

# Thoughts on GPUs as First-Class Citizens



Johannes Doerfert <[jdoerfert@llnl.gov](mailto:jdoerfert@llnl.gov)>

**Why this talk?**

# “Recent” Improvements

- Function Attr: nosync
- AMD GPU buildbot (OpenMP offload)
- Unified driver, embedding, tooling for OpenMP, CUDA (opt-in), HIP (opt-in)

---

# Ongoing Improvements

- Function memory effect “thread\_id”
  - GPU libraries (libm.a, libc.a, ...)
  - Atomics intrinsics and expansion support
  - Intel GPU support (via SPIR-V)
-

# Forever Ongoing Improvements

- Replace Function Attr: convergent
- Add GPU tests into LLVM-Test Suite
- Debug Metadata and GPUs

---

# “Out-there” Features

- “In-house” alternatives to vendor tools
  - Transparent execution on remote GPUs (*Remote GPU Offloading @ ISC’22*)
  - Host execution of GPU code (*VGPU paper @ LLPP’21*)
  - GPU execution of host code (*Direct GPU Compilation @ LLVM-HPC’22, noon in Monterey*)
  - Portability layer for GPU code (*CUDA-OMP @ PACT’22 [last talk]*)
-

# Some Missing Features

- Testing, incl. buildbots and unit tests
  - Side-effect API to take synchronization and termination into account
  - Convergence on GPU-centric analysis and optimization passes
  - GPU-aware defaults for the pass manager and pass options
  - Unified host-device optimization pipeline
  - Testing, incl. Buildbots and unit tests
-

# Action Items



**Join the LLVM-GPU Working Group**

**Create GPU Tests - Regression, Executable, ...**

# Setup a GPU Buildbot

# Develop Portable GPU Tooling in LLVM

# Tune Pass Parameters and Pipelines for GPUs

Build `libc` and `lib(std)c++` GPU libraries

**Provide “Real” GPU Codes for Test Suites**

# Abstract GPU Driver/Hardware Details in LLVM-IR



**Adjust Core-LLVM(-IR) wrt. (GPU) Parallelism**

# Unify (GPU) Offloading Logic

**Help Upstream Existing GPU Prototypes**

**Talk to us!**

**Discourse,  
LLVM-GPU Meeting,  
E-Mail,**

**...**