Hermes: An Integrated CPU/GPU Microarchitecture for IP Routing

Yuhao Zhu*
Yangdong Deng‡
Yubei Chen‡

* Electrical and Computer Engineering
  University of Texas at Austin

‡ Institute of Microelectronics
  Tsinghua University
Motivation: IP Routing
Motivation: IP Routing

- Challenges in IP Router design
Motivation: IP Routing

- Challenges in IP Router design
  - Internet traffic is still increasing
Motivation: IP Routing

- Challenges in IP Router design

  Throughput & QoS!
Motivation: IP Routing

- Challenges in IP Router design

  - Throughput & QoS!

  - New network services and protocols keep appearing
Motivation: IP Routing

- Challenges in IP Router design
  - Throughput & QoS!
  - Programmability
Motivation: IP Routing

- Challenges in IP Router design
  - Throughput & QoS!
  - Programmability
- Traditional router solutions
Motivation: IP Routing

- Challenges in IP Router design
  - Throughput & QoS!
  - Progammability

- Traditional router solutions
  - Hardware Routers: ASIC, Network Processors
Motivation: IP Routing

- Challenges in IP Router design
  - Throughput & QoS!
  - Progammability

- Traditional router solutions
  - Hardware Routers: ASIC, Network Processors
  - PC based Software Routers
Motivation: IP Routing

- Challenges in IP Router design
  - Throughput & QoS!
  - Progammability

- Traditional router solutions
  - Hardware Routers: ASIC, Network Processors
  - PC based Software Routers

- How about GPUs?
Motivation: IP Routing

- Challenges in IP Router design
  - Throughput & QoS!
  - Programmability

- Traditional router solutions
  - Hardware Routers: ASIC, Network Processors
  - PC based Software Routers

- How about GPUs?
  - High computing power
Motivation: IP Routing

- Challenges in IP Router design

- Traditional router solutions
  - Hardware Routers: ASIC, Network Processors
  - PC based Software Routers

- How about GPUs?
  - High computing power
  - Mass market with strong development support
GPU based Software Router

- Related Work
  - Smith et al. [ISPASS2009]
  - Mu et al. [DATE2010]
  - Han et al. [SIGCOMM2010]
GPU based Software Router

- Related Work
  - Smith et al. [ISPASS2009]
  - Mu et al. [DATE2010]
  - Han et al. [SIGCOMM2010]

- Restrictions
GPU based Software Router

- Related Work
  - Smith et al. [ISPASS2009]
  - Mu et al. [DATE2010]
  - Han et al. [SIGCOMM2010]

- Restrictions
  - CPU/GPU communication overhead hurts overall throughput
GPU based Software Router

- Related Work
  - Smith et al. [ISPASS2009]
  - Mu et al. [DATE2010]
  - Han et al. [SIGCOMM2010]

- Restrictions
  - CPU/GPU communication overhead hurts overall throughput

![Graph showing throughput vs number of string patterns]

Mu et al. [DATE2010]
GPU based Software Router

Related Work

- Smith et al. [ISPASS2009]
- Mu et al. [DATE2010]
- Han et al. [SIGCOMM2010]

Restrictions

- CPU/GPU communication overhead hurts overall throughput
- Batch (warp) processing hurts QoS
GPU based Software Router

- Related Work
  - Smith et al. [ISPASS2009]
  - Mu et al. [DATE2010]
  - Han et al. [SIGCOMM2010]

- Restrictions
  - CPU/GPU communication overhead hurts overall throughput
  - Batch (warp) processing hurts QoS

Worst case delay: \texttt{batch\_transfer\_granularity/line-card\_rate}
Hermes Microarchitecture
Hermes Microarchitecture

- Throughput wins!
Hermes Microarchitecture

- Throughput wins!
  - Shared-memory mitigating communication overhead
Hermes Microarchitecture

- Throughput wins!
  - Shared-memory mitigating communication overhead
- QoS wins!
Throughput wins!
- Shared-memory mitigating communication overhead

QoS wins!
- Adaptive warp scheduler through Task FIFO and DCQ
Shared Memory Hierarchy

- How?

- Why?
Shared Memory Hierarchy

- How?
  - CPU/GPU connected to the shared, centralized memory

- Why?
Shared Memory Hierarchy

- **How?**
  - CPU/GPU connected to the shared, centralized memory

- **Why?**
  - Except throughput…
  - Serves as a large packet buffer – impractical in traditional routers!
Shared Memory Hierarchy

How?
- CPU/GPU connected to the shared, centralized memory
- Execution model compatible with traditional CPU/GPU systems

Why?
- Except throughput…
- Serves as a large packet buffer – impractical in traditional routers!
Shared Memory Hierarchy

How?
- CPU/GPU connected to the shared, centralized memory
- Execution model compatible with traditional CPU/GPU systems

Why?
- Except throughput…
- Serves as a large packet buffer – impractical in traditional routers!
- Avoid consistency issues in shared memory systems
Adaptive Warp Scheduler
Adaptive Warp Scheduler

- Basic idea
Adaptive Warp Scheduler

- Basic idea
  - Deliver packets in an agile way
Adaptive Warp Scheduler

- Basic idea
  - Deliver packets in an agile way

- Mechanism
Adaptive Warp Scheduler

- Basic idea
  - Deliver packets in an agile way

- Mechanism
  - One GPU thread for one packet
Adaptive Warp Scheduler

- **Basic idea**
  - Deliver packets in an agile way

- **Mechanism**
  - One GPU thread for one packet
  - CPU passes #available packets to GPU through Task FIFO
Adaptive Warp Scheduler

- Basic idea
  - Deliver packets in an agile way

- Mechanism
  - One GPU thread for one packet
  - CPU passes #available packets to GPU through Task FIFO
  - GPU monitors the FIFO and starts processing whenever possible
Adaptive Warp Scheduler

- Basic idea
  - Deliver packets in an agile way

- Mechanism
  - One GPU thread for one packet
  - CPU passes available packets to GPU through Task FIFO
  - GPU monitors the FIFO and starts processing whenever possible
  - Tradeoffs in choosing the updating/fetching frequency
Adaptive Warp Scheduler

Basic idea
- Deliver packets in an agile way

Mechanism
- One GPU thread for one packet
- CPU passes #available packets to GPU through Task FIFO
- GPU monitors the FIFO and starts processing whenever possible
- Tradeoffs in choosing the updating/fetching frequency

Enforce in-order commit
Adaptive Warp Scheduler

- Basic idea
  - Deliver packets in an agile way

- Mechanism
  - One GPU thread for one packet
  - CPU passes available packets to GPU through Task FIFO
  - GPU monitors the FIFO and starts processing whenever possible
  - Tradeoffs in choosing the updating/fetching frequency

- Enforce in-order commit
  - Some protocols (UDP, etc.) require in-order packet committing
Adaptive Warp Scheduler

- Basic idea
  - Deliver packets in an agile way

- Mechanism
  - One GPU thread for one packet
  - CPU passes available packets to GPU through Task FIFO
  - GPU monitors the FIFO and starts processing whenever possible
  - Tradeoffs in choosing the updating/fetching frequency

- Enforce in-order commit
  - Some protocols (UDP, etc.) require in-order packet committing
  - ROB-like structure called DCQ
Adaptive Warp Scheduler

- Enforce in-order commit
  - Some protocols (UDP, etc.) require in-order packet committing
  - ROB-like structure called DCQ
Methodology
Methodology

- Benchmarks: hand-coded complete software router in CUDA
  - Checking IP header → Packet classification → Routing table lookup → Decrementing TTL → IP fragmentation and Deep packet inspection
  - Various packet traces with both burst and sparse patterns
Methodology

- Benchmarks: hand-coded complete software router in CUDA
  - Checking IP header → Packet classification → Routing table lookup → Decrementing TTL → IP fragmentation and Deep packet inspection
  - Various packet traces with both burst and sparse patterns

- gpgpu-sim -- cycle accurate CUDA-compatible GPU simulator
  - 8 shader cores
  - 32-wide SIMD, 32-wide warp
  - 1000MHz shared core frequency
  - 16768 registers per shader core
  - 16KByte shared memory per shared core
Methodology

- Benchmarks: hand-coded complete software router in CUDA
  - Checking IP header → Packet classification → Routing table lookup → Decrementing TTL → IP fragmentation and Deep packet inspection
  - Various packet traces with both burst and sparse patterns

- gpgpu-sim -- cycle accurate CUDA-compatible GPU simulator
  - 8 shader cores
  - 32-wide SIMD, 32-wide warp
  - 1000MHz shared core frequency
  - 16768 registers per shader core
  - 16KByte shared memory per shared core

- Maximally allowed concurrent warps (MCW) per core
  - They compete for hardware resources
  - They affect the updating/fetching frequency
Evaluations
Evaluations

- Throughput
Evaluations

- Delay

![Bar chart showing delay in CPU/GPU and Hermes-8, Hermes-16, Hermes-32 for DPI, Classifier, RTL, DecTTL categories, with delays measured in 1K cycles.](chart.png)
Evaluations

Scalability

![Scalability Graph]

- DPI
- Classifier
- RTL
- DecTTL
Conclusion
Conclusion

- Ever-demanding need for high-quality IP Routers
Conclusion

- Ever-demanding need for high-quality IP Routers
  - Throughput, QoS and programmability are important metrics but often not guaranteed at the same time
Conclusion

- Ever-demanding need for high-quality IP Routers
  - Throughput, QoS and programmability are important metrics but often not guaranteed at the same time

- Hermes: GPU-based software router
Conclusion

- Ever-demanding need for high-quality IP Routers
  - Throughput, QoS and programmability are important metrics but often not guaranteed at the same time

- Hermes: GPU-based software router
  - Meet all three at the same time
Conclusion

- Ever-demanding need for high-quality IP Routers
  - Throughput, QoS and programmability are important metrics but often not guaranteed at the same time

- Hermes: GPU-based software router
  - Meet all three at the same time
  - Leverage huge and mature GPU market
Conclusion

- Ever-demanding need for high-quality IP Routers
  - Throughput, QoS and programmability are important metrics but often not guaranteed at the same time

- Hermes: GPU-based software router
  - Meet all three at the same time
  - Leverage huge and mature GPU market
  - Minimal hardware extensions
Conclusion

- Ever-demanding need for high-quality IP Routers
  - Throughput, QoS and programmability are important metrics but often not guaranteed at the same time

- Hermes: GPU-based software router
  - Meet all three at the same time
  - Leverage huge and mature GPU market
  - Minimal hardware extensions

Come to my poster to learn more!