Lattice QCD on QPACE 4

Asia-Pacific Symposium for Lattice Field Theory
APLAT 2020

Peter Georg, **Nils Meyer**, Dirk Pleiter, Stefan Solbrig, Tilo Wettig

August 4, 2020

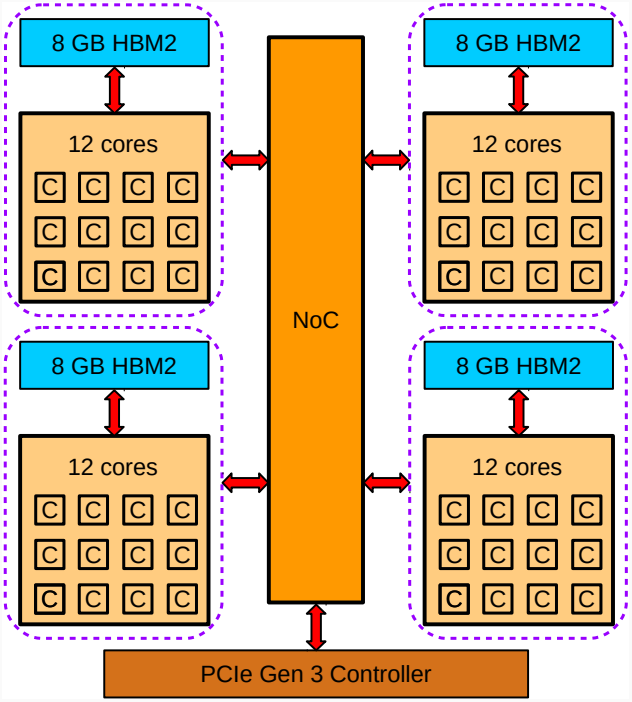
University of Regensburg (Regensburg, Germany)

QPACE 4

- Latest member of QCD PArallel Compute Engine (QPACE) series
- First Fujitsu A64FX installation in Europe
  - Fujitsu PRIMEHPC FX700 [PrimeHPC]
  - Deployed June 2020
- 64 Fujitsu A64FX model FX700 CPUs (1.8 GHz)
  - Armv8.2 + Scalable Vector Extension (SVE) instruction set
  - 177 / 354 TFlop/s peak in double precision (DP) / single precision (SP)
  - 2048 GB HBM2 memory total
  - InfiniBand EDR interconnect (100 Gbit/s)
- Open-source software stack
  - CentOS 8.2, gcc 10.1, OpenMPI 4.0.2
  - GlusterFS parallel filesystem
  - Grid [Boyle et al.:Latt15]
  - Grid Python Toolkit (gpt) [Lehner et al.:20]

Fujitsu A64FX model FX700

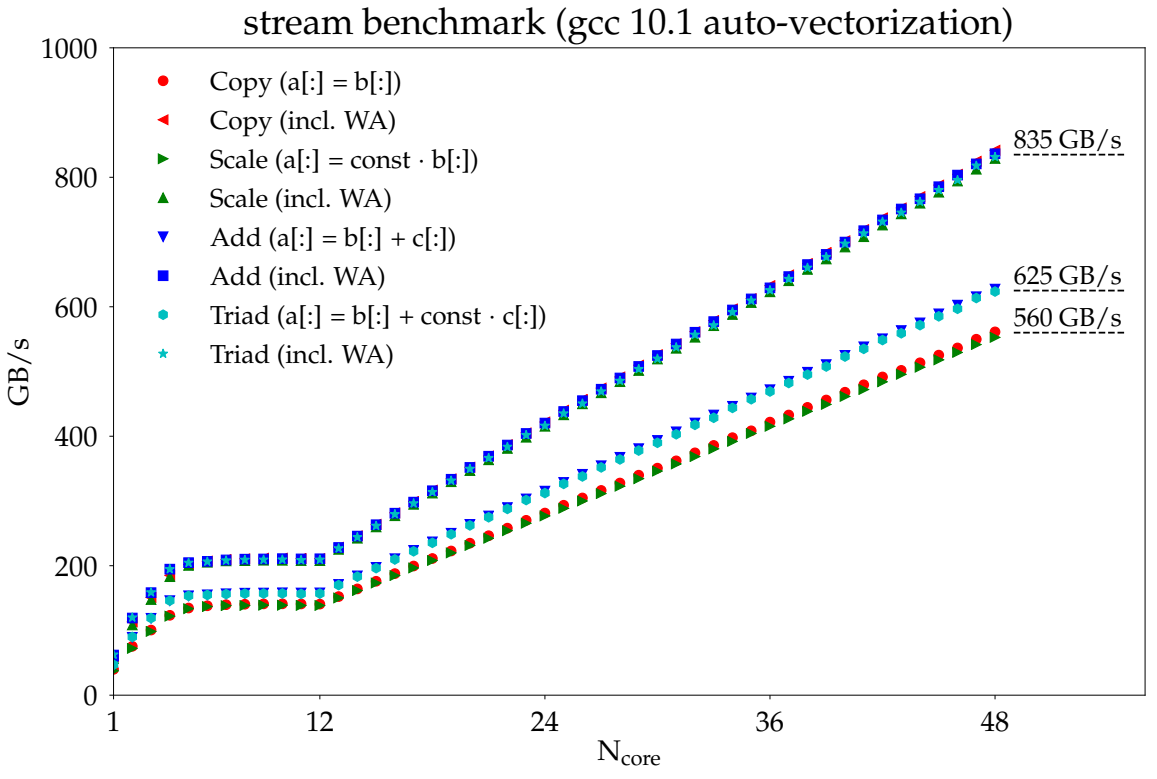
- 4 Core Memory Groups
  - 12 cores each, no assistant core
  - 8 MB shared L2 cache
  - 8 GB HBM2 memory
- Each core
  - 64 kB L1d cache
  - 1 hardware thread
  - 2 × 512-bit vector pipelines
    - dual-issue FMA
    - quarter-issue complex FMA (riri layout[1])
    - single-issue $x \pm iy$ (riri layout)
  - 32 architecture 512-bit vector registers
- 1 PCIe Gen3 x16 interface (no Tofu)



- A64FX Microarchitecture Manual available [A64FX]

[1] Alternating real and imaginary parts

Memory Throughput



- $N_{core} > 12$: data throughput scales with number of cores in use
  - Triad: up to 840 GB/s reported for FX1000
- Caches implement write-back policy
  - Write Allocation (WA): cache block (256 bytes) load from memory on write miss
    - Reduces effective memory throughput

SVE Programming Models

- Arm C Language Extensions (ACLE) for SVE (SVE ACLE) [Arm:SVEACLE]
  - Provides access to sizeless SVE vector types and most SVE instructions in C/C++
  - Inclusion of `arm_sve.h` header file enables SVE ACLE in SVE compilers
- Vector-length agnostic (VLA) programming model
  - Official SVE programming model
  - Size of vector registers unknown at compile time
  - Restrictions on usage of SVE ACLE vectors include, but are not limited to*
    - `sizeof()` not applicable
    - Not allowed as member of unions, structures and classes
    - Not allowed as types of array elements and C++ STL containers like std::vector
- Vector-length specific (VLS) programming model (supported since May 2020)
  - Sized vectors derived from sizeless SVE ACLE vector types (`typedef` declaration)
    - Restrictions (*) do not apply to sized vector
    - Compatible with SVE ACLE functions
    - Mixing VLS and VLA programming models possible
  - Supported as of gcc 10 (released May 2020)
  - Announced for LLVM 11 at ISC 2020 [Arm:ISC20]

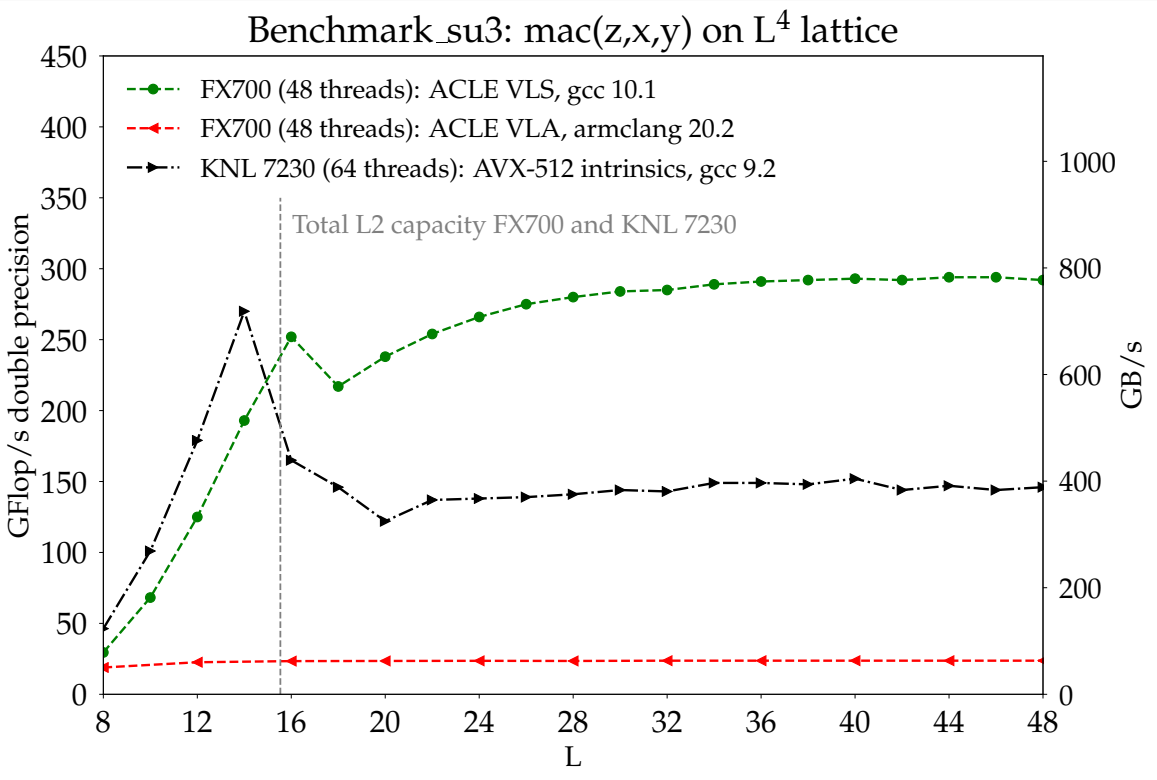
Porting Grid to A64FX

- Research contract with Arm UK maintaining armclang and gcc

Compiler	A64FX support -mcpu=a64fx	ACLE VLA	ACLE VLS -msve-vector-bits=512
armclang 20.2 (LLVM 9)	✓	✓	✗
gcc 10.1	✗	✓	✓

- Core of Grid's architecture abstraction is a template class `Grid_simd` with architecture-specific member data types (`SIMD_*` types)
  - ACLE VLA implementation (2018 [Meyer et al.:Latt18, Meyer et al.:Cluster18])
    - Sizeless SVE ACLE vectors not feasible as `SIMD_*` types
    - Ordinary arrays as member data, SVE ACLE only for data processing within functions
    - Explicit vector load/store accessing arrays
    - Excessive scalar copy operations introduced by compilers handling `Grid_simd` class
  - ACLE VLS implementation (2020)
    - 512-bit sized vectors as `SIMD_*` types
    - `SIMD_*` types as function parameters, in function bodies and as return types
    - Vector load/store generated by compiler
  - VLA and VLS codes fully vectorized, hardware support processing complex (riri layout)
- Access to Fujitsu FX1000 cluster with Tofu interconnect since Dec. 2019

SU(3) Matrix Multiplication



- Independent SU(3) matrix multiplication $z = x \times y$ on each lattice site
- FX700
  - ACLE VLS: good code quality, best performance if data resides in HBM2
  - ACLE VLA: excessive scalar copy operations limit performance
- KNL 7230: performance drop if data exceeds L2 and resides in MCDRAM

Wilson Dslash

- Specialization for A64FX
- Single-node performance in GFlop/s (normalized to 1320 Flop/s per site)

```
$ Benchmark_wilson --grid *.*.*.* --mpi 1.1.1.1|4 --dslash-asm
```

	gcc 10.1 ACLE VLS				armclang 20.2 ACLE VLA			
Volume	1 MPI rank		4 MPI ranks		1 MPI rank		4 MPI ranks	
	DP	SP	DP	SP	DP	SP	DP	SP
$16^3 \times 32$	336	635	275	490	387	746	176	324
$24^3 \times 48$	351	711	312	620	413	843	247	478
$32^3 \times 64$	344	694	317	633	397	814	274	542

- armclang outperforms gcc using 1 MPI rank
  - Performance up to 2.4x KNL 7230
- Communication overhead using 4 MPI ranks (armclang: excessive copies)
- Performance penalty due to L1d cache misses despite software prefetching, likely due to competition for cache ways

Domain Wall

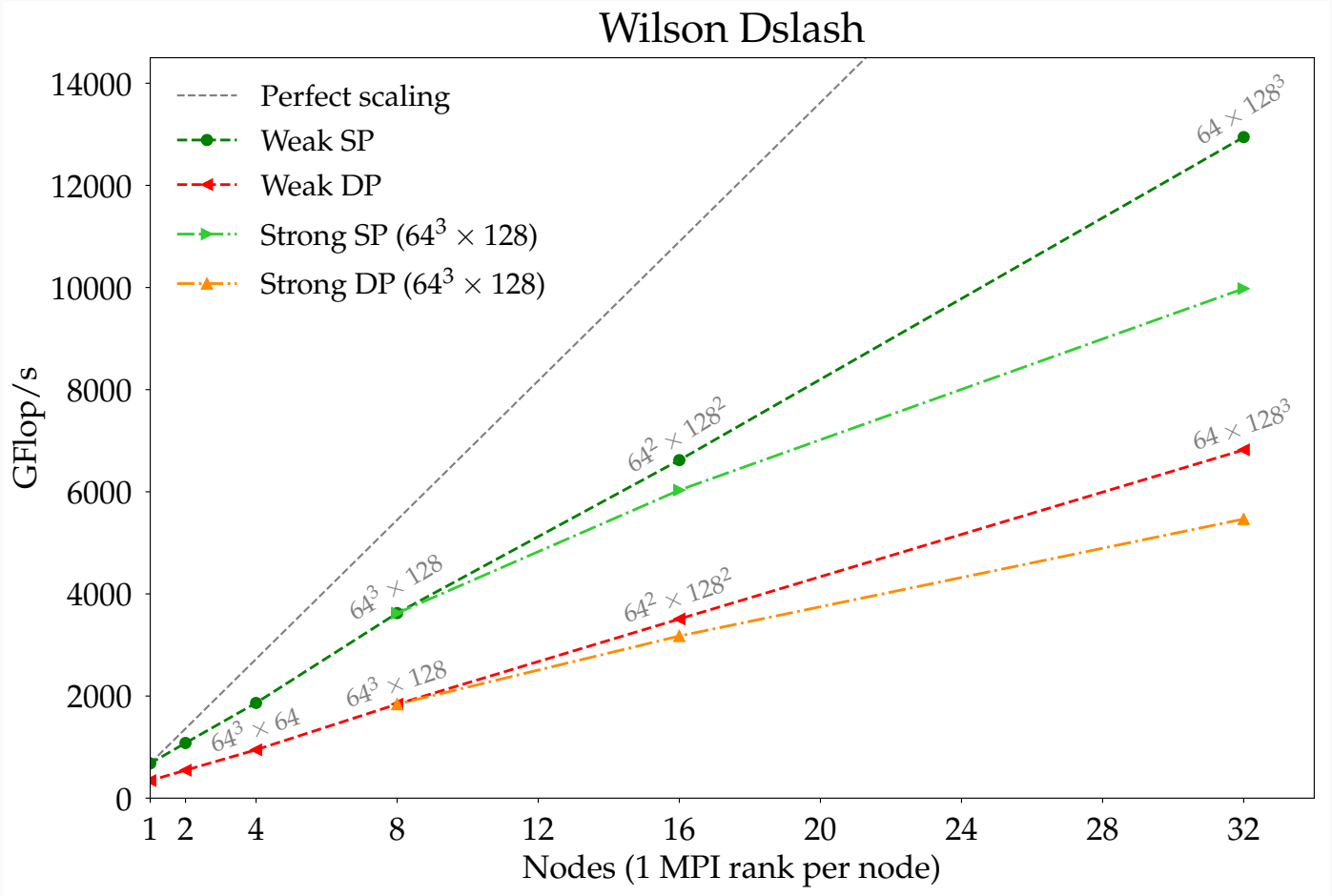
- Implementation details
  - Wilson Dslash kernel (slide 8), 4d gauge and 5d fermion fields
  - Gauge field reuse in L1d cache traversing 5-direction in innermost loop
- Single-node performance in GFlop/s (normalized to 1320 Flop/s per site)

```
$ Benchmark_dwf --grid *.*.*.* -Ls * --mpi 1.1.1.1|4 --dslash-asm
```

	gcc 10.1 ACLE VLS				armclang 20.2 ACLE VLA			
Volume	1 MPI rank		4 MPI ranks		1 MPI rank		4 MPI ranks	
	DP	SP	DP	SP	DP	SP	DP	SP
$16^3 \times 32 \times 8$	362	712	320	627	467	923	233	444
$16^3 \times 32 \times 16$	363	724	323	642	466	928	234	455
$24^3 \times 48 \times 8$	359	723	314	673	463	927	280	554

- Again armclang outperforms gcc using 1 MPI rank
  - Performance up to 2x KNL 7230
- Again communication overhead using 4 MPI ranks (armclang: excessive copies)
- Again performance penalty due to L1d cache misses
  - Gauge field reuse not as beneficial as intended

MPI Scaling



- ACLE VLS, gcc 10.1, OpenMPI 4.0.2
- QPACE 4 network topology: 2 partitions with 32 nodes each
  - Weak scaling feasible
  - Only slight performance penalty for strong scaling

Summary and Outlook

- QPACE 4 is latest member of QPACE series
  - 64 Fujitsu A64FX model FX700
  - Open-source software stack
- ACLE VLS superior to VLA for Grid
  - VLS support announced for LLVM 11
- Optimization opportunities
  - Elimination of write allocation inserting `DC ZVA` (Zero Fill)
  - Mitigation of cache miss penalties using Sector Cache feature, disclosure of details announced for Sep. 2020 by Fujitsu
- Grid for A64FX sources available

  https://github.com/nmeyer-ur/Grid/tree/feature/a64fx-2

  `$ configure --enable-simd=A64FX` (gcc 10 only)
- Grid is ready for testing on Fugaku

We thank the HPC tools team at Arm UK for support
and Mitsuhisa Sato and Yuetsu Kodama at RIKEN Japan for valuable discussions

References

Fujitsu. FUJITSU Supercomputer PRIMEHPC. 2020 [https://www.fujitsu.com/global/products/computing/servers/supercomputer/index.html].

Peter Boyle, Azusa Yamaguchi, Guido Cossu, and Antonin Portelli. Grid: A next generation data parallel C++ QCD library. *Proceedings of LATTICE 15* (2015), 023 [arXiv:1512.03487].

Christoph Lehner et al. GPT - Grid Python Toolkit. 2020 [https://github.com/lehner/gpt].

Fujitsu. A64FX. 2020 [https://github.com/fujitsu/A64FX].

Arm. ARM C Language Extensions for SVE. 2020 [https://developer.arm.com/documentation/100987/latest].

Will Lovett. SVE in LLVM. *ISC 2020* [https://hps.vi4io.org/_media/events/2020/llvm-cth20_lovett.pdf].

Nils Meyer, Dirk Pleiter, Stefan Solbrig, and Tilo Wettig. Lattice QCD on upcoming Arm architectures. *Proceedings of LATTICE 18* (2019), 316 [arXiv:1904.03927].

Nils Meyer, Peter Georg, Dirk Pleiter, Stefan Solbrig, and Tilo Wettig. SVE-enabling Lattice QCD Codes. *IEEE International Conference on Cluster Computing (CLUSTER)* (2018), 623 [arXiv:1901.07294].

Nils Meyer. Grid for A64FX. 2020 [https://github.com/nmeyer-ur/Grid/tree/feature/a64fx-2].