XenGT: a Software Based Intel Graphics Virtualization Solution

Oct 22, 2013

Haitao Shan, haitao.shan@intel.com

Kevin Tian, kevin.tian@intel.com

Eddie Dong, eddie.dong@intel.com

David Cowperthwaite, david.j.cowperthwaite@intel.com
Agenda

• Background
• Existing Arts
• XenGT Architecture
• Performance
• Summary
Background
Graphics Computing

• Entertainment applications
  • Gaming, video playback, browser, etc.

• General purpose windowing
  • Windows Aero, Compiz Fusion, etc

• High performance computing
  • Computer aided designs, weather broadcast, etc.

Same capability required, when above tasks are moved into VM
Graphics Virtualization

- Performance vs. multiplexing
  - Consistent and rich user experience in all VMs
  - Share a single GPU among multiple VMs

<table>
<thead>
<tr>
<th>Client</th>
<th>Rich Virtual Client</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server</td>
<td>VDI, transcoder, GPGPU</td>
</tr>
<tr>
<td>Embedded</td>
<td>Smartphone, tablet, IVI</td>
</tr>
</tbody>
</table>
Existing Arts
Device Emulation

- Only for legacy VGA cards
  - E.g. Cirrus logic VGA card

- Limited graphics capability
  - 2D only
  - Optimizations on frame buffer operations
    - E.g. PV framebuffer

- Impossible to emulate a modern GPU
  - Complexity
  - Poor performance
Split Driver Model

- Frontend/Backend drivers
  - Forward OpenGL/DirectX API calls
  - Implementation specific for the level of forwarding
  - E.g. VMGL, VMware vGPU, Virgil

- Hardware agnostic

- Challenges on forwarding between host/guest graphics stacks
  - API compatibility
  - CPU overhead
Direct Pass-Through/SR-IOV

- Best performance with direct pass-through
  - However no multiplexing
XenGT Architecture
XenGT

- A mediated pass-through solution for graphics virtualization
  - Pass-through performance critical resources
  - Trap-and-emulate privileged operations
    - Maintain a device model per VM

- Run native graphics driver in VM

- Achieve good performance and moderate multiplexing capability

```
Device Emulation
Split Driver Model
Mediated Pass-Through
Direct Pass-Through
Performance
Multiplexing
```
XenGT Architecture

*Dom0 Gfx driver are de-privileged.
Intel Processor Graphics

- **Graphics memory**
  - Virtual memory address spaces
    - A single global virtual memory (GVM) space
    - Multiple per-process virtual memory (PPVM) spaces
  - Backed by system memory through GTTs

- **Render engine**
  - Fulfill the acceleration capability through fixed pipelines and execution units

- **Display engine**
  - Route data from graphics memory to external monitors

- **Global state**
  - Represent remaining circuits, including initialization, PM, etc.
Mediated Pass-Through Policies

- Access frequency on GPU interfaces

![Graph showing access frequency on GPU interfaces for OpenArena and UrbanTerror.]

- Policies

**Pass-through**

- Graphics Virtual Memory Spaces
- Command Buffers

**Mediation**

- MMIO registers
- GTTs
- PCI configuration space
- Legacy VGA I/O ports
Global Virtual Memory Space

- The single GVM space is partitioned
  - Access to VM’s own GVM region is **passed through**
  - Classical memory virtualization challenge
    - Host view vs. guest view
  - Address space ballooning with driver cooperation

- GGTT accesses are **mediated**
  - Access to its own GGTT entries is translated
    - GPFN <-> MFN
  - Access to others’ entries is virtualized
Per-Process Virtual Memory Spaces

- Each VM manages its own PPVM spaces
  - Active space pointed by PP_DIR_BASE
  - Accesses are passed through

- PPGTT accesses are write-protected
  - Shadow PPGTT table
  - Switch PP_DIR_BASE at render context switch
Command Buffers

- Command buffer access is **passed through**
  - Reside in virtual memory spaces

- Command submission request is **mediated**
  - Through MMIO register (ring tail)
  - Render scheduler makes the decision
    - Render owner request is submitted to render engine
    - Non-render owner request is blocked
Render Engine Sharing

- 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 Engine Sharing

**Direct display model**
- Display engine points to the frame buffer of the foreground VM
- vGT driver configures display engine for foreground/background switch

**Indirect display model**
- vGT driver provides interface to decode VM frame buffer location/format
- An OpenGL app composites VM frame buffers
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
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
Summary

• Sustain consistent and rich user experience in VM
  • Running native graphics driver in VM

• Achieve good performance
  • Minimum impact on performance critical operations

• Support moderate multiplexing capability
  • Trap-and-emulate privileged operations

• 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
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 © 2013 Intel Corporation. All rights reserved.