManufacturer           1
  iProduct                2
  iSerial                 3
  bNumConfigurations      1
  Configuration Descriptor:
    bLength                 9
    bDescriptorType         2
    wTotalLength          153
    bNumInterfaces          4
    bConfigurationValue     1
    iConfiguration          0
    bmAttributes         0xa0
      (Bus Powered)
      Remote Wakeup
    MaxPower              500mA
    Interface Descriptor:
      bLength                 9
      bDescriptorType         4
      bInterfaceNumber        0
      bAlternateSetting       0
      bNumEndpoints           2
      bInterfaceClass       255 Vendor Specific Class
      bInterfaceSubClass     93
      bInterfaceProtocol      1
      iInterface              0
      Endpoint Descriptor:
        bLength                 7
        bDescriptorType         5
        bEndpointAddress     0x81  EP 1 IN
        bmAttributes            3
          Transfer Type            Interrupt
          Synch Type               None
          Usage Type               Data
        wMaxPacketSize     0x0020  1x 32 bytes
        bInterval               4
      Endpoint Descriptor:
        bLength                 7
        bDescriptorType         5
        bEndpointAddress     0x02  EP 2 OUT
        bmAttributes            3
          Transfer Type            Interrupt
          Synch Type               None
          Usage Type               Data
        wMaxPacketSize     0x0020  1x 32 bytes
        bInterval               8
    Interface Descriptor:
      bLength                 9
      bDescriptorType         4
      bInterfaceNumber        1
      bAlternateSetting       0
      bNumEndpoints           4
      bInterfaceClass       255 Vendor Specific Class
      bInterfaceSubClass     93
      bInterfaceProtocol      3
      iInterface              0
      Endpoint Descriptor:
        bLength                 7
        bDescriptorType         5
        bEndpointAddress     0x83  EP 3 IN
        bmAttributes            3
          Transfer Type            Interrupt
          Synch Type               None
          Usage Type               Data
        wMaxPacketSize     0x0020  1x 32 bytes
        bInterval               2
      Endpoint Descriptor:
        bLength                 7
        bDescriptorType         5
        bEndpointAddress     0x04  EP 4 OUT
        bmAttributes            3
          Transfer Type            Interrupt
          Synch Type               None
          Usage Type               Data
        wMaxPacketSize     0x0020  1x 32 bytes
        bInterval               4
      Endpoint Descriptor:
        bLength                 7
        bDescriptorType         5
        bEndpointAddress     0x85  EP 5 IN
        bmAttributes            3
          Transfer Type            Interrupt
          Synch Type               None
          Usage Type               Data
        wMaxPacketSize     0x0020  1x 32 bytes
        bInterval              64
      Endpoint Descriptor:
        bLength                 7
        bDescriptorType         5
        bEndpointAddress     0x05  EP 5 OUT
        bmAttributes            3
          Transfer Type            Interrupt
          Synch Type               None
          Usage Type               Data
        wMaxPacketSize     0x0020  1x 32 bytes
        bInterval              16
    Interface Descriptor:
      bLength                 9
      bDescriptorType         4
      bInterfaceNumber        2
      bAlternateSetting       0
      bNumEndpoints           1
      bInterfaceClass       255 Vendor Specific Class
      bInterfaceSubClass     93
      bInterfaceProtocol      2
      iInterface              0
      Endpoint Descriptor:
        bLength                 7
        bDescriptorType         5
        bEndpointAddress     0x86  EP 6 IN
        bmAttributes            3
          Transfer Type            Interrupt
          Synch Type               None
          Usage Type               Data
        wMaxPacketSize     0x0020  1x 32 bytes
        bInterval              16
    Interface Descriptor:
      bLength                 9
      bDescriptorType         4
      bInterfaceNumber        3
      bAlternateSetting       0
      bNumEndpoints           0
      bInterfaceClass       255 Vendor Specific Class
      bInterfaceSubClass    253
      bInterfaceProtocol     19
      iInterface              4
      UNRECOGNIZED:  06 41 00 01 01 03
```

Speculation

- Rumors that both the wired gamepad and wireless dongle share the same interface, but probably won't have the same USB device IDs.
- The last six bytes of the input descriptor are for analog face buttons. The information on the web is contradictory. I know that the controller did have pressure sensitive face buttons originally. Some web sites now say that it does not, so they must have been scrapped. Others say that it still does have them. If it does not the bytes are just a relic, but if the controler does have the analog buttons then there must be some form of toggle mechanism.

Retrieved from "http://www.free60.org/wiki/Gamepad"

Categories: Hardware

- This page was last modified 13:41, 17 Feb 2008.
-