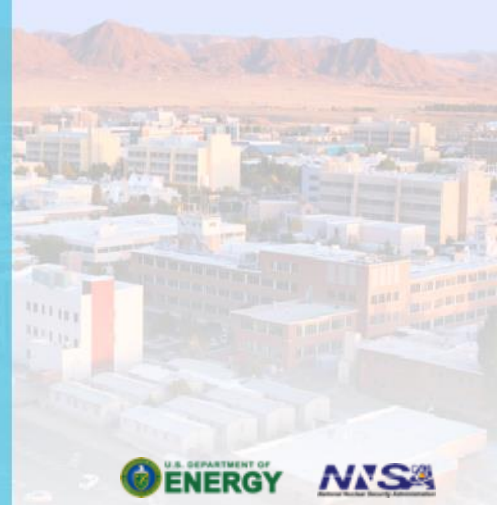
SAND2022-7054C

Sandia National Laboratories

MultiGrid on FPGA Using Data Parallel C++

PRESENTED BY

Christopher Siefert, Stephen Olivier, Gwendolyn Voskuilen and Jeffery Young



U.S. DEPARTMENT OF ENERGY | NNSA

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

What's Will This Talk Cover?

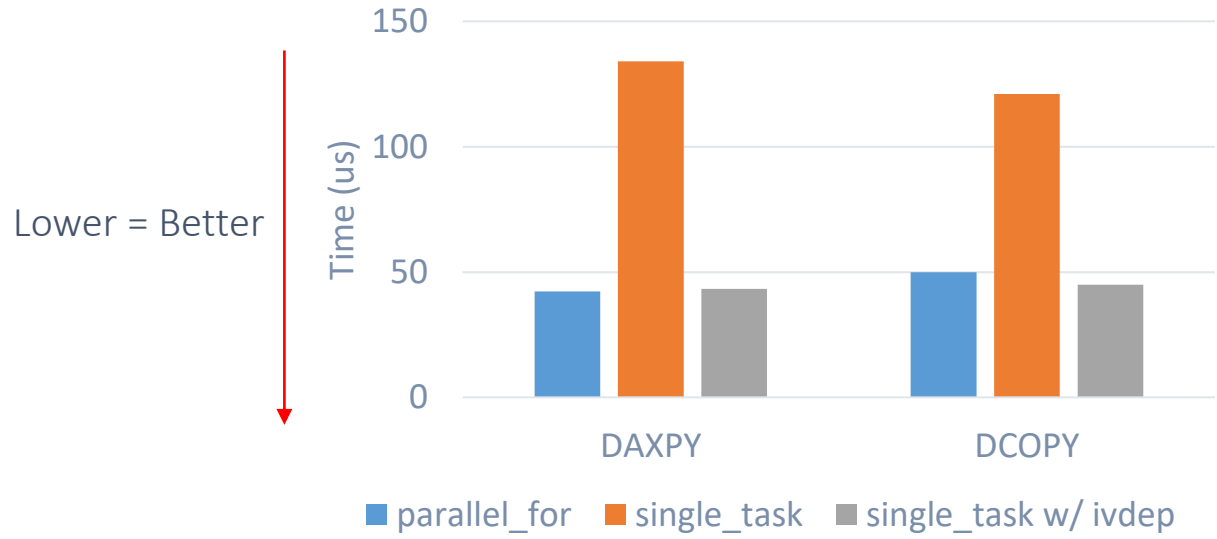
- Field Programmable Gate Arrays (FPGAs) are software-configurable circuits.
- Common apps: embedded computing, video processing, telecommunications, radar and crypto.
- Traditionally, FPGAs were programmed at a very low level (System Verilog anyone?).
- oneAPI is Intel's SYCL-based write-once / run-anywhere* approach for CPUs/GPUs/FPGAs.
- Question: oneAPI makes FPGAs accessible to HPC developers. But is it worth it?
- We'll answer that question looking at a modified version of the HPCG Benchmark.

* SYCL primarily supports Intel hardware (via oneAPI), but backends for HIP, CUDA and Xilinx FPGAs are under development.

Image from: https://www.intel.com/content/www/us/en/products/details/fpga/platforms/pac/d5005.html

parallel_for vs. single_task

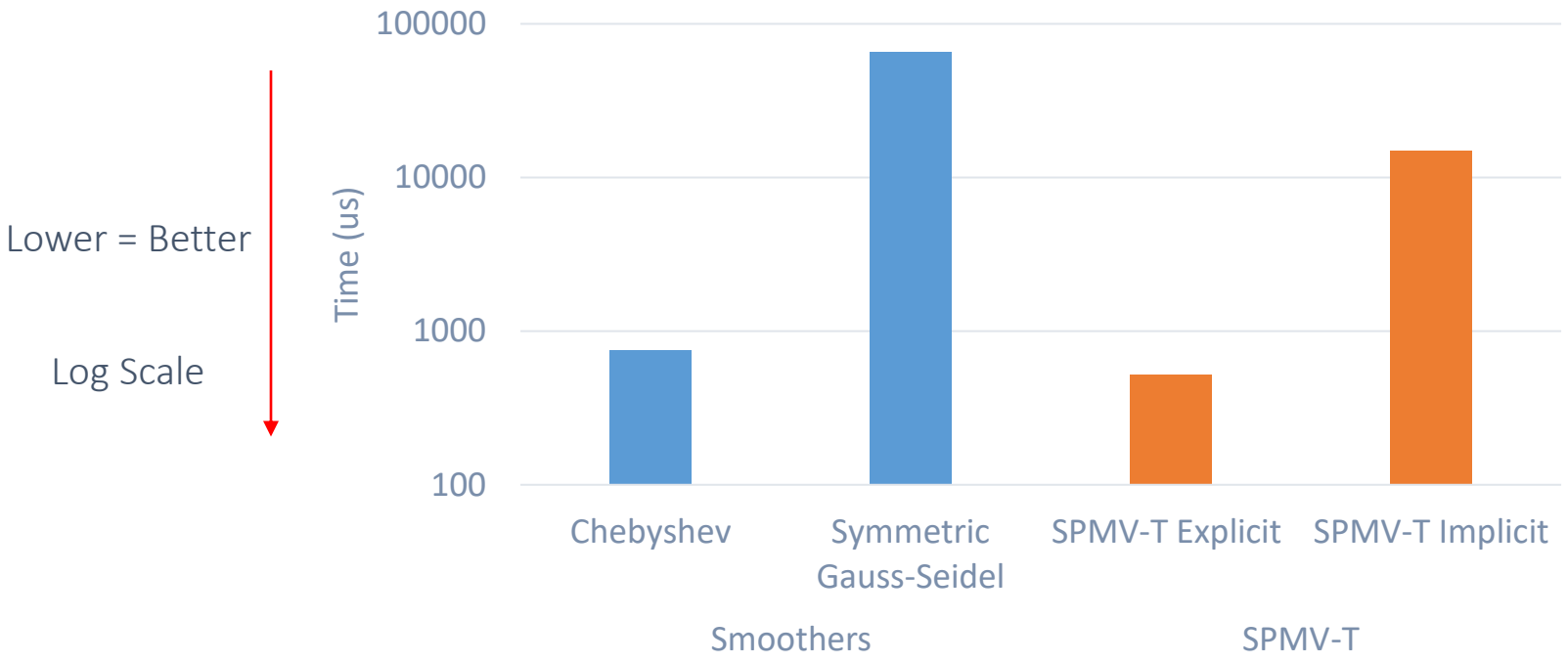
- To get single_task to work as advertised you need compiler hints!
- Either [[intel::ivdep]] on the loop or [[intel::kernel_restrict_args]] on the single_task itself.
- Example for small BLAS-1 style kernels: Cost per kernel for vector size 100



- NOTE: Not all kernels lend themselves to parallel_for / ivdep…

Performance Implications of Lack of Write Caching

- Smoothers: Chebyshev can be parallel_for. SGS has loop-carried dependencies.
- SPMV-T: Explicit has random access reads. Implicit has random access read/writes.



- Takeaway #1: Loop-carried dependency inhibits pipelining.
- Takeaway #2: Random access read/writes inhibit pipelining.

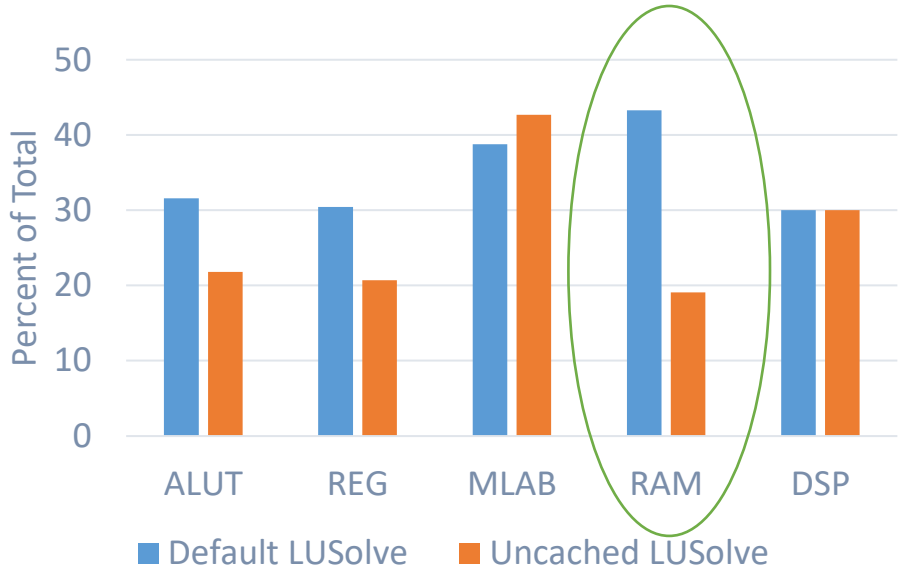
Resource Reduction w/ Load-Store Unit Options

- LUsolve is 0.4% of total time on CPU Laplace2D run.

ALUT = Adaptive Lookup Tables
REG = Registers
MLAB = Memory Local Array Blocks
RAM = Memory
DSP = Digital Signal Processor Blocks

- Laplace2D Tests

Lower = Better



- How much slower is the uncached LUSolve?

Kernel	Mean	Stdev
Default LUSolve	4250 us	11 us
Uncached LUSolve	4350 us	8 us

- Conclusion: Uncached LSUs have a negligible performance impact for real resource savings.

CPU vs. FPGA Bake-off

Laplace2D: 5pt stencil, 10k unknowns. Brick3D: 27pt stencil, 64k unknowns

Time per multigrid preconditioned CG solve

	Laplace2D	Brick3D
oneAPI-1	1,430 us	8,504 us
oneAPI-18	1,906 us	3,006 us
FPGA	7,760 us	600,000 us

FPGAs take 4x to 70x longer than oneAPI on CPU.

Power/Energy for HPCG run on Laplace2D

	Energy	Peak Power
oneAPI-1	134 J	165W / 198W
oneAPI-18	272 J	165W / 198W
FPGA	853 J	67 W

Takeaway #1: Long runtimes swamp lower power.

Takeaway #2: Lots of optimization left to do!

CPU energy via RAPL, power is TDP/PL2.

FPGA board power via OPAE.

Lessons Learned & Future Directions

- FPGAs via oneAPI are substantially easier to program than System Verilog, but…
  - Programming bottlenecks (e.g. reductions, compiler directives for single_task)
  - Launch/wait latency, host/device transfer too expensive.
  - Lack of write caching hurts.
  - Lack of 64-bit atomics forces us back to less-performant single_task w/o directives.

- Where we want to go
  - Kernel replication (data parallelism) to use all of the memory bandwidth.
  - Comparing HBM vs. DDR for on-board memory.
  - Pipelining between kernels to reduce memory access costs (not easy).
  - FPGA/MPI interaction, both via the PCIe bus and on-board NICs.