Paravirtualized USB Support for Xen: Status Update

Noboru Iwamatsu
n_iwamatsu@jp.fujitsu.com
FUJITSU LABORATORIES LTD.

Xen Summit at Oracle Feb 24-25, 2009
Contents

- Background and Motivation
- Development Status
- PVUSB Design
- Demo (Video)
- Test Results
In client virtualization, special Service-VMs work in the background of User-VM, and provide various functions.

Both User-VM and Service-VMs require using the USB devices at the same time by the same USB host controller.
Motivation

USB support for Xen today

- Qemu-dm UHCI emulation
  - USB 1.1 only
- PCI pass-through with IOMMU
  - Works well. But the entire controller is assigned to a single domain and can’t be used from other domains.

Current options are not suitable for client virtualization!

- We have started development to achieve a good performance and flexibility!

  Aug. 2008  Started development.
  Proposed in XCI (Xen Client Initiative).
  Sep. 2008  “PV USB support” was on the roadmap of Xen 3.4.
  Nov. 2008  Just started working!
  Xen Summit @Tokyo 2008
Development Status

CHANGES from the last summit

- **What we did**
  - Emulation of the CLEAR_HALT command added in the backend.
  - RING operations and queue usage changed.
  - Initial hotplug code added.

- **Results**
  - Various devices worked (HID, Storage, Serial, Ethernet)
  - Performance improved (6MB/s to 33MB/s on USB storage)
  - Hotplug worked, but unplug not worked yet :-(

- **What we left**
  - No code posted yet!
  - Disconnection and canceling functions not completed.
  - Not well tested.
PVUSB Design: Driver Architecture

- **Frontend driver (usbfront)** - Implemented as a USB host controller driver
- **Backend driver (usbbback)** - Implemented as a USB function driver

**urb (USB Request Block structure):**
In the Linux kernel, all USB drivers communicate with urb. (described in include/linux/usb.h).

---

**Driver domain kernel-space**

- **usbbback**
  - Xen backend interface
  - xenbus
  - RING

**USB Function Driver**

**USB Core**

**Host Controller Driver**

**Hardware** (USB Host Controller)

**Guest domain kernel-space**

- **USB Function Driver**
  - USB Core
  - Virtual USB Host Controller Driver
  - RING
  - xenbus

**Xen frontend interface**

**RING**

**usb_alloc_urb()**

**Free or reuse**

**urb_enqueue()**

**usb_hcd_giveback_urb()**

**urb_complete**

**Free or reuse**

**usb_submit_urb()**

**Free**

**usb_alloc_urb()**

**Cloned urb**

**Cloned urb**

**RING request**

**RING response**
Hotplug-rule is set from sysfs interface in the backend driver.

- The hotplug-rule format
  - `<usbbusname>:<domid>:<vusb number>:<virtual port number>`

- Example settings
  - `% echo 1-2.3:1:0:3 > /sys/bus/usb/drivers/usback/new_vport`
  - `% echo 1-4:2:0:1 > /sys/bus/usb/drivers/usback/new_vport`
  - `% echo 1-2.1:2:0:2 > /sys/bus/usb/drivers/usback/new_vport`
PVUSB Design: Source Code Tree

Xen USB public I/O header
usbfif header file

Xen USB backend driver
Backend interface management
Makefile
USB backend RING operations and URB transmissions
usbfback header file
USB stub driver – grabbing and managing USB devices
Xenbus operations

Xen USB frontend driver
Makefile
usbffront header file
Debugging functions for Host Controller
USB2.0 Host Controller interface
Root Hub emulations for Host Controller
USB frontend RING operations and URB transmissions
Xenbus operations
Hotplug Keyboard, Mouse, Flash drive, HDD and Ethernet.

All devices successfully connected to guest domain and work fine.
## Results: Tested Devices (USB2.0)

<table>
<thead>
<tr>
<th>Device Type</th>
<th>Name</th>
<th>Manufacturer</th>
<th>Transfer Type</th>
<th>Driver</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flash drive</td>
<td>RUF2-R2GS</td>
<td>Buffalo</td>
<td>Bulk</td>
<td>usb-storage</td>
<td>✔️</td>
</tr>
<tr>
<td>Flash drive</td>
<td>RUF-C1G/U2</td>
<td>Buffalo</td>
<td>Bulk</td>
<td>usb-storage</td>
<td>✔️</td>
</tr>
<tr>
<td>Flash drive</td>
<td>DataTraveler DTI/1G</td>
<td>Kingston</td>
<td>Bulk</td>
<td>usb-storage</td>
<td>✔️</td>
</tr>
<tr>
<td>HDD</td>
<td>HDCN-U500</td>
<td>IO DATA</td>
<td>Bulk</td>
<td>usb-storage</td>
<td>✔️</td>
</tr>
<tr>
<td>Memory Card Reader/Writer</td>
<td>BSCRA38U2</td>
<td>Buffalo</td>
<td>Bulk</td>
<td>usb-storage</td>
<td>✔️</td>
</tr>
<tr>
<td>Ethernet (100Base-TX)</td>
<td>ETX2-US2</td>
<td>IO DATA</td>
<td>Bulk/Intr</td>
<td>pegasus</td>
<td>✔️</td>
</tr>
<tr>
<td>Ethernet (1000Base-T)</td>
<td>ETG2-US2</td>
<td>IO DATA</td>
<td>Bulk/Intr</td>
<td>asix</td>
<td>❌ [1]</td>
</tr>
<tr>
<td>VGA</td>
<td>USB2.0 SVGA Adapter</td>
<td>KAIREN</td>
<td>Bulk/Intr</td>
<td>sisusbvga</td>
<td>⚠️ [2]</td>
</tr>
</tbody>
</table>

- ✔️ Works
- ⚠️ Works with issues
- ❌ Not work

[1] Interrupt URB fails, and the device goes into a stall.
[2] No URB fails, but fails to output the correct screen.
# Results: Tested Devices (USB1.1)

<table>
<thead>
<tr>
<th>Device Type</th>
<th>Name</th>
<th>Manufacturer</th>
<th>Transfer Type</th>
<th>Driver</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keyboard</td>
<td>FMV-KB333</td>
<td>Fujitsu</td>
<td>Intr</td>
<td>usbhid</td>
<td>✓</td>
</tr>
<tr>
<td>Keyboard</td>
<td>FKB-108-EU</td>
<td>FILCO</td>
<td>Intr</td>
<td>usbhid</td>
<td>✓</td>
</tr>
<tr>
<td>Mouse</td>
<td>Cordless Notebook Mouse</td>
<td>Logitech</td>
<td>Intr</td>
<td>usbhid</td>
<td>✓</td>
</tr>
<tr>
<td>Mouse</td>
<td>Optical Mouse USB</td>
<td>Logitech</td>
<td>Intr</td>
<td>usbhid</td>
<td>✓</td>
</tr>
<tr>
<td>FDD</td>
<td>USB-FDU</td>
<td>Y-E DATA</td>
<td>Bulk/Intr</td>
<td>usb-storage</td>
<td>✓</td>
</tr>
<tr>
<td>Serial</td>
<td>USB-RSAQ5</td>
<td>IO DATA</td>
<td>Bulk/Intr</td>
<td>pl2303</td>
<td>✓</td>
</tr>
<tr>
<td>Webcam</td>
<td>WebCam 3 USB</td>
<td>Creative Labs</td>
<td>Isoc</td>
<td>ov511</td>
<td><img src="https://example.com" alt="3" /></td>
</tr>
<tr>
<td>Webcam</td>
<td>WebCam NX pro</td>
<td>Creative Labs</td>
<td>Isoc</td>
<td>gspca</td>
<td><img src="https://example.com" alt="4" /></td>
</tr>
</tbody>
</table>

- ✓ Works
- ![3](https://example.com) Works with issues
- ![4](https://example.com) Not work

[3] No URB fails, but incorrect horizontal stripes are into the video image.
[4] No URB fails, but the video image has incorrect blinking and flickers.
Results: Performance (USB Storage)

USB Mass Storage Read/Write Throughput (MB/s)

<table>
<thead>
<tr>
<th>USB Device</th>
<th>Read Throughput</th>
<th>Write Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td>USB Flash Drive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read</td>
<td>29.0</td>
<td>31.5</td>
</tr>
<tr>
<td>Write</td>
<td>20.3</td>
<td>24.9</td>
</tr>
<tr>
<td>USB HDD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read</td>
<td>33.3</td>
<td>33.8</td>
</tr>
<tr>
<td>Write</td>
<td>24.9</td>
<td>25.2</td>
</tr>
</tbody>
</table>

Paravirtualized USB vs Dom0 native driver:
- Read: 92% vs native
- Write: 82% vs native
- Read: 99% vs native
- Write: 99% vs native

Test environment:
- CPU: Intel Core 2 Duo 6700 2.66GHz
- Kernel: linux-2.6.18-xen (c/s 789), x86_64
- Xen: 3.3.0 release, x86_64
Results: Performance (USB Ethernet)

TCP throughput measured by ttcp command.

USB Ethernet: IODATA ETX2-US2

Test environment:

CPU: Intel Core 2 Duo 6700 2.66GHz
Kernel: linux-2.6.18-xen (c/s 789), x86_64
Xen: 3.3.0 release, x86_64
Results: CPU Utilization (USB Storage)

Xentop average while Read/Writ with dd command.

USB HDD: IODATA HDCN-U500

Test environment:
- CPU: Intel Core 2 Duo 6700 2.66GHz
- Kernel: linux-2.6.18-xen (c/s 789), x86_64
- Xen: 3.3.0 release, x86_64
Results: CPU Utilization (USB Ethernet)

CPU utilization: USB Ethernet Rx/Tx

Xentop average while using ttcp command.

Test environment:

CPU: Intel Core 2 Duo 6700 2.66GHz
Kernel: linux-2.6.18-xen (c/s 789), x86_64
Xen: 3.3.0 release, x86_64

USB Ethernet: IODATA ETX2-US2
TODO & Future Work

**TODO (for posting the code)**
- Complete the device disconnection and the URB canceling functions.
  - Current PVUSB can hotplug, but can’t unplug, and can’t cancel the URBs that already transferred to the backend.
  - When the device fails and actually stops, current PVUSB can do nothing.
- Add Xend support.

**Future Work (after initial posted)**
- Isochronous transfer support (for WebCAM)
  - Driver has no errors, the problem seems timing related.
- Some optimizations
  - Reducing the buffer allocation overhead
  - Better handling of the large buffer transfer
  - Netchannel2 support

We take a month to post the code.
Conclusion

Paravirtualized USB achieved:
- Various devices worked.
  - Worked devices
    - Keyboard(2), Mouse(2), FDD(1), Serial(1), Flash drive(3), HDD(1), Ethernet(1)
- Performance improved.
  - USB Storage: 99 % of the native driver
  - USB Ethernet: 67% - 86% of the native driver
- Hotplug worked.

Toward the initial release
- Complete the several essential functions.
- More test
  - SMP(Quad-core), multiple-domain, heavy load, …
Appendix: dmesg output

dmesg output after usbfront loaded.
Appendix: lsusb output

lsusb output after connected the devices.
Appendix: Webcam(ov511) output

xawtv video streaming output
Appendix: USB-VGA(sisusbvga) output

USB-VGA output: CentOS 5.2 default login screen (res. 1024x768)