Final report for the UNC subcontract with JLabs for the LQCD SciDAC 4.

May 31, 2019

Robert Fowler -- PI for UNC Subcontract.

This is a collection of raw text fragments to be integrated into JLab's report.

Note that there are embedded comments <<Bracketed like this>> that

can/should be removed from the final.

Period of reporting for the UNC sub-contract 12/5/20188 to 5/31/2019.

<<Materials below organized by the requirements list from Ted.>>

1) Project participants:

Robert Fowler, Ph.D. UNC/RENC . Director of High Performance Computing

at RENCI and Adjunct Professor of Computer Science at UNC Chapel Hill.

Project role: Research contributions as Ph.D. advisior to Diptorup Deb.

Active participation in the design and implementation of the QUARC

framework.

Fowler was supported at 14.5% from 12/18 through 5/18.

Diptorup Deb, UNC/RENCI. Ph.D. Candidate in Computer Science at

UNC Chapel Hill. Principal researcher in the design and implementation of

QUARC. This is the subject of Deb's Ph.D. dissertation “Automating

Data-Layout Decisions In Domain-Specific Languages” which was successfully

defended in Spring of 2019. Diptorup graduated from UNC in May 2019.

He will be taking a job with Intel Corporation.

One hundred percent of Diptorup’s graduate RA stipend was paid by this project.

2) Progress

<<Intro material from the UNC SOW >>

QUARC represents the application to LQCD of methods for implementing statically

compiled data parallel domain-specific programming languages. In QUARC,

a domain-specific extension to C++ is defined using template meta-programming.

Unlike other approaches in which template expansion is used to perform localized

code and loop instantiation in the compiler front end, the instantiated templates

are not meant to be compiled into executable code using a standard compiler,

rather they are used to pass high-level abstract constructs and attributes of those

constructs transparently through a standard compiler front-end.

In prior work we demonstrated performance for QUARC-generated parallel

code competitive with QPhiX on shared memory domains using

Intel Xeon and Xeon Phi platforms.

In first year of the UNC subcontract we extended and generalized

QUARC to the full set of constructs needed to program full LQCD analysis codes.

During the second year we redesigned and re-implemented parts of QUARC

to facilitate the integration of modules written in QUARC with existing

LQCD code bases. On CPU-based systems, including

Xeon Phi Knights Landing, QUARC has been demonstrated to generate validated

code for several important LQCD kernel with performance comparable to that obtained

using the QphiX library.

Highlights of the implementation:

* QUARC is based on using innovative compilation techniques for embedding domain specific languages into C++ for compilation using the existing LLVM compiler infrastructure. LLVM is competitive with the best commercial compilers and it is supported by many major corporations and DOE national labs. QUARC uses much of LLVM’s native optimization infrastructure for both high- and low-level optimization.
* A unique contribution of QUARC is that it supports an implicitly data-parallel array language model. This allows analyses to be performed at an abstract level. This is used to drive data layout transformations to enable generation of cache-friendly code and extremely efficient vectorized code that is up to 2X faster than standard commercial methods.
* QUARC code generation uses polyhedral compilation methods to partition the application problem for parallel execution. This includes automatic identification of halo data to be shared, buffer management, data marshaling, and message transport/sharing. Because an appropriate Open MP for LLVM did not exist for most of the project, QUARC uses MPI-3 shared memory on a node and MPI-2 single-sided communication across nodes.
* The QUARC array transformation system supports modules that can be embedded in full programs rather than single-kernel benchmarks. This has included a simplification and eneralization of the notation for the transformations into an array transformation language ATL) that can be generated from a variety of sources to optimize the code. This facilitates

autotuning.

Some highlights of the performance results:

* We determined that appropriate data layout transformations such as used in QPhiX and QUARC are essential for getting the best performance. For most architectures we determined that there is an intuitive set of transformations that are equivalent to one another within a few percent. Small differences occur due to the number of instructions needed to deal with boundaries between subproblems, cache friendliness, and other details. Thus, a good guess will result in “really good” code, but to get the last few percent will require either empirical experimentation (autotuning) or the use of a detailed architectural description of the system.
* For Wilson and K-S Dslash kernels, QUARC generates code within a few percent (either way)

of QphiX. QphiX can be 15 to 20 percent faster than QUARC because QphiX co-schedules adjacent fragments of the lattice to take advantage of shared L2 caches.

* A complete conjugant gradient solver using the K-S Dslash was implemented for MILC. This solver is 2X faster than the solver in the MILC distribution. This solver is a drop in replacement for the one in the distribution. Because a data layout transformation is performed at the beginning and end of each solve, the amount of available speedup increases as the size of the problem increases.
* We are pursuing an effort beyond the funding of this project to use QUARC in other domains.

Status of the code:

* QUARC is open source.
* It is currently in a GIT repository on Bitbucket. Diptorup is pursuing the option of relocating it to the Jlab Github.

(3) QUARC Publications .

Diptorup Deb, “Automating Data-Layout Decisions In Domain-Specific Languages”, Ph.D. Dissertation, Department of Computer Science, University of North Carolina at Chapel Hill,

May 2019. (Acknowledges funding on this project and previous SciDAC 3 funding.)

<< The following appeared during the overall project period, but before our

funding started in December.>>

Diptorup Deb, Robert J. Fowler, and Allan Porterfield. QUARC: an optimized DSL

framework using LLVM. In The Fourth Workshop on the LLVM Compiler

Infrastructure in HPC (LLVM-HPC2017), Denver, CO, November 2017. ACM SIGHPC.

(acknowledges SciDAC3.)

Diptorup Deb, Robert Fowler, and Allan Porterfield. QUARC: An Array Programming Approach to High Performance Computing, volume 10136 of Lecture Notes in Computer Science. Springer International Publishing, Rochester, NY, September 2016. (acknowledges SciDAC 3.)

(4) Reports and other Documents.

RENCI TR-18-01 QUARC: A DSL Framework to Achieve Mixed-mode

Parallelism Using a System of Array Transformations,

Diptorup Deb, Robert Fowler, and Allan Porterfield, January, 2018. (acknowledges this project)