



# XenGT: A Full GPU Virtualization Solution with Mediated Pass-Through

March 27, 2014

Zhiyuan Lv, [zhiyuan.lv@intel.com](mailto:zhiyuan.lv@intel.com)

Kevin Tian, [kevin.tian@intel.com](mailto:kevin.tian@intel.com)

Eddie Dong, [eddie.dong@intel.com](mailto:eddie.dong@intel.com)

David Cowperthwaite,  
[david.j.cowperthwaite@intel.com](mailto:david.j.cowperthwaite@intel.com)



ANDROID FOR INTEL ARCHITECTURE INTEL LINUX WIRELESS GUPNP KVM POKY  
TIZEN OPENSTACK POWERTOP YOCTO CONNMAN XEN POFONO LINUX KE  
INTEL LINUX GRAPHICS SYNCEVOLUTION SIMPLE FIRMWARE INTERFACE (SFI) ENTERPRISE SECURITY IN

# Agenda



- Why GPU Virtualization
- How GPU Virtualization Works
- How XenGT Works
- Demo
- Current Status
- Summary



# GPU Use Cases



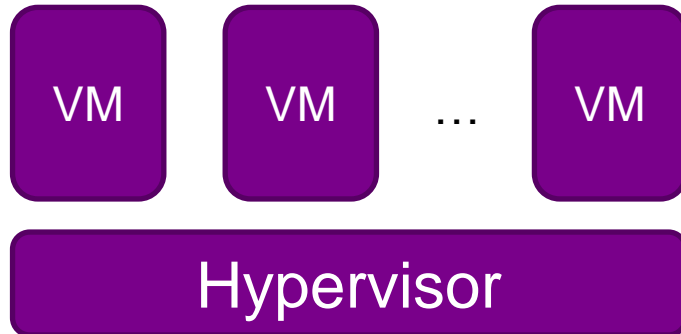
3D Graphics

Media

Compute



# Virtualization Use Cases



## Use Cases

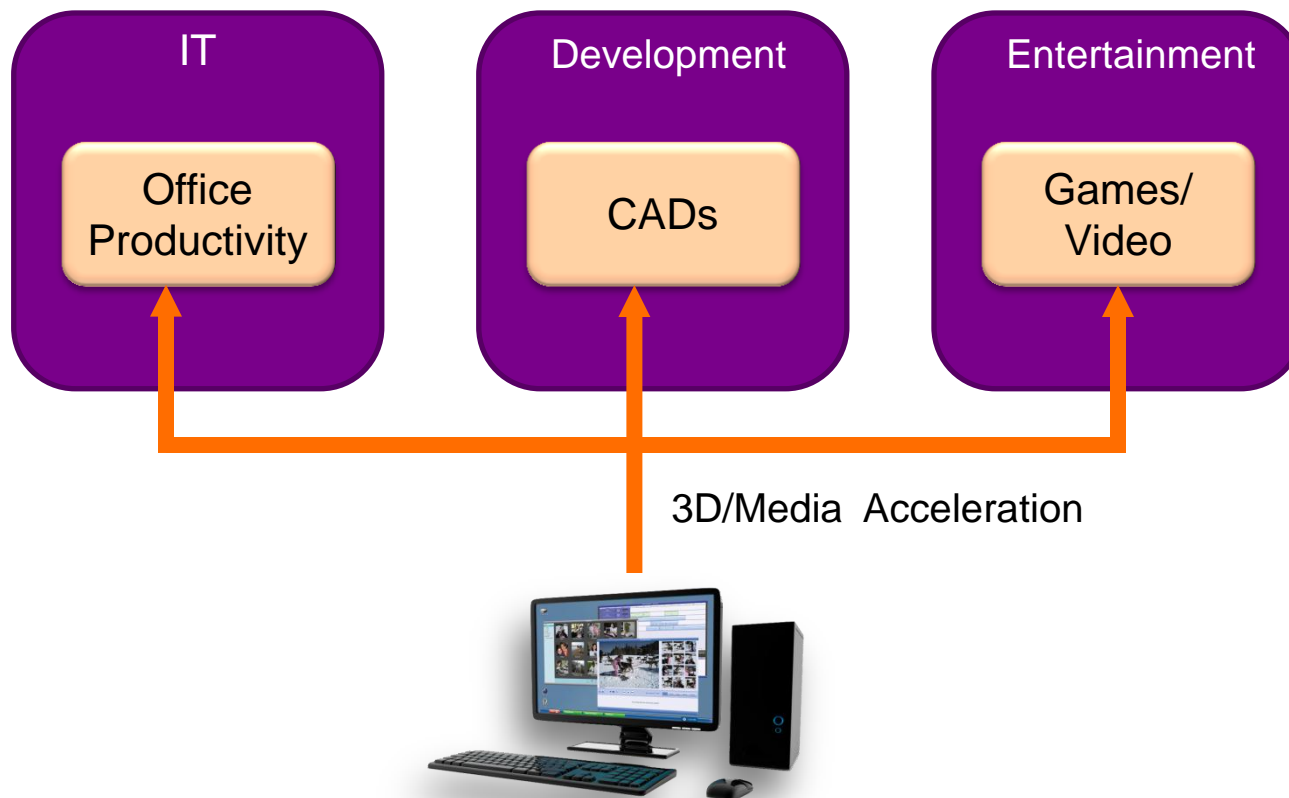
Virtual Data Center  
Cloud  
Remote Virtual Desktop  
Rich Virtual Client  
Bring Your Own Device  
Smart TV  
Multi-Screen Infotainment  
Secure e-Payment

...

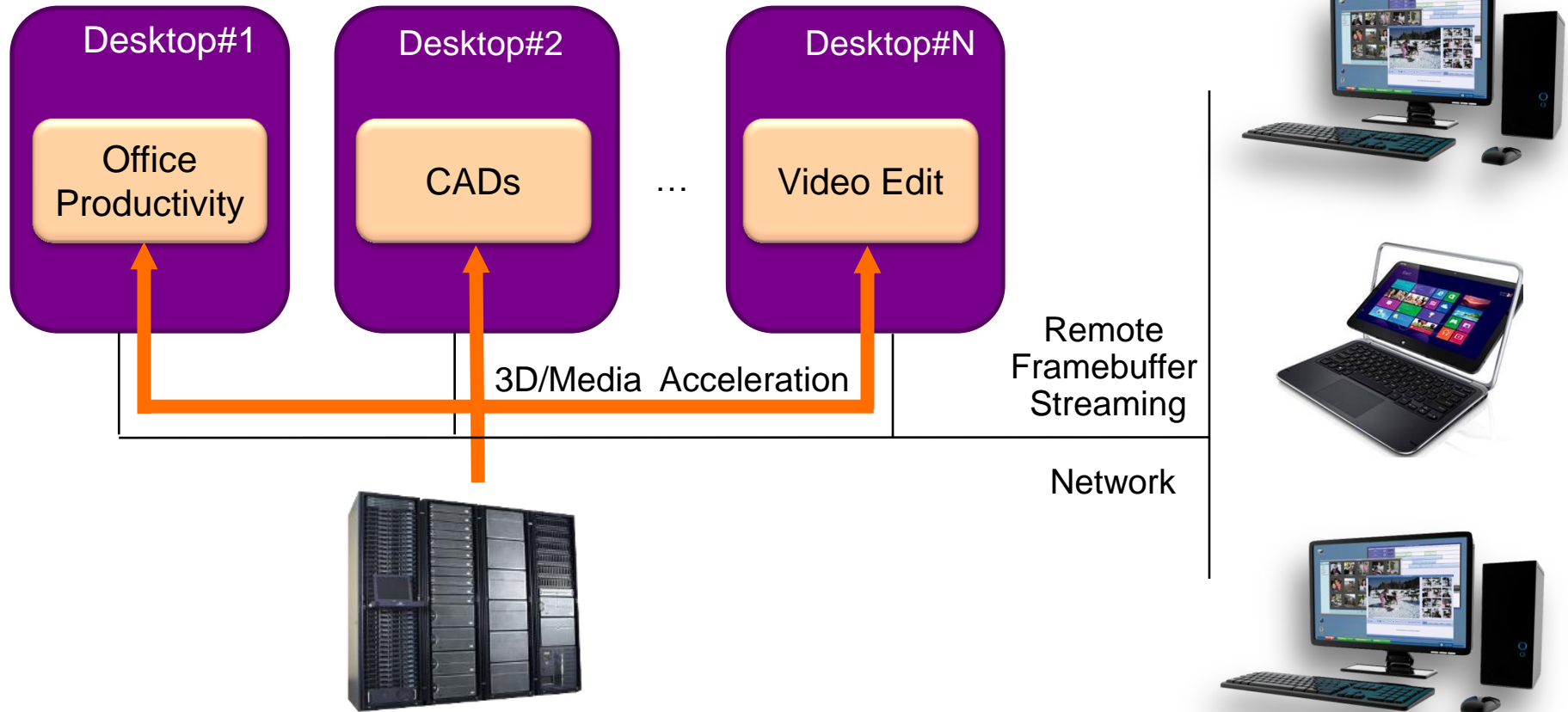


# GPU Virtualization Gains Momentum

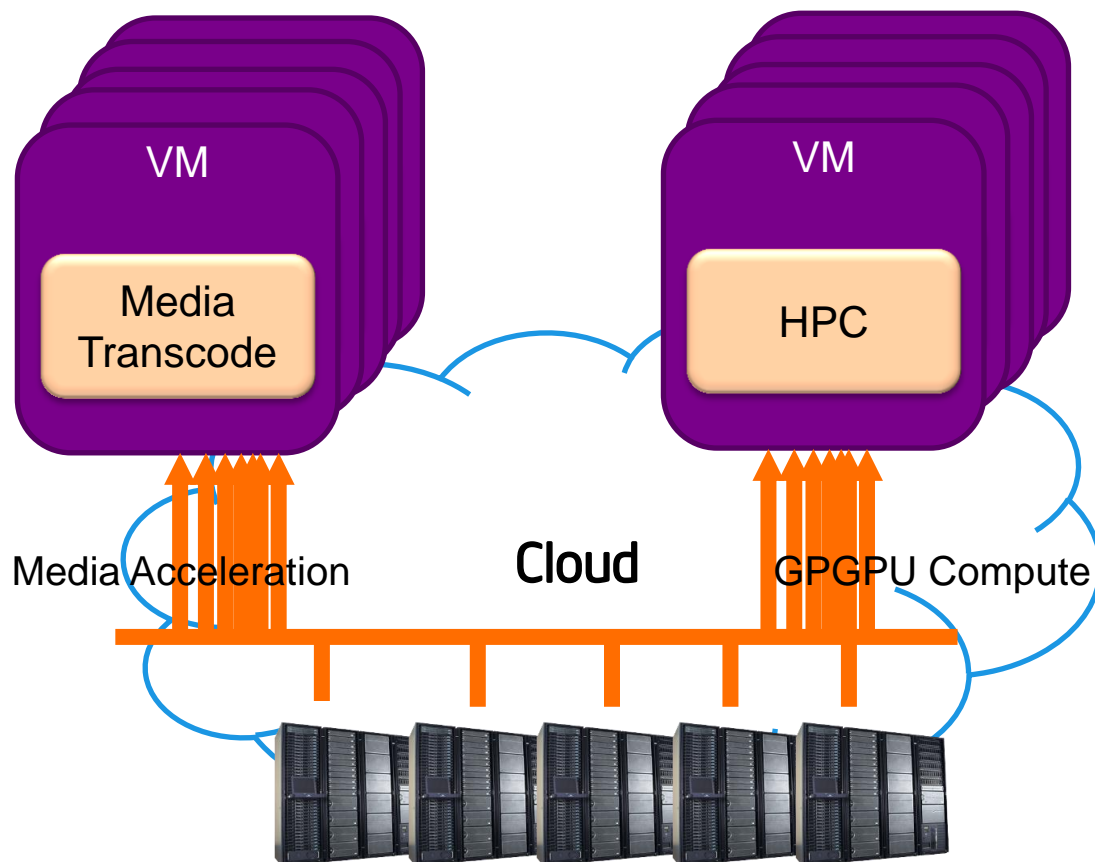
# Scenario-1: Rich Virtual Client



# Scenario-2: Remote Virtual Desktop



# Scenario-3: GPU As A Service





And many other virtualization scenarios  
which run GPU-accelerated tasks in VM

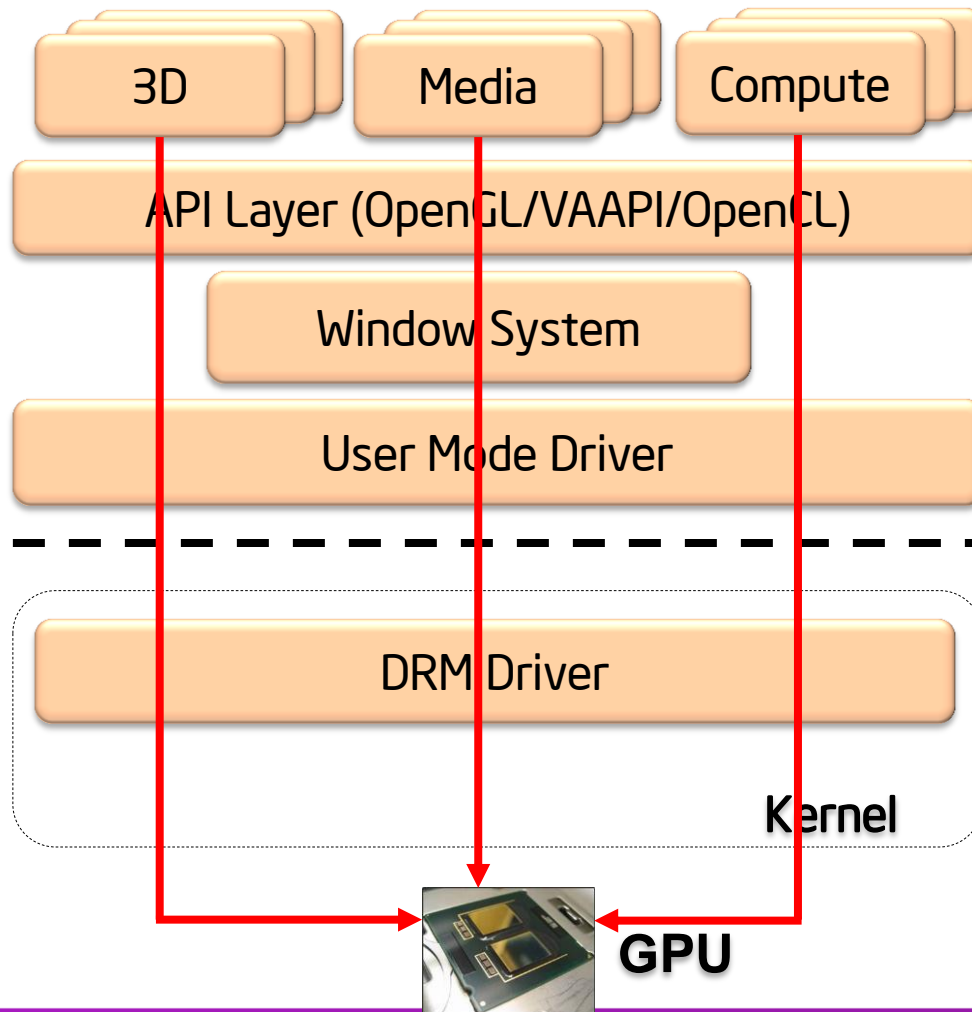
# Agenda



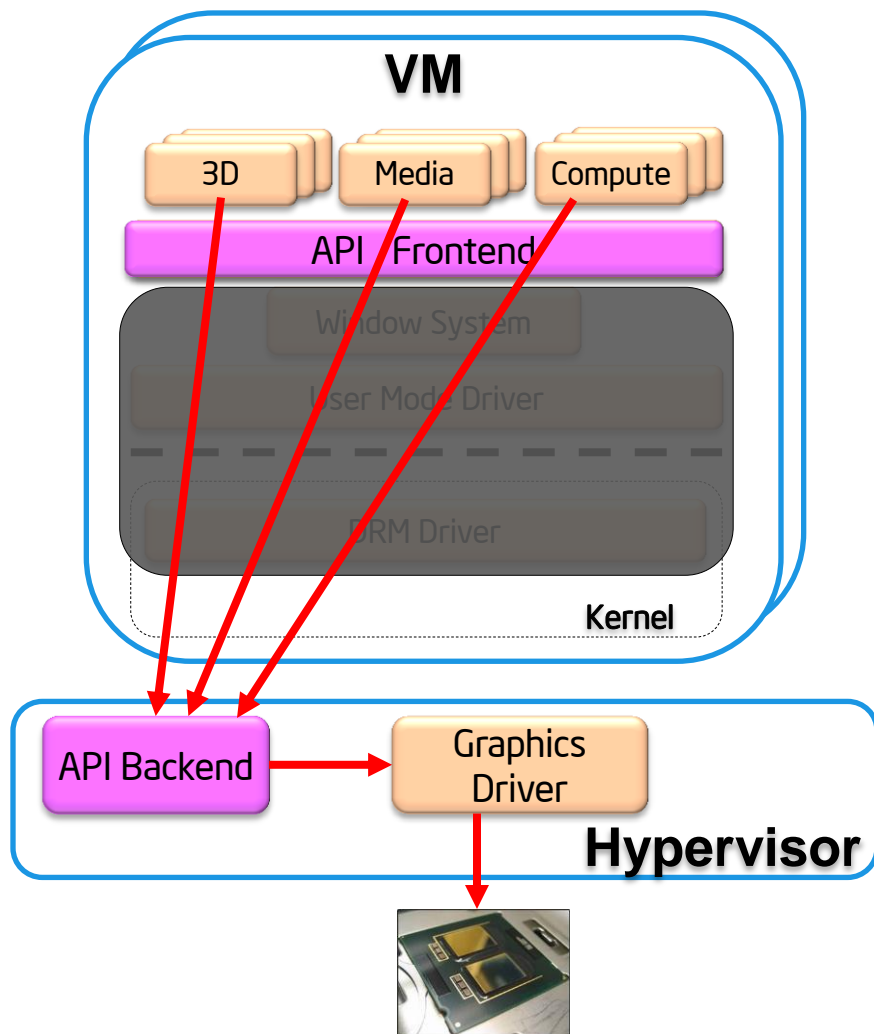
- Why GPU Virtualization
- How GPU Virtualization Works
- How XenGT Works
- Demo
- Current Status
- Summary



# Linux Graphics Stack

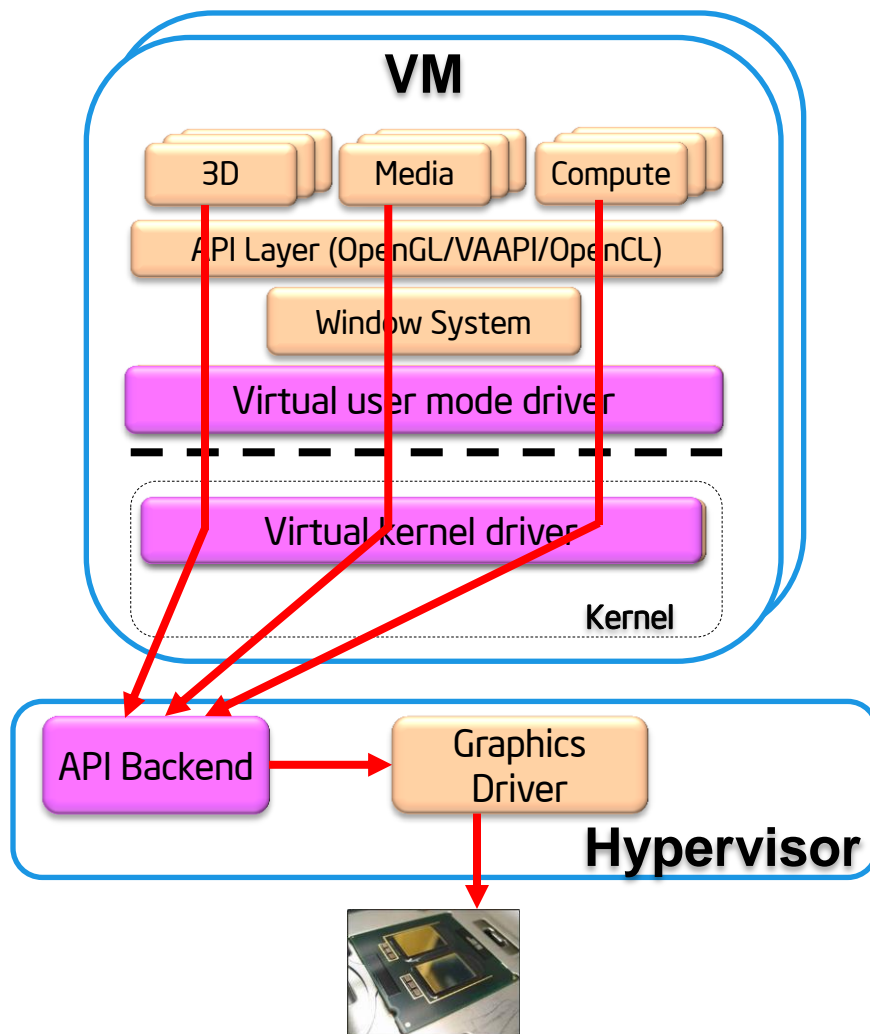


# Virtualization in API Level



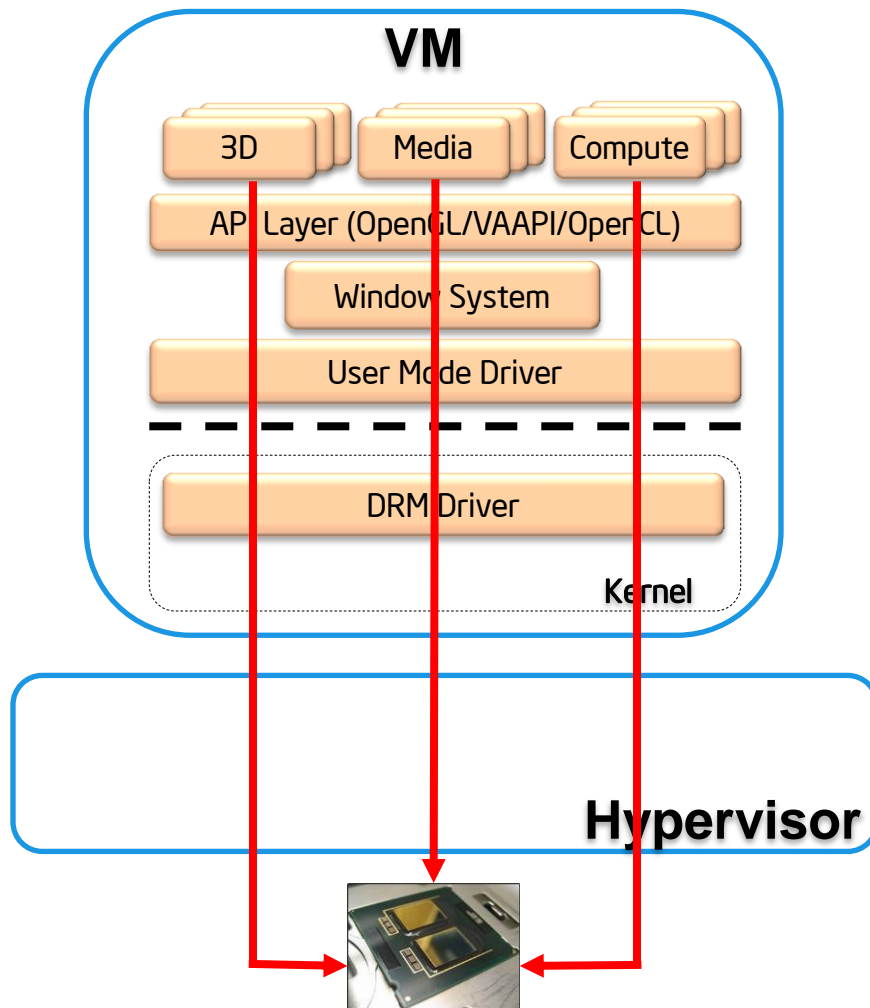
- API forwarding between frontend and backend driver
- Gain GPU acceleration capability in VMs
- However, lagging features and API compatibility

# Virtualization in Driver Level



- Enhance VGA device model with GPU resources
- New user mode and kernel drivers for the specific virtual device, but still like API forwarding
- So, lagging features and API compatibility too

# Virtualization in Device Level



- Assign GPU to a specific VM
- Near-native performance
- Full features
- However, no sharing capability

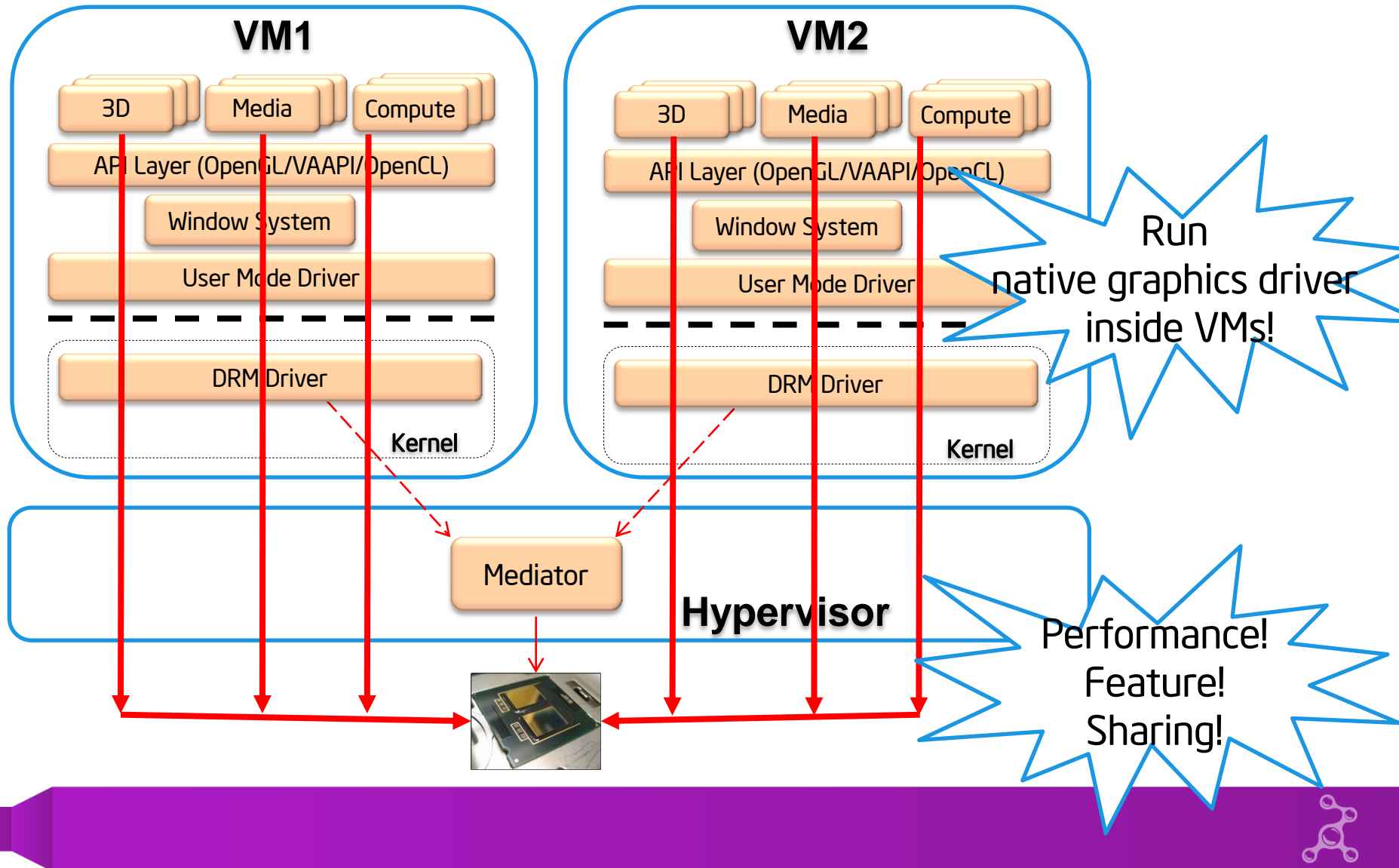
**So... none of above techniques can meet all requirements of GPU Virtualization:**

**Performance**

**Feature**

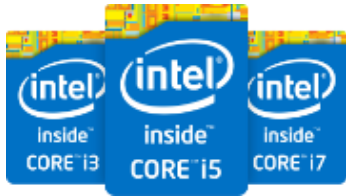
**Sharing**

# Here Comes Full GPU Virtualization!





# XenGT Target: Intel® Processor Graphics



22nm

Build-in into 4<sup>th</sup> generation Intel® Core™ processors



GT3

2x computational shader power with new GT3 - Intel® Iris™



EDRAM

128MB fast cache for bandwidth saving with GT3e - Intel® Iris™ Pro



QSV

High Speed Video Decode & Encode H.264/MPEG-4 AVC, VC-1



# Agenda

- Why GPU Virtualization
- How GPU Virtualization Works
- How XenGT Works
- Demo
- Current Status
- Summary



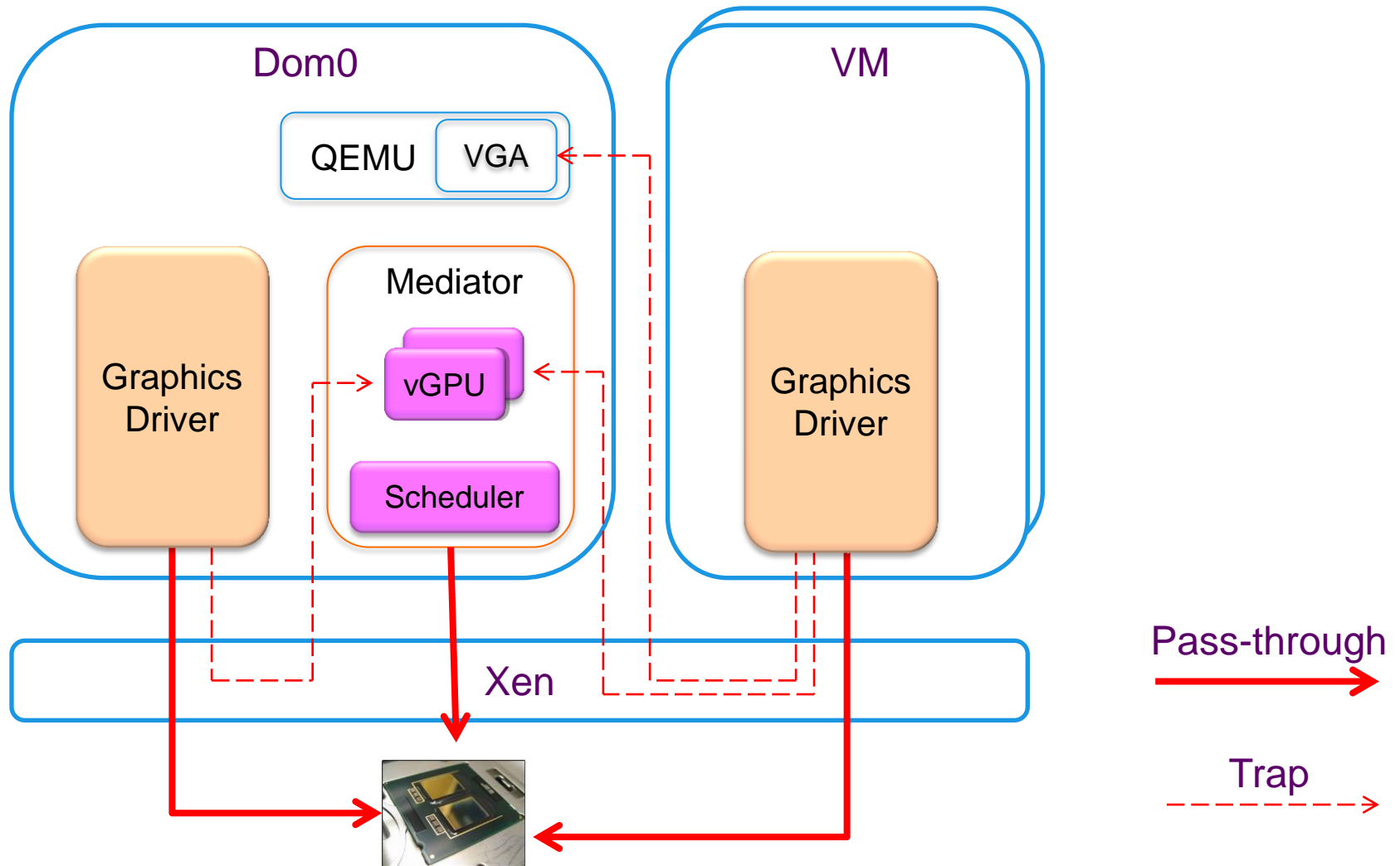
# XenGT: A Full GPU Virtualization Solution



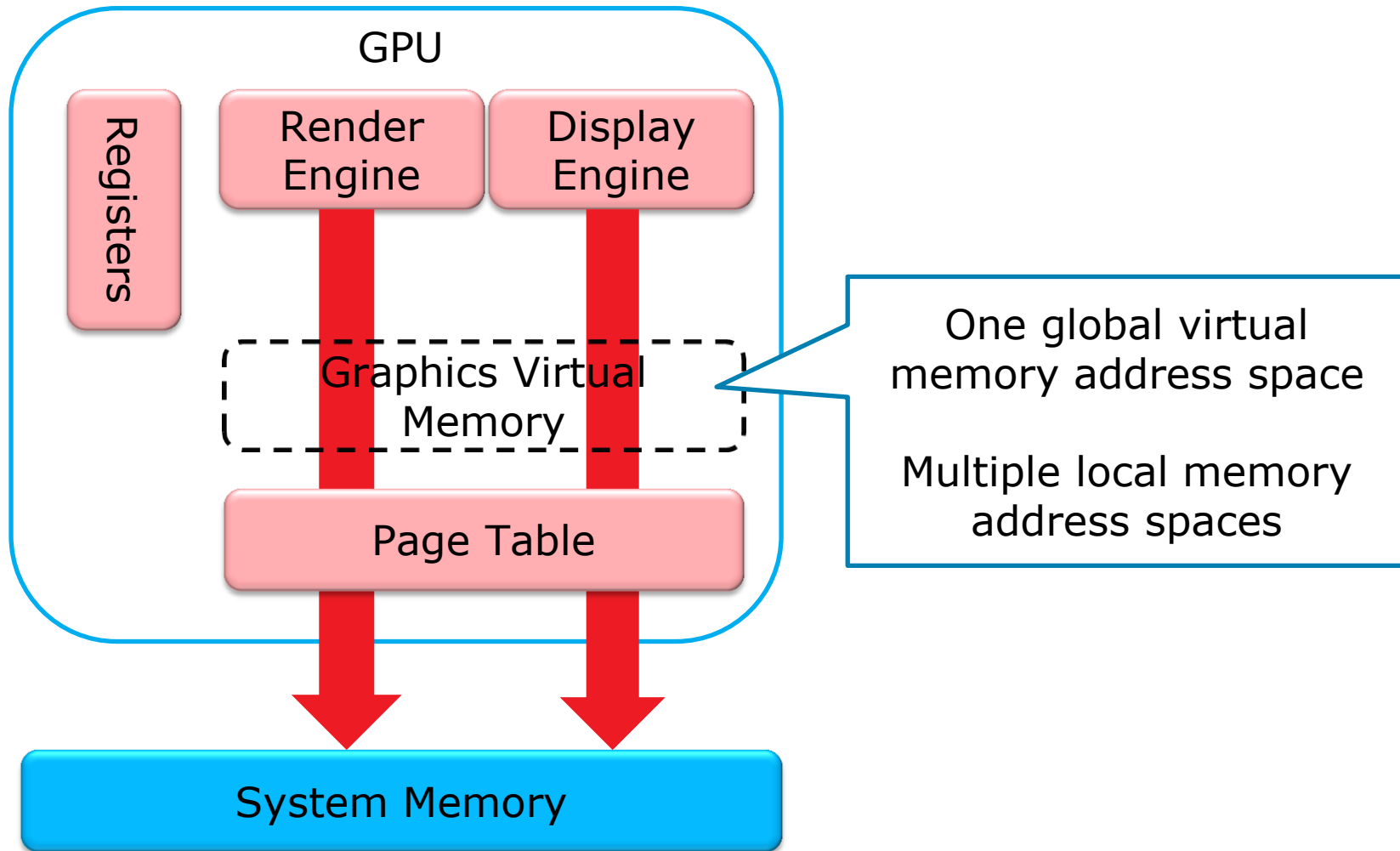
- Built on a mediated pass-through framework
  - Privileged I/O operations are trap-and-emulated
  - Performance critical operations are passed through
- Virtual GPU (vGPU) device model
  - Equivalent features as physical Intel Processor Graphics
- Running native graphics driver inside VMs
  - Leverage existing driver optimizations and stability fixes
- First implementation on Xen hypervisor
  - Core device model reusable in other hypervisors



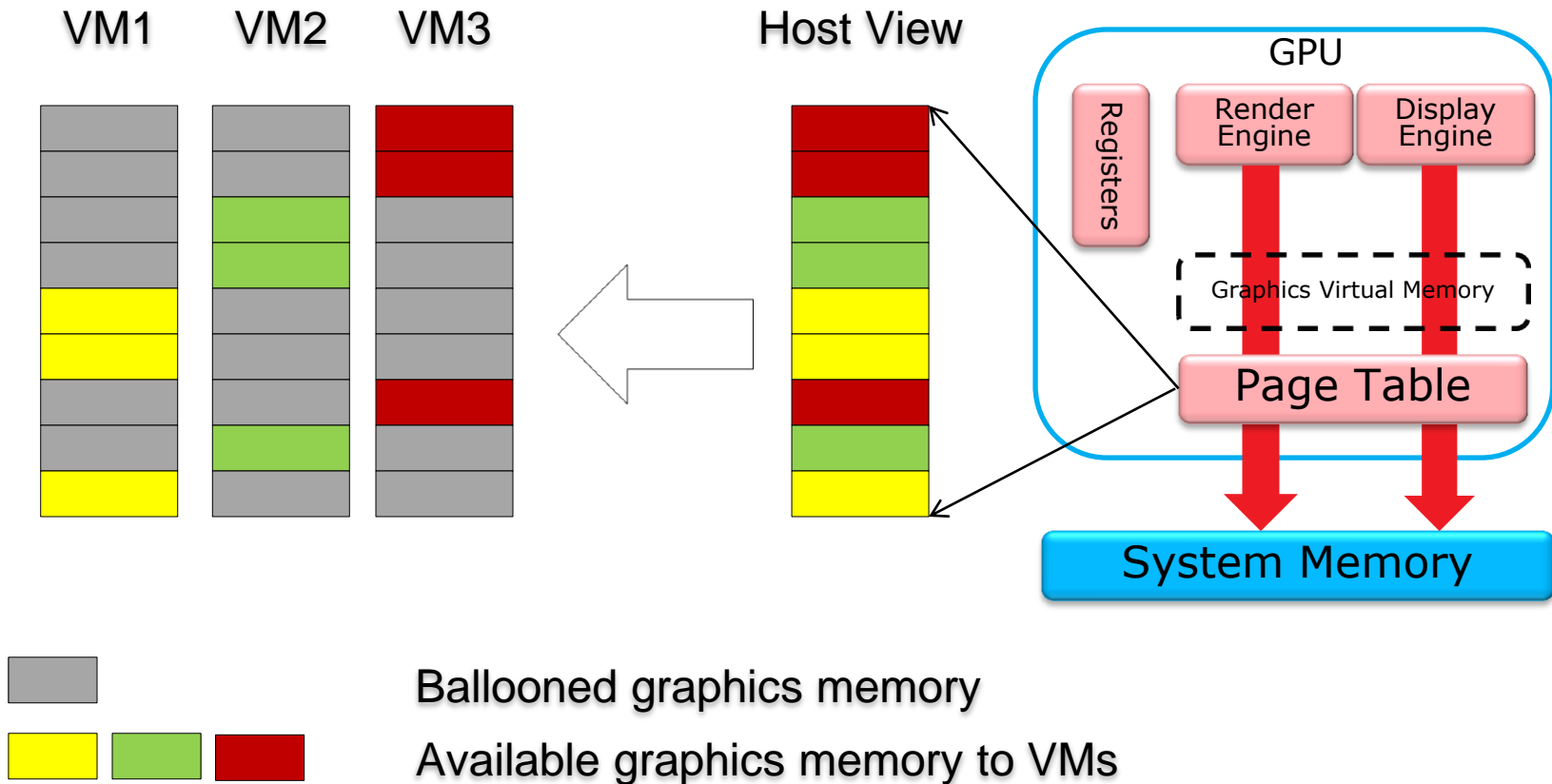
# XenGT Architecture



# Intel® Processor Graphics Architecture



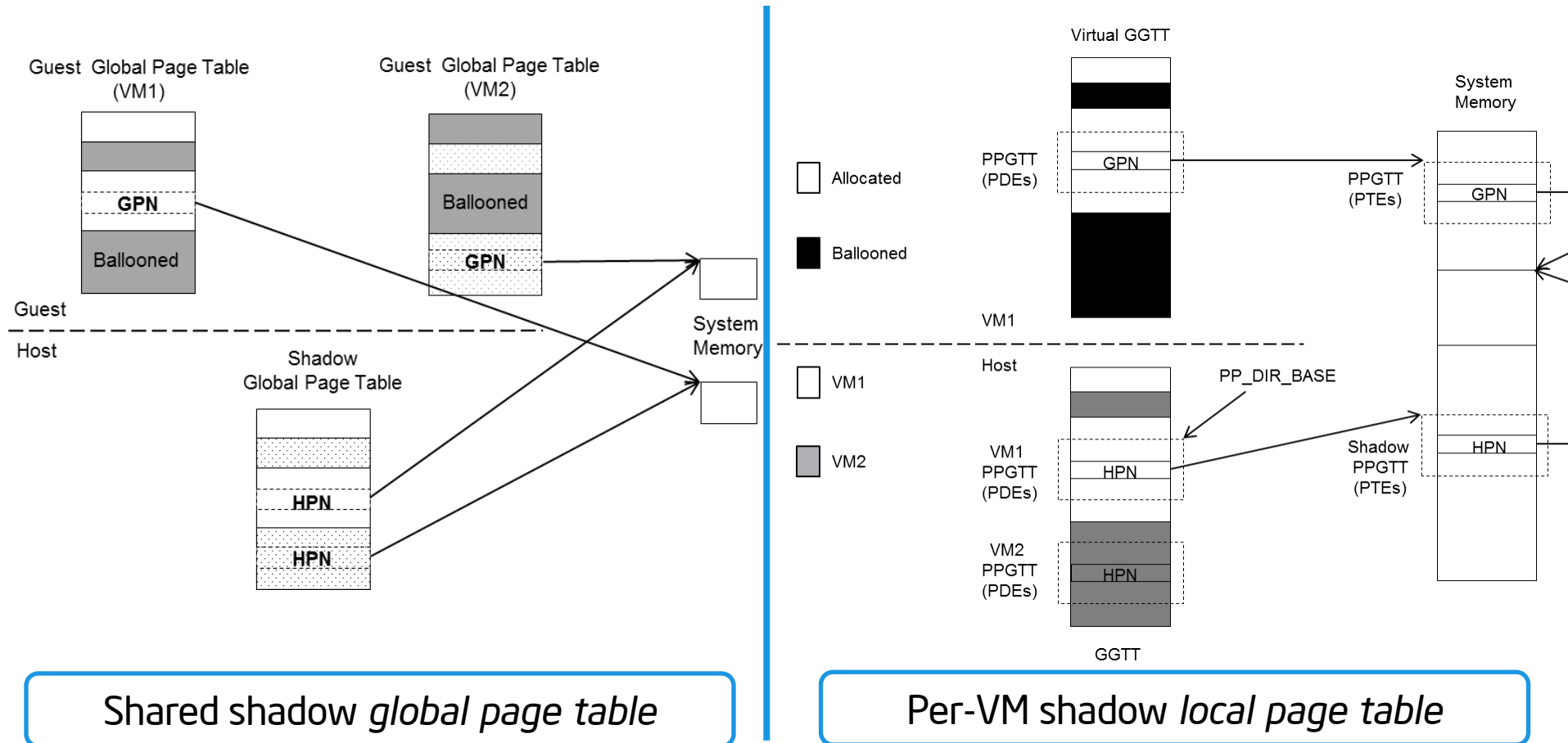
# XenGT Graphics Memory Partition



# GPU Page Table Virtualization



- Shadow GPU page tables (GPN<->HPN)



Shared shadow *global page table*

Per-VM shadow *local page table*



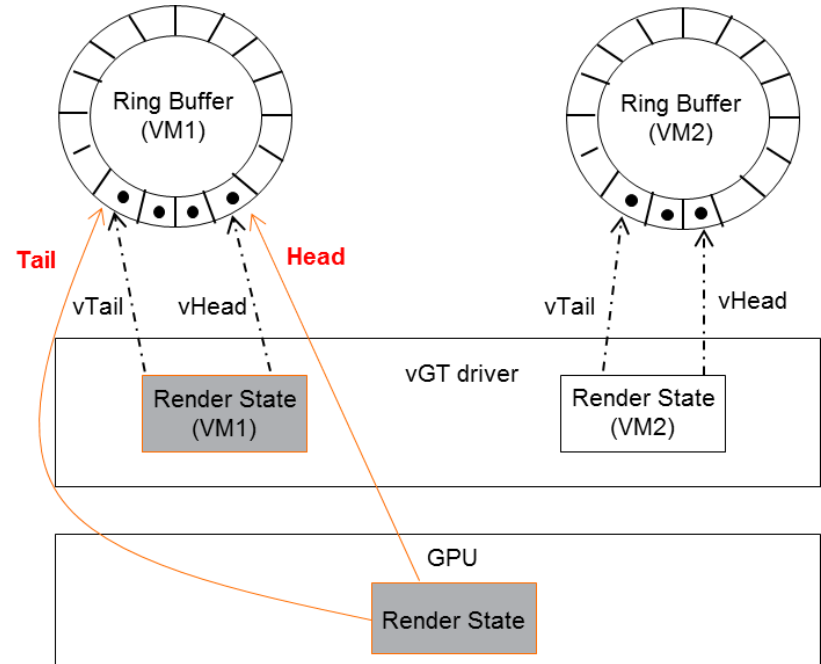
# Render Engine Virtualization

- A simple round-robin scheduler
  - In 16ms epoch
- Render owner access is trap-and-forwarded to the render engine
- Non-render owner access is trap-and-emulated



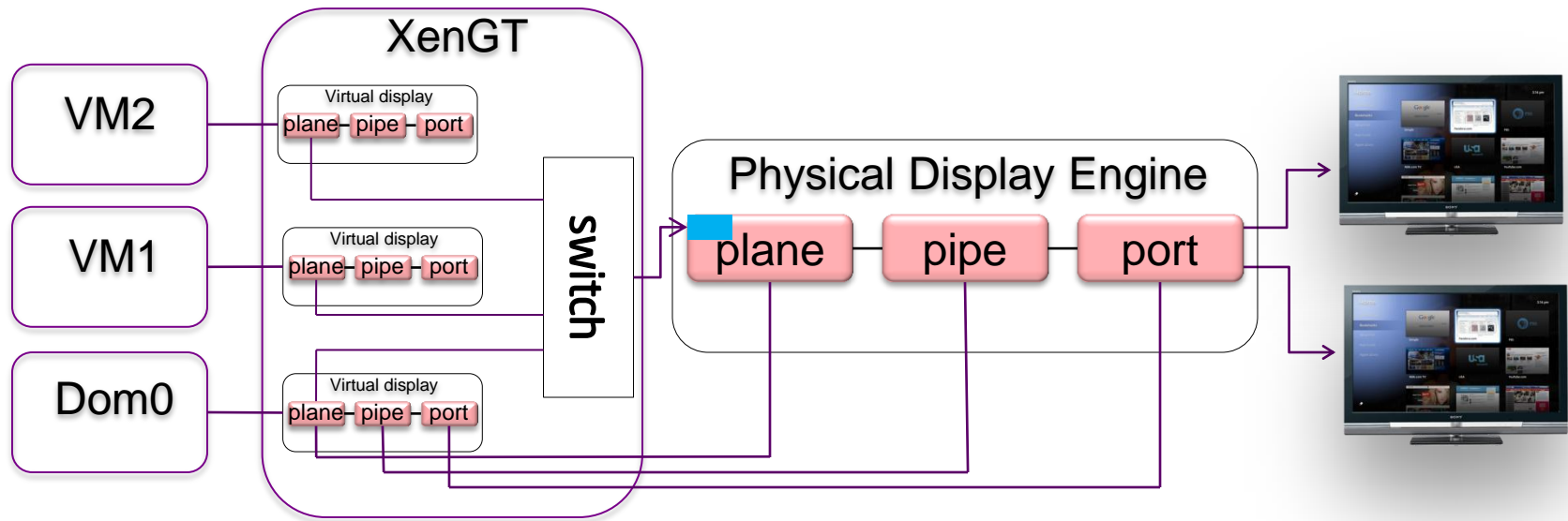
## Render context switch flow

1. Wait VM1 ring buffer becoming empty
2. Save render MMIO registers for VM1
3. Flush internal TLB/caches
4. Hardware context switch
5. Restore render MMIO registers for VM2
6. Submit previously queued commands





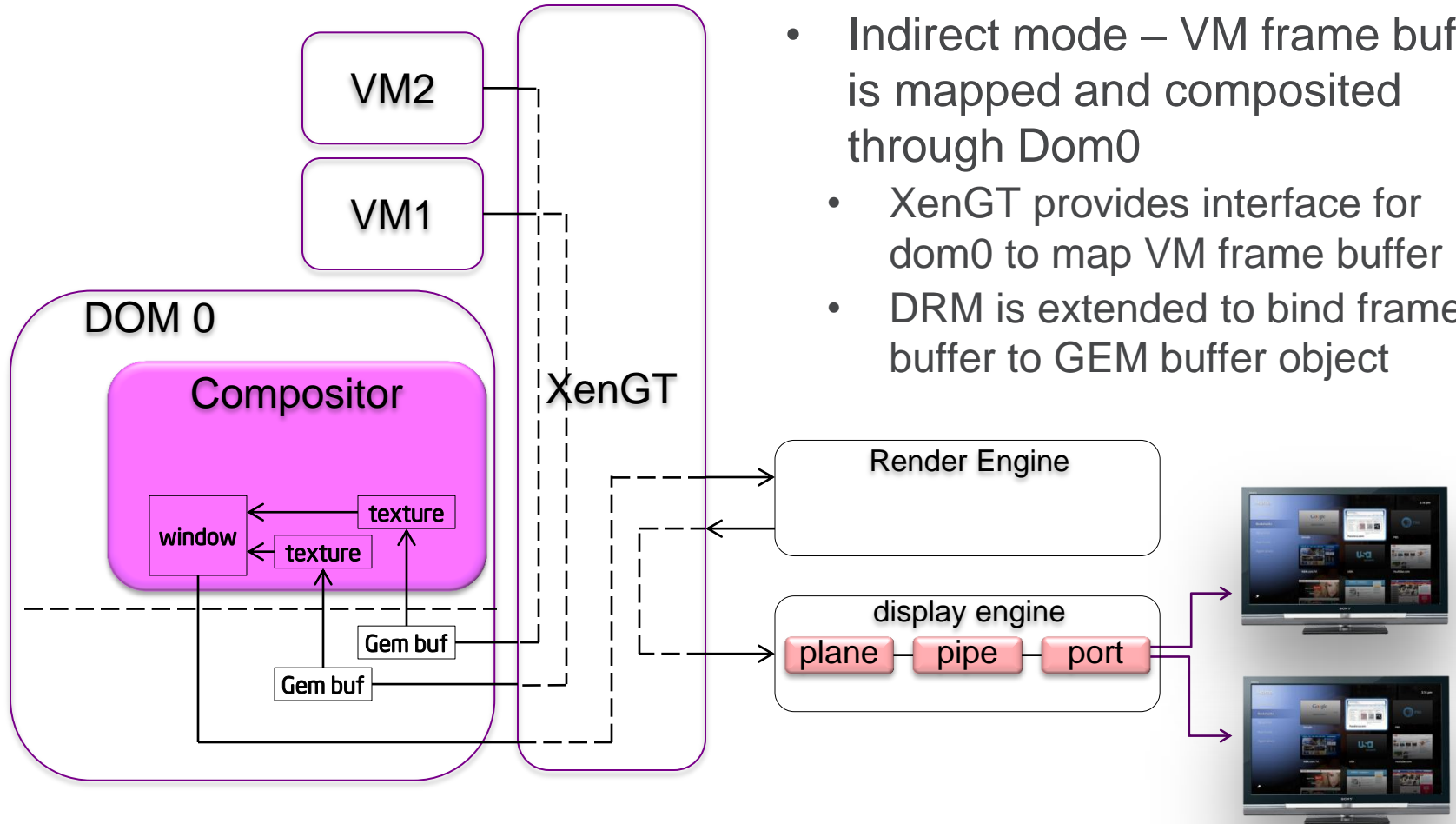
# Display Virtualization – Direct Mode



- Direct mode – directly program VM frame buffer to display engine
- Display Switch
  - Simple as frame buffer flip
  - Panel fitting for VMs with different resolutions



# Display Virtualization – Indirect Mode



# Agenda

- Why GPU Virtualization
- How GPU Virtualization Works
- How XenGT Works
- Demo
- Current Status
- Summary



# Agenda

- Why GPU Virtualization
- How GPU Virtualization Works
- How XenGT Works
- Demo
- Current Status
- Summary

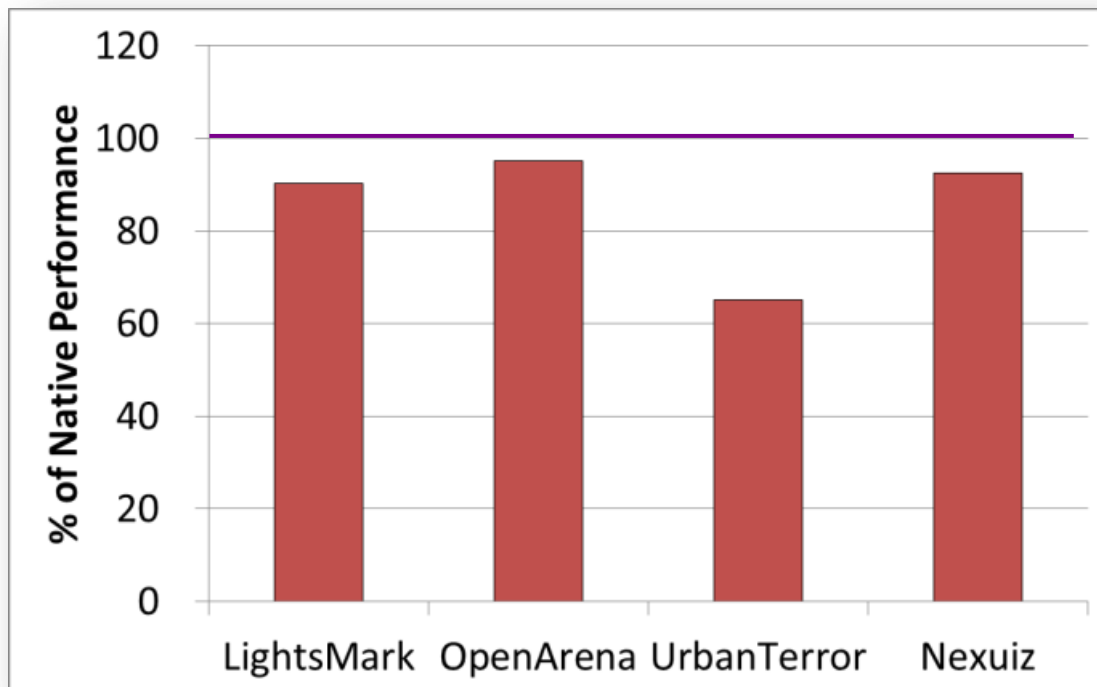


# Current Status

- Initial release in 09/2013, including all major features
  - Multiple VMs (Dom0 + 3 VMs) running simultaneously
  - Display switch among VMs
- Second release in 03/2014
  - More stability improvement
  - Support guest resolution changes
  - Enhanced support for multiple display and hotplug
  - Preliminary support for GPU recovery
- More developing/tuning work ongoing



# Performance



Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information about performance and benchmark results, visit [www.intel.com/benchmarks](http://www.intel.com/benchmarks)



# Agenda

- Why GPU Virtualization
- How GPU Virtualization Works
- How XenGT Works
- Demo
- Current Status
- Summary



# Summary



- GPU virtualization gains momentum
- Full GPU virtualization provides a good balance among performance, feature and sharing capability
- XenGT is a full GPU virtualization solution, on Intel® Processor Graphics, running native graphics driver inside VMs
- Call for action - try and feedback
  - <https://github.com/01org/XenGT-Preview-kernel>
  - <https://github.com/01org/XenGT-Preview-xen>
  - <https://github.com/01org/XenGT-Preview-qemu>





# Notices and Disclaimers



INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL® PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. INTEL PRODUCTS ARE NOT INTENDED FOR USE IN MEDICAL, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS.

Intel may make changes to specifications and product descriptions at any time, without notice.

All products, dates, and figures specified are preliminary based on current expectations, and are subject to change without notice.

Intel, processors, chipsets, and desktop boards may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

No computer system can provide absolute security under all conditions. Intel® Trusted Execution Technology (Intel® TXT) requires a computer with Intel® Virtualization Technology, an Intel TXT-enabled processor, chipset, BIOS, Authenticated Code Modules and an Intel TXT-compatible measured launched environment (MLE). Intel TXT also requires the system to contain a TPM v1.s. For more information, visit <http://www.intel.com/technology/security>

Intel, Intel logo, Xeon, and Xeon Inside are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

\*Other names and brands may be claimed as the property of others.

Copyright © 2014 Intel Corporation. All rights reserved.

