KVM: PCI device assignment

Chris Wright
Red Hat
August 10, 2010

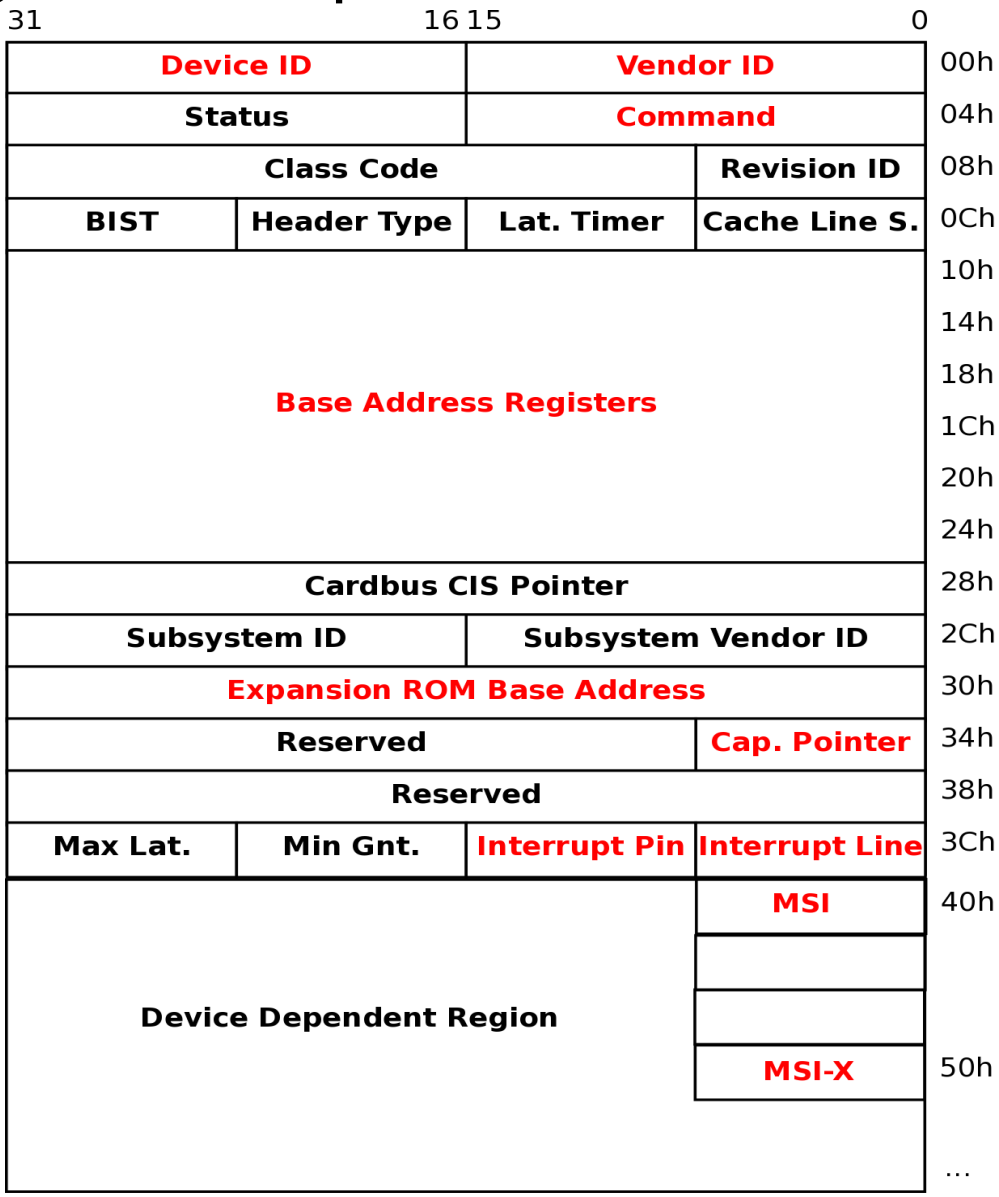
Agenda

- Anatomy of a PCI device
- Current mechanism
- Shortcomings
- Future

PCI: Device Anatomy

- PCI Configuration space
  - Header + Device Dependent Region
- Device-specific registers
  - Configuration space, PIO, MMIO
- Interrupts
  - INTx, MSI, MSI-X
- DMA

PCI: Configuration Space



PCI: Device-specific registers

"It is strongly recommended that PCI Express devices place no registers in Configuration Space other than those in headers or Capability structures architected by applicable PCI specifications." -- PCIe 2.1

PCI: Device-specific registers

"It is strongly recommended that PCI Express devices place no registers in Configuration Space other than those in headers or Capability structures architected by applicable PCI specifications." -- PCIe 2.1

And I want a pony

PCI: Interrupts

- INTx shared interrupts
- MSI, MSI-X
  - 0xFEE0_0000h

PCI: DMA

- Program device with bus addresses
- Device capable of issuing PCI memory transactions
- IOMMU required for any isolation/integrity

PCI: IOMMU

- DMA isolation
- Interrupt protection
- Routing ID issues
- ACS

KVM Device Assignment

- Goal: guest owns and drives device
- Requirements: maintain isolation
- Ideal world: maintain mobility

KVM Device Assignment: Mechanism

- Libvirt
- Qemu
- KVM

KVM Device Assignment: libvirt

- Handles complex reset logic
- Handles ACS filtering
- Unbind physical driver, bind pci-stub
- Set proper security context for sysfs files
- Rest pushed to qemu

KVM Device Assignment: qemu

- Add device to guest pci bus
- Manages config space access
  - PCI sysfs files
- Calls KVM ioctl interface

KVM Device Assignment: ioctl

- KVM_ASSIGN_PCI_DEVICE
  - enable pci device, reserve pci resources
  - reset device
  - create iommu domain, map guest
  - attach device to iommu domain
- KVM_DEASSIGN_PCI_DEVICE
  - detach device from iommu domain
  - reset device
  - release pci resources, disable pci device

KVM Device Assignment: ioctl

- KVM_ASSIGN_DEV_IRQ
  - enable host irq (INTx, MSI, MSI-X)
  - enable guest irq (INTx, MSI, MSI-X)
- KVM_DEASSIGN_DEV_IRQ
  - disable host irq
  - disable guest irq
- KVM_ASSIGN_SET_MSIX_NR
- KVM_ASSIGN_SET_MSIX_ENTRY

KVM Device Assignment: Shortcomings

- Solved issues
  - < 4k BAR (slow map)
  - Deprivileged QEMU: sysfs resource files, ioport access
  - hot unplug
- Unsolved issues
  - Capabilities mess (PCI and PCIe)
  - Topology disconnect
  - Memory locking
  - Device whitelist
  - ROM
  - SR-IOV management
  - Shared interrupts
  - KVM as device driver

KVM Device Assignment: Future

- VFIO
- PRI
- graphics?

Future: VFIO

- UIO based
  - UIO provides crude interrupt support
  - rest via PCI sysfs files (config, BAR)
  - no MSI or IOMMU support
- VFIO
  - Tom Lyon posted v3 in July
  - PCI config space access and virtualization
  - BARs, read/write/mmap for MMIO, read/write for PIO
  - INTx, MSI, MSI-X interrupts via eventfd
  - IOMMU support via UIOMMU
  - works with qemu and userspace drivers

Future: VFIO interfaces

- VFIO_DMA_MAP_IOVA
- VFIO_EVENTFD_{IRQ,MSI,MSIX}
- VFIO_BAR_LEN
- VFIO_DOMAIN_{,UN}SET
- Magic values for BAR/config space access via read/write/mmap

Future: VFIO qemu

- Alex Williamson posted in July 2010
- Can replace existing implementation
  - not KVM only
- Interrupts through qemu
- PCI 2.3 only for INTx

Future: PRI

- PCI specification update
- Requires I/O device hardware support
- Can eliminate memory locking

Future: Graphics

- PCI device assignment is generic
- Graphics devices are special


redhat
Virtualibation