Introduction of AMD Advanced Virtual Interrupt Controller

XenSummit 2012

Wei Huang
August 2012

What is AVIC?

- AVIC is ***Advanced Virtual Interrupt Controller***
- A virtual APIC to guest OSs with hardware acceleration
- Enhancement to the AMD SVM architecture
- Changes to CPU, NorthBridge, and IOMMU

Agenda

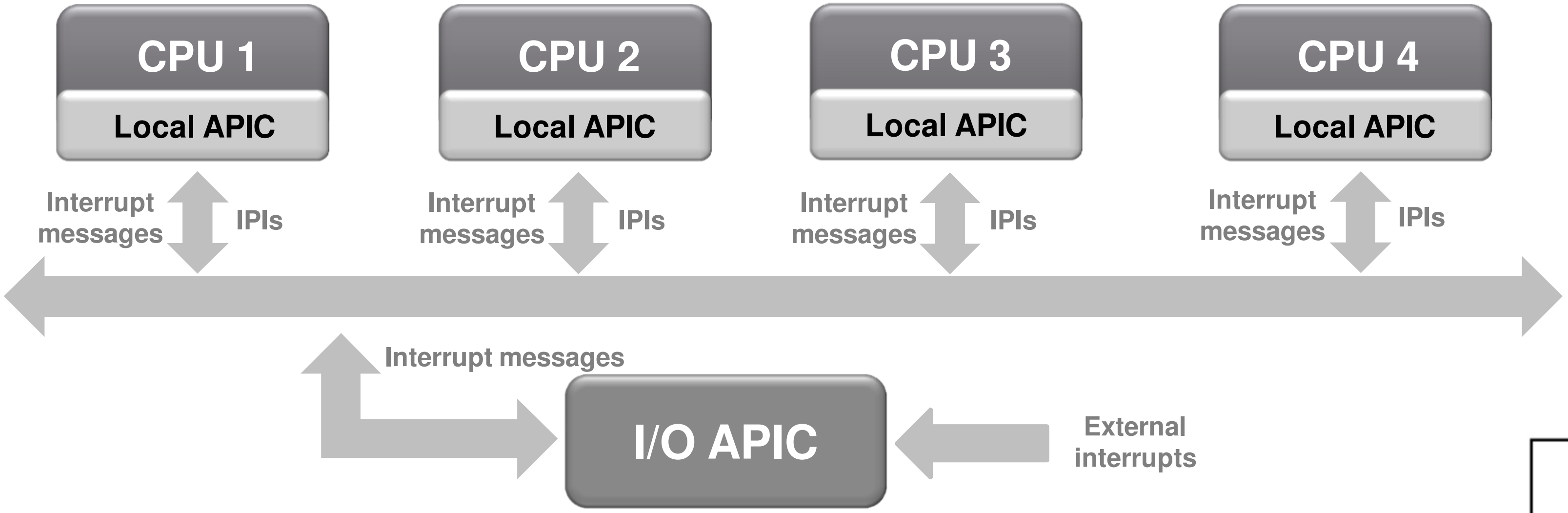
- Motivation for AVIC
- AVIC Architecture
  - APIC backing page
  - Physical & logical APIC ID tables
  - Door bell & IOMMU extension
- Hypervisor Design for AVIC
- Summary

Agenda

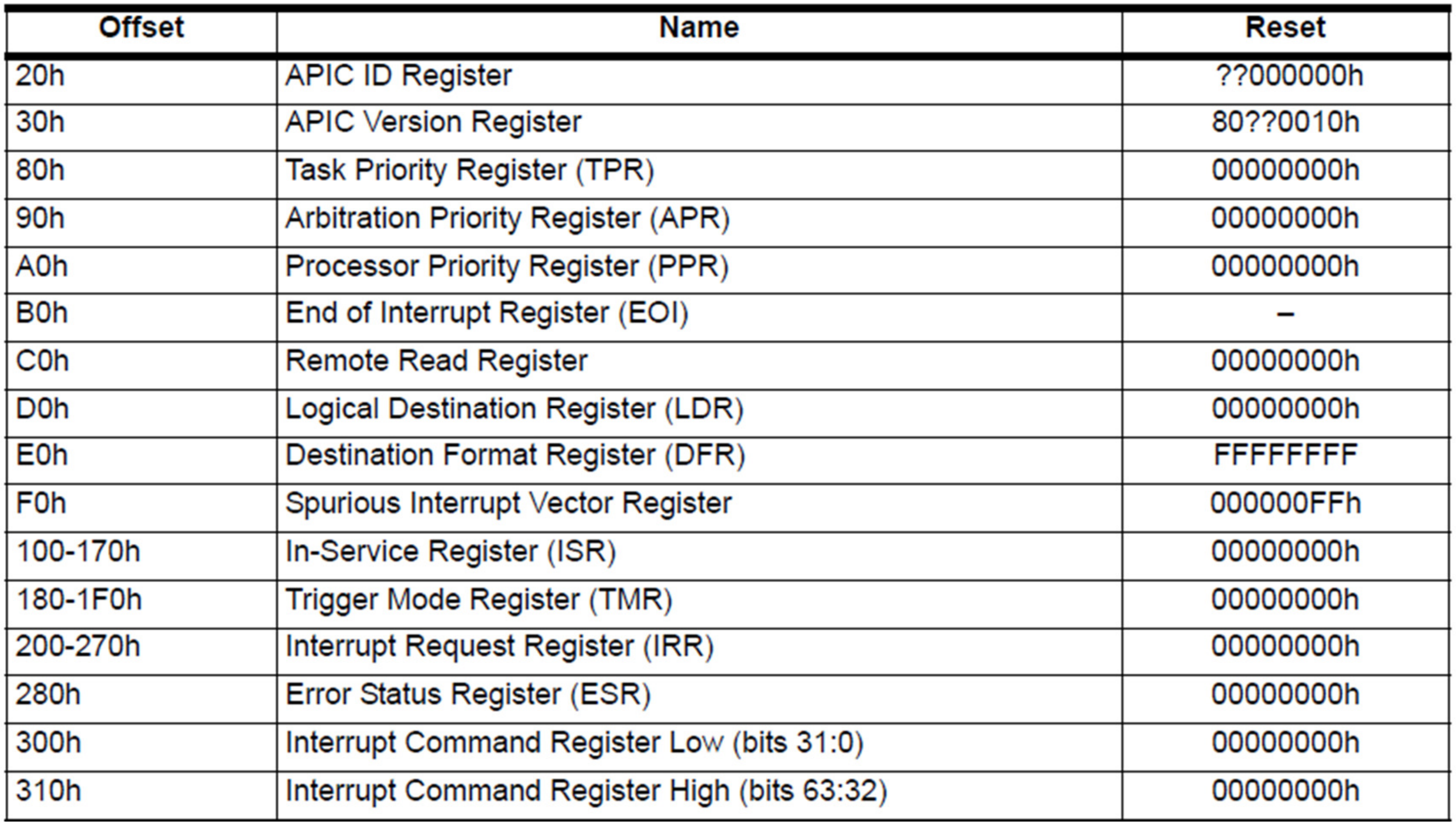
- Motivation for AVIC
- AVIC Architecture
  - APIC backing page
  - Physical & logical APIC ID tables
  - Door bell & IOMMU extension
- Hypervisor Design for AVIC
- Summary

Local APIC in x86

- Local APIC (LAPIC) is vital for x86 SMP system
- Handles various interrupt sources:
  - Inter-processor interrupts (IPIs)
  - I/O devices
  - APIC timer-generated interrupts
  - Internal events



Many LAPIC Registers…

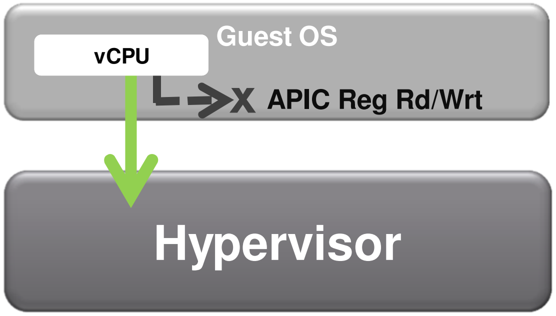


Offset	Name	Reset
20h	APIC ID Register	??000000h
30h	APIC Version Register	80??0010h
80h	Task Priority Register (TPR)	00000000h
90h	Arbitration Priority Register (APR)	00000000h
A0h	Processor Priority Register (PPR)	00000000h
B0h	End of Interrupt Register (EOI)	–
C0h	Remote Read Register	00000000h
D0h	Logical Destination Register (LDR)	00000000h
E0h	Destination Format Register (DFR)	FFFFFFFF
F0h	Spurious Interrupt Vector Register	000000FFh
100-170h	In-Service Register (ISR)	00000000h
180-1F0h	Trigger Mode Register (TMR)	00000000h
200-270h	Interrupt Request Register (IRR)	00000000h
280h	Error Status Register (ESR)	00000000h
300h	Interrupt Command Register Low (bits 31:0)	00000000h
310h	Interrupt Command Register High (bits 63:32)	00000000h

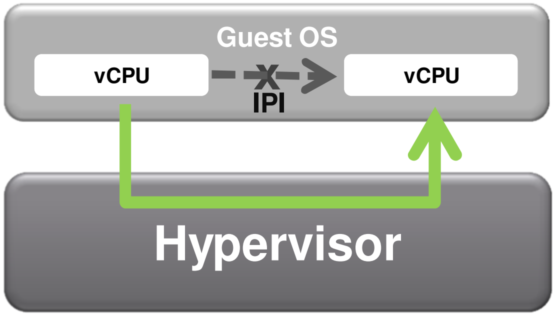
*AMD APM Vol 2

LAPIC + Virtualization = Slow

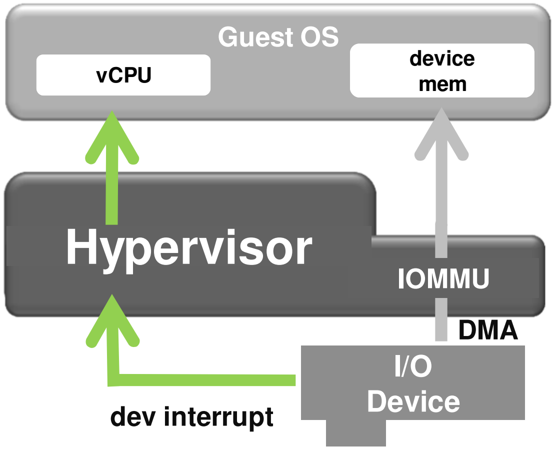
1. APIC register reads/writes



2. Inter-processor Interrupts



3. I/O interrupts from peripherals (e.g., pass-through device)



Attacking Problems with AVIC

- We define new architectural components to present a virtualized APIC to guests, thus allowing most APIC accesses and interrupt delivery into the guests directly.
- AVIC components:

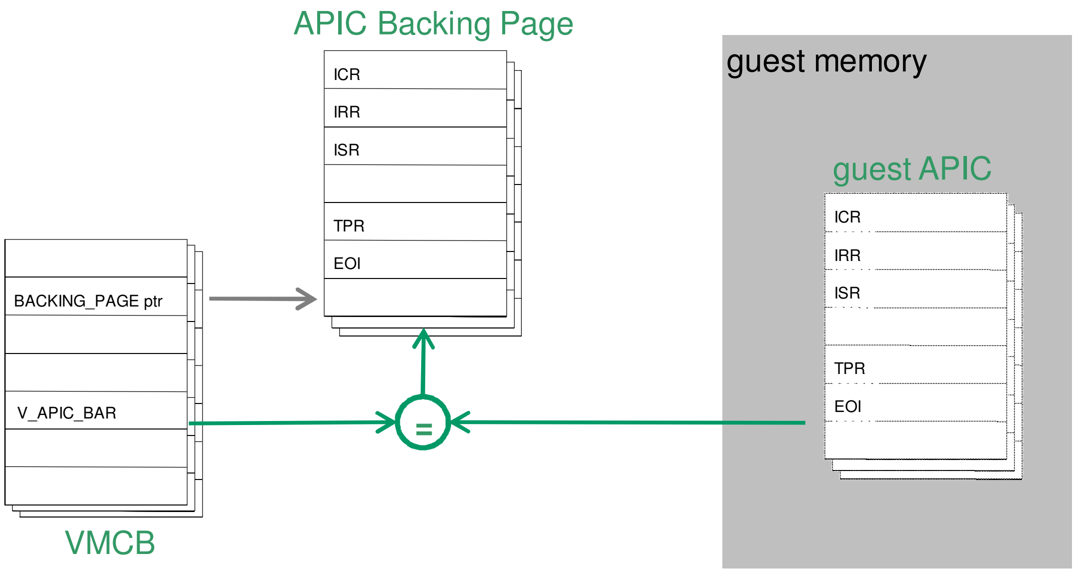
Component	Problem Solved
vAPIC	Direct access to APIC registers
Physical & **logical APIC ID tables**	Mapping of physical & virtual cores (scheduling)
Doorbell interrupt	Interrupt delivery
IOMMU extension	I/O interrupt delivery

Agenda

- Motivation for AVIC
- AVIC Architecture
  - APIC backing page
  - Physical & logical APIC ID tables
  - Doorbell & IOMMU extension
- Hypervisor Design for AVIC
- Summary

vAPIC

- vAPIC is a virtual APIC to the guest
  - Backed by a 4KB memory page
  - Allocated and initialized by hypervisor
  - vAPIC pages are allocated on a per-vCPU basis
  - Contains storage for guest APIC values for the vCPU
  - Mapped into the guest physical address space

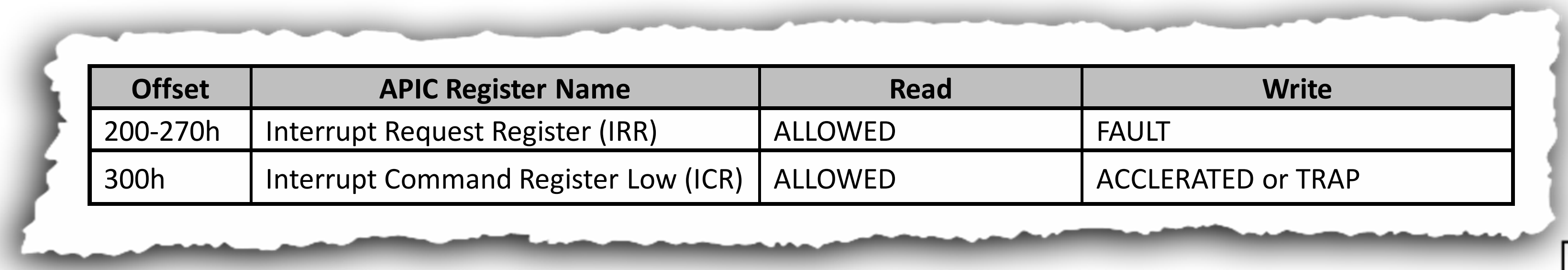


Handling Guest APIC Accesses

- Four types of actions for guest APIC accesses:

Action	VMEXIT?	Details
ALLOWED	NO	The field will be updated (for Write) or read (for Read).
ACCLERATED	NO	The field will be updated and CPU will take further actions, such as sending IPI to CPUs.
TRAP	YES	The field will be updated and CPU will take VMEXIT immediately **after** access.
FAULT	YES	The field will **NOT** be updated and CPU will take VMEXIT immediately **before** the access.

- Example



Offset	APIC Register Name	Read	Write
200-270h	Interrupt Request Register (IRR)	ALLOWED	FAULT
300h	Interrupt Command Register Low (ICR)	ALLOWED	ACCLERATED or TRAP

Regarding APIC Acceleration…

- Three performance-critical areas are accelerated
  - TPR reads and writes
  - EOI writes
  - ICRL writes

Agenda

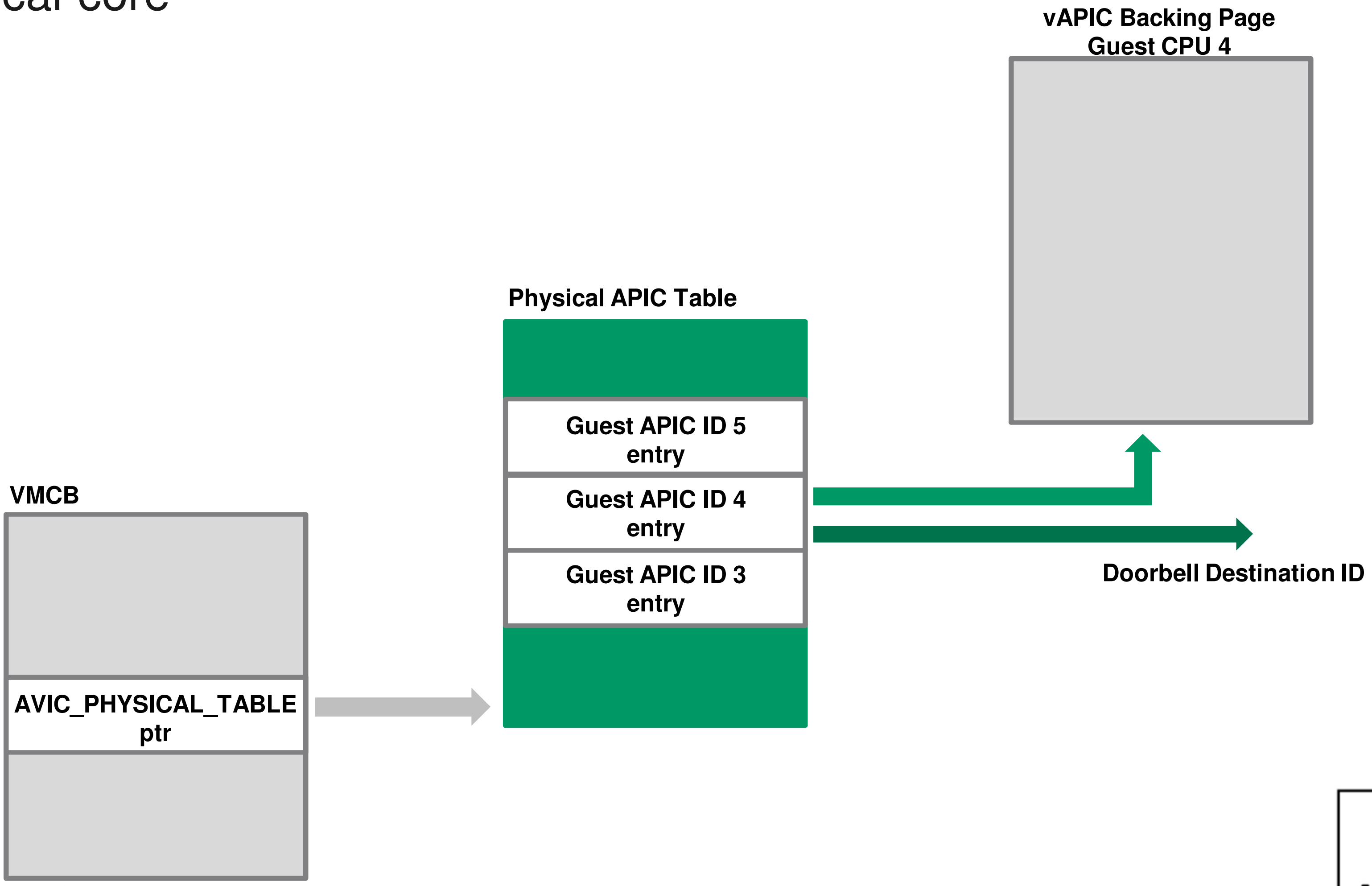
- Motivation for AVIC
- AVIC Architecture
  - APIC backing page
  - Physical & logical APIC ID tables
  - Door bell & IOMMU extension
- Hypervisor Design for AVIC
- Summary

How About vCPU Scheduling?

- Hypervisor schedules guest VM’s VCPUs on-the-fly
  - VCPUs can be ON or OFF
  - VCPUs can be migrated
- AVIC uses ***physical*** and ***logical APIC ID tables*** to show VCPU’s mapping relationship with physical cores

Physical APIC ID Table

- Purpose:
  Used by the hardware to route the virtual interrupt messages to the proper physical core



vAPIC Backing Page
Guest CPU 4

Physical APIC Table

Guest APIC ID 5 entry

Guest APIC ID 4 entry

Guest APIC ID 3 entry

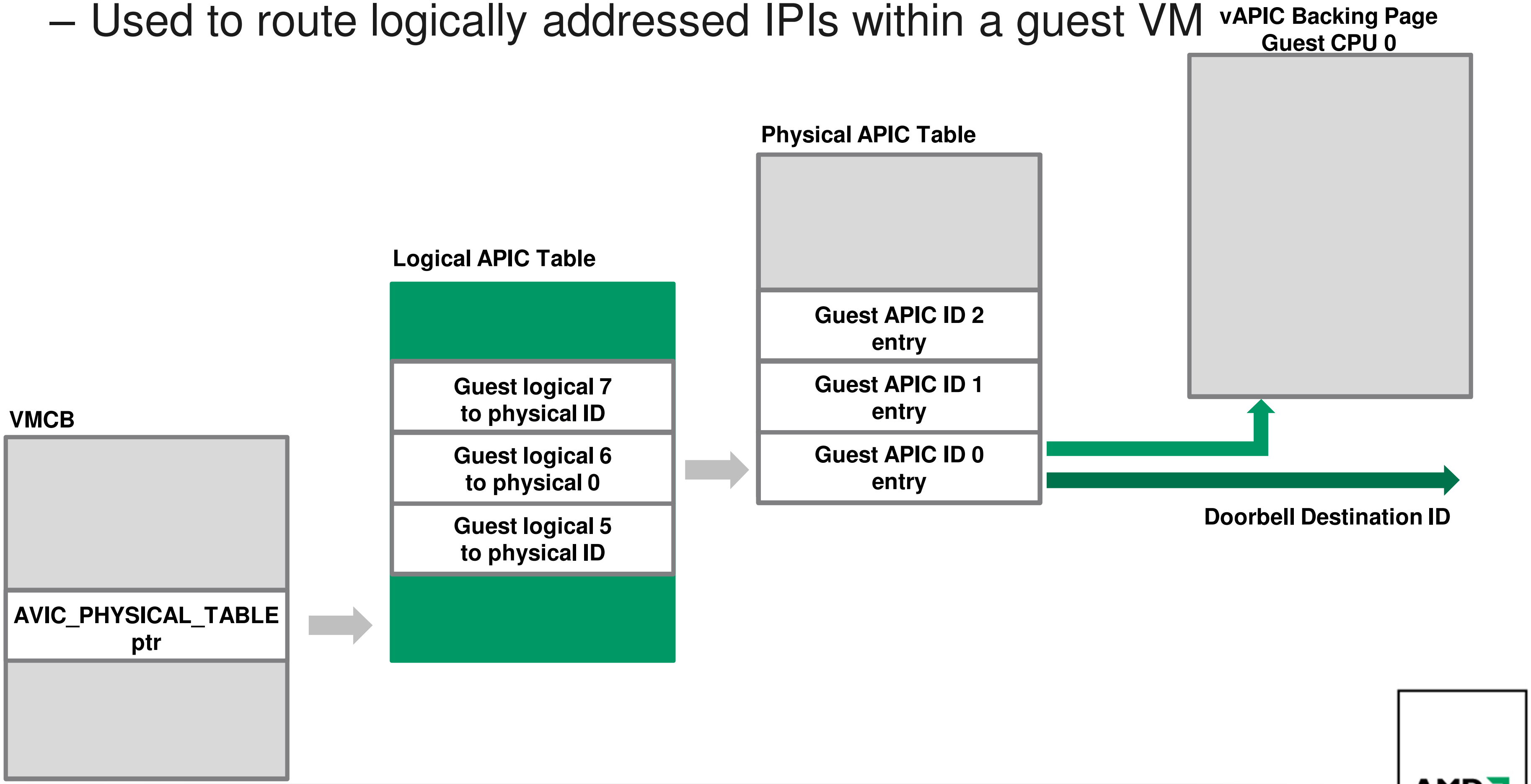
Doorbell Destination ID

VMCB

AVIC_PHYSICAL_TABLE ptr

Logical APIC ID Table

- Purpose:
  - Maps guest logical APIC IDs to guest physical APIC IDs
  - Used to route logically addressed IPIs within a guest VM



Agenda

- Motivation for AVIC
- AVIC Architecture
  - APIC backing page
  - Physical & logical APIC ID tables
  - Doorbell & IOMMU extension
- Hypervisor Design for AVIC
- Summary

Doorbell

- A new interrupt mechanism that is used to deliver guest interrupts to a specific physical core
- Doorbell interrupts are sent in three ways:

Doorbell Source	Reason
AVIC hardware	In response to a guest ICRL write for a supported IPI type
IOMMU	An incoming I/O interrupt is remapped to an AVIC mode guest
System software	Writing to the doorbell interrupt MSR

IOMMU Extension

- The AVIC architecture leverages the existing IOMMU interrupt redirection mechanism to provide a new guest-delivered interrupt type
- A new field in the IOMMU device table specifies whether AVIC is available
- IOMMU uses doorbell mechanism to delivery interrupts into guest VMs

Agenda

- Motivation for AVIC
- AVIC Architecture
  - APIC backing page
  - Physical & logical APIC ID tables
  - Door bell & IOMMU extension
- Hypervisor Design for AVIC
- Summary

How to Design a Hypervisor for AVIC?

- At guest start-up:
  - Allocate and initialize one vAPIC page for each vCPU
  - Allocate one physical APIC table and logical APIC table for the guest
  - Set up VMCB pointers for these structures and enable AVIC mode in VMCB
  - Set up IOMMU tables for direct-assigned I/O devices
- While running:
  - Handle un-accelerated cases similar to current approach
  - Update physical APIC table and IOMMU IRTE entries to change running status whenever a vCPU is moved to or from a physical CPU

Setting up AVIC Tables

- vAPIC backing pages and physical/logical ID tables are allocated in host physical memory
- Related VMCB fields:

VMCB Field	Details
V_APIC_BAR	Guest physical base address of the virtual APIC.
AVIC_BACKING_PAGE	Host physical address of the APIC backing page for the associated VM.
AVIC_LOGICAL_PAGE	Host physical address of the logical APIC look-up table for the associated VM.
AVIC_PHYSICAL_PAGE	Host physical address for the physical APIC look-up table for the associated VM.

Handle New VMEXITs

- VMEXIT 401h
  - For incomplete IPI Delivery
  - This VMEXIT describes the specific reason for the IPI delivery failure
- VMEXIT 402h
  - For ccess to un-accelerated vAPIC field
  - This VMEXIT indicates the offset of the un-accelerated vAPIC register, as well as whether a read or write operation was attempted

Summary

- AVIC is an hardware extension to AMD SVM for APIC acceleration
- AVIC targets critical APIC operations for optimal performance
  - APIC read/write accesses
  - Interrupt delivery
- The design of AVIC is simple for fast hypervisor integration
- We expect AVIC to eliminate lots of overhead introduced by virtual local APIC

Questions?

Trademark Attribution