PROCEEDINGS OF SCIENCE

Porting Lattice QCD benchmark to upcoming STX stencil/tensor accelerator

Simon Schlepphorst[a,*] **and Stefan Krieg**[a,b]

[a] *Jülich Supercomputing Centre (JSC) & Center for Advanced Simulation and Analytics (CASA), Forschungszentrum Jülich, 54245 Jülich, Germany*

[b] *Helmholtz-Institut für Strahlen- und Kernphysik (HISKP), Rheinische Friedrich-Wilhelms-Universität Bonn, 53115 Bonn, Germany*

E-mail: s.schlepphorst@fz-juelich.de, s.krieg@fz-juelich.de

Developed under the European Processor Initiative (EPI), the STX stencil/tensor accelerator aims to achieve a 5-10x higher energy efficiency over general purpose compute units. The architecture consists of specialized MIMD compute units which are supported and controlled by RISC-V cores. We describe a co-design effort between hardware, software, and application development focused around porting a LQCD benchmark to this new architecture.

The 41st International Symposium on Lattice Field Theory (LATTICE2024)
28 July - 3 August 2024
Liverpool, UK

*Speaker

PoS(LATTICE2024)451

1. Project and Hardware

The STX Project is a joint effort between the Fraunhofer Institute for Integrated Circuits IIS, the Fraunhofer ITWM and partners from industry to develop the Stencil and Tensor Accelerator (STX). Funded by the European Processor Initiative (EPI), this will build expertise on accelerator design and manufacturing in Europe. Another aspect of this development is early co-design between hard- and software, so both can improve in iterative design changes. Being part of this co-design effort we can share preliminary results, even though the hardware is not in production yet.

Starting from the smallest unit, the STX is built on specialized floating point engines (SPU). Eight of those SPUs, together with RISCV 32-bit management cores, a DMA unit and 128 kB of user managed scratch pad memory form a MIMD unit (Cluster), see figure 1. This scratchpad memory (TCDM) is a drop in replacement for cache usually found on other architectures, with the caveat that the user has to manage all the prefetching. For energy efficiency reasons, the usual hardware prefetcher is omitted.

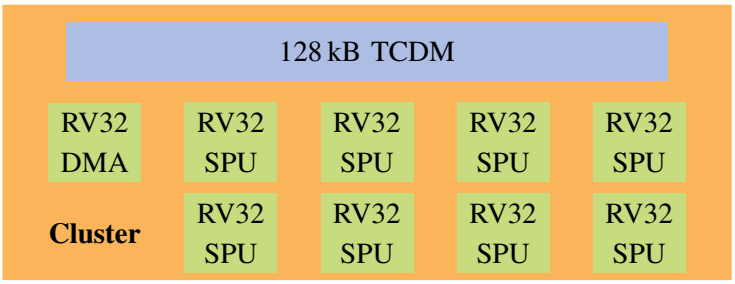

Figure 1: Cluster MIMD core topology

The full STX chip then contains 16 Microtiles with 4 Cluster each, 16 GB of HBM2e memory and an interface to the PCIe Bus. Figure 2 depicts the topology as well as the expected bandwidth between each component. Clocking at 1 GHz, the expected performance for 32-bit floats is 1 TFLOP per chip.

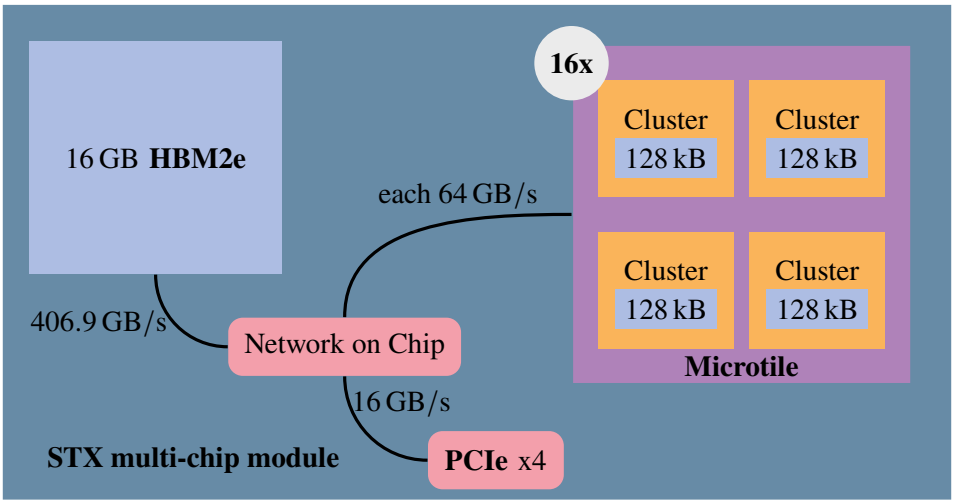

Figure 2: STX multi-chip module bandwidth flowchart

Finally four of these STX Chips can fit on one PCIe card. The Prototype shown in figure 3 was presented at ISC 2024. This card will have a power envelope of 200 W.

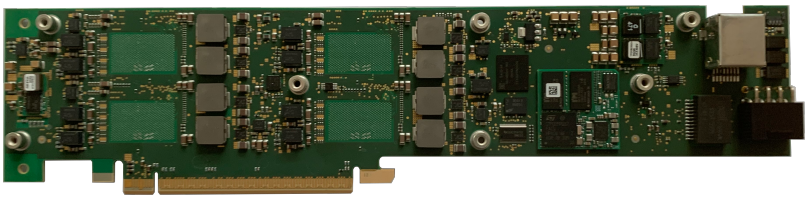

Figure 3: Prototype PCIe card with space for 4 STX multi-chip modules, unveiled on ISC 2024.

2. Programming Challenges

Since the main target is energy efficiency, the hardware has special features programmers must utilize. Address calculation for each element accessed by a SPU is done by a specialized hardware circuit. This circuit can handle up to 3D arrays of structs, which are composed of at most 32 basic data types.

To save chip area, SPU compute cores can only process floating point numbers. Thus most integer arithmetic has to be done at compile time. An example how to achieve 4D indexing is shown in figure 4.

```
using chi_t = std::array<double, 6>;
const auto Nx{8u}, Ny{2u}, Nz{2u}, Nt{2u};

void process(){
  const int num_cluster = stx::get_num_clusters();
  auto &cluster = stx::get_cluster();
  spu_data_3d<chi_t> data = cluster.memory_alloc_3d<chi_t>(Nx, Ny, Nz * Nt);
#pragma omp parallel
  {
    if (cluster.is_compute_core()) {
#pragma omp target
      {
#pragma stx worksharing(chunks)
        for (auto x = 0; x < Nx; x++) {
          for (auto y = 0; y < Ny; y++) {
            for (auto t = 0; t < Nt; t++) {
#pragma unroll
              for (auto z = 0; z < Nz; z++) {
                data[x][y][z * Nt + t][0] = 3.14;
                data[x][y][z * Nt + t][1] = 6.28;
}}}}}}}}
```

Figure 4: Example code using 4D access on arrays

3. Implementation strategies

For this study we focus on the Kogut-Susskind (staggered) fermion definition of the Dirac operator [1] [2] . We benchmark the stencil application, which is given by:

$$\chi(n) = \sum_{\mu=1}^{4} U_\mu(n)\chi(n+\hat{\mu}) - U_\mu^\dagger(n-\hat{\mu})\chi(n-\hat{\mu}) \tag{1}$$

$$\mu = 1, 2, 3, 4 \qquad \chi \in \mathbf{C}^3 \qquad U \in \mathrm{SU}(3) \in \mathbf{C}^{3\times 3}$$

In Equation 1, constant factors have been suppressed. The gauge fields can be compressed using two rows of the matrix:

$$\begin{pmatrix} U^{00} & U^{01} & U^{02} \\ U^{10} & U^{11} & U^{12} \\ U^{20} & U^{21} & U^{22} \end{pmatrix} = \begin{pmatrix} a \\ b \\ (a \times b)^* \end{pmatrix}$$

which allows us to trade floating point operations for memory bandwidth. The Arithmetic Intensity per lattice site for both the full stored gauge field and the two row compressed one, is summarized in table 1.

dtype	FLOP	Bytes moved	AI
double	570	1584	0.36
float	570	792	0.72
double	1530	1200	1.28
float	1530	600	2.55

Table 1: General Arithmetic Intensity (AI [FLOP/B]) of a staggered fermion kernel for a single site. Top: full gauge field format. Bottom: two row compressed gauge field.

To mask memory latency we want to use at least double buffering. This cuts the available local TCDM cache in half, leaving us with just 64 kB to work with. We choose to work on small 4D chunks in local cache, but cycle through the gauge field one direction μ at a time. For the example of a [8, 4, 3, 2] output volume, we need to load gauge field slices of volume $[9, 4, 3, 2]_{\mu=1}$, $[8, 5, 3, 2]_{\mu=2}$, $[8, 4, 4, 2]_{\mu=3}$ and $[8, 4, 3, 3]_{\mu=4}$ one after another.

4. Results

Assuming we will be able to saturate 80% of the specified HBM bandwidth, we can extrapolate the expected performance with regard to the memory subsystem. This is shown in table 2.

dtype	volume	sites	Bytes moved	AI	% TCDM	sites/s	sites/W
double	[8, 2, 2, 2]	64	85 632	0.43	40.14	2.43×10^8	4.87×10^6
float	[8, 4, 3, 2]	192	105 408	1.04	45.64	5.93×10^8	11.86×10^6
double	[8, 3, 2, 2]	96	93 312	1.57	47.91	3.35×10^8	6.70×10^6
float	[8, 4, 4, 2]	256	103 680	3.78	49.85	8.04×10^8	16.08×10^6

Table 2: Arithmetic Intensity for local volumes still fitting in half of the TCDM. Sites/s and sites/W are extrapolated assuming we can saturate 80% of the specified HBM Bandwidth. Top: full gauge field format. Bottom: two row compressed gauge field.

The early development-stage compiler is accompanied by a basic software simulator of one cluster, which reports cycle and instruction counters. Using this information, we can predict the compute performance of our implementation as shown in table 3.

dtype	volume	sites	FLOP/site	cycles	sites/s
double	[8, 2, 2, 2]	64	624	3317	12.34×10^8
float	[8, 4, 3, 2]	192	576	10 945	11.23×10^8
double	[8, 3, 2, 2]	96	1711	13 484	4.56×10^8
float	[8, 4, 4, 2]	256	1583	35 344	4.64×10^8

Table 3: Performance counters reported by the software simulator. Here the total FLOPs are normalized to FLOP/site so the values are comparable to table 1. Sites/s are extrapolated from the specified chip clock rate of 1 GHz and 64 Clusters per chip. Top: full gauge field format. Bottom: two row compressed gauge field.

The SPU compute core has a two lane SIMD capability when operating in float mode. That feature wasn't used for this work. We plan to explore it in future works, since this could give us the edge to utilize the chip to the fullest when using two row compressed gauge fields. Another feature to explore in the future is using the network-on-chip for halo exchanges between Cluster units. This could give us another tool for saving the scarce bandwidth of the HMB memory.

Acknowledgments

- The STX Project received funding from the EPI project under Specific Grant Agreement No 101036168 (EPI SGA2).
- The STXDemo Project received funding from the German Federal Ministry of Education and Research.
- This work was supported in part by DFG project 460248186 (PUNCH4NFDI).

References

[1] C. Gattringer and C.B. Lang, *Quantum chromodynamics on the lattice*, vol. 788, Springer, Berlin (2010), 10.1007/978-3-642-01850-3.

[2] J. Kogut and L. Susskind, *Hamiltonian formulation of Wilson's lattice gauge theories*, *Phys. Rev. D* **11** (1975) 395.