Sprzętowe wsparcie dla wirtualizacji

w architekturach IA-32 i Intel 64

Julian Pszczołowski
Instytut Informatyki UWr

“All problems in computer science can be solved by another level of indirection” — David Wheeler

“All problems in computer science can be solved by another level of indirection” — David Wheeler

“... except for the problem of too many layers of indirection.” — Kevlin Henney

Dlaczego lubimy wirtualizację?

- sandboxing
- mimo że awaria nadzorcy, który dogląda wielu maszyn wirtualnych, spowoduje ogromne szkody, przy wsparciu sprzętowym kod hipervisora ma rzędy wielkości mniej kodu (i błędów)
- oszczędność kosztów (np. system Borg w Google)
- migawki (snapshot) - przyrostowy backup
- łatwa migracja na inny sprzęt
- testowanie oprogramowania na różnych systemach
- modne słowa
  - cloud, Docker, Kubernetes, serverless, skalowalność
  - firmy wykorzystujące AWS: Netflix, Airbnb, Slack, Atlassian, NASA, Spotify, i wiele innych

Dlaczego lubimy sprzętowe wsparcie w wirtualizacji?

- procesor jest świadomy wirtualizacji - udostępnia nowe mechanizmy, nowe instrukcje
  - większa prostota kodu, np. nadzorcy
- tłumaczenie kodu w locie nie jest już potrzebne
  - zmieniono semantykę niektórych operacji w user space wirtualnej maszyny, część akcji generuje (zależnie od naszej konfiguracji) przejście do nadzorcy - mniejsze problemy z wychwytywaniem instrukcji wrażliwych, które nie są uprzywilejowane
- szybsze przechodzenie tablicy stron mapującej adresy fizyczne gościa na prawdziwe adresy fizyczne
- izolacja adresów przy używaniu DMA zwiększa bezpieczeństwo (maszyna wirtualna nadpisująca dane innych maszyn)
- nie lubimy emulować software'owo kontrolera przerwań czy trasowania przerwań do wirtualnych maszyn
- mniejszy narzut, większa wydajność

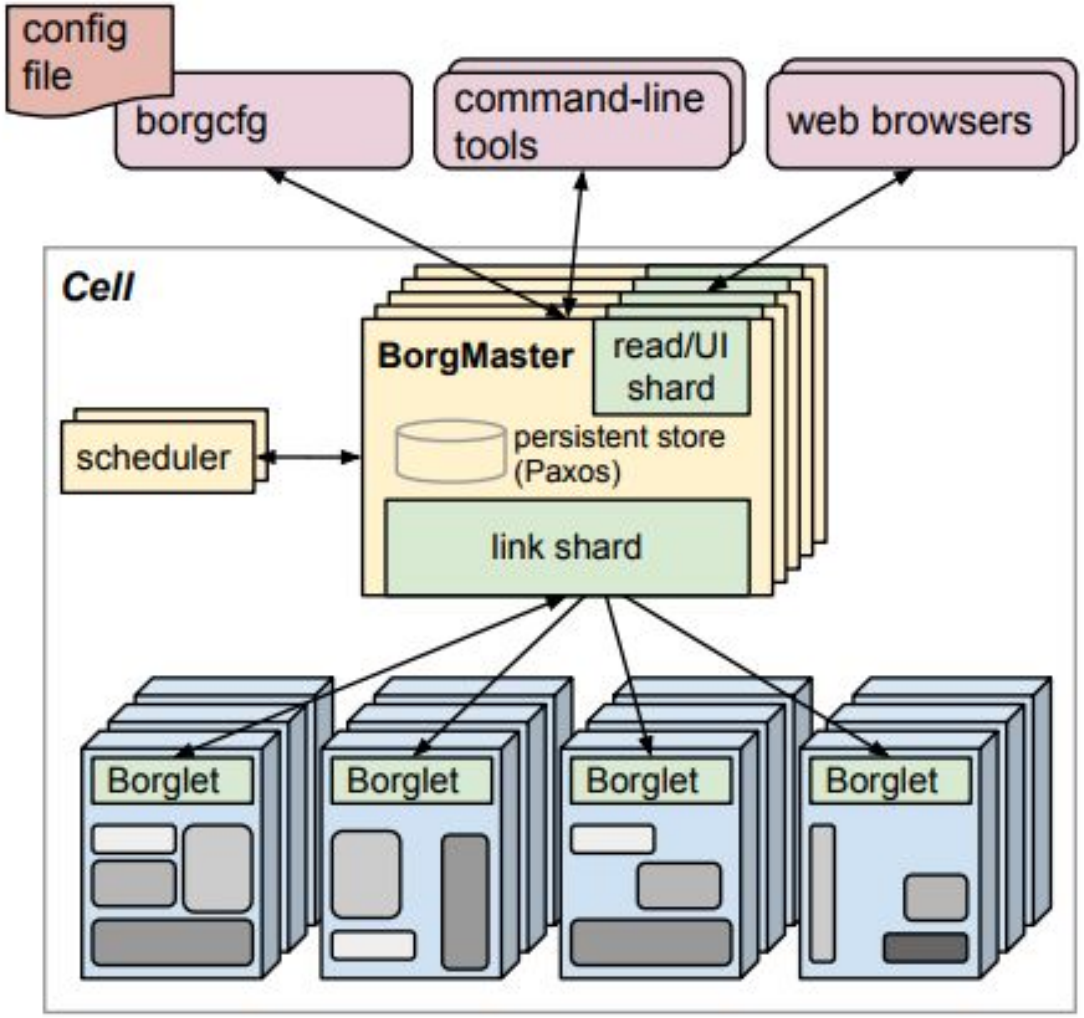


Figure 1: The high-level architecture of Borg. *Only a tiny fraction of the thousands of worker nodes are shown.*

źródło: https://research.google.com/pubs/pub43438.html

```
$ cat /proc/cpuinfo

...

vendor_id : GenuineIntel

model name : Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz

flags : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge
mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht
tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc art
arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc
cpuid aperfmperf tsc_known_freq pni pclmulqdq dtes64 monitor
ds_cpl vmx est tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid sse4_1
sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx
f16c rdrand lahf_lm abm 3dnowprefetch cpuid_fault epb
invpcid_single pti intel_pt tpr_shadow vnmi flexpriority ept
vpid fsgsbase tsc_adjust bmi1 avx2 smep bmi2 erms invpcid
mpx rdseed adx smap clflushopt xsaveopt xsavec xgetbv1
xsaves dtherm ida arat pln pts hwp hwp_notify hwp_act_window
hwp_epp
```

Instrukcja CPUID

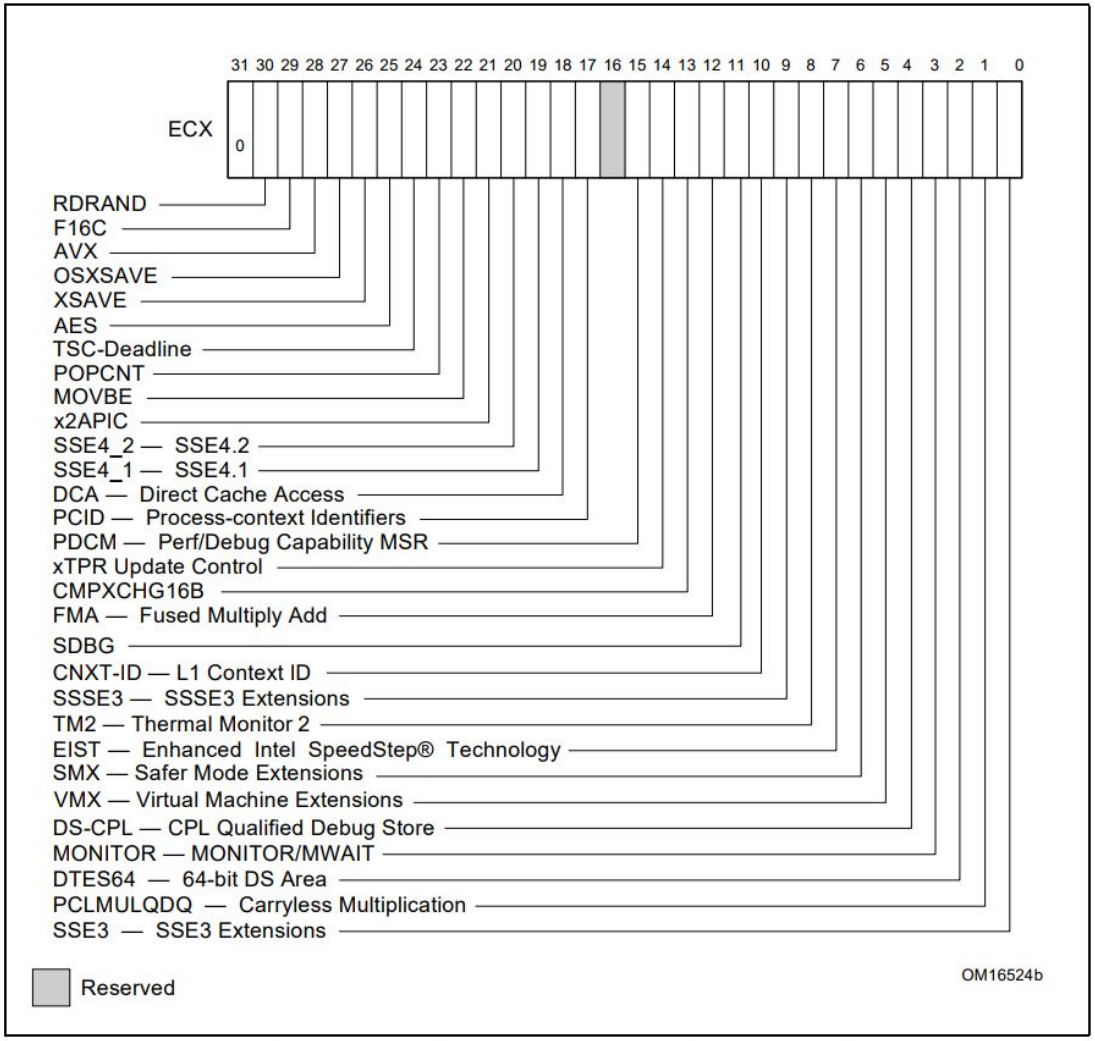


Figure 3-7. Feature Information Returned in the ECX Register

Architektura maszyny wirtualnej (w IA-32 i Intel 64)

- monitor maszyny wirtualnej (VMM) posiada pełną kontrolę nad fizycznym procesorem i zasobami, prezentuje oprogramowaniu gościa abstrakcję wirtualnego procesora (vCPU) oraz zarządza pamięcią, przerwaniami, wejściem i wyjściem
- oprogramowanie gościa składa się systemu operacyjnego i aplikacji
  - operuje na sprzęcie używając takiego samego interfejsu co zwykły system bez wirtualizacji
  - transparentne dla innych wirtualnych maszyn

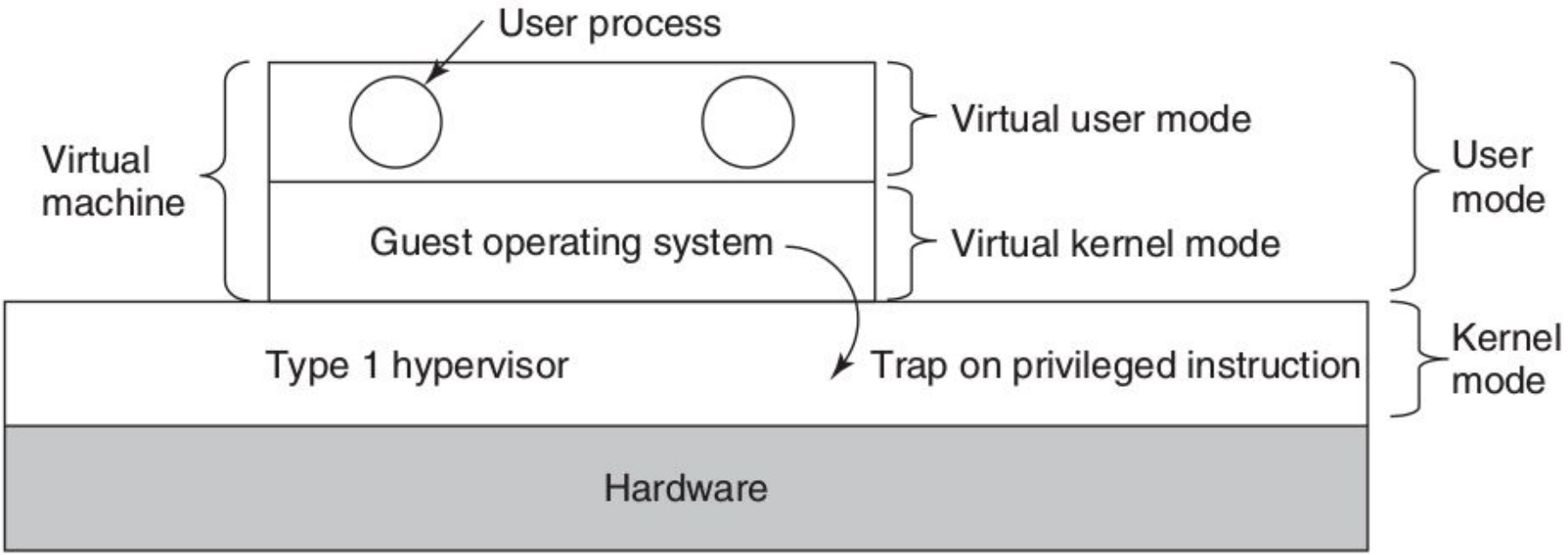


Figure 7-3. When the operating system in a virtual machine executes a kernel-only instruction, it traps to the hypervisor if virtualization technology is present.

Dwa nowe tryby działania procesora

- VMX root operation - tryb dla monitora maszyny wirtualnej
  - nowe instrukcje, np. INVEPT (Invalidate Cached EPT Mappings), VMCLEAR (Clear Virtual-Machine Control Structure), VMPTRST (Store Pointer to Virtual-Machine Control Structure)
- VMX non-root operation - tryb dla oprogramowania gościa
  - ograniczona funkcjonalność, tzn. niektóre instrukcje powodują przejście do VMM
  - nowe instrukcje, np. VMCALL (Call to VM Monitor), VMFUNCTION (Invoke VM function)
  - nie istnieje bit widoczny dla oprogramowania który mówi, czy jesteśmy w VMX non-root
- wejście do non-root nazywamy VM entry
- wyjście z non-root do root nazywamy VM exit

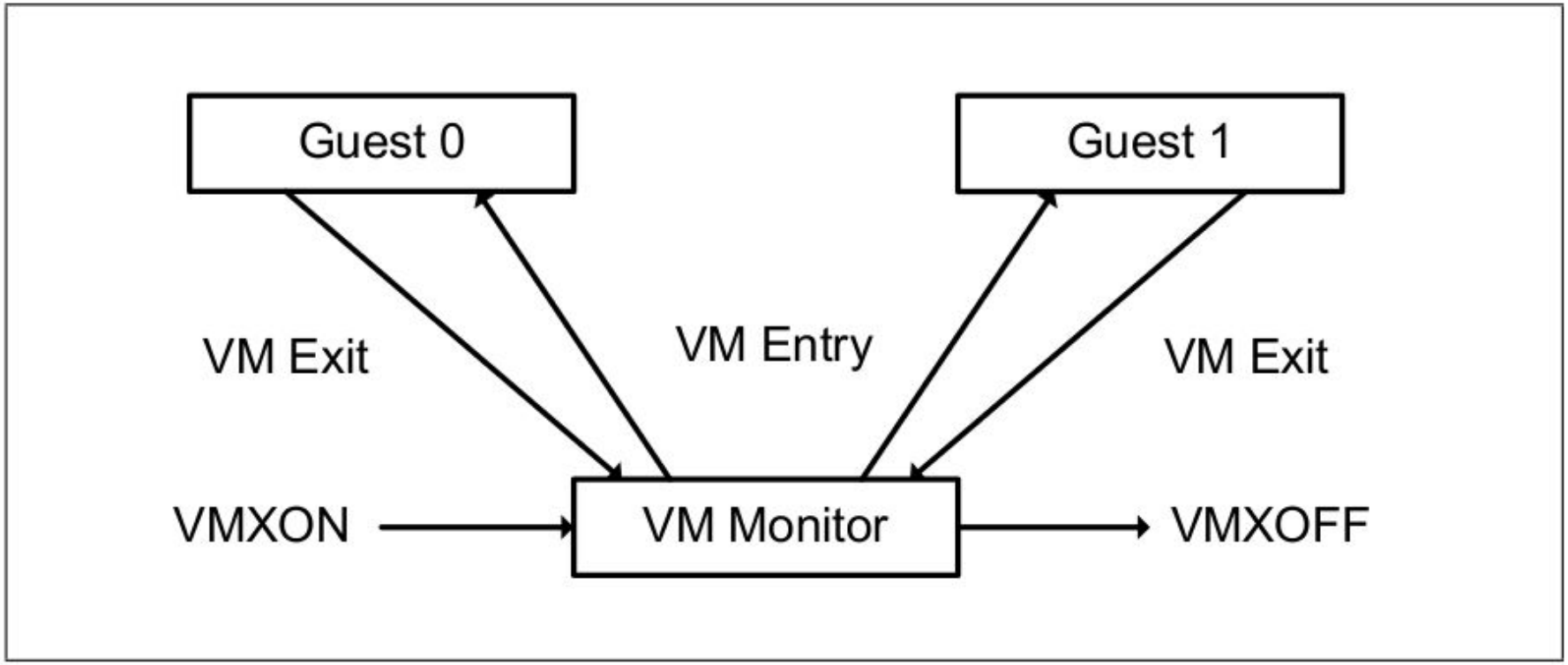


Figure 23-1. Interaction of a Virtual-Machine Monitor and Guests

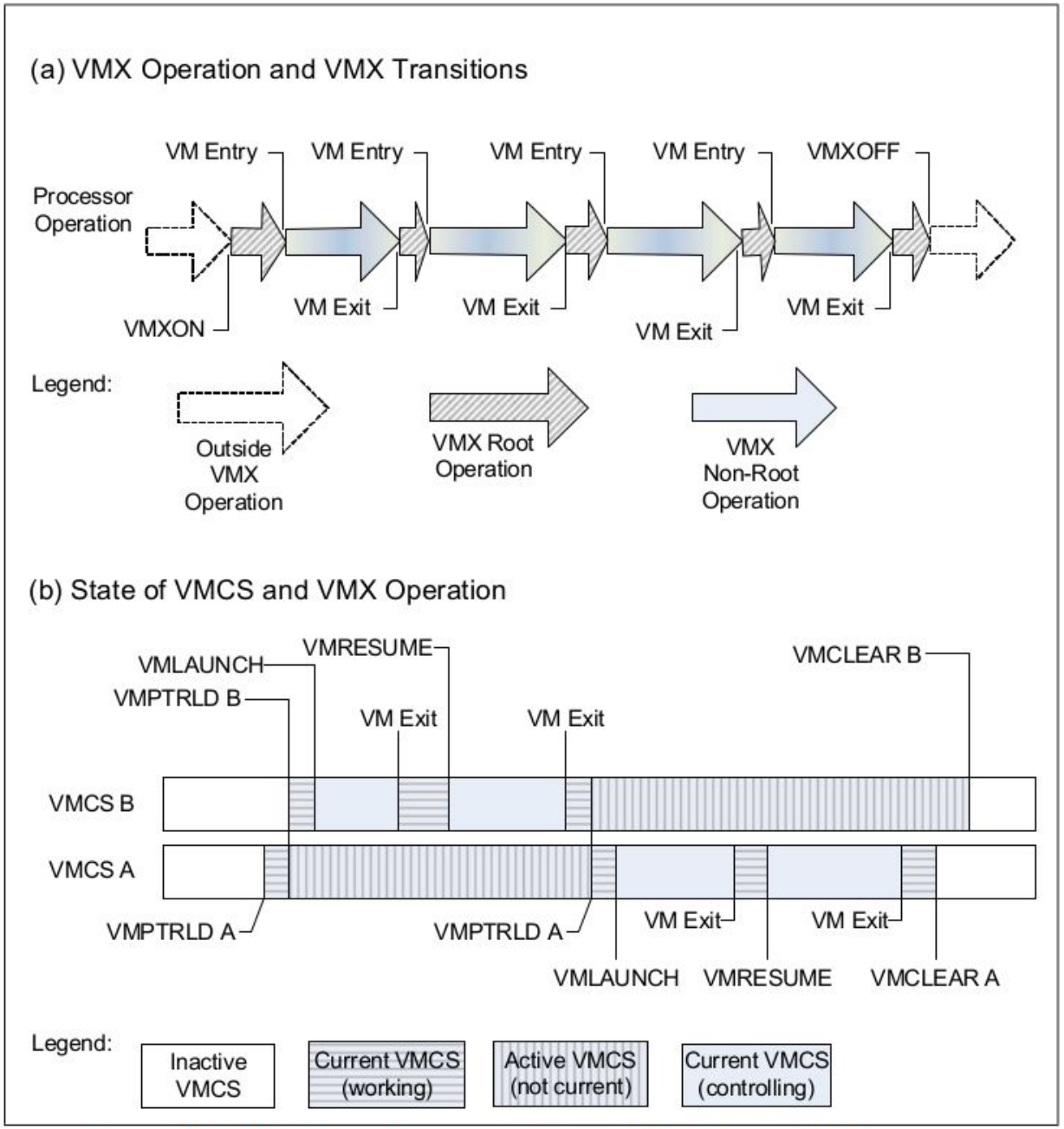


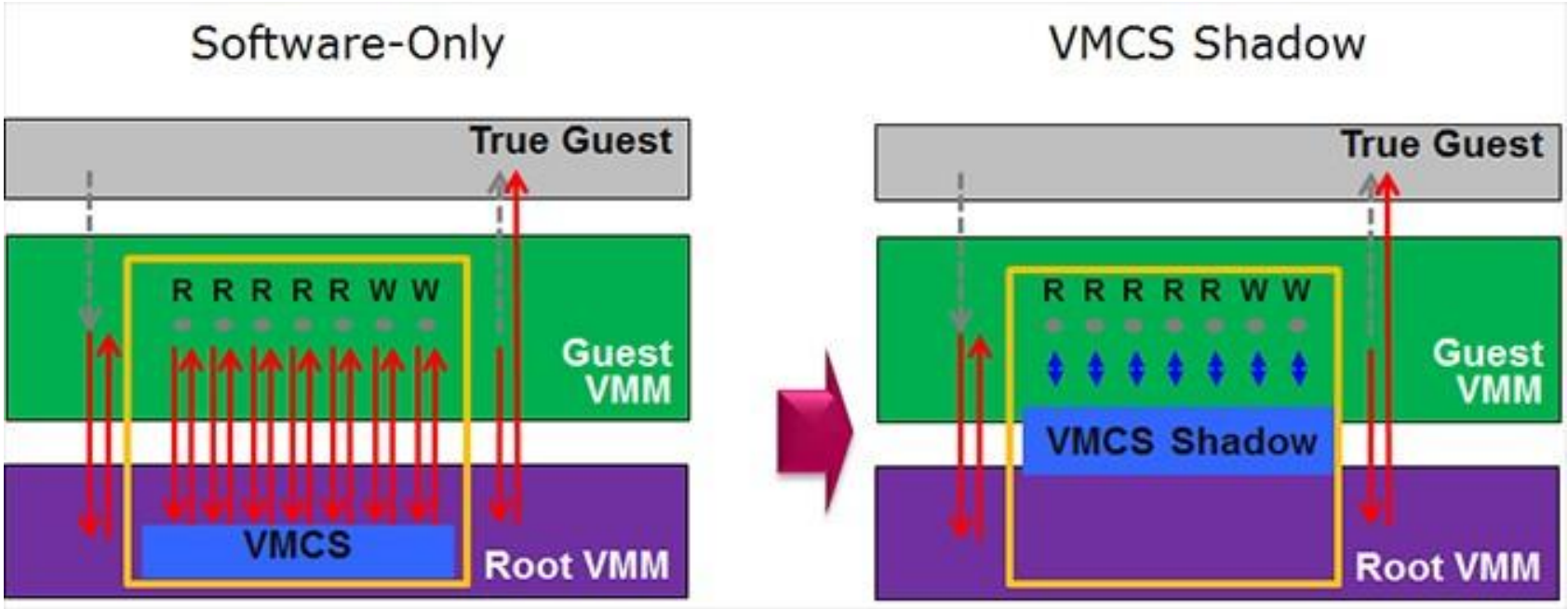
Figure 31-1. VMX Transitions and States of VMCS in a Logical Processor

VM Control Structure

- struktura danych do trzymania informacji o stanie maszyny wirtualnej
  - stan uruchomienia maszyny - czy włączyć ją używając VMRESUME czy VMLAUNCH?
  - stan procesora zapisywany przy VM exit i VM entry
  - informacja o ostatnim wydarzeniu (np. dlaczego zaszło VM exit?)
  - etc.
- modyfikowana przez VMM instrukcjami VMREAD, VMWRITE, VMCLEAR
- wskaźnik na VMCS modyfikowany przez VMM instrukcjami VMPTRST, VMPTRLD

Shadow VM Control Structure

- pozwala na modyfikację struktury danych będąc wewnątrz VMX non-root
- przydatne do zagnieżdżonej wirtualizacji



Powody VM exit

- instrukcje, bezwarunkowo: CPUID, VMCALL, VMXON, VMXOFF, …
- instrukcje, warunkowo, zależnie od konfiguracji: LIDT (Load Interrupt Descriptor Table Register) jeżeli ustawimy “descriptor-table exiting”; PAUSE (Spin Loop Hint - np. oszczędza energię), w zależności od “PAUSE exiting” i “PAUSE-loop exiting”; VMWRITE, ...
- wyjątki (w przypadku page fault możemy konfigurować)
- zewnętrzne przerwania zależnie od konfiguracji
- VMX-preemption timer (VMM może ustawić timer robiąc VM entry)
- Monitor Trap Flag (narzędzia do debugowania, pozwala na wychodzenie gdy zostaną spełnione określone wcześniej warunki)
- translacja adresu używając EPT (extended page table)
- APICv (wirtualny Advanced Programmable Interrupt Controller)
- zmiana EPTP (wskaźnik na EPT) - jeżeli wybierzemy spoza listy dozwolonych
- etc.

Obsługa VM exit w przypadku wyjątku

- VMM może celowo powodować wyjątki w maszynie wirtualnej
  - np. może chcieć wychwytywać zapisy do jakiejś strony, więc uczyni ją read-only
  - w takim wypadku VMM odpowiednio obsługuje VM exit i powraca do maszyny
- wyjątek może nie mieć nic wspólnego z VMM
  - wtedy następuje odbicie (reflect) wyjątku i wirtualna maszyna odbiera (przetwarza) go tak jakby była uruchomiona na zwykłym sprzęcie

Wsparcie w translacji adresów

- VPID (Virtual Processor Identifier)
  - przedtem (pierwsze implementacje VMX): wymagane czyszczenie TLB podczas np. VM entry czy VM exit
- EPT (Extended Page Table)

Extended Page Table

- co by było bez EPT?
- sprzęt rozumie różnicę pomiędzy adresem fizycznym gościa, a prawdziwym adresem fizycznym
- procesor przechodzi rozszerzoną tablicę stron by przetłumaczyć adres fizyczny gościa na prawdziwy adres fizyczny
- 4 poziomy
- VM exit gdy:
  - EPT misconfiguration - procesor natknie się na niewspieraną wartość lub sprzeczne informacje zakodowane w PDE lub PTE (np. zgaszony bit możliwości odczytu, lecz zapalony bit możliwości zapisu)
  - EPT violation - wpis jest nieważny (wszystkie flagi RWX zgaszone) lub błąd uprawnień RWX
- niektóre procesory wspierają bit dirty i bit accessed

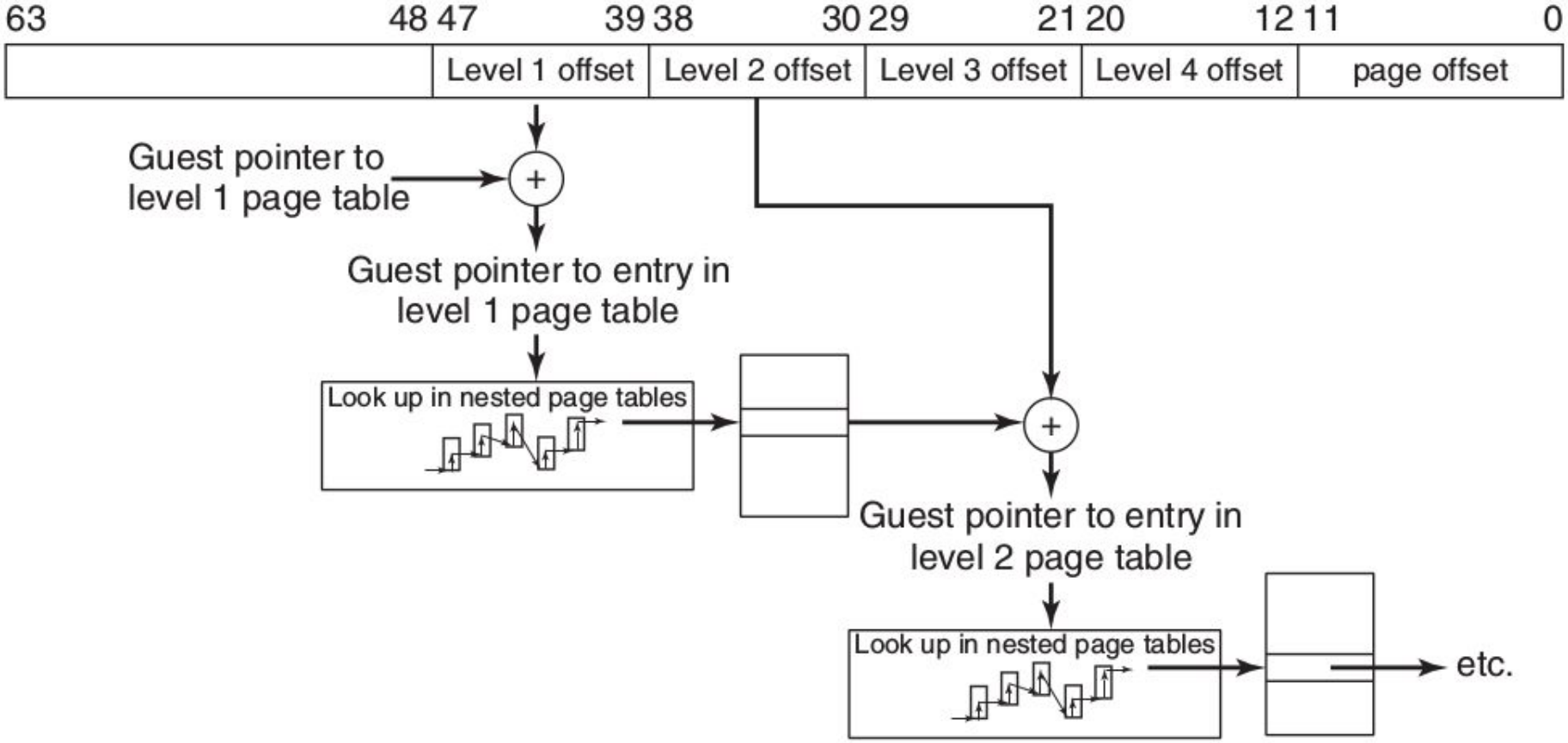


Figure 7-7. Extended/nested page tables are walked every time a guest physical address is accessed—including the accesses for each level of the guest's page tables.

Extended Page Table

- sprzętowa możliwość śledzenia modyfikacji pamięci robionych przez maszynę wirtualną (a dokładniej zapisów do adresów fizycznych gościa) - Page-Modification Log zawierający 512 wpisów po 64 bity; gdy bufor pełny, VM exit
- cache'owanie w TLB, trzy możliwe rodzaje wpisów dotyczące: tablicy stron gościa, EPT, mieszane

Wirtualizowanie pamięci

- gdy maszyna wirtualna próbuje się dostać do adresów fizycznych, pod które podmapowane jest jakieś urządzenie (wirtualne), page fault + VM exit sprawiają, że VMM może uruchomić kod emulujący zachowanie urządzenia
- w jaki sposób zachować kontrolę nad pamięcią maszyny, zatem także nad translacją adresów, ale pozwolić jej normalnie działać?
- VMM nie kontroluje tablicy stron gościa, za to kontroluje rejestr CR3 (Control Register), w którym znajduje się adres tablicy stron

Obsługa tablicy stron gościa, podejście brute force

- znamy adres tablicy stron
- VMM zakazuje dostępu do tablicy stron (np. robi całą tablicę stron read only)
- gdy gość chce użyć lub zmodyfikować tablicę, page fault + VM exit sprawiają, że VMM może zaemulować tę operację
- VMM trzyma własną wersję tablicy stron gościa
- nieefektywne

Obsługa tablicy stron gościa, podejście drugie

- gość ma dowolność w używaniu swojej tablicy stron
- VMM utrzymuje swoją wersję tablicy stron, niekoniecznie taką samą jak gość (nie kontroluje zmian gościa)
- tablica stron VMM to cache dla tablicy stron gościa (tzn. wirtualny TLB)
- procesor korzysta z tablicy stron VMM

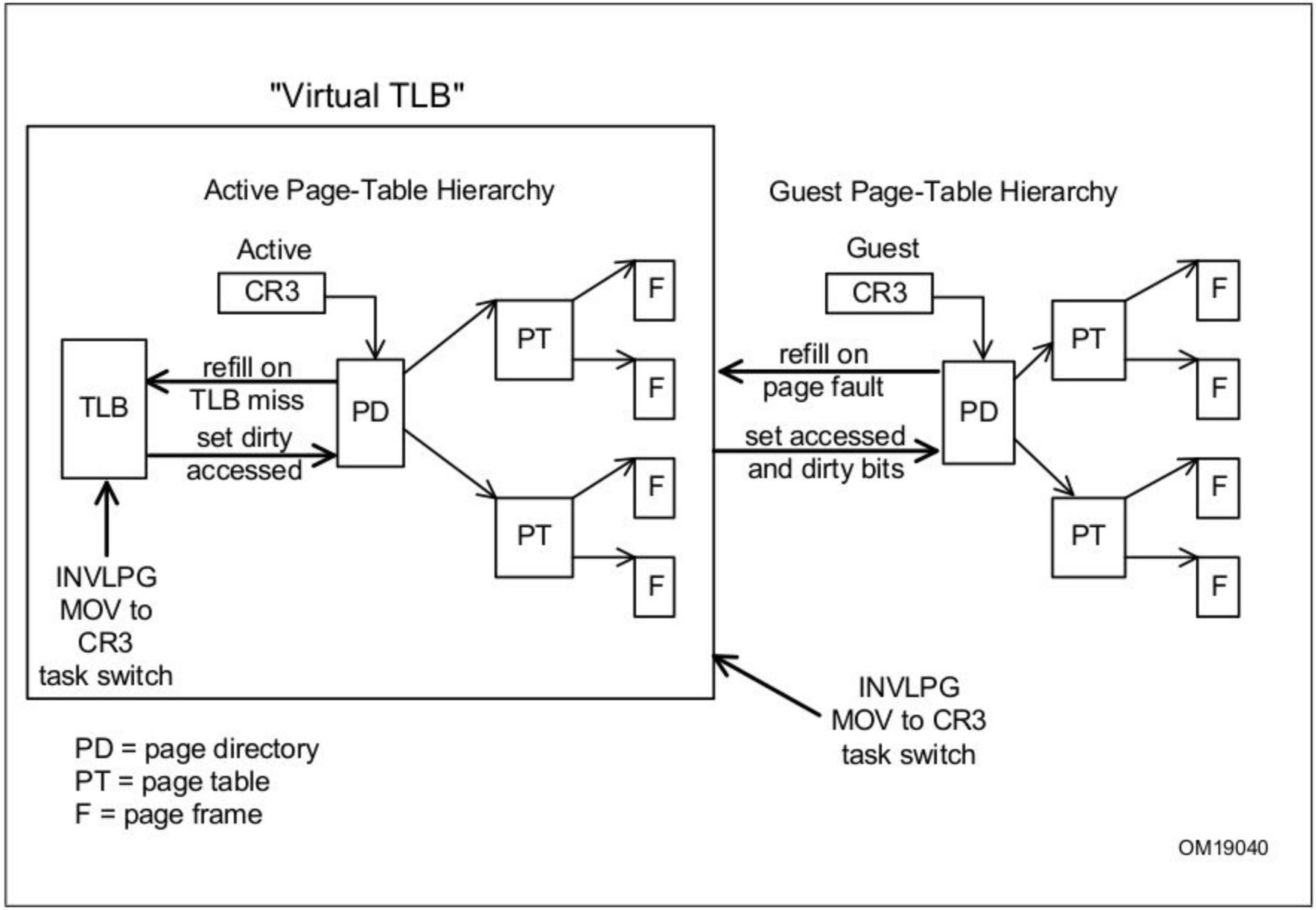


Figure 32-1. Virtual TLB Scheme

Wirtualne TLB

- załóżmy że PT gościa pozwala na większy dostęp niż PT VMM
  - jeżeli zostanie wykonany dostęp prawidłowy ze względu na gościa, ale nieprawidłowy ze względu na VMM, dostaniemy page fault (“TLB miss”), a VMM uzupełni swoją tablicę stron w oparciu o struktury gościa (“TLB fill”)
- załóżmy że PT gościa ma większe restrykcje niż PT VMM
  - tylko jeżeli gość zredukuje dostęp do istniejącego wpisu
  - sytuacja dozwolona, tak samo jak w przypadku pobrania starych danych z prawdziwego TLB
  - jeżeli system zmniejsza uprawnienia, powinien wywołać np. INVLPG (Invalidate TLB Entries), który spowoduje VM exit
- jak ustawiać bity A i D w tablicy stron gościa?
  - procesor sam zapala A i D w tablicy stron VMM
  - ustawianiem bitów u gościa zajmuje się VMM

Advanced Programmable Interrupt Controller

- kontroler służący do obsługi przerwań w procesorach Intel
  - dwie części: I/O APIC i local APIC (patrz kolejny slajd)
- wysyłanie przerwań pomiędzy procesorami
- niektóre rejestry
  - TPR (Task Priority Register) - określamy próg priorytetu przerwań, które chcemy wyłączyć
  - PPR (Processor Priority Register) - rejestr tylko do odczytu określający priorytet z jakim procesor wykonuje zadania w danym momencie
  - EOI (End Of Interrupt) - do tego rejestru musimy wykonać zapis aby zasygnalizować koniec wykonywania procedury przerwania (przed IRET)
- przerwania generowane przez timer
- przerwania generowane gdy procesor się przegrzewa

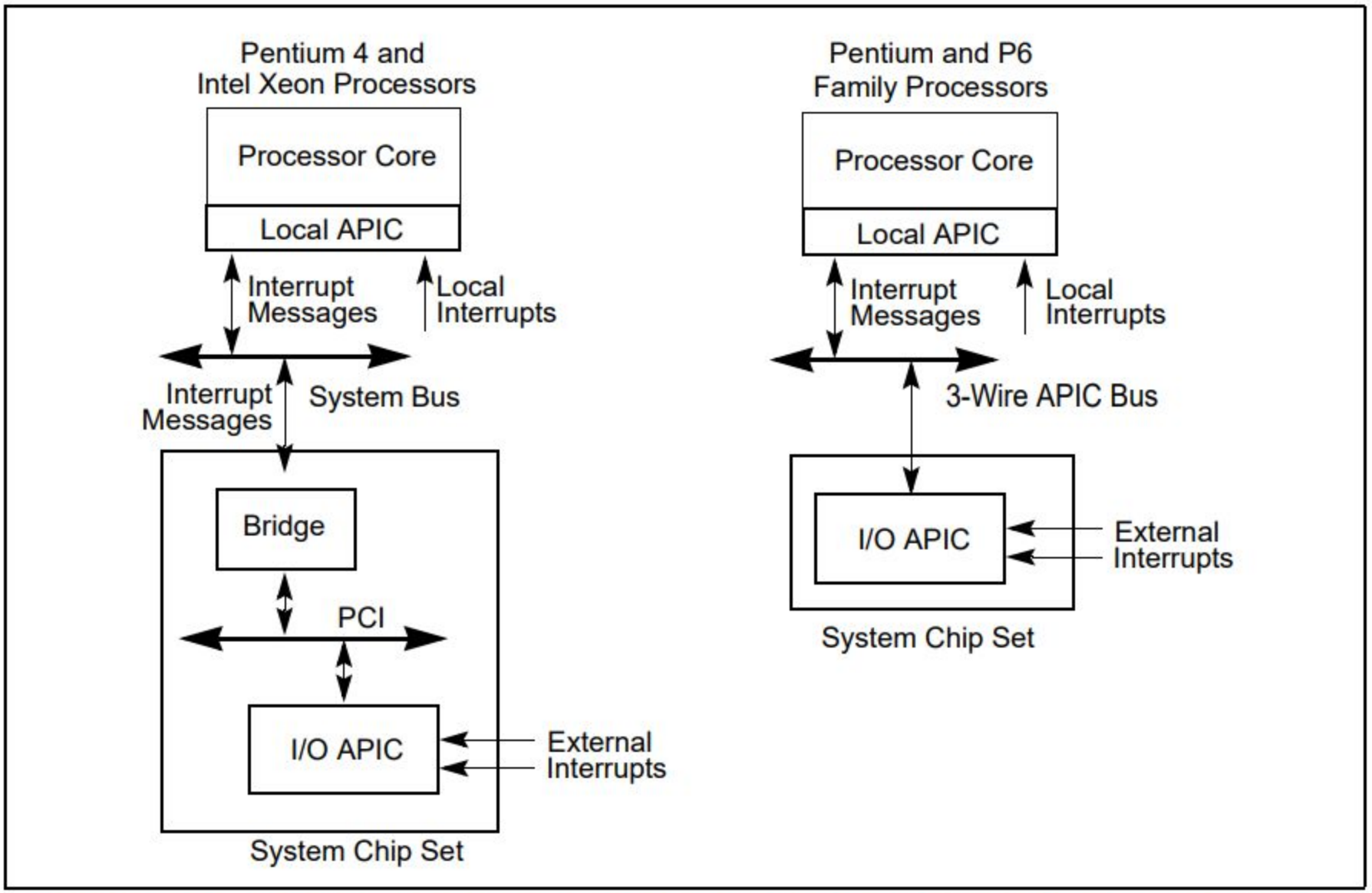


Figure 10-1. Relationship of Local APIC and I/O APIC In Single-Processor Systems

APICv

- dostarczanie wirtualnych przerwań
  - VM exit nie jest wymagane
  - mogą obudzić procesor po instrukcji HLT tak jak zwykłe przerwania
  - sprawdzenie czy są dostępne przerwania wykonywane podczas VM entry, TPR i EOI virtualization, …
  - możliwość ustawienia interrupt-window, by zakolejkować VM exit służące do dostarczenia przerwania, jeżeli gość nie jest w stanie interruptible
  - możliwość ustawienia NMI-window, by zakolejkować przerwanie NMI (Non-Maskable), jeżeli gość nie jest gotowy go przyjąć
- emulowanie odczytów i zapisów do pamięci APIC
  - zwykle VM exit
  - niektóre żądania procesor może zwirtualizować i nie wychodzić do VMM (jeśli odczyt/zapis dotyczy co najwyżej 4 bajtów, jest odpowiednio wyrównany i zachodzi szereg innych wymagań)
- możliwość przesyłania przerwań do innych wirtualnych procesorów

Pochodzenie wirtualnych przerwań

- od wirtualnych urządzeń, emulowanych przez VMM
- bezpośrednio (direct-mapped) od fizycznych urządzeń

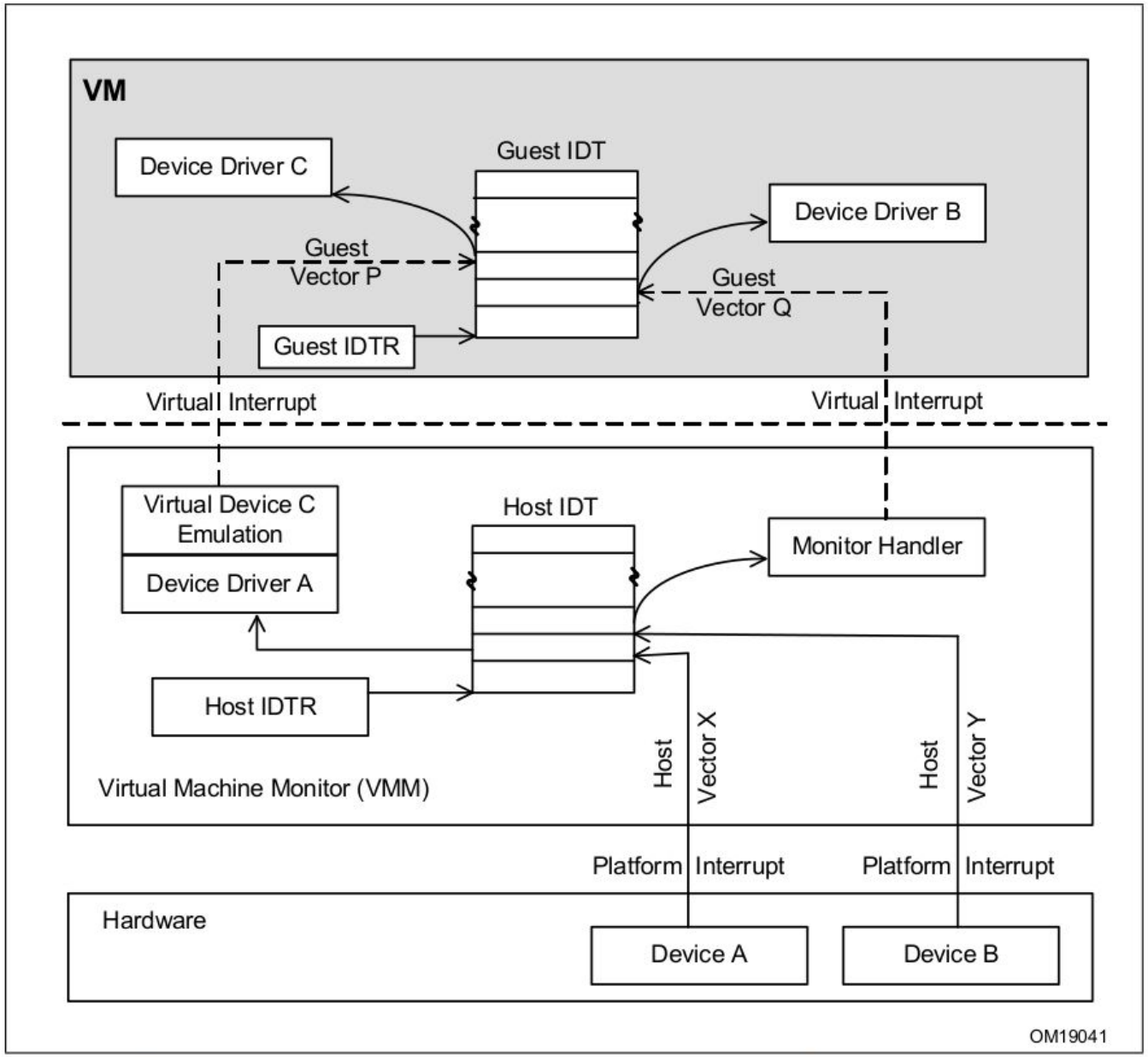


Figure 33-1. Host External Interrupts and Guest Virtual Interrupts

Systemy wieloprocesorowe

- jeżeli VMM posiada kilka fizycznych procesorów, może odpalać wirtualne maszyny na różnych z nich - potrzebny planista
- problem jeżeli procesory posiadają inny zestaw rozszerzeń VMX
- potrzeba np. emulacji instrukcji CPUID

32 vs 64 bit

- możliwość wirtualizowania systemu 32-bitowego na 64-bitowym sprzęcie

Table 31-1. Operating Modes for Host and Guest Environments

Capability	Guest Operation in IA-32e mode	Guest Operation Not Requiring IA-32e Mode
IA-32e mode VMM	Yes	Yes
32-bit VMM	Not supported	Yes

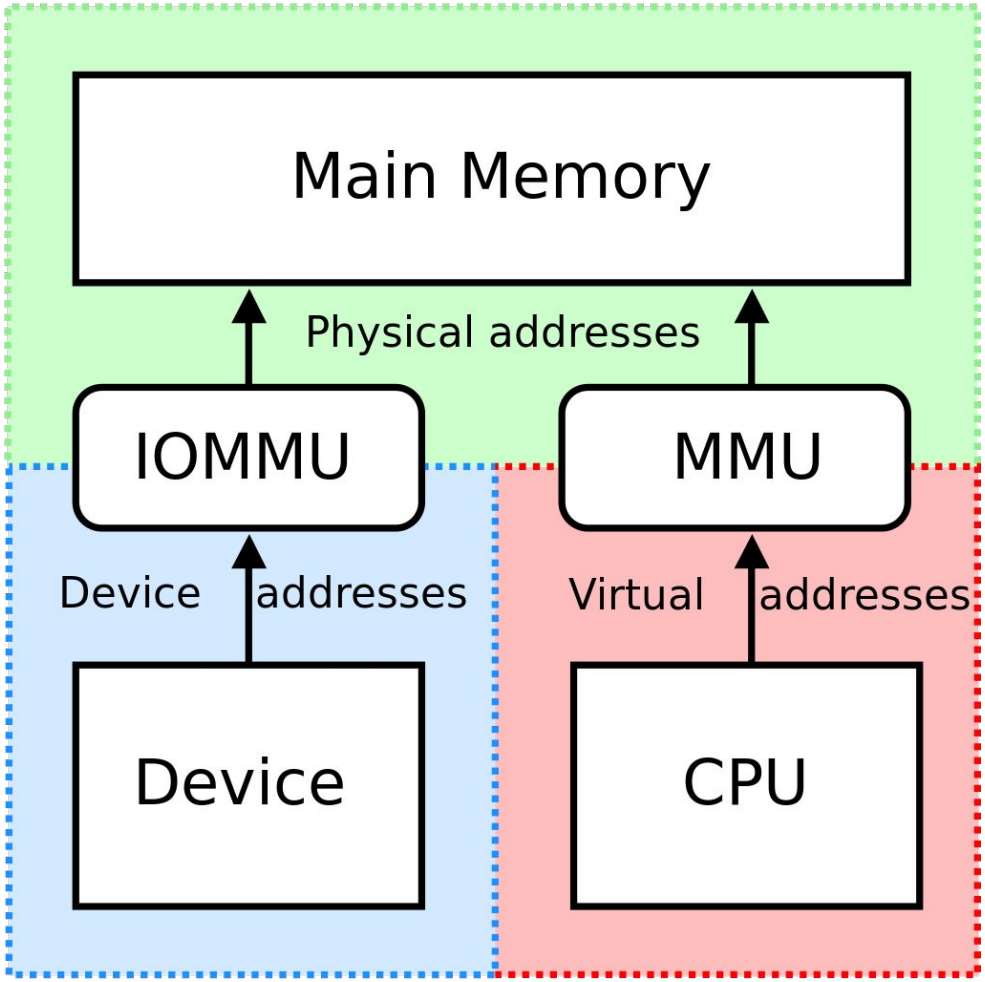
Intel Processor State

- wsparcie dla odpluskwiania/profilowania
- pozwala na zbieranie informacji o stanie wirtualnej maszyny, by móc potem odtworzyć dokładny przebieg wykonania kodu
  - wartość instruction pointer
  - skoki warunkowe: kierunek, adres skoku
  - power management events
  - wartość rejestru C3
  - timestamp
  - bus clock ratio
  - zdarzenia asynchroniczne (np. przerwania)

Wirtualizacja wejścia/wyjścia

- co jeżeli maszyna wirtualna sprawdza, jakie urządzenia są do niej podłączone? VM exit
- jeśli VMM da dostęp do prawdziwych urządzeń, gość może zacząć pisać do cudzej pamięci (myśląc, że jest jedynym systemem)
- w przypadku dysku można stworzyć abstrakcję za pomocą osobnego pliku dla każdego gościa
- można całkowicie emulować dane urządzenie
- można powiedzieć maszynie, że ma dostęp do dysku X, choć realnie jest to dysk Y (jeżeli wymieniamy sprzęt i wolimy zmienić kod nadzorcy, a nie wielu wirtualnych maszyn)

Intel Virtualization Technology For Directed I/O (a.k.a. I/O MMU, a.k.a. Intel VT-d)



źródło: https://en.wikipedia.org/wiki/Input%E2%80%93output_memory_management_unit

I/O MMU

- translacja adresów urządzeń na adresy fizyczne przy użyciu tablic stron, jak przy zwykłym MMU; nadzorca tworzy takie mapowanie, by maszyna wirtualna mogła korzystać z DMA i miała dostęp tylko do swojej pamięci (device isolation, DMA remapping)
- możemy przypisać urządzenie bezpośrednio maszynie wirtualnej, bez tworzenia warstwy abstrakcji (device pass through)
- tłumaczenie i trasowanie przerwań, by mogły dochodzić do konkretnej maszyny wirtualnej (interrupt remapping)
- wsparcie dla dostarczania przerwań (interrupt posting)

Single Root I/O Virtualization

- możemy już przypisać każdej maszynie wirtualnej odrębną fizyczną kartę sieciową
- jeżeli wirtualizujemy kilka systemów, nie robi to problemu
- jeżeli wirtualizujemy 64 systemy, nie znajdziemy komputera pod stosem kabli
- można próbować emulować software'owo współdzielenie urządzenia przez kilka systemów/nadzorców
- taka emulacja byłaby wolna oraz musielibyśmy stworzyć ją sami (czyt. chcemy wsparcia sprzętowego)

Single Root I/O Virtualization

- współdzielenie urządzania przez kilka maszyn wirtualnych bez udziału nadzorcy
- urządzenie posiada zbiór niezależnych przestrzeni adresowych, przerwań, kanałów DMA, buforów
- urządzenie wspiera dwa sposoby komunikacji
  - PF (physical functions) - pełen zestaw funkcji wyłącznie dla administratora urządzenia
  - VF (virtual functions) - funkcje dla wirtualizowanych systemów (zwykle podobne do PF lecz bez możliwości konfiguracji)

Single Root I/O Virtualization

- seria kart sieciowych Intela I350 wspiera 8 buforów cyklicznych do wysyłania i odbierania danych

źródło: https://www.ceneo.pl/15339451

DMA remapping

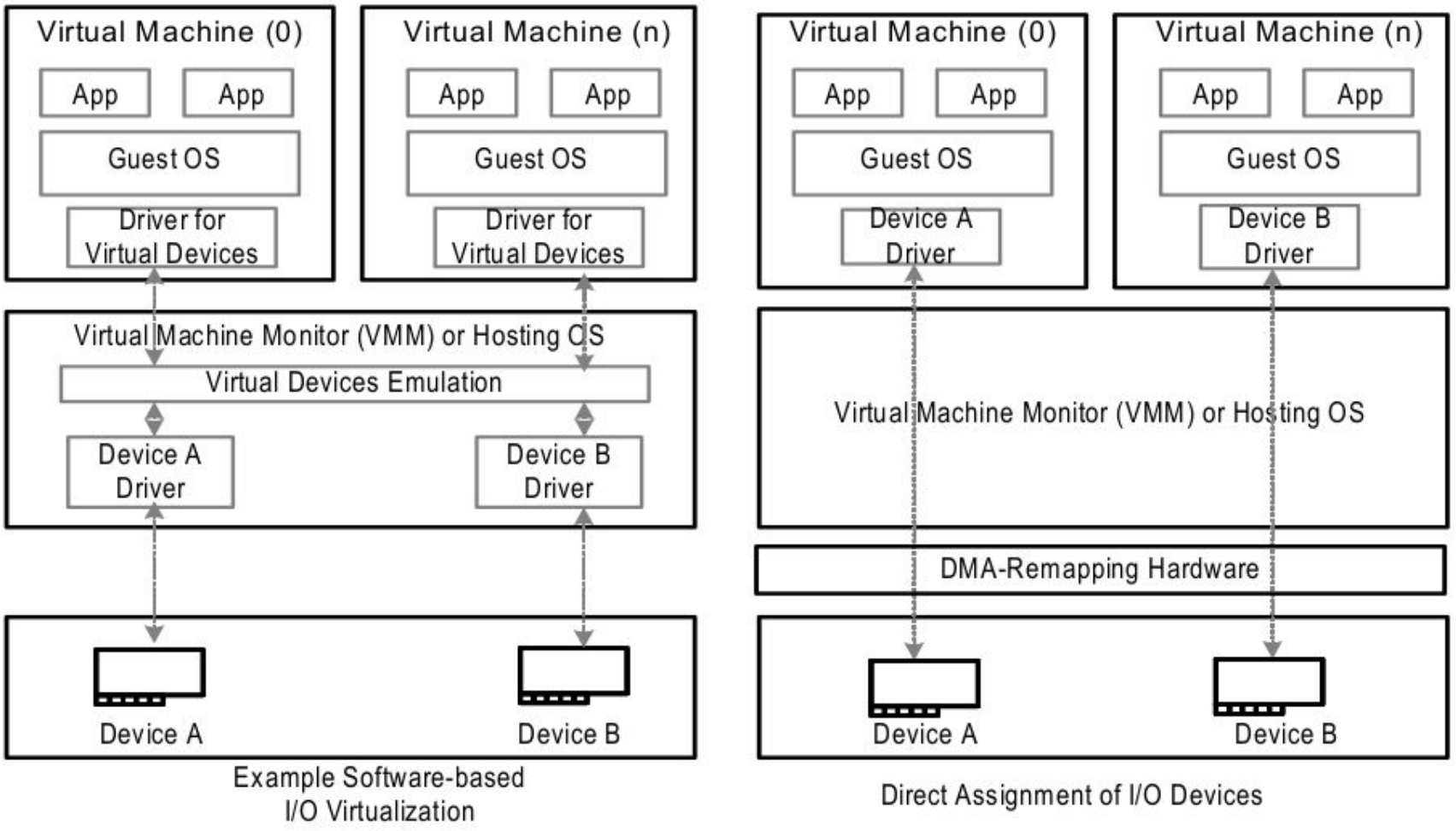


Figure 2-3. Example Virtualization Usage of DMA Remapping

DMA remapping - inne zastosowania

- system operacyjny może chronić wrażliwe struktury danych przed szkodliwymi procesami
- 32-bitowe urządzenia mogą korzystać z pamięci powyżej 4GB
- jeżeli urządzenie posiada dużo pamięci, nie musimy jej podmapowywać w ciągłą przestrzeń adresową
- możliwość udostępniania przestrzeni adresowej procesów urządzeniom, np. GPU

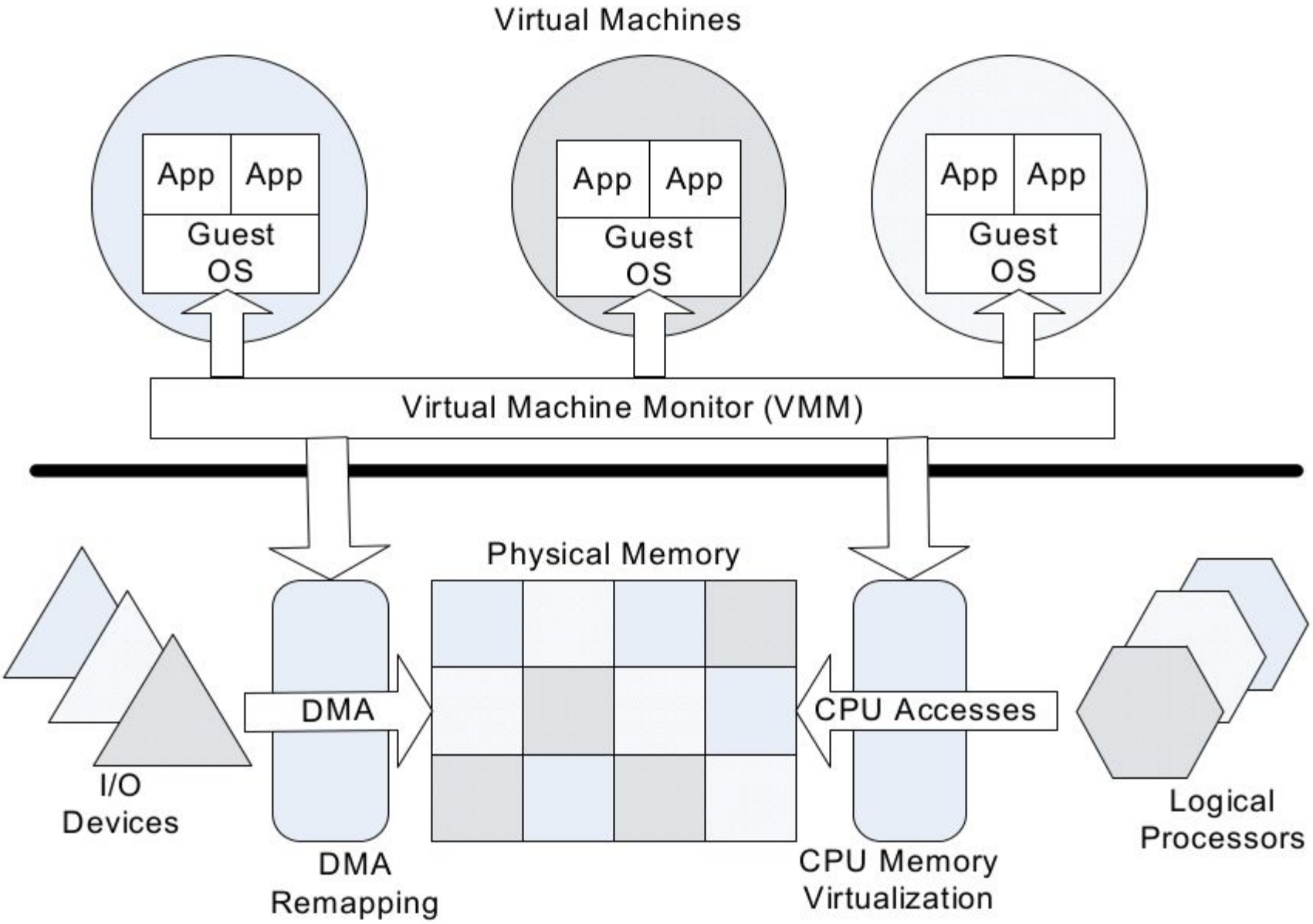


Figure 2-4. Interaction Between I/O and Processor Virtualization

Interrupt remapping

- izolacja
  - wykrywanie, do kogo zostało wysłane przerwanie (konkretna maszyna wirtualna)
  - VMM może konfigurować i zmieniać parametry dostarczanych przerwań, np. usuwać atrybuty z zewnętrznych przerwań by można było je odróżnić od przerwań IPI (Inter-Processor Interrupt)
- migracja
  - dynamiczne trasowanie przerwań, jeżeli nagle maszyna wirtualna została przeniesiona na inny vCPU
  - bez tej opcji wymagane byłoby ponowne wysłanie przerwania
  - może być używane do load-balancingu przerwań na kilka procesorów (przez OS lub VMM) podczas dużego obciążenia I/O

Interrupt posting

- skalowalność
  - wirtualizacja wraz z użyciem SR-IOV wymaga więcej wektorów przerwań niż w standardowym systemie
  - wsparcie sprzętowe tworzy wirtualną przestrzeń dla wektorów przerwań dla każdej VM
- wydajność
  - bez wsparcia sprzętowego przerwania przychodzące do VM zawsze przechodzą przez VMM
  - ze wsparciem, VMM konfiguruje przerwania, które następnie dochodzą bezpośrednio do vCPU
    - jeżeli vCPU działa, przerwanie dochodzi
    - jeżeli vCPU jest obecnie wywłaszczone, przerwanie trafia do bufora
    - jeżeli vCPU jest obecnie wywłaszczone, ale przerwanie wymaga przetworzenia w czasie rzeczywistym, procesor przekazuje kontrolę VMM by włączyć vCPU, a następnie przerwanie jest dostarczane

Device domains (Tanenbaum)

- inny sposób na rozwiązanie problemu interakcji maszyny ze światem
- maszyna zamiast wysyłać typowe żądanie I/O, np. pisząc do rejestrów urządzenia, kontaktuje się z nadzorcą
- nadzorca dostaje wysokopoziomowe żądanie “przeczytaj blok 42 z dysku 7”, a następnie sam tłumaczy je używając odpowiedniego sterownika

NASA & Intel case study

- w 2011 NASA we współpracy z Intelem przetestowali szybkość komunikacji pomiędzy VM używając różnych technologii
  - “Bare-metal - data transfer between non-virtualized servers”
  - “Software-only virtualization - data transfer between virtual machines (VMs)”
  - “Virtualized I/O - data transfer between VMs with OS-based paravirtualization”
  - “Single-Root I/O Virtualization (SR-IOV) - Data transfer between VMs using SR-IOV”
- użyli serwerowni w której NASA analizowało zmiany klimatyczne
  - 30.000 Intel Xeon E5520 @ 2.27 GHz (quad-core) + 64 GB RAM
  - 64 GPU
  - łącznie 24 petabajty (1 PB to 1024 TB) powierzchni dyskowej
  - Ubuntu 11.04
- wnioski
  - “SR-IOV is a Key Requirement for Moving Cluster Applications to the Cloud.”
  - “The core conclusion from this testing is that cloud-based high-performance computing is a viable possibility. SR-IOV, as supported by Intel Ethernet Server Adapters, is a core enabling technology that helps overcome performance limitations associated with virtualization.”

Table 2. Nuttcp results, which demonstrate that SR-IOV helps attain virtualized throughput near wire speed, similar to that with bare-metal servers.

Bare Metal-to-Bare Metal	VM-to-VM (Software Virtualization)	VM-to-VM (Virtualized I/O)	VM-to-VM (with SR-IOV)
4418.8401 Mbps	137.3301 Mbps	5678.0625 Mbps	8714.4063 Mbps
8028.6459 Mbps	138.5963 Mbps	5692.8146 Mbps	8958.5032 Mbps
9341.4362 Mbps	141.8702 Mbps	5746.2926 Mbps	9101.7356 Mbps
9354.0999 Mbps	145.6024 Mbps	5864.0557 Mbps	9151.5769 Mbps
9392.7072 Mbps	145.7500 Mbps	5955.8176 Mbps	9193.1103 Mbps
9414.7318 Mbps	146.1043 Mbps	5973.2256 Mbps	9228.5370 Mbps
9414.8207 Mbps	146.1092 Mbps	6223.4034 Mbps	9251.8453 Mbps
9414.9368 Mbps	146.2758 Mbps	6309.8478 Mbps	9313.8894 Mbps
9415.1618 Mbps	146.3042 Mbps	6311.3896 Mbps	9348.2984 Mbps
9415.2675 Mbps	146.4449 Mbps	6316.7924 Mbps	9408.0323 Mbps

Bibliografia

- Intel® 64 and IA-32 architectures software developer's manual, volumes 2, 3A and 3C, December 2017
- Intel® Virtualization Technology for Directed I/O Architecture Specification, November 2017
- Andrew S. Tanenbaum, Modern Operating Systems, 4th edition
- https://software.intel.com/en-us/blogs/2014/12/12/enabling-virtual-machine-control-structure-shadowing-on-a-nested-virtual-machine
- https://www.intel.com/content/dam/www/public/us/en/documents/case-studies/10-gigabit-ethernet-nasa-case-study.pdf