Implementation and Experimental Evaluation of a CUDA Core under Single Event Effects

Werner Nedel, Fernanda Kastensmidt, José Rodrigo Azambuja

Universidade Federal do Rio Grande do Sul (UFRGS)
Instituto de Informática – PPGC

Universidade Federal do Rio Grande (FURG)
Centro de Ciências Computacionais – PPGCOMP
What Color is a CUDA Core?

- CUDA: Compute Unified Device Architecture
  - Programming model
  - Parallel computing platform
  - Created by NVIDIA
- Implemented by Graphic Processing Units (GPUs)
- What is a GPU? Why should I use it?
CPU vs GPU

W. Nedel, LATW 2014
Why Should I Care About GPUs?

- Specialized massively parallel many-core processors
  - Thread-Level Parallelism (TLP)
- Small portion assigned to data caching and flow control circuitry
  - Majority of the silicon area to data processing
- Great for manipulating high amounts of memory
  - Image processing, oil exploration, linear algebra, statistics, etc…
- Significant programming support

- Up to 624 GFLOPS (against 90 from an i7)
GPU Technology

- Technology down to 28nm
  - Higher transistor density
- Operating clock frequency up to 1GHz
  - Less latch window masking
- Lower voltage supply
  - Particles with smaller LET can affect the silicon

Current studies show that such GPUs are prone to radiation-induced errors at ground level (Rech et al., 2013)
GPUs and the Radiation Environment

- How are GPUs sensitive to radiation?
- How do Single Event Effects (SEE) interact with GPUs?
- Can we implement a GPU on an FPGA? Does it fit?
- How can we harden GPUs?

- Can we launch a FPGA implemented GPU into space?

W. Nedel, LATW 2014
Outline

- Implementation
  - GPU Structure
  - Streaming Multiprocessor (SM)
  - CUDA Core
- Fault Injection Campaign
- Results
- Conclusions and Future Work
GPU Structure Evolution (GT200 vs Fermi)

W. Nedel, LATW 2014
Fermi Structure

- 512 CUDA cores
- Unified address space: full C++ support
- ECC memory
- Dual warp scheduler
- OPCodes - Asfermi project (assembler compiler)

Implemented instructions:
- FP: FADD, FADD32I, FMUL, FMUL32I, FCMP, MUFU, DADD, DMUL, DFMA
- INT: IADD, IADD32I, IMUL, IMUL32I, IMAD, ISCADD, ISETP, ICMP
Streaming Multiprocessor (SM)

Host

Streaming Multiprocessor

Instruction Cache

Register File

Core  Core  LD/ST  SFU
Core  Core  LD/ST  SFU
Core  Core  LD/ST  SFU

Interconnect Network

Cache

W. Nedel, LATW 2014
Streaming Multiprocessor (SM)

- Instruction cache
- Register File
- CUDA Core
- Load/Store unit
- Special functions unit: sin, cosin
- Interconnect Network
- Cache
- Single Instruction Multiple Data (SIMD)
CUDA Core

- **Dispatch port**: distribute control signals
- **Operand collector**: interface with the register bank
- **Floating point unit**
- **Integer unit**
- **Result queue**: avoids conflicts of writes in the register file
## Implementation Results

- XC5VLX110T part Virtex5 FPGA
- Implemented using only the FPGA’s fabric
- We could roughly implement 18 CUDA Cores

<table>
<thead>
<tr>
<th></th>
<th>INT Unit (occupation)</th>
<th>FP Unit (occupation)</th>
<th>CUDA Core (occupation)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slice LUTs</td>
<td>2,098 (2.6%)</td>
<td>1,258 (1.8%)</td>
<td>3,656 (5.3%)</td>
</tr>
<tr>
<td>Slice Registers</td>
<td>1,812 (3.0%)</td>
<td>1,306 (1.9%)</td>
<td>3,417 (4.9%)</td>
</tr>
<tr>
<td>Frequency (MHz)</td>
<td>100</td>
<td>300</td>
<td>100</td>
</tr>
</tbody>
</table>
Fault Injection Campaign

- Injected 40,000 faults by simulation
- Faults were injected in the post-synthesis VHDL code by forcing bit-flips with the ModelSim simulator
- Faults were injected both in sequential (SEU) and combinational (SET) structures
- Most of the 5,841 signals describing the CUDA Core were upset
Fault Injection Campaign Results

<table>
<thead>
<tr>
<th></th>
<th>Faults Injected</th>
<th>Correct Results</th>
<th>Incorrect Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>INT Unit</td>
<td>20,000</td>
<td>16,313 (81.6%)</td>
<td>3,687 (18.4%)</td>
</tr>
<tr>
<td>FP Unit</td>
<td>20,000</td>
<td>15,746 (78.8%)</td>
<td>4,254 (21.2%)</td>
</tr>
<tr>
<td>Total</td>
<td>40,000</td>
<td>32,059 (80.2%)</td>
<td>7,491 (19.8%)</td>
</tr>
</tbody>
</table>

- Fault tolerance techniques are mandatory!
- Floating point unit is more sensitive than the integer unit
Conclusions and Future Work

- We have implemented a CUDA Core with an occupation of 5.3% of a Virtex5 FPGA board and operating frequency of 100MHz.
- Fault injection results have been performed showing that fault tolerance techniques are mandatory for GPUs.
- Area improvements are necessary.
- We intend to implement the entire SM structure followed by the entire GPU.
- We are looking to irradiate the CUDA Core with neutrons and Cobalt-60.

W. Nedel, LATW 2014
Thank You!