FORTH
Institute of Computer Science

GPU-Disasm: A GPU-based x86 Disassembler

ISC 2015

Evangelos Ladakis ,
Giorgos Vasiliadis,
Michalis Polychronakis,
Sotiris Ioannidis, George Portokalidis

First Impressions

reddit **REVERSEENGINEERING** comments related

43

GPU-Disasm: A GPU-based x86 Disassembler by Ladakis, Evangelos, Giorgos Vasiliadis, Michalis Polychronakis, Sotiris Ioannidis, and Georgios Portokalidis [PDF] (necoma-project.eu)

submitted 10 days ago by turnersr

3 comments **share**

all 3 comments

sorted by: **best**

[–] **fwork** 30 points 10 days ago

my first response to this, as captured in a technical IRC channel:

> if you have so much x86 binary that you need to accelerate disassembling it with GPUs, you have made bad life choices

permalink

[–] **Docmandu** 1 point 9 days ago

Think they were trying to come up with reasons why they need that NVidia GTX Titan card to play QuakeWorld on during lunch.

permalink **parent**

First Impressions

reddit REVERSEENGINEERING comments related

43

GPU-Disasm: A GPU-based x86 Disassembler by Ladakis, Evangelos, Giorgos Vasiliadis, Michalis Polychronakis, Sotiris Ioannidis, and Georgios Portokalidis [PDF] (necoma-project.eu)

submitted 10 days ago by turnersr

3 comments share

all 3 comments

sorted by: best

[–] fwork 30 points 10 days ago

my first response to this, as captured in a technical IRC channel:

> if you have so much x86 binary that you need to accelerate disassembling it with GPUs, you have made bad life choices

permalink

[–] Docmandu 1 point 9 days ago

Think they were trying to come up with reasons why they need that NVidia GTX Titan card to play QuakeWorld on during lunch.

permalink parent

First Impressions

reddit REVERSEENGINEERING comments related

43 GPU-Disasm: A GPU-based x86 Disassembler by Ladakis, Evangelos, Giorgos Vasiliadis, Michalis Polychronakis, Sotiris Ioannidis, and Georgios Portokalidis [PDF] (necoma-project.eu)

submitted 10 days ago by turnersr

3 comments share

all 3 comments

sorted by: best

[–] **fwork** 30 points 10 days ago

my first response to this, as captured in a technical IRC channel:

> if you have so much x86 binary that you need to accelerate disassembling it with GPUs, you have made bad life choices

permalink

[–] **Docmandu** 1 point 9 days ago

Think they were trying to come up with reasons why they need that NVidia GTX Titan card to play QuakeWorld on during lunch.

permalink parent

Outline

- Background
- Architecture
- Optimization
- Evaluation
- Conclusion

Disassembly

Software Reverse Engineering

- *Mandatory when source code is not available*
  - Bad guys
    - Find vulnerabilities
    - Bypass protection mechanisms
  - Good guys
    - Find malicious code
    - Debug and patching
    - Apply protection mechanisms
- Techniques
  - Linear
  - Recursive

```
push   %rbp
mov    %rsp,%rbp
push   %rbx
sub    $0x8,%rsp
mov    0x200868(%rip),%rax        # 600e28 <__CTOR_LIST__>
cmp    $0xffffffffffffffff,%rax
je     4005df <__do_global_ctors_aux+0x2f>
mov    $0x600e28,%ebx
nopl   0x0(%rax,%rax,1)
sub    $0x8,%rbx
callq  *%rax
mov    (%rbx),%rax
cmp    $0xffffffffffffffff,%rax
jne    4005d0 <__do_global_ctors_aux+0x20>
add    $0x8,%rsp
pop    %rbx
pop    %rbp
retq
nop
nop
```

Binary Stores

- Large number of binaries
  - 1.6 million Google play
  - 1.5 million app store
- Updated occasionally

From a security aspect:

- Analysis time and cost are essential

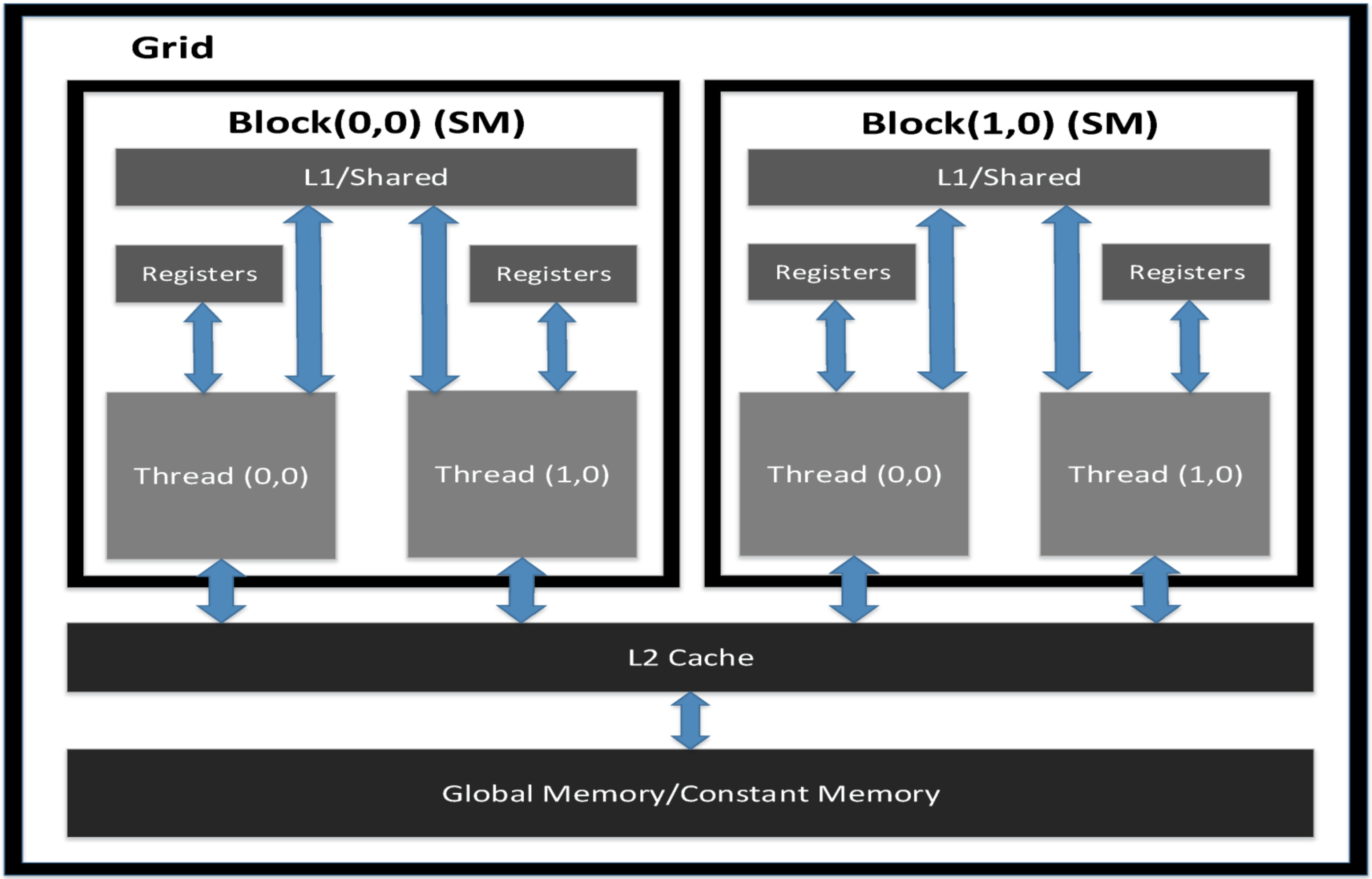
Motivation

- How can we build a fast and cheap Disassembler for large scale analysis?
- Can we use GPU’s to accelerate the decoding process?
- Why GPUs?

General-Purpose Programming on GPUs (GPGPU)

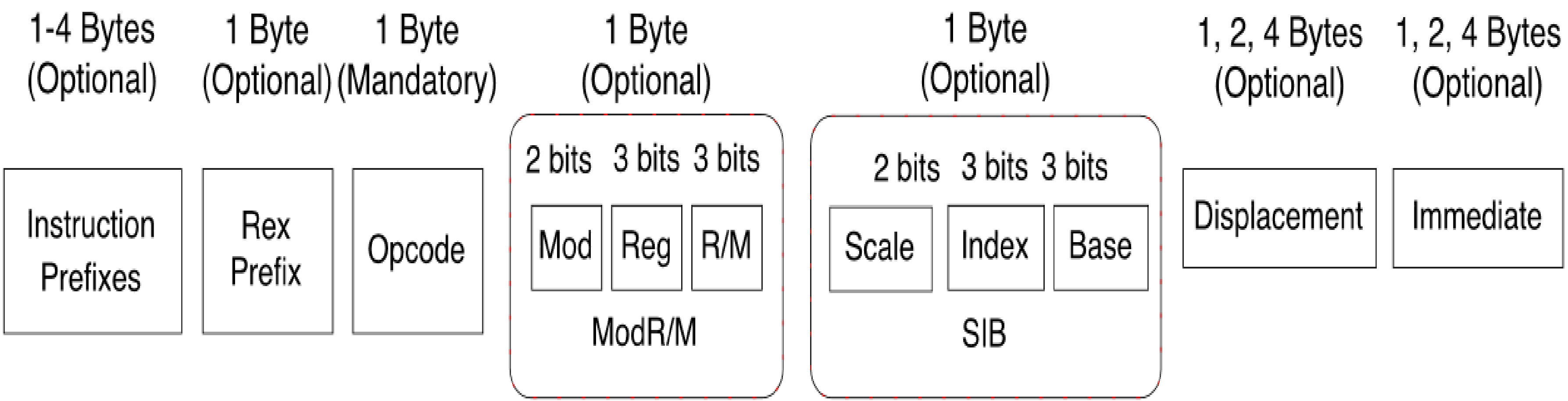
- Powerful co-processors for General Purpose Programming
- Commodity hardware, relative cheap
- Compute capabilities increasing
- Familiar API CUDA and OpenCl

GPU memory model



X86-ISA

- CISC architecture
- 1~15 Bytes instructions



Why x86?

- Widely used
- More challenges to address
- Applying to RISC is easier

GPU-Disasm Arch.

GPU-based Disassembler of the x86 architecture

Two modes:

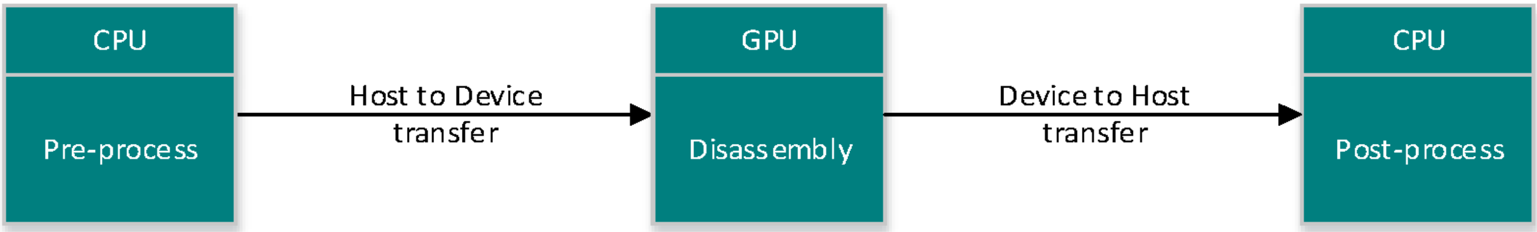
- Linear disassembly
  - Each thread is assigned a binary
- Exhaustive disassembly
  - Each thread decodes one instruction of the same binary but from a different offset

Challenges

- Arbitrary accesses to Global
  - X86 nature
- Load balancing and correctness
  - Utilize threads fairly with same size buffers
  - Start disassembling where we left
- Large number of static and constant values
  - Fast memory interfaces are small in capacity
  - Store the most frequently used

GPU-Disasm Arch.

GPU-Disasm Components:

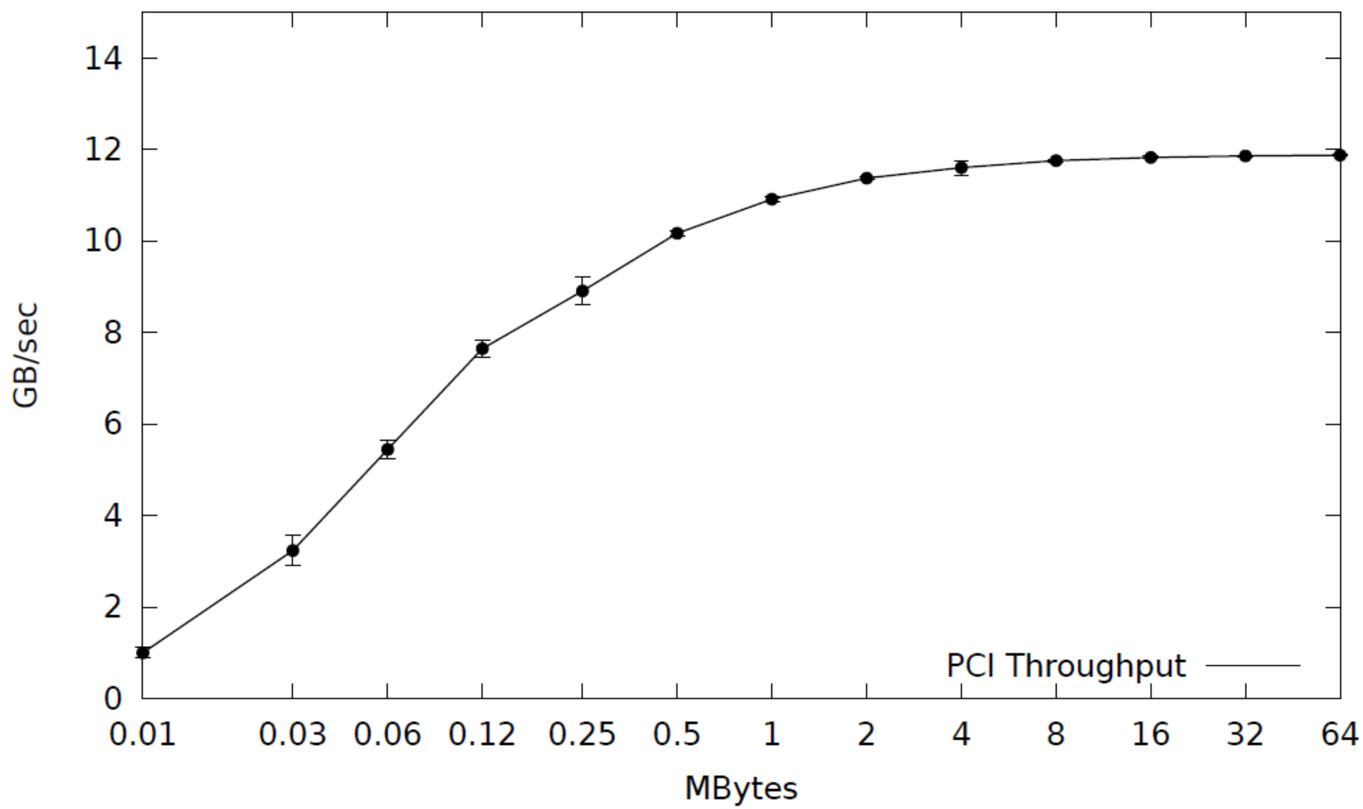


How to achieve high performance:

- Optimize transfers
- Optimize the Disassembly process
- Pipeline the operations

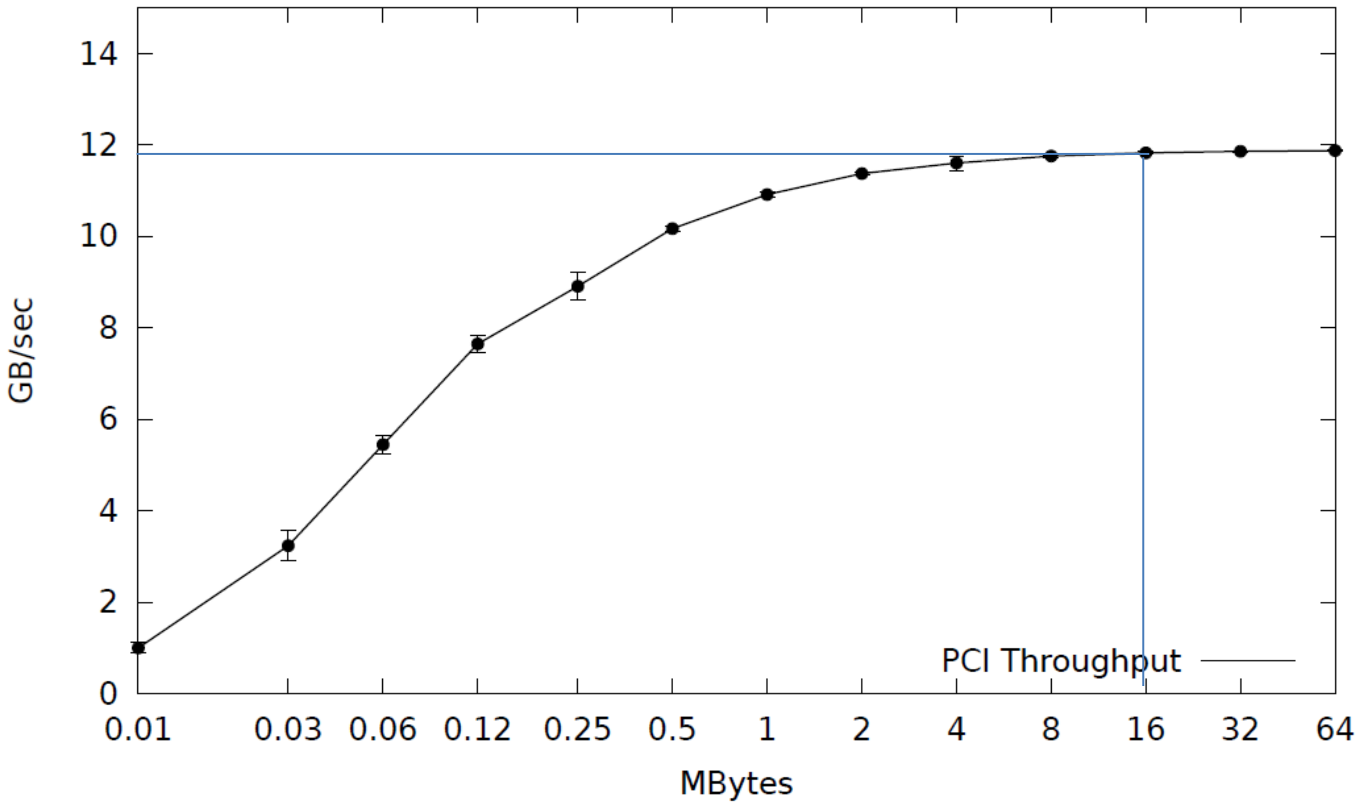
PCI Throughput

- PCI 3.0 throughput evaluation



PCI Throughput

- Maximum throughput on 16MB of data

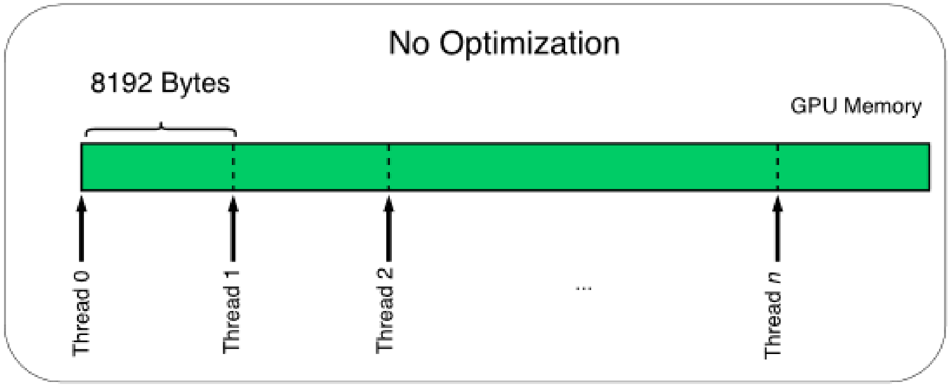


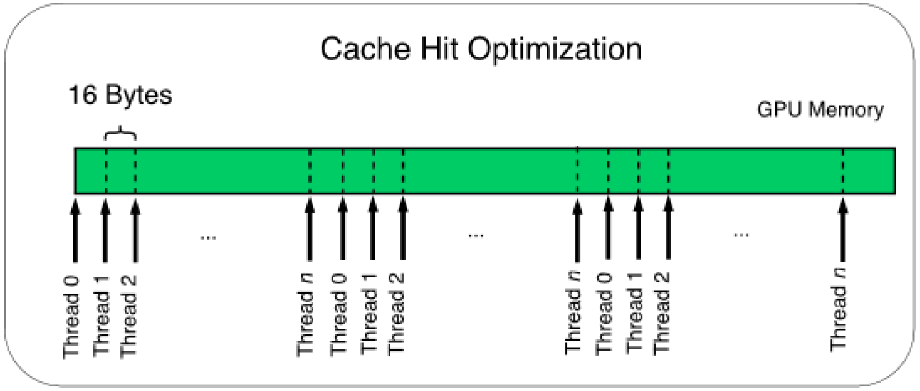
Optimize Transfers

1. Pre-allocate page-locked I/O buffers to the host (*cudaMallocHost)*
2. Place I/O to single buffers
   - Greater of 16 MB for PCI max throughput
3. Minimize the PCI transfer API calls

Optimize Disassembly

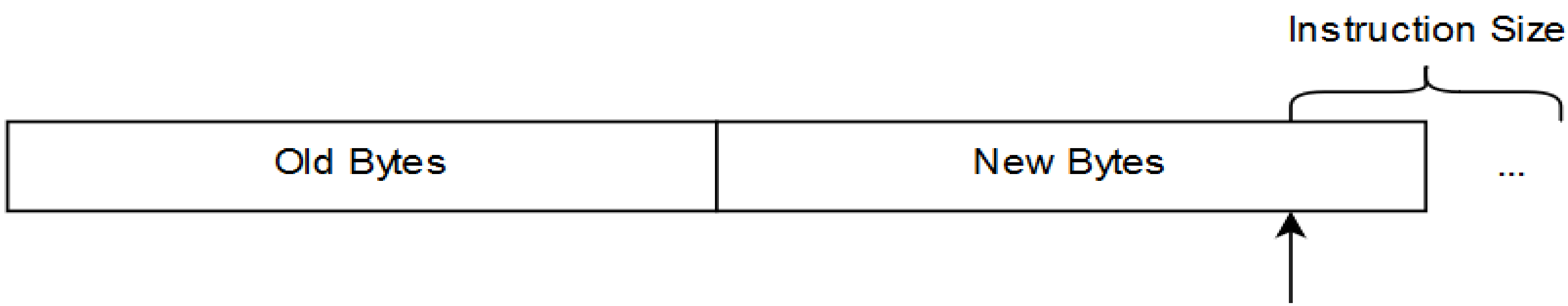
- Store Look-up-tables to Constant & Shared mem.
- Pre-fetch input data to registers
- Improve cache hits in L2
  - Divide input into small buffers
  - Move threads as groups inside memory



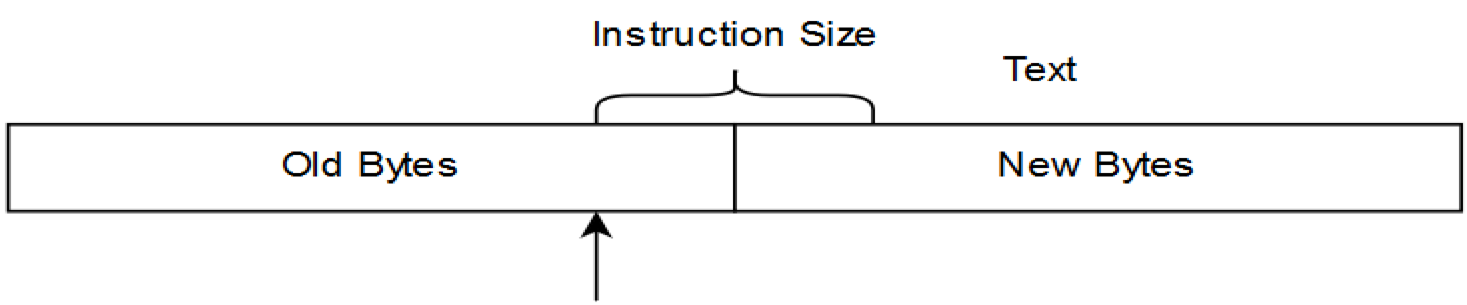


Correctness

- We keep a copy of old decoded bytes and the upcomming bytes



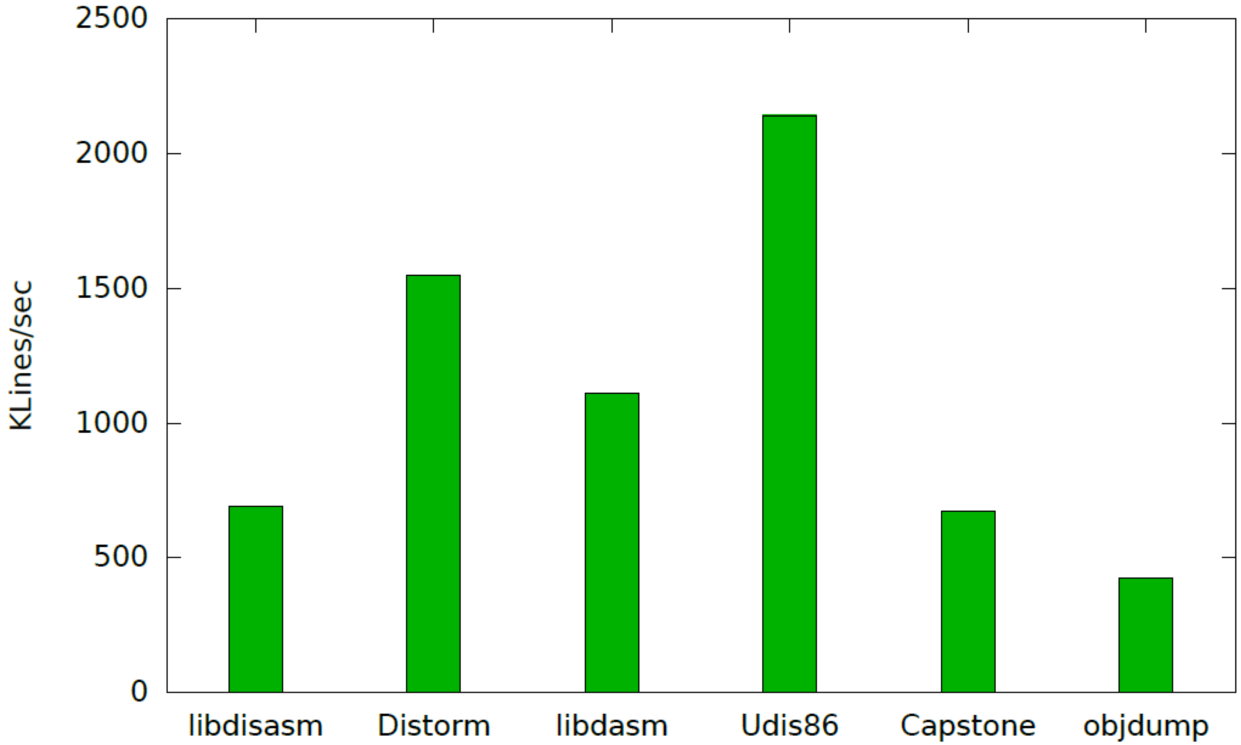
- So that we can continue decoding where we left



Evaluation

- Implementation in CUDA
- System:
  - GPU: NVIDIA GTX 770 $396
  - CPU: intel i7 $305
  - Total cost $1120
- Dataset from usr of ubuntu 12.04
- Performance measured in *Lines/sec*

Disassemblers Evaluation



- Single threaded, discard disk I/O
- Performance divergence due to output construction

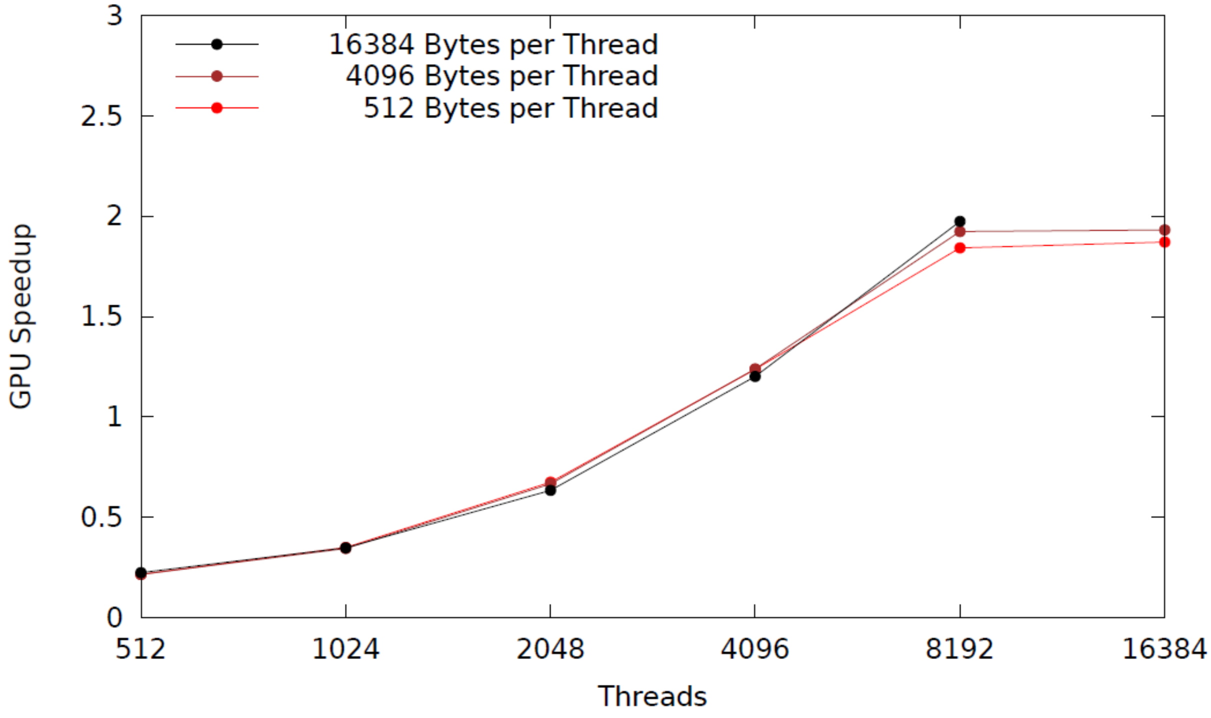
GPU-Disasm on crafted bins

Buffer Size (Bytes)	Average Hit Rate % (L1 to L2)
16	58.7
32	53.65
64	45.26

- Decode 2 Bytes Instructions
- Impact of L2 optimization
  - 25.85 % more performance

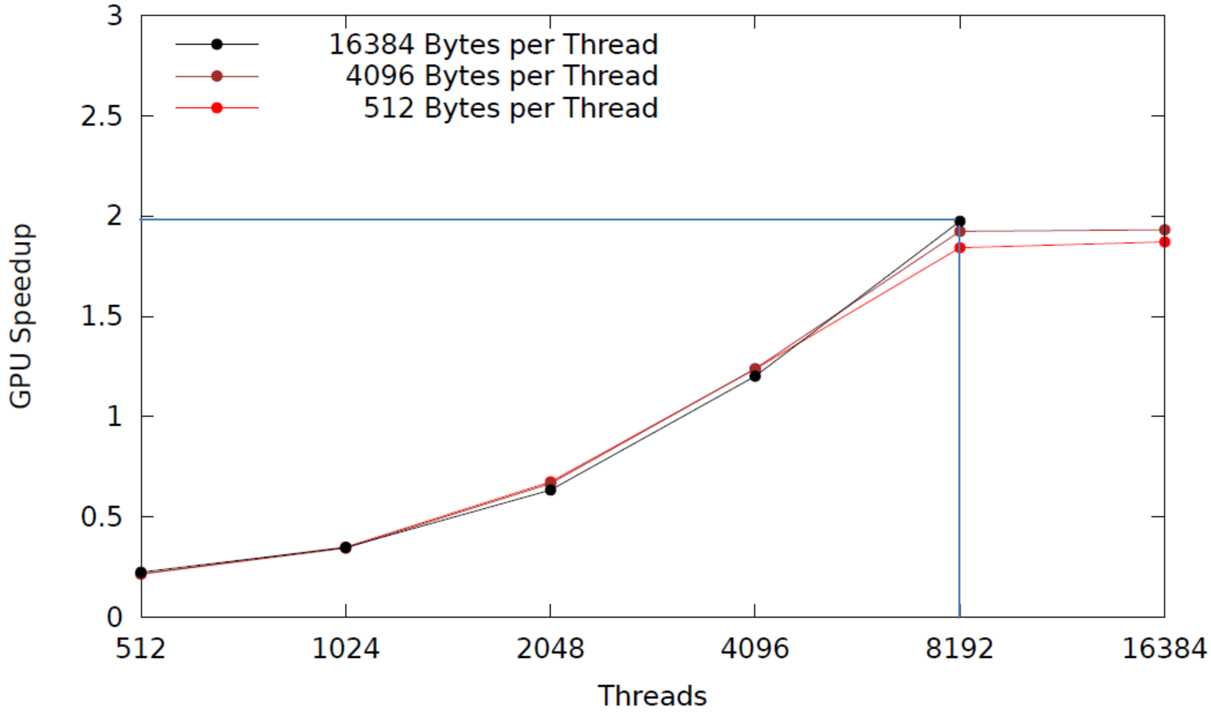
GPU-Disasm on Binaries

Comparing only the disassembly process



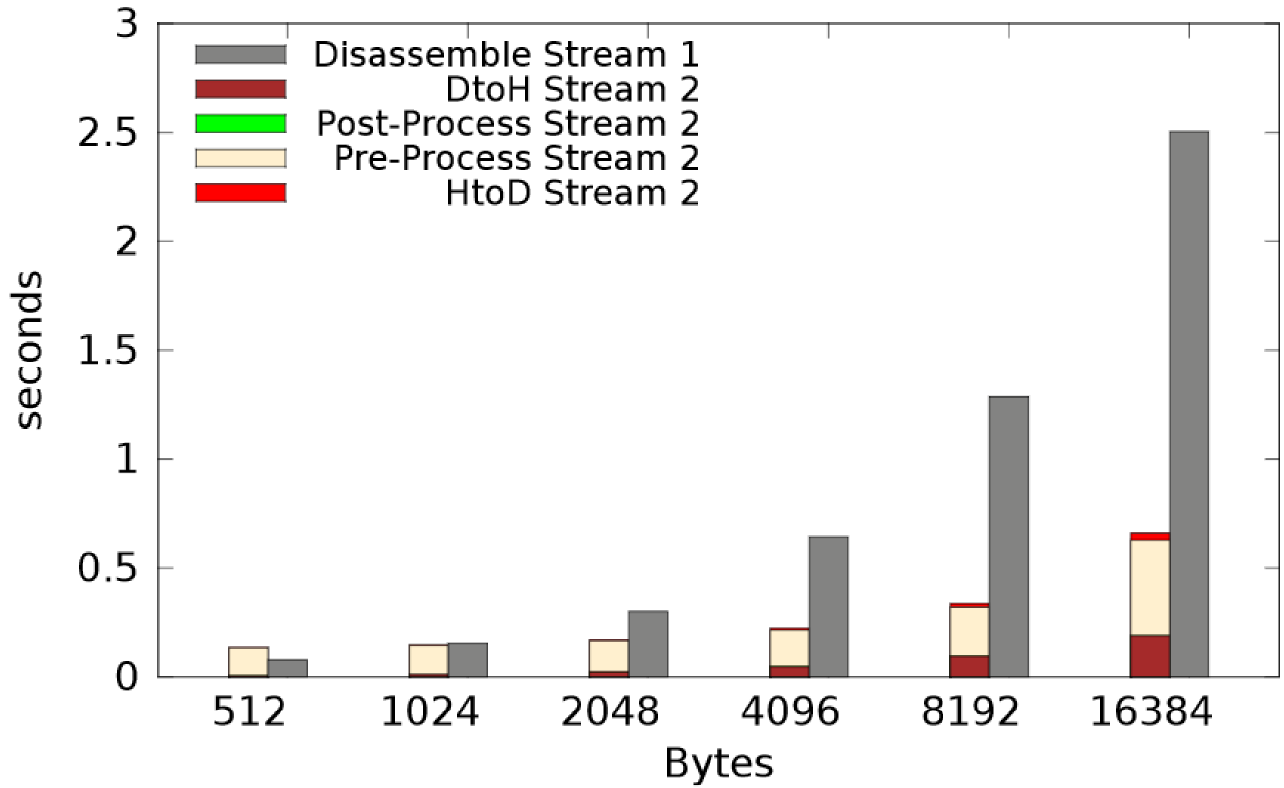
GPU-Disasm on Binaries

Comparing only the disassembly process



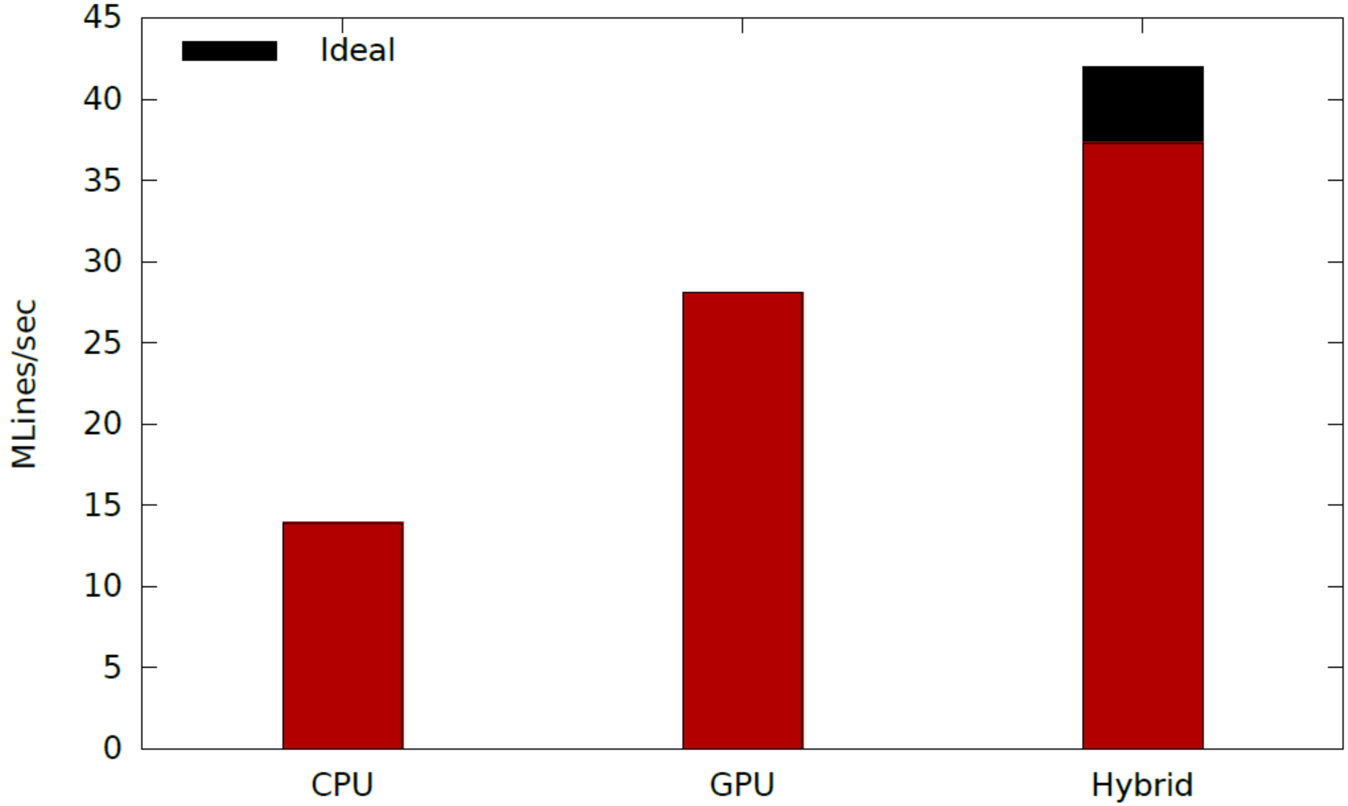
- Linear disassembly 2 times faster
- Exhaustive average 4.4 times faster

Pipeline Components



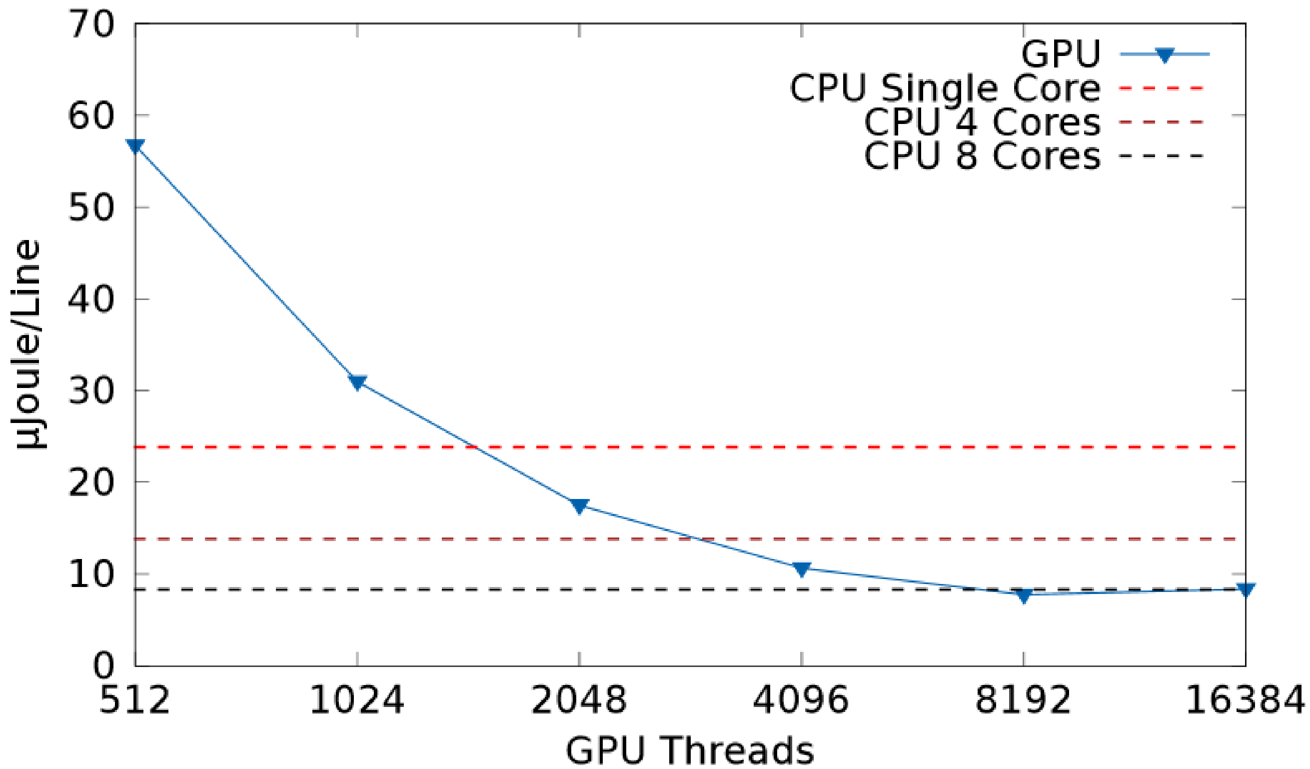
- After 1024 batch size, disassembly becomes the bottleneck

Hybrid (CPU & GPU)



- Hybrid has 7 CPU threads and the GPU
  - 1 thread is needed as the GPU controller

Power evaluation



- Metrics include CPU, RAM, and peripherals power consumption
  - Measured internally with sensors

Conclusion

- Presented a GPU-based implementation of an x86 disassembler
- 2 times faster in linear disassembly and 4.4 in exhaustive
- Similar power consumption with the CPU implementation

Thank you