VOLUME MANAGEMENT DEVICE (VMD) SUBDEVICE ASSIGNMENT

Jon Derrick

Linux Plumber's Conference 2020

8/26/2020

Volume Management Device Technical QRD

PCIe-to-PCIe bridge to new PCIe segment

- Intended for NVMe devices + Root Ports and Bridges
- Subdevice Requester-ID changed to VMD endpoint
- Subdevice MSI/X Interrupts remapped into VMD endpoint MSI/X table
- VMD driver collects IRQs and iterates into subdevice ISRs
- 1-3 in Xeon CPUs, recent client CPUs
- Entire VMD domain existing in one IOMMU group per VT-D rules on bridges

Passthrough Benefits

Passing through VMD endpoint (Possible today)

- Guest OS driver enumerates VMD domain
- Native hotplug support if hotplug port is on VMD domain
- Cross-OS RAID support with Intel drivers/tools

Passing through individual subdevices

- Separating VMD subdevices to individual guests
- Better hypervisor support for known type 0 devices (NVMe)
- OEM BIOS don't need to add a VMD enable/disable knob

Technical Challenges

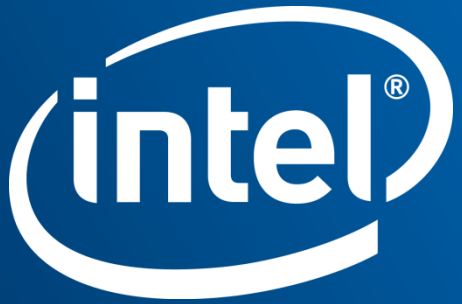
DMA Remapping

- Past DMAR model indirected subdevice ops to VMD endpoint's struct device
- < v5.6, DMAR model clears whole SID's remapping context on device removal
- VMD subdevices now handled by DMAR code. Keeps contexts programmed until 'parent' (VMD endpoint) device is removed (pci_real_dma_dev())
- For subdevice guest passthrough, is it safe to hold DMA remapping contexts after subdevice removal or should the individual contexts be cleared?
- Would it be preferable to use PASID routing?

Technical Challenges

Interrupt Remapping

- Subdevice interrupt would still dispatch interrupt from Host's VMD endpoint
- Host VMD driver ISR would be unaware of guest subdevice ISR
- Add VFIO ISR to pass interrupts from Host VMD driver to guest?
- Should MSI/X be avoided?
- Eg, poll-based interrupts or IMS?


intel®