Heterogeneous Isolated Execution for Commodity GPUs

Insu Jang\textsuperscript{1}, Adrian Tang\textsuperscript{2}, Taehoon Kim\textsuperscript{1}, Simha Sethumadhavan\textsuperscript{2}, and Jaehyuk Huh\textsuperscript{1}

\textsuperscript{1} KAIST, School of Computing
\textsuperscript{2} Columbia University, Department of Computer Science
Architecture Trend: Heterogeneous Computing

• Heterogeneous computing is emerging (GPUs, FPGAs, etc)

- Machine Learning
- Image Processing
- Complex Calculations

\[ \int \frac{dy}{dx} \]
Architecture Trend: Heterogeneous Computing

- Heterogeneous computing is emerging (GPUs, FPGAs, etc)

- Problem: lack of trusted execution environment in devices
Architecture Trend: Heterogeneous Computing

- Heterogeneous computing is emerging (GPUs, FPGAs, etc)

- Problem: lack of trusted execution environment in devices

Machine Learning

Image Processing

Complex Calculations

Untrusted Kernel Space

Device Driver
Architecture Trend: Heterogeneous Computing

- Heterogeneous computing is emerging (GPUs, FPGAs, etc)

- Problem: lack of trusted execution environment in devices
Architecture Trend: Heterogeneous Computing

- Heterogeneous computing is emerging (GPUs, FPGAs, etc)

- Problem: lack of trusted execution environment in devices
Architecture Trend: Heterogeneous Computing

- Heterogeneous computing is emerging (GPUs, FPGAs, etc)

- Problem: lack of trusted execution environment in devices
Architecture Trend: Heterogeneous Computing

- Heterogeneous computing is emerging (GPUs, FPGAs, etc)

- Problem: lack of trusted execution environment in devices
Architecture Trend: Heterogeneous Computing

- Heterogeneous computing is emerging (GPUs, FPGAs, etc)

- Problem: lack of trusted execution environment in devices
Architecture Trend: Heterogeneous Computing

- Heterogeneous computing is emerging (GPUs, FPGAs, etc)

- Problem: lack of trusted execution environment in devices
Architecture Trend: Heterogeneous Computing

- Heterogeneous computing is emerging (GPUs, FPGAs, etc)

- Problem: lack of trusted execution environment in devices
Architecture Trend: Heterogeneous Computing

- Heterogeneous computing is emerging (GPUs, FPGAs, etc)

- Problem: lack of trusted execution environment in devices
How to Provide TEE to Devices?

- Problem: lack of trusted execution environment in devices
How to Provide TEE to Devices?

• Problem: lack of trusted execution environment in devices

• Existing works regarding TEE for peripheral devices
  – SGXIO [Weiser, CODASPY’17]: use a trusted hypervisor
  – Graviton [Volos, OSDI’18]: use a modified GPU with a root of trust
How to Provide TEE to Devices?

- **Problem:** lack of trusted execution environment in devices
- **Existing works regarding TEE for peripheral devices**
  - **SGXIO** [Weiser, CODASPY’17]: use a trusted hypervisor
  - **Graviton** [Volos, OSDI’18]: use a modified GPU with a root of trust
How to Provide TEE to Devices?

• Problem: lack of trusted execution environment in devices

• Existing works regarding TEE for peripheral devices

  – SGXIO [Weiser, CODASPY’17]: use a trusted hypervisor

  – Graviton [Volos, OSDI’18]: use a modified GPU with a root of trust
Our Approach: Securing I/O Path

- All device I/O accesses from software are handled by CPU

* x86 architecture based
Our Approach: Securing I/O Path

- All device I/O accesses from software are handled by CPU

* x86 architecture based
Our Approach: Securing I/O Path

- All device I/O accesses from software are handled by CPU
Our Approach: Securing I/O Path

- All device I/O accesses from software are handled by CPU

* x86 architecture based
Our Approach: Securing I/O Path

- All device I/O accesses from software are handled by CPU

* x86 architecture based
Our Approach: Securing I/O Path

- All device I/O accesses from software are **handled by CPU**

* x86 architecture based
Our Approach: Securing I/O Path

- All device I/O accesses from software are **handled by CPU**

* x86 architecture based
Our Approach: Securing I/O Path

- All device I/O accesses from software are handled by CPU

* x86 architecture based
Our Approach: Securing I/O Path

- All device I/O accesses from software are handled by CPU

Idea: Prevent I/O from Attackers by Securing I/O Path!
HIX: Heterogeneous Isolated Execution

PCI Express (PCIe) Interconnect Architecture
HIX: Heterogeneous Isolated Execution

- Implementation based on Intel SGX (basic TEE necessary)
- Extend TEE to I/O path (from SGX enclave to the device)
HIX: Heterogeneous Isolated Execution

- Implementation based on Intel SGX (basic TEE necessary)
- Extend TEE to I/O path (from SGX enclave to the device)
HIX: Heterogeneous Isolated Execution

- Implementation based on Intel SGX (basic TEE necessary)
- Extend TEE to I/O path (from SGX enclave to the device)
HIX: Heterogeneous Isolated Execution

- Implementation based on Intel SGX (basic TEE necessary)
- Extend TEE to I/O path (from SGX enclave to the device)
HIX: Heterogeneous Isolated Execution

- Implementation based on Intel SGX (basic TEE necessary)
- Extend TEE to I/O path (from SGX enclave to the device)

Memory-mapped I/O (MMIO)

Trusted Enclave

Untrusted Process

Trusted Device Driver

PCIe Root Complex

CPU

GPU

Protection scope by Intel SGX

Extended Protection scope by HIX

Only accessible by the trusted device driver
HIX: Heterogeneous Isolated Execution

- Implementation based on Intel SGX (basic TEE necessary)
- **Extend TEE to I/O path (from SGX enclave to the device)**

![Diagram showing HIX architecture]

- **Protection scope by Intel SGX**
- **Extended Protection scope by HIX**
- Only accessible by the trusted device driver
Contributions and Threat Model
Contributions and Threat Model

• Provide **confidentiality and integrity** to user data in GPU

• **No GPU modifications** are required
  – Provide GPU TEE by securing I/O path
  – No protection against physical attacks; software based attacks prevented

• **Threat Model**
  – Attackers have all privileged permission on software level
  – Not consider physical attacks on any hardware
  – Protect the system from **privileged software attacks**
HIX Architecture

- Trusted GPU Device Driver: GPU Enclave
- MMIO Protection
- Inter-Enclave Communication
HIX: Architecture Overview

Three communication paths to be protected:

- User Enclave
- GPU Enclave
- Memory-mapped I/O (MMIO)
- HIX-enabled CPU
- GPU

Untrusted Process
HIX: Architecture Overview

- Three communication paths to be protected

• Inter-enclave communication
• Trusted GPU device driver
• MMIO access validation
• Guaranteed PCIe packet routing
HIX: Architecture Overview

Three communication paths to be protected:

- Inter-enclave communication
- Trusted GPU device driver
- MMIO access validation
- Guaranteed PCIe packet routing

Diagram:
- User Enclave
- GPU Enclave
- Memory-mapped I/O (MMIO)
- HIX-enabled CPU
- GPU
- Untrusted Process
HIX: Architecture Overview

- Three communication paths to be protected

- Inter-enclave communication
- Trusted GPU device driver
- MMIO access validation
- Guaranteed PCIe packet routing
HIX: Architecture Overview

- Three communication paths to be protected

  - Inter-enclave communication
  - Trusted GPU device driver
  - MMIO access validation
  - Guaranteed PCIe packet routing

User Enclave → GPU Enclave → Memory-mapped I/O (MMIO) → HIX-enabled CPU → GPU
HIX: Architecture Overview

Three communication paths to be protected

- Inter-enclave communication
- Trusted GPU device driver
- MMIO access validation
- Guaranteed PCIe packet routing
HIX: Architecture Overview

Three communication paths to be protected

- Inter-enclave communication
- Trusted GPU device driver
- MMIO access validation
- Guaranteed PCIe packet routing
HIX: Architecture Overview

Three communication paths to be protected:

- Inter-enclave communication
- Trusted GPU device driver
- MMIO access validation
- Guaranteed PCIe packet routing
HIX: Architecture Overview

- Three communication paths to be protected

- Inter-enclave communication
- Trusted GPU device driver
- MMIO access validation
- Guaranteed PCIe packet routing
HIX: Architecture Overview

- Inter-enclave communication
- Trusted GPU device driver
- MMIO access validation
- Guaranteed PCIe packet routing
HIX: Architecture Overview

- Three communication paths to be protected

- Inter-enclave communication
- Trusted GPU device driver
- MMIO access validation
- Guaranteed PCIe packet routing
HIX: Architecture Overview

Three communication paths to be protected

- Inter-enclave communication
- Trusted GPU device driver
- MMIO access validation
- Guaranteed PCIe packet routing
HIX: Architecture Overview

- Three communication paths to be protected

- Inter-enclave communication
- Trusted GPU device driver
- MMIO access validation
- Guaranteed PCIe packet routing
HIX: Architecture Overview

Three communication paths to be protected

- Inter-enclave communication
- Trusted GPU device driver
- MMIO access validation
- Guaranteed PCIe packet routing
GPU Enclave: Trusted Device Driver

- Move device driver from untrusted kernel space to trusted enclave
- Extended SGX enclave that owns and controls GPU in TEE

Untrusted Kernel Space | User Space
---|---
Kernel | GPU Enclave Process

GPU Enclave

Trusted Device Driver

Exclusively access to GPU in the system through MMIO
MMIO Access Validation

- **Exclusively** access to GPU in the system through MMIO: **How?**
MMIO Access Validation

- **Exclusively** access to GPU in the system through MMIO: How?
- Extend SGX EPC access validation mechanism for MMIO
  - Validate *address translation information* during TLB misses
MMIO Access Validation

- **Exclusively** access to GPU in the system through MMIO: How?
- Extend SGX EPC access validation mechanism for MMIO
  - Validate *address translation information* during TLB misses
MMIO Access Validation

• **Exclusively** access to GPU in the system through MMIO: **How?**

• Extend SGX EPC access validation mechanism for MMIO
  – Validate *address translation information* during **TLB misses**
**MMIO Access Validation**

- **Exclusively** access to GPU in the system through MMIO: *How?*

- Extend SGX EPC access validation mechanism for MMIO
  - Validate *address translation information* during *TLB misses*
MMIO Access Validation

• **Exclusively** access to GPU in the system through MMIO: How?

• Extend SGX EPC access validation mechanism for MMIO
  – Validate *address translation information* during **TLB misses**

Virtual Address Space

Physical Address Space
MMIO Access Validation

• **Exclusively** access to GPU in the system through MMIO: How?

• Extend SGX EPC access validation mechanism for MMIO
  – Validate *address translation information* during TLB misses

Virtual Address Space

Physical Address Space

GPU

DRAM
MMIO Access Validation

- **Exclusively** access to GPU in the system through MMIO: How?
- Extend SGX EPC access validation mechanism for MMIO
  - Validate *address translation information* during TLB misses

![Diagram showing virtual and physical address spaces](image-url)
MMIO Access Validation

• Exclusively access to GPU in the system through MMIO: How?
• Extend SGX EPC access validation mechanism for MMIO
  – Validate *address translation information* during TLB misses

![Diagram showing virtual and physical address spaces]
MMIO Access Validation

- **Exclusively** access to GPU in the system through MMIO: **How?**
- Extend SGX EPC access validation mechanism for MMIO
  - Validate *address translation information* during **TLB misses**

---

Virtual Address Space

Physical Address Space

- **GPU**
- **DRAM**
- **Main Memory**
MMIO Access Validation

- **Exclusively** access to GPU in the system through MMIO: How?
- Extend SGX EPC access validation mechanism for MMIO
  - Validate *address translation information* during *TLB misses*
MMIO Access Validation

- **Exclusively** access to GPU in the system through MMIO: How?
- Extend SGX EPC access validation mechanism for MMIO
  - Validate *address translation information* during TLB misses

Virtual Address Space

Physical Address Space

```
<table>
<thead>
<tr>
<th>Virtual Address Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMIO VA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Physical Address Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMIO PA</td>
</tr>
<tr>
<td>Main Memory</td>
</tr>
<tr>
<td>GPU</td>
</tr>
<tr>
<td>DRAM</td>
</tr>
</tbody>
</table>
```
MMIO Access Validation

- **Exclusively** access to GPU in the system through MMIO: **How?**

- Extend SGX EPC access validation mechanism for MMIO
  - Validate *address translation information* during **TLB misses**
MMIO Access Validation

- **Exclusively** access to GPU in the system through MMIO: How?
- Extend SGX EPC access validation mechanism for MMIO
  - Validate *address translation information* during TLB misses
MMIO Access Validation

- **Exclusively** access to GPU in the system through MMIO: **How?**
  - Extend SGX EPC access validation mechanism for MMIO
    - Validate *address translation information* during **TLB misses**
**MMIO Access Validation**

- **Exclusively** access to GPU in the system through MMIO: How?
- Extend SGX EPC access validation mechanism for MMIO
  - Validate *address translation information* during TLB misses

![Diagram showing GPU Enclave, Virtual Address Space, Physical Address Space, MMIO VA, MMIO PA, Main Memory, GPU, DRAM]
MMIO Access Validation

- **Exclusively** access to GPU in the system through MMIO: How?
- Extend SGX EPC access validation mechanism for MMIO
  - Validate *address translation information* during TLB misses

![Diagram showing MMIO access validation]

**Virtual Address Space**
- MMIO VA
- Text, data, etc

**Physical Address Space**
- MMIO PA
- Main Memory

**GPU Enclave**

**GPU**

**DRAM**
MMIO Access Validation

- **Exclusively** access to GPU in the system through MMIO: **How?**
- Extend SGX EPC access validation mechanism for MMIO
  - Validate *address translation information* during **TLB misses**
MMIO Access Validation

- **Exclusively** access to GPU in the system through MMIO: How?
- Extend SGX EPC access validation mechanism for MMIO
  - Validate *address translation information* during **TLB misses**

![Diagram of MMIO Access Validation]

- **GPU Enclave**
- **Virtual Address Space**
  - MMIO VA
  - Text, data, etc
- **Physical Address Space**
  - MMIO PA
  - Main Memory
- **TLB**
- **0x24f3**
- **GPU**
- **DRAM**

*Diagram shows the interaction between virtual and physical address spaces, highlighting the process of accessing memory through MMIO.*
**MMIO Access Validation**

- **Exclusively** access to GPU in the system through MMIO: How?
- Extend SGX EPC access validation mechanism for MMIO
  - Validate **address translation information** during **TLB misses**
MMIO Access Validation

- **Exclusively** access to GPU in the system through MMIO: How?
- Extend SGX EPC access validation mechanism for MMIO
  - Validate **address translation information** during **TLB misses**

![Diagram](image_url)

- MMIO VA: MMIO Virtual Address
- MMIO PA: MMIO Physical Address
- Text, data, etc
- Main Memory
- GPU
- DRAM
- TLB
- TLB MISS
- 0x24f3

Virtual Address Space

Physical Address Space
MMIO Access Validation

- **Exclusively** access to GPU in the system through MMIO: How?
- Extend SGX EPC access validation mechanism for MMIO
  - Validate *address translation information* during TLB misses
MMIO Access Validation

- **Exclusively** access to GPU in the system through MMIO: How?

- Extend SGX EPC access validation mechanism for MMIO
  - Validate *address translation information* during TLB misses

![Diagram showing MMIO access validation process]

- Validation before adding PTE into TLB
  - PTE not corrupted?
  - Process is the GPU enclave?
PCIe Packet Routing

Virtual Address Space

MMIO VA

Secured by MMIO access validation

Physical Address Space

MMIO PA

GPU Enclave

GPU
PCIe Packet Routing

Virtual Address Space

| MMIO VA |

Physical Address Space

| MMIO PA |

Secured by MMIO access validation

Packet guaranteed to be routed to GPU??

GPU Enclave

GPU
PCle Packet Routing

Virtual Address Space

MMIO VA

Secured by MMIO access validation

Physical Address Space

MMIO PA

Packet guaranteed to be routed to GPU??

GPU Enclave

Untrusted Device

GPU
PCIe Packet Routing

- **GPU Enclave**
  - **Virtual Address Space**: MMIO VA
    - Secured by **MMIO access validation**
  - **Physical Address Space**: MMIO PA
    - **Packet guaranteed to be routed to GPU??**

- **Untrusted Device**
- **GPU**
PCIe Packet Routing: Introduction

- Use PCIe hardware registers for packet routing (e.g. BARs*)

* BAR: Base Address Register
PCle Packet Routing: Introduction

- Use PCle hardware registers for packet routing (e.g. BARs*)

I/O write: 0x1 @ 0x1042

* BAR: Base Address Register
PCle Packet Routing: Introduction

- Use PCle hardware registers for packet routing (e.g. BARs*)

I/O write: 0x1 @ 0x1042

CPU

PCle Root Complex

Switch

NVMe SSD

Other Device...

GPU

Registers used for routing

0x0100 ~ 0x01af

0x0500 ~ 0x05ff

0x0f00 ~ 0x0f4f

0x1000 ~ 0x1fff

* BAR: Base Address Register
PCle Packet Routing: Introduction

- Use PCle hardware registers for packet routing (e.g. BARs*)

I/O write: 0x1 @ 0x1042

0x0100 ~ 0x01af
0x0500 ~ 0x05ff
0x0f00 ~ 0x0f4f
0x1000 ~ 0x1fff

* BAR: Base Address Register
PCle Packet Routing: Challenge

- PCle hardware registers can be manipulated by software
PCle Packet Routing: Challenge

- PCle hardware registers can be manipulated by software

---

**Expected routing**

```
PA  | Dev MMIO | GPU MMIO
---|----------|----------
CPU |          |          
```

**Actual routing with manipulation**

```
PA  | Dev MMIO | GPU MMIO
---|----------|----------
CPU |          |          
```

---
PCIe Packet Routing: Challenge

- PCIe hardware registers can be manipulated by software
**PCle Packet Routing: Challenge**

- PCle hardware registers can be manipulated by software

**Expected routing**

- **CPU**
  - PCIe Root Complex
  - 0x04~0x0f
  - 0x24~0x4f

- **Untrusted Device**
  - 0x04~0x0f

- **GPU**
  - GPU MMIO
  - 0x24~0x4f

**Actual routing with manipulation**

- **CPU**
  - PCIe Root Complex
  - 0x24~0x4f
  - 0x04~0x0f

- **Untrusted Device**
  - 0x24~0x4f

- **GPU**
  - GPU MMIO
  - 0x04~0x0f

---

115
MMIO Lockdown

- PCIe hardware registers can be manipulated by software

**Solution:** freeze MMIO routing information (MMIO lockdown)
MMIO Lockdown

- PCIe hardware registers **can be manipulated** by software

- **Solution:** freeze MMIO routing information (MMIO lockdown)

```
Privileged Process

“Modify GPU register value to 0x2000~0x2fff”
```

```
Core
```

```
CPU
```

```
PCle Root Complex
```

```
GPU Regs
0x1000~0x1fff
```
MMIO Lockdown

- PCIe hardware registers can be manipulated by software
- **Solution:** freeze MMIO routing information (MMIO lockdown)

```
Privileged Process
“Modify GPU register value to 0x2000~0x2fff”

Core

CPU

PCle Root Complex

PCle Packet
Dest: GPU register
Value: 0x2000~0x2fff

GPU
Regs
0x1000~0x1fff
```
MMIO Lockdown

- PCIe hardware registers can be manipulated by software

**Solution:** freeze MMIO routing information (MMIO lockdown)

Privileged Process

“Modify GPU register value to 0x2000~0x2fff”

He’s trying to modify trusted-GPU’s PCIe routing information.

Discard it
MMIO Lockdown

- PCIe hardware registers can be manipulated by software
- **Solution**: freeze MMIO routing information (MMIO lockdown)

```
Privileged Process

“Modify GPU register value to 0x2000~0x2fff”

He’s trying to modify trusted-GPU’s PCIe routing information.

Discard it
```
Next: Inter-Enclave Communication

Protected by MMIO Access Restriction

Protected by MMIO Lockdown
Inter-Enclave Communication

- Inter-process communication: message queue & shared memory
- Confidentiality & integrity provided by **authenticated encryption**

![Diagram showing communication between User Enclave and GPU Enclave]
Inter-Enclave Communication

- Inter-process communication: message queue & shared memory
- Confidentiality & integrity provided by authenticated encryption
Inter-Enclave Communication

- Inter-process communication: message queue & shared memory
- Confidentiality & integrity provided by authenticated encryption
Inter-Enclave Communication

- Inter-process communication: message queue & shared memory
- Confidentiality & integrity provided by authenticated encryption
Inter-Enclave Communication

- Inter-process communication: message queue & shared memory
- Confidentiality & integrity provided by authenticated encryption
Inter-Enclave Communication

- Inter-process communication: message queue & shared memory
- Confidentiality & integrity provided by authenticated encryption

![Diagram showing inter-enclave communication](image)
Inter-Enclave Communication

• Inter-process communication: message queue & shared memory
• Confidentiality & integrity provided by authenticated encryption
Communication Challenge: DMA

• Challenge
  – DMA from device to enclaves not allowed by SGX
  – Data copy can only be done through (slow) MMIO
Trusted DMA Support

• GPU DMAs encrypted data from shared memory to GPU
• GPU enclave launches in-GPU decryption kernel

(1) DMA
(Command issued by GPU enclave through MMIO)

(2) Launch in-GPU decryption kernel
Evaluation
Evaluation

• Prototype Implementation
  – Hardware changes are emulated in a KVM/QEMU virtual machine
  – GPU enclave implementation is based on Gdev [Kato, ATC’12]

• Performance analysis: Rodinia GPU microbenchmark
  – Measure overheads due to cryptography, etc.
  – Baseline: unmodified Gdev NVIDIA GPU driver

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>HIX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trusted Execution</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Encryption</td>
<td>N/A</td>
<td>AES-OCB [Rogaway ‘14]</td>
</tr>
<tr>
<td>GPU</td>
<td>NVIDIA GeForce GTX 580*</td>
<td></td>
</tr>
</tbody>
</table>

* Newer devices are not supported by Gdev
## Performance Result: Rodinia

<table>
<thead>
<tr>
<th>App Name</th>
<th>SRAD</th>
<th>PF</th>
<th>NN</th>
<th>NW</th>
<th>LUD</th>
<th>HS</th>
<th>GS</th>
<th>BFS</th>
<th>BP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memcpy</td>
<td>48.4MB</td>
<td>256.0MB</td>
<td>501.2KB</td>
<td>192.1MB</td>
<td>32.0MB</td>
<td>12.0MB</td>
<td>64.0MB</td>
<td>46.9MB</td>
<td>159.8MB</td>
</tr>
</tbody>
</table>

### Rodinia Benchmark Execution

- **HIX Average Overhead: 26%**

![Diagram showing execution times and relative execution times for different applications.]
## Performance Result: Rodinia

### Table: Performance Comparison

<table>
<thead>
<tr>
<th>App Name</th>
<th>SRAD</th>
<th>PF</th>
<th>NN</th>
<th>NW</th>
<th>LUD</th>
<th>HS</th>
<th>GS</th>
<th>BFS</th>
<th>BP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memcpy</td>
<td>48.4MB</td>
<td>256.0MB</td>
<td>501.2KB</td>
<td>192.1MB</td>
<td>32.0MB</td>
<td>12.0MB</td>
<td>64.0MB</td>
<td>46.9MB</td>
<td>159.8MB</td>
</tr>
</tbody>
</table>

### Diagram: Execution Time Analysis

- **Baseline** Execution Time: 30ms
- **HIX** Execution Time: 267ms
- **Relative Speedup**:
  - MemcpyDtoH: 1.76x
  - Computation: 2.54x
  - MemcpyHtoD: 0.54x
  - Close: 1.22x

The diagram illustrates the execution time comparison between the baseline and HIX, highlighting the speedup achieved with each component.
## Performance Result: Rodinia

<table>
<thead>
<tr>
<th>App Name</th>
<th>SRAD</th>
<th>PF</th>
<th>NN</th>
<th>NW</th>
<th>LUD</th>
<th>HS</th>
<th>GS</th>
<th>BFS</th>
<th>BP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memcpy</td>
<td>48.4MB</td>
<td>256.0MB</td>
<td>501.2KB</td>
<td>192.1MB</td>
<td>32.0MB</td>
<td>12.0MB</td>
<td>64.0MB</td>
<td>46.9MB</td>
<td>159.8MB</td>
</tr>
</tbody>
</table>

Rodinia Benchmark Execution

- memncpy includes cryptography overheads
### Performance Result: Rodinia

<table>
<thead>
<tr>
<th>App Name</th>
<th>SRAD</th>
<th>PF</th>
<th>NN</th>
<th>NW</th>
<th>LUD</th>
<th>HS</th>
<th>GS</th>
<th>BFS</th>
<th>BP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memcpy</td>
<td>48.4MB</td>
<td>256.0MB</td>
<td>501.2KB</td>
<td><strong>192.1MB</strong></td>
<td>32.0MB</td>
<td>12.0MB</td>
<td>64.0MB</td>
<td>46.9MB</td>
<td>159.8MB</td>
</tr>
</tbody>
</table>

**HIX Average Overhead: 26%**

*memcpy includes cryptography overheads*
## Performance Result: Rodinia

<table>
<thead>
<tr>
<th>App Name</th>
<th>SRAD</th>
<th>PF</th>
<th>NN</th>
<th>NW</th>
<th>LUD</th>
<th>HS</th>
<th>GS</th>
<th>BFS</th>
<th>BP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memcpy</td>
<td>48.4MB</td>
<td>256.0MB</td>
<td>501.2KB</td>
<td>192.1MB</td>
<td>32.0MB</td>
<td>12.0MB</td>
<td>64.0MB</td>
<td>46.9MB</td>
<td>159.8MB</td>
</tr>
</tbody>
</table>

### Rodinia Benchmark Execution

- **memcpy includes cryptography overheads**
- **HIX Baseline**
- **HIX**
- **Close**
- **MemcpyDtoH**
- **MemcpyHtoD**

Execution time (ms) and relative execution time:

- Baseline: 382ms
- HIX: 217ms
- memcpy: 267ms
- Close: 30ms
### Performance Result: Rodinia

<table>
<thead>
<tr>
<th>App Name</th>
<th>SRAD</th>
<th>PF</th>
<th>NN</th>
<th>NW</th>
<th>LUD</th>
<th>HS</th>
<th>GS</th>
<th>BFS</th>
<th>BP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memcpy</td>
<td>48.4MB</td>
<td>256.0MB</td>
<td>501.2KB</td>
<td>192.1MB</td>
<td>32.0MB</td>
<td>12.0MB</td>
<td>64.0MB</td>
<td>46.9MB</td>
<td>159.8MB</td>
</tr>
</tbody>
</table>

**memcpy** includes cryptography overheads.
### Performance Result: Rodinia

<table>
<thead>
<tr>
<th>App Name</th>
<th>SRAD</th>
<th>PF</th>
<th>NN</th>
<th>NW</th>
<th>LUD</th>
<th>HS</th>
<th>GS</th>
<th>BFS</th>
<th>BP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memcpy</td>
<td>48.4MB</td>
<td>256.0MB</td>
<td>501.2KB</td>
<td>192.1MB</td>
<td>32.0MB</td>
<td>12.0MB</td>
<td>64.0MB</td>
<td>46.9MB</td>
<td>159.8MB</td>
</tr>
</tbody>
</table>

**HIX Average Overhead:** 26%

**memcpy** includes cryptography overheads
Performance Result: Rodinia

<table>
<thead>
<tr>
<th>App Name</th>
<th>SRAD</th>
<th>PF</th>
<th>NN</th>
<th>NW</th>
<th>LUD</th>
<th>HS</th>
<th>GS</th>
<th>BFS</th>
<th>BP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memcpy</td>
<td>48.4MB</td>
<td>256.0MB</td>
<td>501.2KB</td>
<td>192.1MB</td>
<td>32.0MB</td>
<td>12.0MB</td>
<td>64.0MB</td>
<td>46.9MB</td>
<td>159.8MB</td>
</tr>
</tbody>
</table>

HIX Average Overhead: 26%
## Performance Result: Rodinia

<table>
<thead>
<tr>
<th>App Name</th>
<th>SRAD</th>
<th>PF</th>
<th>NN</th>
<th>NW</th>
<th>LUD</th>
<th>HS</th>
<th>GS</th>
<th>BFS</th>
<th>BP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memcpy</td>
<td>48.4MB</td>
<td>256.0MB</td>
<td>501.2KB</td>
<td>192.1MB</td>
<td>32.0MB</td>
<td>12.0MB</td>
<td>64.0MB</td>
<td>46.9MB</td>
<td>159.8MB</td>
</tr>
</tbody>
</table>

**Rodinia Benchmark Execution**

- **HIX Average Overhead:** 26%

**Large Amount of Data → High Cryptography Overheads**
Performance Result: Rodinia

Execution occupies 88% of the entire operation time
### Performance Result: Rodinia

#### Rodinia Benchmark Execution

- **Execution occupies 88% of the entire operation time**
- **HIX Average Overhead: 26%**
- **High Computational Ratio → Cryptography Overhead Ratio Reduced**

<table>
<thead>
<tr>
<th>App Name</th>
<th>SRAD</th>
<th>PF</th>
<th>NN</th>
<th>NW</th>
<th>LUD</th>
<th>HS</th>
<th>GS</th>
<th>BFS</th>
<th>BP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memcpy</td>
<td>48.4MB</td>
<td>256.0MB</td>
<td>501.2KB</td>
<td>192.1MB</td>
<td>32.0MB</td>
<td>12.0MB</td>
<td>64.0MB</td>
<td>46.9MB</td>
<td>159.8MB</td>
</tr>
</tbody>
</table>

**HIX Average Overhead: 26%**

**High Computational Ratio → Cryptography Overhead Ratio Reduced**
Performance Result: Rodinia

<table>
<thead>
<tr>
<th>App Name</th>
<th>SRAD</th>
<th>PF</th>
<th>NN</th>
<th>NW</th>
<th>LUD</th>
<th>HS</th>
<th>GS</th>
<th>BFS</th>
<th>BP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memcpy</td>
<td>48.4MB</td>
<td>256.0MB</td>
<td>501.2KB</td>
<td>192.1MB</td>
<td>32.0MB</td>
<td>12.0MB</td>
<td>64.0MB</td>
<td>46.9MB</td>
<td>159.8MB</td>
</tr>
</tbody>
</table>

Rodinia Benchmark Execution

HIX Average Overhead: 26%
## Performance Result: Rodinia

<table>
<thead>
<tr>
<th>App Name</th>
<th>SRAD</th>
<th>PF</th>
<th>NN</th>
<th>NW</th>
<th>LUD</th>
<th>HS</th>
<th>GS</th>
<th>BFS</th>
<th>BP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memcpy</td>
<td>48.4MB</td>
<td>256.0MB</td>
<td>501.2KB</td>
<td>192.1MB</td>
<td>32.0MB</td>
<td>12.0MB</td>
<td>64.0MB</td>
<td>46.9MB</td>
<td>159.8MB</td>
</tr>
</tbody>
</table>

### HIX Performance Overheads

\[ \alpha \begin{align*}
\text{Data Copy Ratio} \\
\text{Computational Ratio}
\end{align*} \]

---

**Rodinia Benchmark Execution**

- Close
-MemcpyDtoH
-Computation
-MemcpyHtoD

**HIX**

**Baseline**

Execution time (ms) and relative execution time
Conclusion

- HIX: Provide trusted execution environment to commodity GPUs
Conclusion

• HIX: Provide trusted execution environment to commodity GPUs
Conclusion

- HIX: Provide trusted execution environment to commodity GPUs
Conclusion

- HIX: Provide trusted execution environment to commodity GPUs
Conclusion

- HIX: Provide trusted execution environment to commodity GPUs
Conclusion

• HIX: Provide trusted execution environment to commodity GPUs
Conclusion

• HIX: Provide trusted execution environment to commodity GPUs
Conclusion

- HIX: Provide trusted execution environment to commodity GPUs
Heterogeneous Isolated Execution for Commodity GPUs

Thank you for Listening!

Q&A