An Introduction to PS/2 and ps2emu

Stephen Chandler “Lyude” Paul
What is PS/2?

• Old protocol for mice and keyboard
• Still used in most laptops
• Low bandwidth
• Only allows for two finger multitouch
Why is it still used?

- Designed when PS/2 was the only option
- Older versions of Windows still need it
- Mouse support in UEFI (PS/2 doesn't need special drivers)
Alternatives

- I²C: Used by Chromebooks, some Dells
- RMI4/SMBus: Supported by Lenovo ThinkPads and some HP notebooks. Linux support is a WIP.
- Not widely used, but some laptops have them
PS/2 in the kernel

1. Keyboard
2. TouchPad
3. TrackPoint

- i8042 serio driver
  - serio0 (Keyboard)
    - PS/2 Keyboard Driver
  - serio1 (TouchPad)
    - Synaptics Driver
  - serio2 (Trackpoint thru PS/2 Passthrough)
    - TrackPoint Driver
The basic PS/2 protocol for mice

• Each packet is 11 bits
• Payload is 8 bits
• Packets are sent in groups of 3 for normal 3 button mice, groups of 4 for mice with 5 buttons
• There are some commands you can send:
  • 0xFF – Reset the mouse
  • 0xFE – Resend
  • 0xF6 – Set defaults
  • Etc.
The basic PS/2 protocol for mice

![PS/2 relative motion packet diagram]

Figure 3-2. PS/2 relative motion packet
Problems with PS/2

• i8042 controllers are picky, reject anything that isn't PS/2
• Some even modify packets
• Can't do absolute positioning, pressure, etc.
Working around PS/2

- PS/2 alone wasn't sufficient for touchpads
- Manufacturers made their own protocols over PS/2
- Add more information into PS/2 packets
- Looks like normal PS/2 events to the i8042
Example of a packet from SynPS/2

<table>
<thead>
<tr>
<th>Byte</th>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte 1</td>
<td>1</td>
<td>0</td>
<td>W value</td>
<td>3..2</td>
<td>0</td>
<td>W val 1</td>
<td>Right</td>
<td>Left</td>
</tr>
<tr>
<td>Byte 2</td>
<td></td>
<td></td>
<td>Y position</td>
<td>11..8</td>
<td></td>
<td>X position</td>
<td>11..8</td>
<td></td>
</tr>
<tr>
<td>Byte 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Z pressure</td>
<td>7..0</td>
<td></td>
</tr>
<tr>
<td>Byte 4</td>
<td>1</td>
<td>1</td>
<td>Y pos 12</td>
<td>X pos 12</td>
<td>0</td>
<td>W val 0</td>
<td>R/D</td>
<td>L/U</td>
</tr>
<tr>
<td>Byte 5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X position</td>
<td>7..0</td>
<td></td>
</tr>
<tr>
<td>Byte 6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y position</td>
<td>7..0</td>
<td></td>
</tr>
</tbody>
</table>

Figure 3-4. PS/2 absolute X/Y/Z/W motion packet (Wmode = 1)
Working around PS/2

- Touchpads don't use extended protocols by default, so they work without special drivers
- Activated with special command sequences
- For Synaptics: 0xE8 (set sample rate) with the two bit argument containing part of the special command
Working around PS/2

• For some touchpads, PS/2 is a secondary protocol (e.g. RMI4)
• Doesn't always get the same QA testing
• Quirks for these touchpads are very common
• Quirks both in the protocol and the touchpad
ps2emu

• A new thing I made!
• My intern project at Red Hat
• Allows us to record PS/2 devices
• Recordings can be replayed on other machines
• Lets us reproduce bugs in PS/2 drivers locally
• Has helped fix various bugs in touchpad drivers
ps2emu

- Two commands:
  - `ps2emu-record`: Records the PS/2 devices
  - `ps2emu-replay`: Replays the PS/2 devices

- One kernel module:
  - `userio`: Allows for userspace to create virtual serio ports to communicate to the kernel with
ps2emu-record

**Diagram:**
- **Keyboard**
- **TouchPad**
- **TrackPoint**
- **i8042 serio driver**
- **ps2emu-record**
- **serio0 (Keyboard)**
  - **PS/2 Keyboard Driver**
- **serio1 (TouchPad)**
  - **Synaptics Driver**
- **serio2 (Trackpoint thru PS/2 Passthrough)**
  - **TrackPoint Driver**
ps2emu File Format

T: A
S: Init
E: 0       S f2 # (parameter)
E: 2725    R fa # (interrupt, 1, 12)
E: 3900    R 00 # (interrupt, 1, 12)
E: 4356    S f2 # (parameter)
E: 7038    R fa # (interrupt, 1, 12)
E: 8265    R 00 # (interrupt, 1, 12)
E: 8479    S f6 # (parameter)
E: 11181   R fa # (interrupt, 1, 12)
E: 11378   S f3 # (parameter)
E: 14214   R fa # (interrupt, 1, 12)
...

Demonstration!
Limitations of ps2emu

• Doesn't emulate a touchpad, just repeats whatever is in the log
• Interaction between driver and ps2emu must be the same as the log
  • Recordings may not work between kernel versions
Regression tests?

- No working solution yet
- ps2emu-replay isn't enough for this, we need to be smarter
- Must not break whenever the driver changes
Regression tests?

- Mapping TouchPad registers:
  - Not all TouchPads expose registers
  - Complex
  - Need every register value or it won't work
Regression tests?

- Emulating a TouchPad
  - Very complex
  - Dummy device might be prone to bugs
  - Need one recording for each TouchPad
Summary

• PS/2
  • Most common protocol in use for touchpads
  • Old, very backwards compatible
  • Transparent to the i8042
  • Manufacturer specific protocols
• ps2emu
  • Records and replays PS/2 devices
  • Potential for regression tests
Links

1) Diagrams of packet layouts taken from *Synaptics PS/2 TouchPad Interfacing Guide*

• ps2emu userspace tools available here:
  • Git: https://github.com/Lyude/ps2emu
  • Fedora copr:
    https://copr.fedoraproject.org/coprs/lyude/ps2emu-tools/

• userio available here:
  • Git: https://github.com/Lyude/ps2emu-kmod
Summary

- PS/2
  - Most common protocol in use for touchpads
  - Old, very backwards compatible
  - Transparent to the i8042
  - Manufacturer specific protocols
- ps2emu
  - Records and replays PS/2 devices
  - Potential for regression tests
Summary

- PS/2
  - Most common protocol in use for touchpads
  - Old, very backwards compatible
  - Transparent to the i8042
  - Manufacturer specific protocols
- ps2emu
  - Records and replays PS/2 devices
  - Potential for regression tests