Simulating *nanoscale* dragons

Towards fully open source GPU accelerated molecular dynamics simulation

Vedran Miletić, HITS gGmbH
Szilárd Páll, KTH
Frauke Gräter, HITS gGmbH
Molecular dynamics simulation

- GROMACS simulates proteins, i.e. nanoscale dragons*

*careful, they can still bite
Layers of GPU computing

- GPU accelerated app (e.g. GROMACS) and libraries
Layers of GPU computing

GPU accelerated app (e.g. GROMACS) and libraries

CUDA

OpenCL
Layers of GPU computing

GPU accelerated app (e.g. GROMACS) and libraries

<table>
<thead>
<tr>
<th>CUDA</th>
<th>OpenCL</th>
</tr>
</thead>
<tbody>
<tr>
<td>NVIDIA proprietary compiler</td>
<td>AMD proprietary compiler</td>
</tr>
<tr>
<td>NVIDIA proprietary driver</td>
<td>AMD proprietary driver</td>
</tr>
</tbody>
</table>
Layers of GPU computing

GPU accelerated app (e.g. GROMACS) and libraries

- **CUDA**
- **OpenCL**

- NVIDIA proprietary compiler
- AMD proprietary compiler
- NVIDIA proprietary driver
- AMD proprietary driver
- Clang and libclc
- LLVM, Mesa/DRM, and radeon/nouveau
Layers of GPU computing

GPU accelerated app (e.g. GROMACS) and libraries

- CUDA
- OpenCL

Contributed by StreamComputing

- NVIDIA proprietary compiler
- AMD proprietary compiler
- Clang and libclc

- NVIDIA proprietary driver
- AMD proprietary driver
- LLVM, Mesa/DRM, and radeon/nouveau
Layers of GPU computing

GPU accelerated app (e.g. GROMACS) and libraries

- Contributed by StreamComputing
- Our work

- CUDA
  - NVIDIA proprietary compiler
  - NVIDIA proprietary driver

- OpenCL
  - AMD proprietary compiler
  - AMD proprietary driver

- LLVM, Mesa/DRM, and radeon/nouveau
  - Clang and libclc

Contributed by StreamComputing
Our work
If using proprietary stack is OK, NVIDIA/Intel do it better than AMD.

Source: top500.org statistics for November 2015.
Open source OpenCL stack

- Anyone can improve code
  - Hopefully less bugs, better performance
Open source OpenCL stack

- Anyone can improve code
  - Hopefully less bugs, better performance
- Develop what you care about, no “company priorities”

NVIDIA rep at #SC15 acknowledged that lots of customers are asking for #OpenCL, but certain individuals inside the company are pushing back.
Road to openness

- Remove the unused OpenCL image functions from GROMACS
  - Done by Szilard Pall, accepted upstream ✔
Road to openness

- Remove the unused OpenCL image functions from GROMACS
  - Done by Szilard Pall, accepted upstream ✔

- Implement global atomic compare-and-swap in LLVM AMDGPU target
  - Custom lowering to \{BUFFER,FLAT\}_ATOMIC_CMPSWAP
  - Reviewed by AMDGPU target maintainers 😊😊😊😊😊
  - Will be merged as soon as it has tests
Work in progress

- Add erf() / erff() to libclc
Work in progress

- Add erf() / erff() to libclc
- Handle struct arguments in OpenCL kernels correctly
  - Few possible approaches, cf. Beignet for Intel iGPUs
Work in progress

- Add erf() / erff() to libclc
- Handle struct arguments in OpenCL kernels correctly
  - Few possible approaches, cf. Beignet for Intel iGPUs
- Running, with result correctness issues
Joys of AMDGPU target development

- LLVM ABI breakage, LLVM doesn't compile, ...
- Occasional target-specific regressions
Joys of AMDGPU target development

- LLVM ABI breakage, LLVM doesn't compile, ...
- Occasional target-specific regressions
- Limitations regarding Volcanic Islands
- Overall resemblance to early days of Mozilla
The future is open and is here and now

<table>
<thead>
<tr>
<th>GPU accelerated app (e.g. GROMACS) and libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUDA</td>
</tr>
<tr>
<td>NVIDIA proprietary compiler</td>
</tr>
<tr>
<td>NVIDIA proprietary driver</td>
</tr>
</tbody>
</table>

- CUDA: OpenCL, C++17
- NVIDIA proprietary compiler: Clang, libclc, HCC/HIP
- NVIDIA proprietary driver: LLVM, Mesa/DRM, and radeon/nouveau
Acknowledgments

- Tom Stellard, AMD
- Matt Arsenault, AMD
- Edward O'Callaghan, Freenode channel #radeon
- Serge Martin, Freenode channel #radeon