Open Source Technology Center

Intel Virtualization Technology Overview

Yu Ke

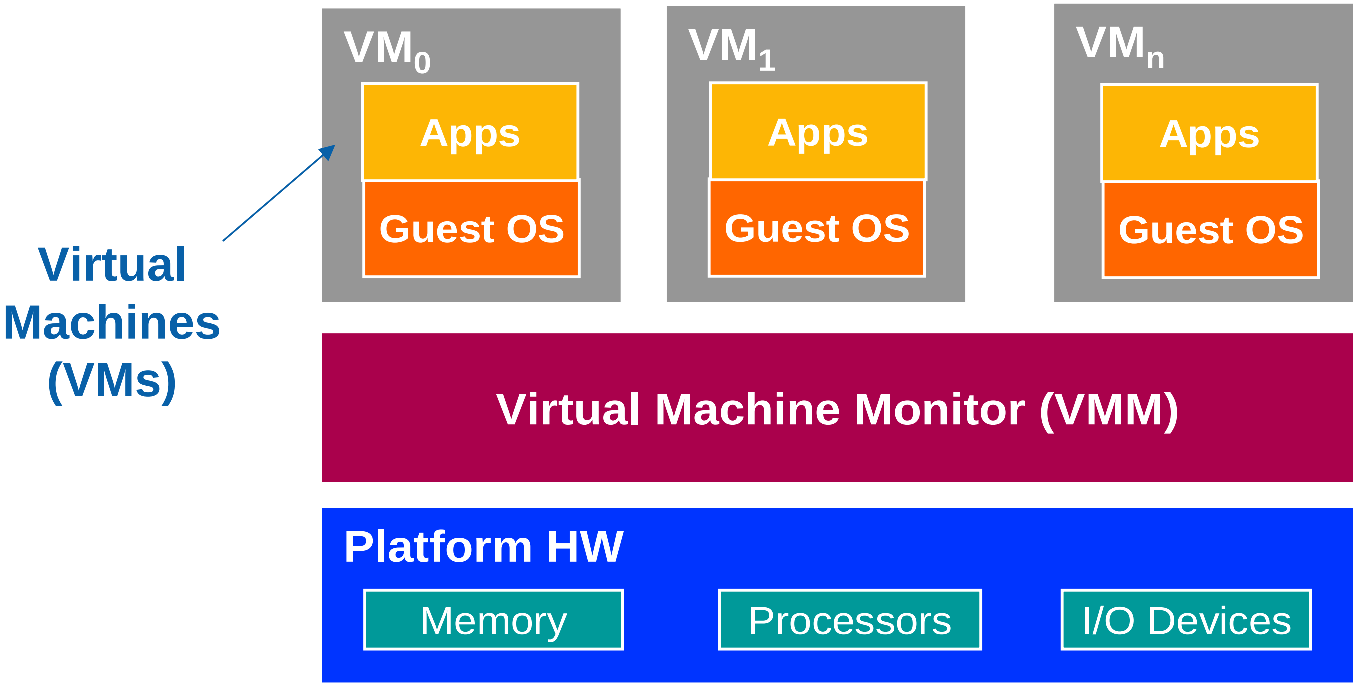
SSG System Software Division

intel Software

Agenda

- Virtualization Overview
- Intel Virtualization Technology

What is Virtualization

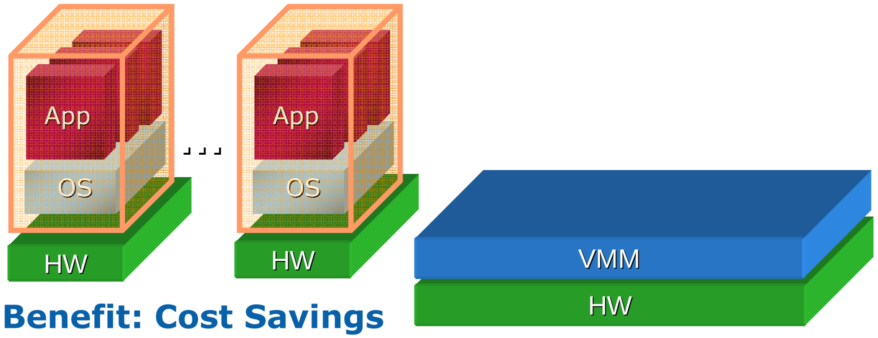


- **"Any problem in computer science can be solved with another layer of indirection." – Butler Lampson (1992 Turing Award Lecture)**
- **VMM is a new layer of system software**
  - Support multiple guest OSes
  - De-privilege each OS to run as Guest OS

Server Virtualization Usages

Today

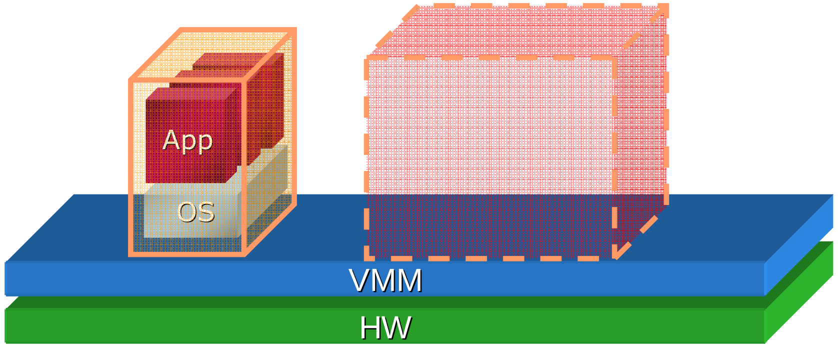
Server Consolidation



Benefit: Cost Savings

- **Power and Cooling**
- **Hardware, Software, Management**

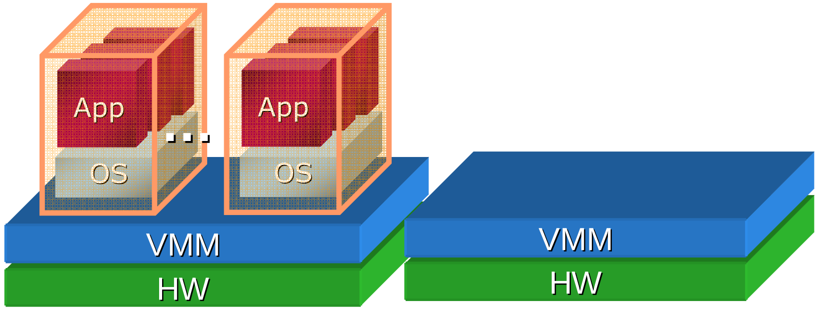
R&D Production



Benefit: Business Agility and Productivity

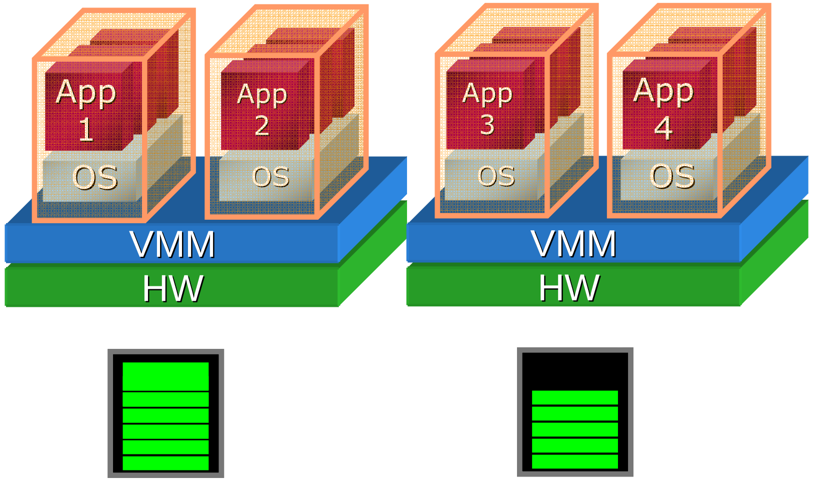
Emerging

Disaster Recovery



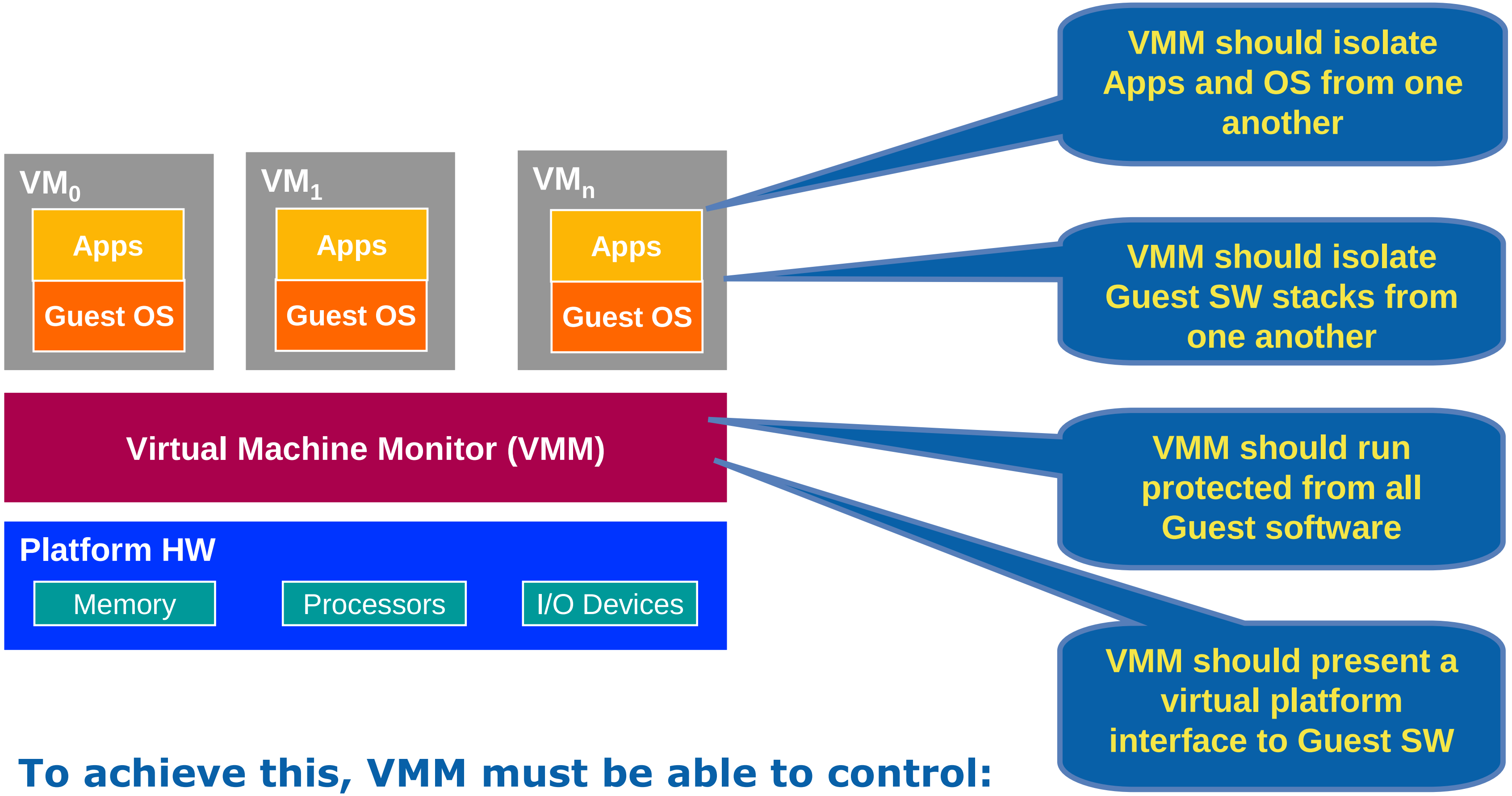
Benefit: Business Continuity and Operational Efficiency

Dynamic Load Balancing



- Benefit: Productivity

VMM Requirements



VMM should isolate Apps and OS from one another

VMM should isolate Guest SW stacks from one another

VMM should run protected from all Guest software

VMM should present a virtual platform interface to Guest SW

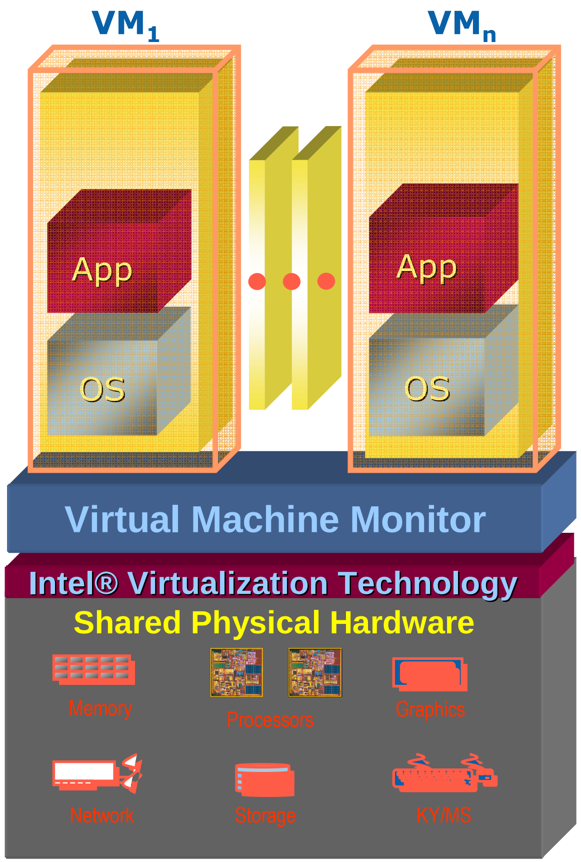
To achieve this, VMM must be able to control:

- CPU(s)
- Memory
- I/O devices

Agenda

- Virtualization Overview
- **Intel Virtualization Technology**
  - ➢ CPU Virtualization - VT-x
  - ➢ Memory Virtualization: EPT, VPID
  - ➢ I/O Virtualization: VT-d, VT-c (VMDq, SR-IOV)

Intel Virtualization Technology



Intel Virtualization Technology is a new hardware layer in Intel CPU/chipset/platform.

- **Make VMM implementation simplified**
- **Improve VMM efficiency**
- **Support full virtualization to be able to run unmodified guest**

"We are on record as saying that VT is the most significant change to PC architecture this decade"

Martin Reynolds, Gartner Senior Analyst – eWeek September 9, 2004

Intel® Virtualization Technology Evolution

	Yesterday: No HW Support	2005-2006 With CPU Support	2007-2008 With Chipset Support & IO improvements	
Vector 3: IO Device Focus				•Assists for IO sh • PCI IOV co • VMDq: Multi-co MA t ning • IO virtualization assists **IOV** **VMDq**
Vector 2: Chipset Focus			Core support for IO robustness & device assig DMA remapp **VT-d**	Interrupt filtering & rema VT- s to track PCI SIG IOV **VT-d2**
Vector 1: Processor Focus		Close basic proc “vi holes” in Intel® 64 & Itanium CPUs **VT-x**	Performance extensio /VTi EPT, VPID, ECRR, APIC-V **VT-x2**	Perf improvements for inte intens faster VM boot **VT-x3**
VMM Software Evolution	Software-only VMMs Binary translation Paravirtualization Device emulations	Simpler and more secure VMM through use of hardware VT support	Better IO/CPU perf and functionality via hardware-mediated access to memory	Richer IO-device functionality and IO resource sharing

CPU Virtualization

- **Goal: present functional virtual CPU to Guest OS**
- **CPU from OS point of view:**
  - ➢A set of hardware resource: general register (EAX, EBX), FPU register, control register (EFLAG, EIP, CR3…)
  - ➢Support several privilege: Ring 0 ~ Ring 3
  - ➢Run instruction with pre-defined semantic:
    - privileged instruction
    - non-privileged instruction
  - ➢Support several address space: logical address, linear address, physical address (memory virtualization)

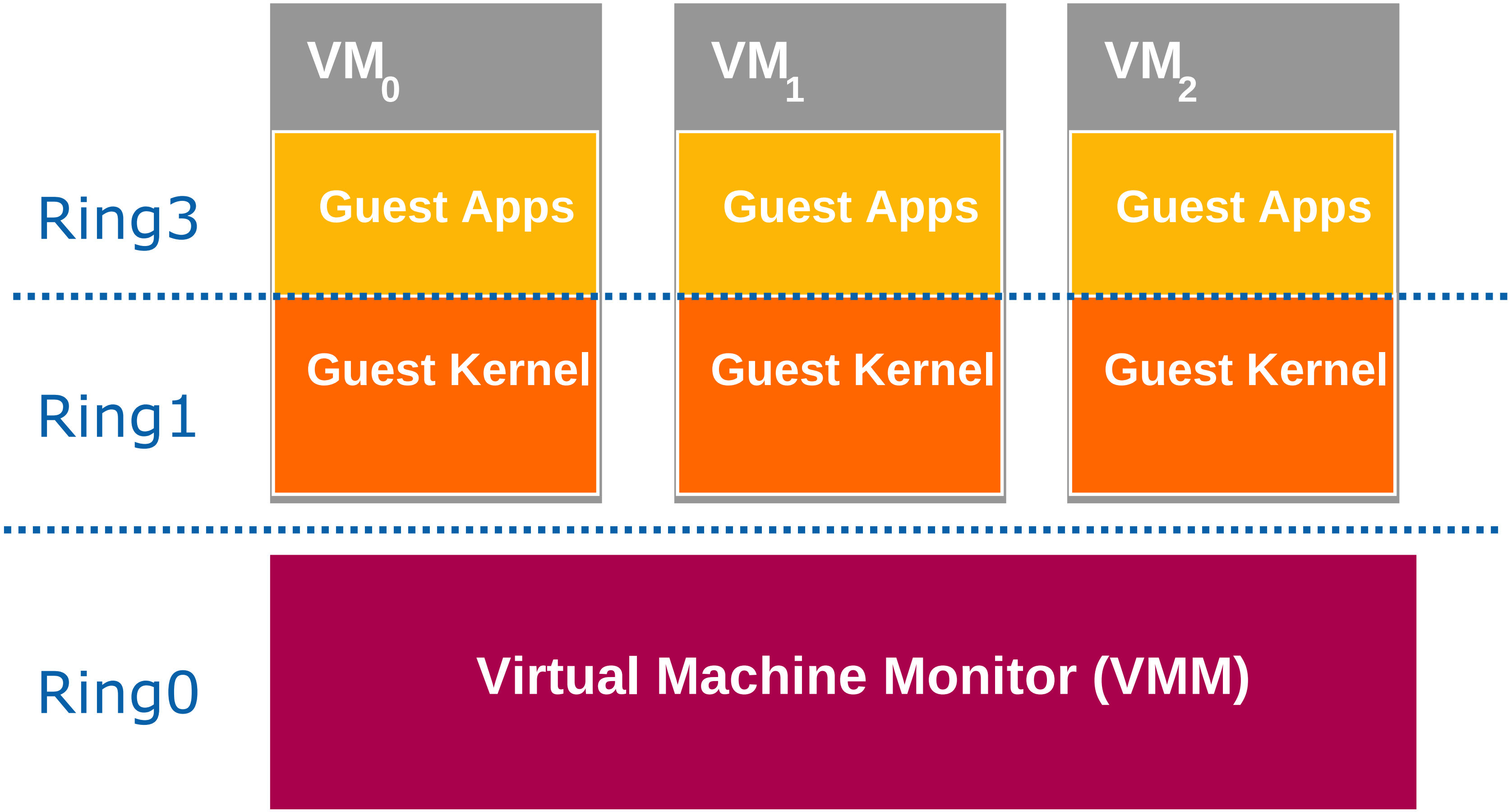
- **VCPU (virtual CPU)**
  - ➢A scheduling entity, containing all the state for virtualized CPU

- **Key to CPU virtualization: Trap and Emulation**
  - ➢Non-privileged instruction: untrap and run in native
  - ➢Privileged instruction: Trap and Emulation

CPU Virtualization (Cont)

- **Example:**
  - STI
  - CLI

Traditional IA32 CPU Virtualization: Ring Compression



IA32 Processor Virtualization Holes

- **Some instructions are hard to be virtualized**
- **e.g. pushf/popf**

```
pushf     //save EFLAG to stack
cli       //disable interrupt, i.e. clear EFLAG.IF
……
popf      //restore EFLAG from stack, restore EFLAG.IF
```

- 17 similar instructions

Addressing IA-32 “Virtualization Holes”

- **Method 1: Paravirtualization techniques**
  - Modify guest OS to work around virtualization holes
  - Typically limited to OSes that can be modified (e.g., Linux)

- **Method 2: Binary translation or patching**
  - Modify guest OS binaries “on-the-fly”
  - Extends range of supported OS’s but introduces new complexities
  - E.g., consider self-modifying code, translation caching, etc.
  - Certain forms of excessive trapping remain

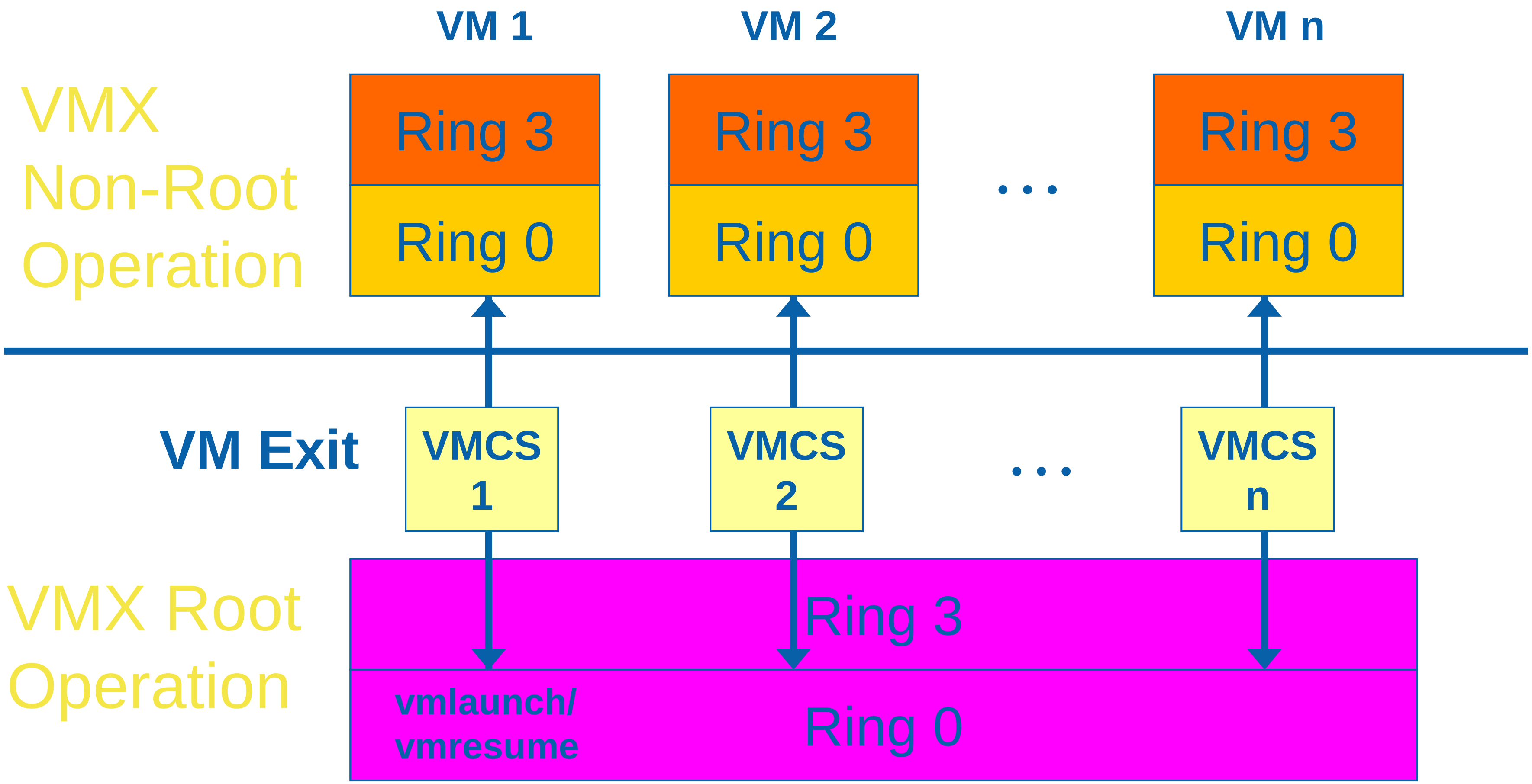
- **Goal for Hardware-assisted Virtualization**
  - Simplify VMM software by closing virtualization holes by design
  - Eliminate need for paravirtualization and binary translation

VT-x: Key Features

- **New mode of operation for guest**
  - Allows VMM control of guest operation
  - Need not use segmentation to control guest
  - Guest can run at its intended ring

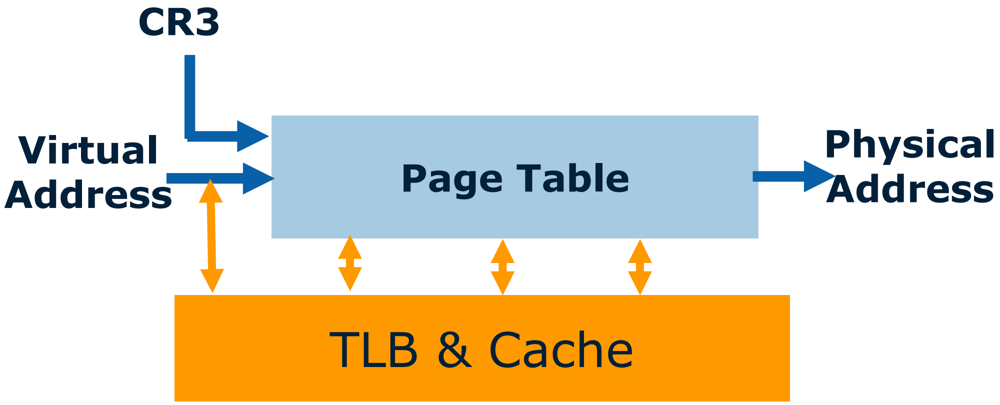
- **New structure controls CPU operation**
  - VMCS: virtual-machine control structure
  - Resides in physical-address space
  - Need not be in guest’s linear-address space

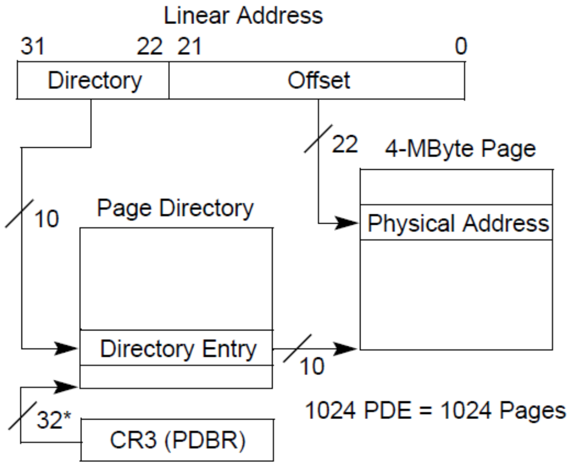
VT-x Operation



Memory Virtualization

- **Goal: Present Virtual Memory to Guest OS**
- **Memory from OS point of view**
  - ➢A set of memory unit (e.g. 2G memory)
  - ➢Support different address space: Virtual Address, Physical Address





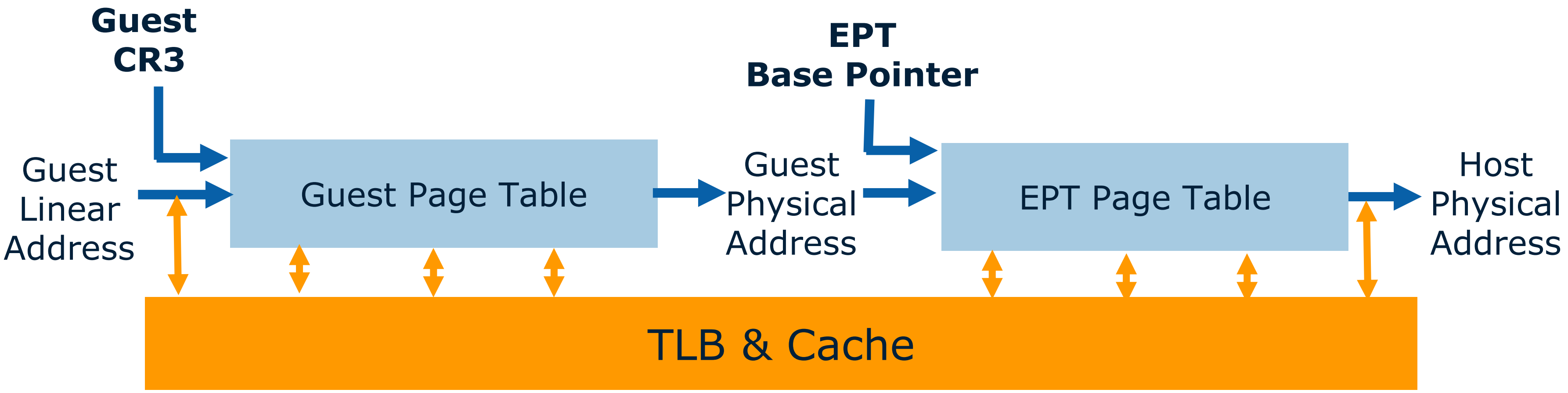
- **Memory Virtualization**
  - ➢Guest Virtual Address
  - ➢Guest Physical Address
  - ➢Machine Physical Address (Host Physical Address)

Extended Page Tables: Motivation

- **VMM needs to retain control of physical-address space**
  - ➢With Intel® 64, paging is main mechanism for protecting that space
  - ➢Intel® VT provides hooks for page-table virtualization…
  - ➢… but page-table virtualization in software is a major source of overhead

- **Extended Page Tables (EPT)**
  - ➢A new CPU mechanism for remapping guest-physical memory references
  - ➢Allows guest to retain control of legacy Intel® 64 paging
  - ➢Reduces frequency of VM exits to VMM

Extended Page Tables: Overview



- **Guest can have full control over page tables / events**
  - ➢CR3, CR0, CR4 paging bits, INVLPG, page fault
- **VMM controls Extended Page Tables**
- **CPU uses both tables**
- **EPT (optionally) activated on VM entry**
  - ➢When EPT active, EPT base pointer (loaded on VM entry from VMCS) points to extended page tables
  - ➢EPT deactivated on VM exit

Extended Page Tables: Performance

- **Estimated EPT benefit is very dependent on workload**
  - Typical benefit estimated up to 20%[1]
  - Outliers exist (e.g., forkwait, Cygwin gcc, > 40%)
  - Benefit increases with number of virtual CPUs (relative to MP page table virtualization algorithm)
- **Secondary benefits of EPT:**
  - No need for complex page table virtualization algorithm
  - Reduced memory footprint compared with shadow page-table algorithms
    - Shadow page tables required for each guest user process
    - Single EPT supports entire VM

EPT improves memory virtualization performance

[1]

VPID: Motivation

- First generation of Intel® VT forces flush of Translation Lookaside Buffer (TLB) on each VMX transition

- Performance loss on all VM exits

- Performance loss on most VM entries
  - ➢Most of the time, the VMM has not modified the guest page tables and does not require TLB flushing to occur
  - ➢Exceptions include emulating MOV CR3, MOV CR4, INVLPG
  - ➢Better VMM software control of TLB flushes is beneficial

VPID: New Support for Software Control of TLB

- **VPID activated if new "enable VPID" control bit is set in VMCS**

- **New 16-bit virtual-processor-ID field (VPID) field in VMCS**
  - ➢VMM allocates unique value for each guest OS
  - ➢VMM uses VPID of 0x0000, no guest can have this VPID

- **Cached linear translations are tagged with VPID value**

- **No flush of TLBs on VM entry or VM exit if VPID active**

TLB Management by the VMM: INVVPID

- **New instruction to allow VMM to flush guest mappings**

- **Three operands:**
  - The flush extent (see below)
  - The 16-bit VPID indicating the VPID context to be flushed
  - The 64-bit guest-linear address to be flushed

- **Flush extent operand chooses:**
  - Address-specific: invalidation of translations associated with VPID and address operands
  - Context-wide: invalidation of all translations associated with VPID operand
  - Context-wide preserving global translations: invalidation of all non-global translations associated with VPID operand
  - All-context: invalidation of all translations associated with all VPID values

- **Allows VMM to emulate Intel® 64 paging faithfully**

VPID Performance

- **VPID benefit is very dependent on workload and memory virtualization mechanism**

- **Without EPT:**
  - Most stressful of CPU-intensive workloads (e.g., gzip) show only small improvements with VPID
  - Process and memory-intensive workloads gain an estimated 1.5% - 2%[1]
  - Worst-case synthetic benchmarks gain an estimated 3%-4%[1]

- **With EPT:**
  - VM-exit frequency decreases but the cost of TLB fills increases
  - VPIDs required to make EPT effective under stressful loads

  - For process/memory-intensive workloads gain an estimated >2%[1]
  - Worst-case synthetic benchmarks gain an estimated 10%-15%[1]

VPID improves TLB performance

with small VMM development effort

[1]

I/O Virtualization

- **Present virtual I/O device to Guest OS**
- **I/O device from OS point of view**
  - A set of resource: I/O port, MMIO, Interrupt
  - Can execute I/O command with predefined semantic
- **Key to I/O Virtualization**
  - Base on CPU virtualization

Software Approaches to I/O Virtualization

- **Device Emulation**
  - ➢Virtualization software emulates real hardware device
  - ➢VMs run same driver for the emulated hardware device
  - ➢Good legacy software compatibility
  - ➢However emulation overheads can limit performance

- **I/O Para-virtualization**
  - ➢Uses abstract interfaces and protocols for I/O services
  - ➢VMs run virtualization-aware I/O stacks and drivers
  - ➢Offers improved performance over emulation
  - ➢Requires new I/O stack/driver in guest OSs

Software approaches offer I/O sharing with H/W transparency, at a performance cost

I/O Virtualization: Direct Assigned I/O

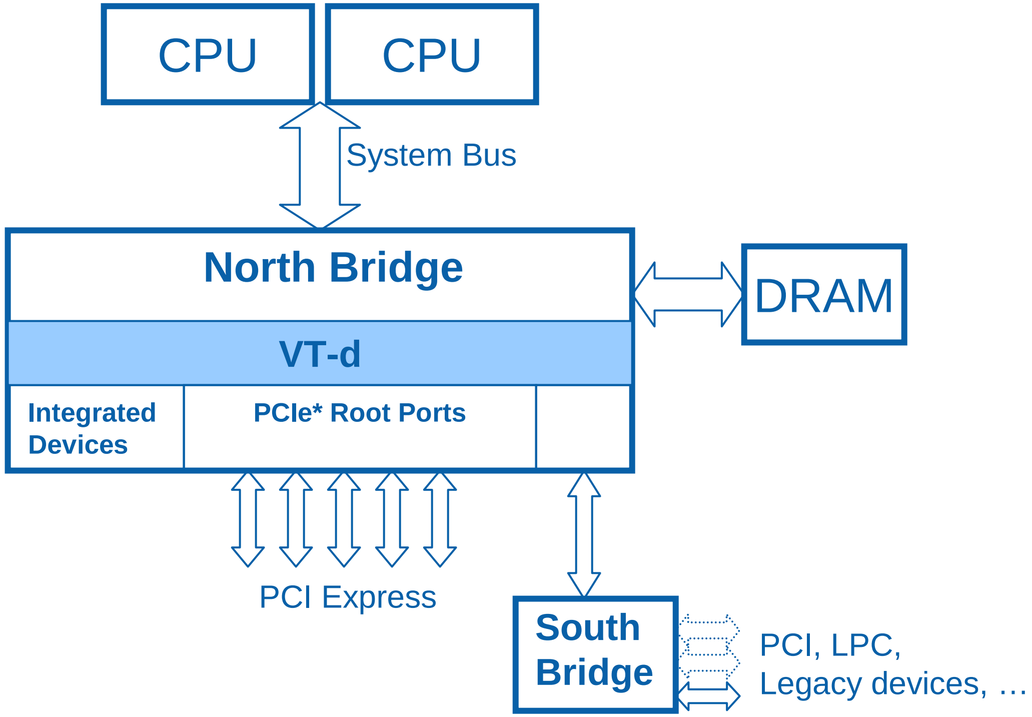
- **Directly assigned I/O device to Guest**
  - Guest OS access I/O device resource directly
  - High performance and low CPU utilization

- **Problem: DMA address mismatch**
  - Guest set guest physical address
  - DMA hardware only accept machine physical address

- **Solution: DMA Remapping (a.k.a IOMMU)**
  - I/O page table is introduced
  - DMA engineer will translate guest physical address to host physical address according to I/O page table

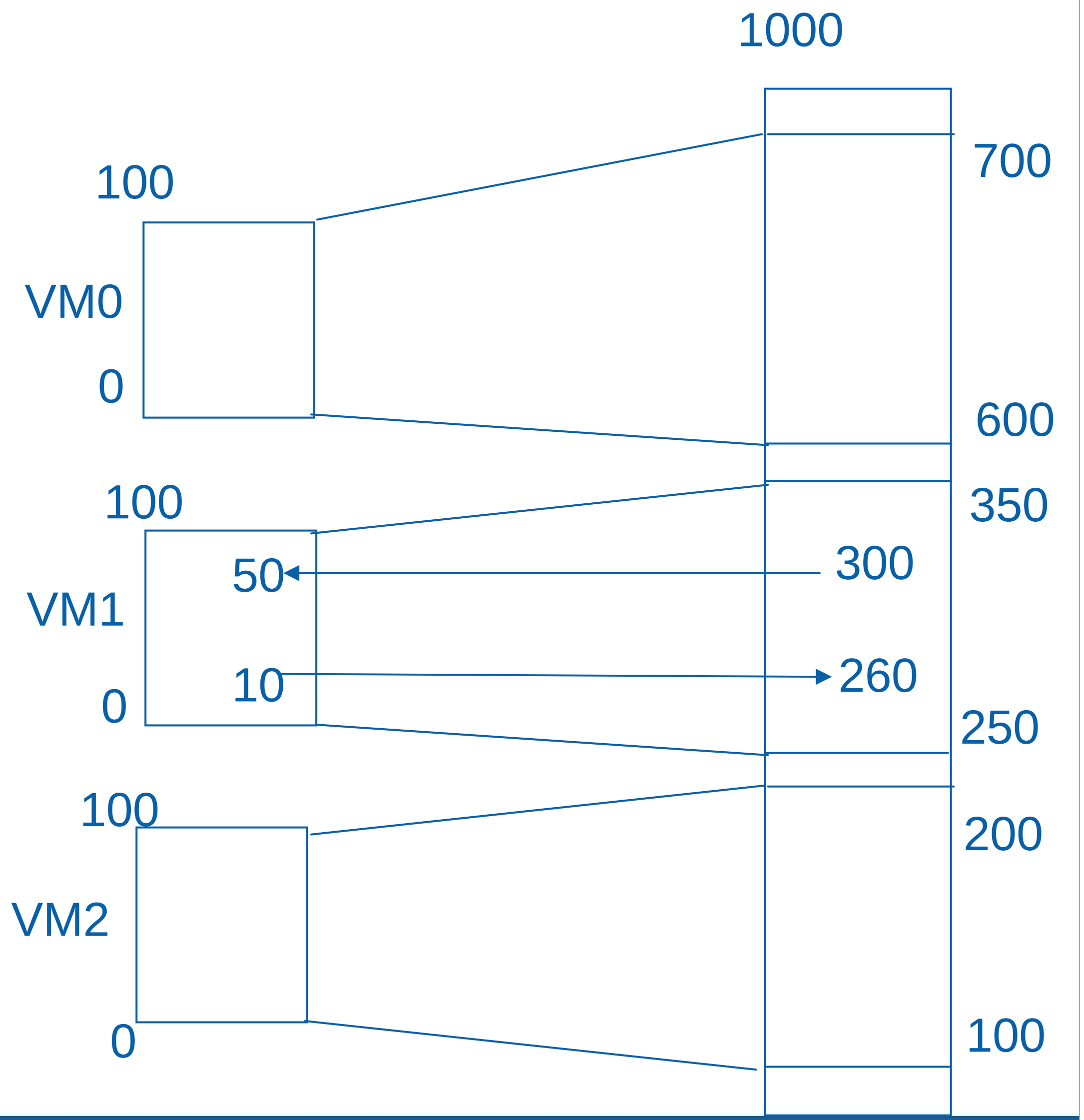
VT-d Overview

- **VT-d provides infrastructure for I/O virtualization**
  - ➢Defines architecture for DMA remapping
  - ➢Common architecture across IA platforms
  - ➢Will be supported broadly across Intel® chipsets

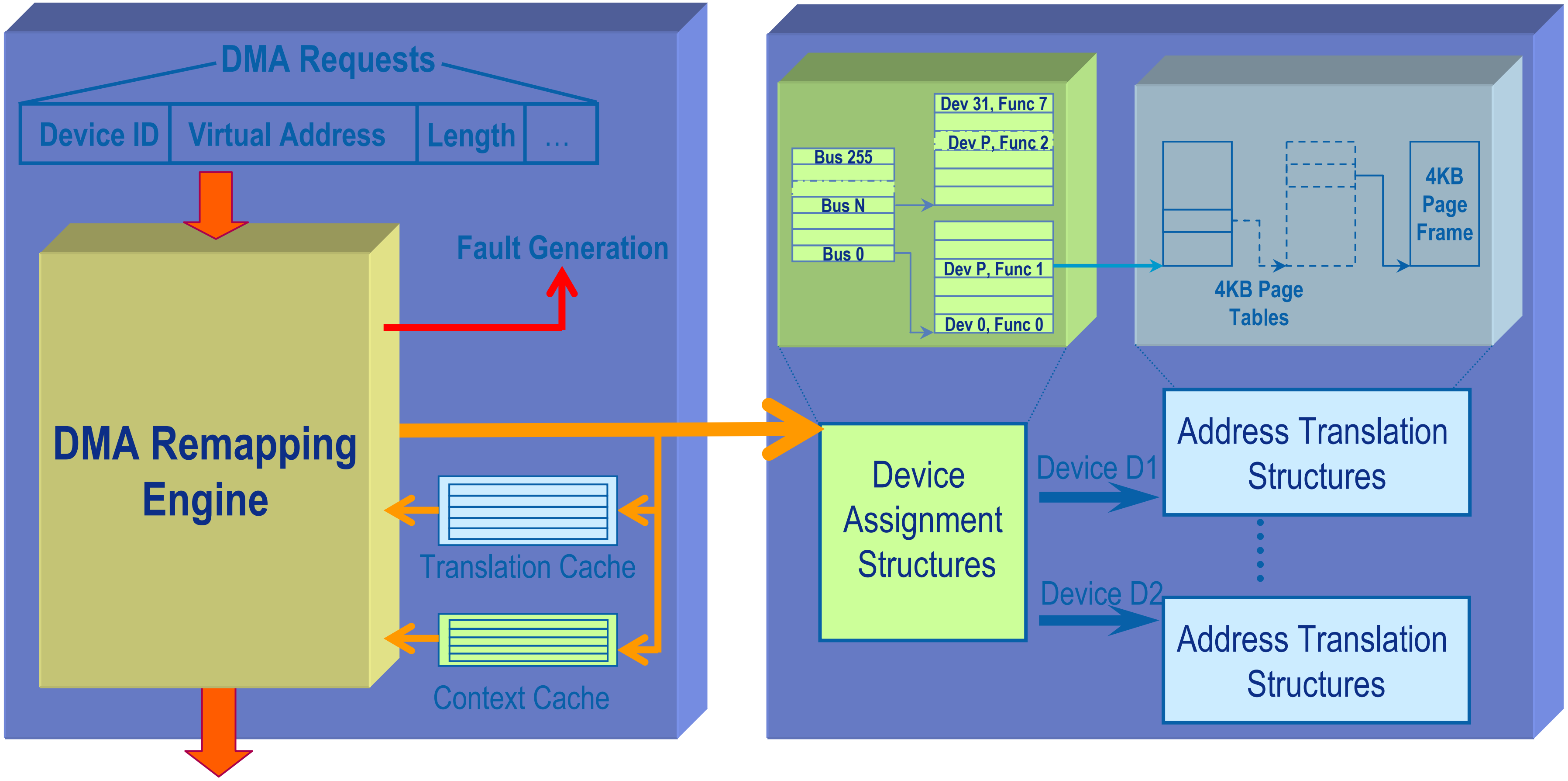


How VT-d works?

- **Each VM thinks it is 0 address based**
  - GPA (Guest Physical Address)
- **But mapped to a different address in the system memory**
  - HPA (Host Physical Address)
- **VT-d does the address mapping between GPA and HPA**
- **Catches any DMA attempt to cross VM memory boundary**



DMA Remapping: Hardware Overview



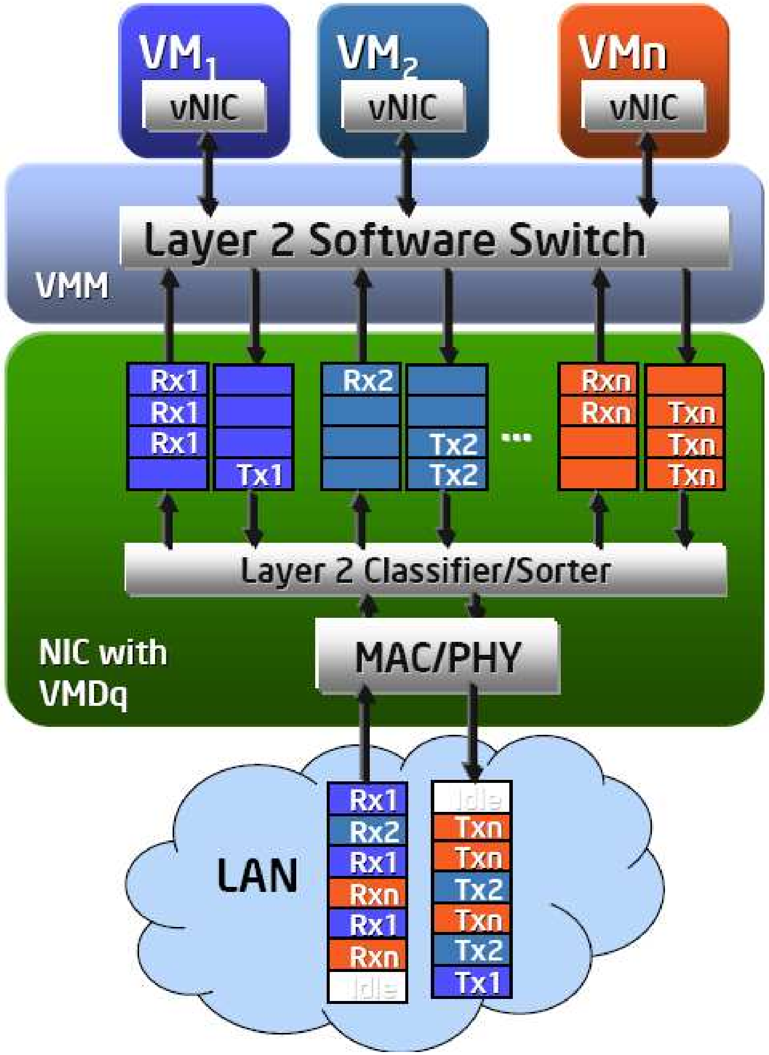
Memory Access with Host Physical Address

Memory-resident Partitioning & Translation Structures

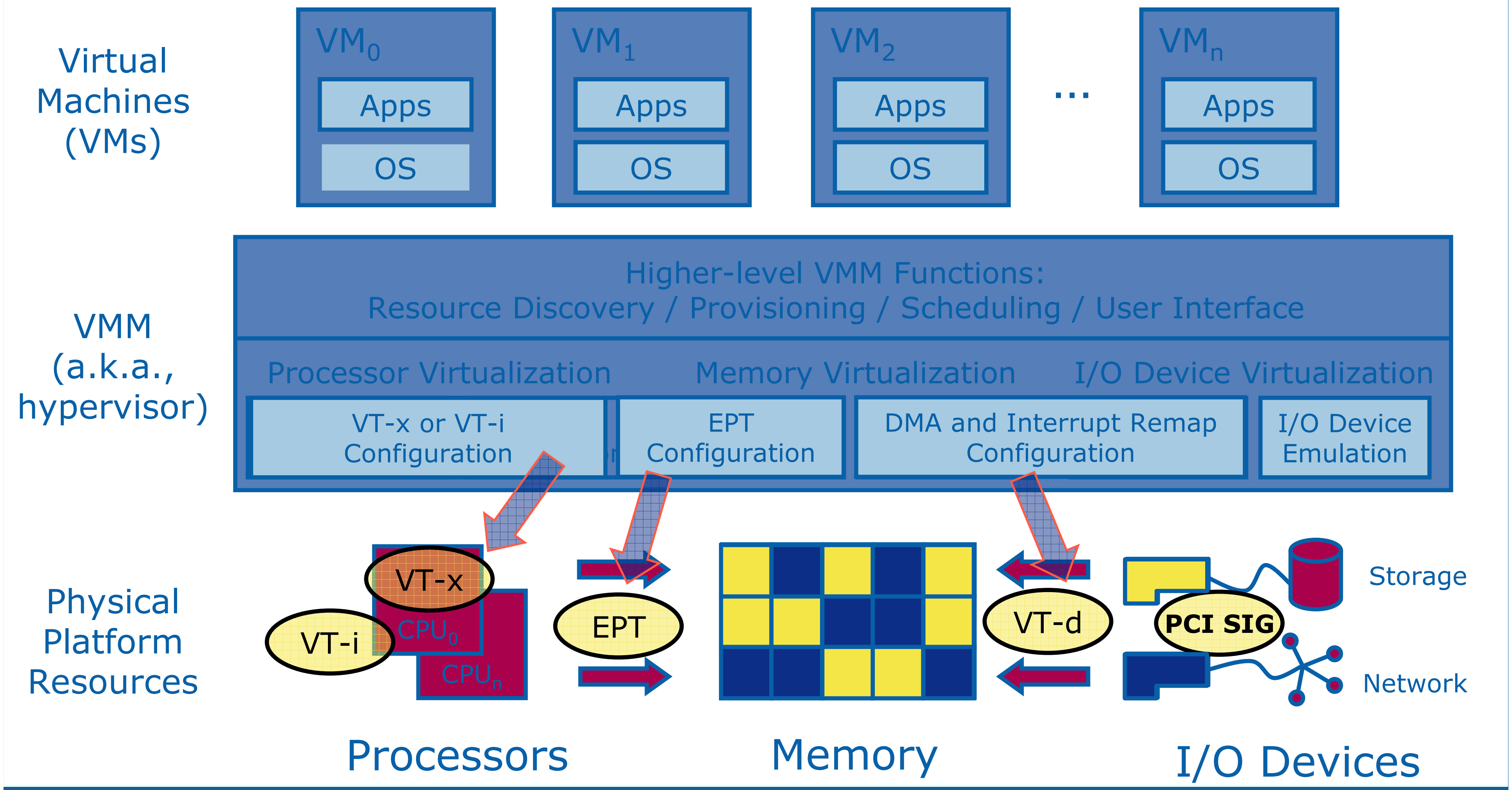
Intel VT For Connectivity: VMDq

- **Network Interface Card with Virtual Machine Device Queues (VMDq)**
  - ➢Multiple Send/Receive Queues
  - ➢Pre-sort Incoming Packets

- **Benefits**
  - ➢Reduce CPU Utilization
  - ➢Higher throughput



Putting it all together…



Virtual Machines (VMs)

VM_0 Apps OS

VM_1 Apps OS

VM_2 Apps OS

…

VM_n Apps OS

VMM (a.k.a., hypervisor)

Higher-level VMM Functions:
Resource Discovery / Provisioning / Scheduling / User Interface

Processor Virtualization — VT-x or VT-i Configuration

Memory Virtualization — EPT Configuration

I/O Device Virtualization — DMA and Interrupt Remap Configuration; I/O Device Emulation

Physical Platform Resources

Processors

Memory

I/O Devices

Virtualization Technology Forecast

- **More usage model**
  - ➢Client Virtualization (Desktop)
  - ➢Mobile Virtualization (Cell phone)
  - ➢Cloud Computing
- **More Virtualization Software**
  - ➢VMWare
  - ➢MS Hyper-V
  - ➢Xen, KVM
- **Hardware**
  - ➢CPU/Memory Virtualization: higher performance
  - ➢I/O Virtualization: SR-IOV, Graphics Virtualization (3D)

Summary

- Exponential grows in virtualization solutions.
- Hardware-assisted Virtualization Technology can make VMM more efficient, simplified & secure.
- A lot of un-cultivated areas in virtualization, a lot of opportunity.

Resource

- **Intel® VT Web Site:**
  http://www.intel.com/technology/virtualization/

- **Open Source Xen**
  http://www.xen.org/

- **Open Source KVM (Kernel-based Virtual Machine)**
  http://kvm.qumranet.com/

Q/A