Arm A64fx and Post-K: Game Changing CPU & Supercomputer for HPC and its Convergence of with Big Data / AI

Satoshi Matsuoka
Director, Riken Center for Computational Science / Professor, Tokyo Institute of Technology

Hyperion HPC User’s Forum
Santa Fe, New Mexico
20190403
Riken Center for Computational Science (R-CCS)
World Leading HPC Research, active collaborations w/Universities, national labs, & Industry

Sci. of Computing
Foundational research on computing in high performance for K, Post-K, and beyond towards the "Post-Moore" era, including future high performance architectures, new computing and programming models, system software, large scale systems modeling, big data analytics, and scalable artificial intelligence / machine learning

Sci. by Computing
Breakthrough Science & Technology using high performance computing capabilities of K, Post-K and beyond to address the issues of high public concern, in areas such as life sciences, climate & environment, disaster prediction & prevention, advanced manufacturing, applications of machine learning for Society 5.0.

Mutual Synergy

Sci. for Computing
High Resolution, High Fidelity Analysis & Simulation

Novel Future High Performance Computing Architectures & Algorithms

New Materials & Electronic Devices e.g., Photonics, Neuromorphics, Quantum, Reconfigurable

Apr 1 2018 Became Director of Riken-CCS: Science, of Computing, by Computing, and for Computing
Arm64fx & Post-K (to be renamed) are:

- Fujitsu-Riken design A64fx ARM v8.2 (SVE), 48/52 core CPU
  - **HPC Optimized:** Extremely high package high memory BW (1TByte/s), on-die Tofu-D network BW (~400Gbps), high SVE FLOPS (~3Teraflops), various AI support (FP16, INT8, etc.)
  - Gen purpose CPU – Linux, Windows (Word), other SCs/Clouds
  - Extremely power efficient – > **10x power/perf efficiency for CFD benchmark** over current mainstream x86 CPU

- **Largest and fastest supercomputer to be ever built circa 2020**
  - > 150,000 nodes, superseding LLNL Sequoia
  - > 150 PetaByte/s memory BW
  - Tofu-D 6D Torus NW, 60 Petabps injection BW (10x global IDC traffic)
  - 25~30PB NVMe L1 storage
  - ~10,000 endpoint 100Gbps I/O network into Lustre
  - The first ‘exascale’ machine (not exa64bitflops but in apps perf.)
Post-K: The Game Changer

1. Heritage of the K-Computer, HP in simulation via extensive Co-Design
   • High performance: up to x100 performance of K in real applications
   • Multitudes of Scientific Breakthroughs via Post-K application programs
   • Simultaneous high performance and ease-of-programming

2. New Technology Innovations of Post-K
   • High Performance, esp. via high memory BW
     Performance boost by “factors” c.f. mainstream CPUs in many HPC & Society 5.0 apps via BW & Vector acceleration
   • Very Green e.g. extreme power efficiency
     Ultra Power efficient design & various power control knobs
   • Arm Global Ecosystem & SVE contribution
     Top CPU in ARM Ecosystem of 21 billion chips/year, SVE co-design and world’s first implementation by Fujitsu
   • High Perf. on Society 5.0 apps incl. AI
     Architectural features for high perf on Society 5.0 apps based on Big Data, AI/ML, CAE/EDA, Blockchain security, etc.

Global leadership not just in the machine & apps, but as cutting edge IT

Technology not just limited to Post-K, but into societal IT infrastructures e.g. Clouds
Post K A64fx Processor is…

- an Many-Core ARM CPU…
  - 48 compute cores + 2 or 4 assistant (OS) cores
  - Brand new core design
  - Near Xeon-Class Integer performance core
  - ARM V8 --- 64bit ARM ecosystem
  - Tofu-D + PCIe 3 external connection

- ...but also an accelerated GPU-like processor
  - SVE 512 bit vector extensions (ARM & Fujitsu)
    - Integer (1, 2, 4, 8 bytes) + Float (16, 32, 64 bytes)
  - Cache + scratchpad-like local memory (sector cache)
  - HBM2 on package memory – Massive Mem BW (Bytes/DPF ~0.4)
    - Streaming memory access, strided access, scatter/gather etc.
  - Intra-chip barrier synch. and other memory enhancing features

- GPU-like High performance in HPC, AI/Big Data, Auto Driving…
“Post-K” Chronology

(Disclaimer: below includes speculative schedules and subject to change)

- May 2018 A0 Chip came out, almost bug free
- 1Q2019 B0 Chip on hand, bug free, exceeded perf. target
- Mar 2019 “Post-K” manufacturing budget approval by the Diet, actual manufacturing contract signed and starts
- Aug 2019 End of K-Computer operations
- 4Q2019 “Post-K” installation starts
- 1H2020 “Post-K” preproduction operation starts
- 2020~2021 “Post-K” production operation starts (hopefully)
- And of course we move on…

Watch for announcements on “Post-K” technology commercialization by Fujitsu and its partner vendors RSN
Co-design for Post-K

(slides by Mitsuhisa Sato Team Leader of Architecture Development Team)

Deputy project leader, FLAGSHIP 2020 project
Deputy Director, RIKEN Center for Computational Science (R-CCS)

Analysis of applications to devise the most efficient solutions

Started in 2009 (before K)

Co-design from Apps to Architecture

- Architectural Parameters to be determined
  - #SIMD, SIMD length, #core, #NUMA node, O3 resources, specialized hardware
  - cache (size and bandwidth), memory technologies
  - Chip die-size, power consumption
  - Interconnect

- We have selected a set of target applications

- Performance estimation tool
  - Performance projection using Fujitsu FX100 execution profile to a set of arch. parameters.

- Co-design Methodology (at early design phase)
  1. Setting set of system parameters
  2. Tuning target applications under the system parameters
  3. Evaluating execution time using prediction tools
  4. Identifying hardware bottlenecks and changing the set of system parameters
Protein simulation before K

- Simulation of a protein in isolation
  Folding simulation of Villin, a small protein with 36 amino acids

Protein simulation with K

- All atom simulation of a cell interior
- Cytoplasm of Mycoplasma genitalium
Global cloud resolving model with 0.87 km-mesh which allows resolution of cumulus clouds

Month-long forecasts of Madden-Julian oscillations in the tropics is realized.

Heart Simulator

- Heartbeat, blood ejection, coronary circulation are simulated consistently.
- Applications explored
  - congenital heart diseases
  - Screening for drug-induced irregular heartbeat risk

Multi-scale simulator of heart starting from molecules and building up cells, tissues, and heart

UT-Heart, Inc., Fujitsu Limited
Co-design of Apps for Architecture

- **Tools for performance tuning**
  - Performance estimation tool
    - Performance projection using Fujitsu FX100 execution profile
    - Gives “target” performance
  - **Post-K processor simulator**
    - Based on gem5, O3, cycle-level simulation
    - Very slow, so limited to kernel-level evaluation

- **Co-design of apps**
  1. Estimate “target” performance using performance estimation tool
  2. Extract kernel code for simulator
  3. Measure exec time using simulator
  4. Feed-back to code optimization
  5. Feed-back to compiler
ARM for HPC - Co-design Opportunities

- ARM SVE Vector Length Agnostic feature is very interesting, since we can examine vector performance using the same binary.
- We have investigated how to improve the performance of SVE keeping hardware-resource the same. (in “Rev-A” paper)
  - ex. “512 bits SVE x 2 pipes” vs. “1024 bits SVE x 1 pipe”
  - Evaluation of Performance and Power (in “coolchips” paper) by using our gem-5 simulator (with “white” parameter) and ARM compiler.
  - Conclusion: Wide vector size over FPU element size will improve performance if there are enough rename registers and the utilization of FPU has room for improvement.

Note that these researches are not relevant to “post-K” architecture.

A64FX: Summary

- Arm SVE, high performance and high efficiency
  - DP performance: 2.7+ TFLOPS, >90%@DGEMM
  - Memory BW: 1024 GB/s, >80%@STREAM Triad

<table>
<thead>
<tr>
<th>Feature</th>
<th>A64FX</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISA (Base, extension)</td>
<td>Armv8.2-A, SVE</td>
</tr>
<tr>
<td>Process technology</td>
<td>7 nm</td>
</tr>
<tr>
<td>Peak DP performance</td>
<td>2.7+ TFLOPS</td>
</tr>
<tr>
<td>SIMD width</td>
<td>512-bit</td>
</tr>
<tr>
<td># of cores</td>
<td>48 + 4</td>
</tr>
<tr>
<td>Memory capacity</td>
<td>32 GiB (HBM2 x4)</td>
</tr>
<tr>
<td>Memory peak bandwidth</td>
<td>1024 GB/s</td>
</tr>
<tr>
<td>PCIe</td>
<td>Gen3 16 lanes</td>
</tr>
<tr>
<td>High speed interconnect</td>
<td>TofuD integrated</td>
</tr>
</tbody>
</table>

12x compute cores                          1x assistant core

CMG: Core Memory Group  NOC: Network on Chip

PCIe Controller  Tofu Interface
A64FX technologies: Core performance

- High calc. throughput of Fujitsu’s original CPU core w/ SVE
- 512-bit wide SIMD x 2 pipelines and new integer functions

![Core peak performance chart]

\[ C = \sum (A_i \times B_i) + C \]

- 16-bit INT8 partial dot product
- 8-bit INT8 partial dot product
- 32-bit Multiply and add
A64FX technologies: Scalable architecture

- Core Memory Group (CMG)
  - 12 compute cores for computing and an assistant core for OS daemon, I/O, etc.
  - Shared L2 cache
  - Dedicated memory controller

- Four CMGs maintain cache coherence with on-chip directory
  - Threads binding within a CMG allows linear speed up of cores' performance

**CMG configuration**

**A64FX chip configuration**
High Bandwidth

- Extremely high bandwidth in caches and memory
  - A64FX has out-of-order mechanisms in cores, caches and memory controllers. It maximizes the capability of each layer’s bandwidth.
A64FX: L1D cache uncompromised BW

- 128B/cycle sustained BW even for unaligned SIMD load

- “Combined Gather” doubles gather (indirect) load’s data throughput, when target elements are within a “128-byte aligned block” for a pair of two regs, even & odd

Maximizes BW to 32 bytes/cyc.
A64FX: Power monitor and analyzer

- Energy monitor (per chip)
  - Node power via Power API*1 (~msec)
  - Averaged power of a node, CMG (cores, L2 cache, memory) etc.
  *1: Sandia National Laboratory

- Energy analyzer (per core)
  - Power profiler via PAPI*2 (~nsec)
  - Fine grained power analysis of a core, L2 cache, and memory
  *2: Performance Application Programming Interface
A64FX: Power Knobs to reduce power consumption

- “Power knob” limits units’ activity via user APIs
- Performance/W can be optimized by utilizing Power knobs, Energy monitor & analyzer

Decode width: 4 to 2

Frequency reduction

EXA pipeline only

FLA pipeline only

HBM2 B/W adjust (units of 10%)
Preliminary performance evaluation results

- Over 2.5x faster in HPC & AI benchmarks than SPARC64 XIfx

<table>
<thead>
<tr>
<th>HPC</th>
<th>AI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Throughput tests</td>
<td>Application kernels</td>
</tr>
</tbody>
</table>

- Memory BW
- 512-bit SIMD
- Combined gather
- L1$ BW
- L2$ BW
- INT8 partial dot product

Performance increase normalized by SPARC64 XIfx:
- DGEMM: 2.51f
- STREAM Triad: 840 GB/s
- Fluid dynamics: 3.0x
- Atmosphere: 2.8x
- Seismic wave propagation: 3.4x
- Convolution FP32: 2.5x
- Convolution low precision: 9.4x
# Post-K A64fx A0 (ES) performance

<table>
<thead>
<tr>
<th></th>
<th>Performance / CPU</th>
<th>Machine Performance (HPC)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Peak TF (DFP)</td>
<td>Peak Mem. BW</td>
</tr>
<tr>
<td>Post-K A64fx (A0 Eng. Sample)</td>
<td>2.764/3.072</td>
<td>1024GB/s</td>
</tr>
<tr>
<td>Intel KNL</td>
<td>3.0464</td>
<td>600GB/s</td>
</tr>
<tr>
<td>Intel Skylake</td>
<td>1.6128</td>
<td>127.8GB/s</td>
</tr>
<tr>
<td>NVIDIA V100 (DGX-2)</td>
<td>7.8</td>
<td>900 GB/s</td>
</tr>
</tbody>
</table>
NAS Parallel Benchmark of FX100

[Slide by Ikuo Miyoshi, Fujitsu, SSKen2015]

OpenMP版を用いてノードあたり演算性能を評価

FX10に対するノードあたり性能向上比

Note: Haswell:
1 node = 2 chips
32 threads&cores
FX100: 1 node = 1 chip
32 threads&cores

FX100は、FX10比平均3.9倍、Haswell比平均1.3倍のノードあたり演算性能
Fiber (Post-K) MiniApp on FX100
[Slide by Ikuo Miyoshi, Fujitsu, SSKen2015]

■ 評価条件

<table>
<thead>
<tr>
<th>アプリ名</th>
<th>問題サイズ</th>
<th>評価区間</th>
<th>スレッド数 × プロセス数 (左からFX10、FX100、Haswell)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCS QCD</td>
<td>32 × 32 × 32 × 32</td>
<td>BiCGStab</td>
<td>16t × 2p 32t × 1p 16t × 2p</td>
</tr>
<tr>
<td>NICAM-DC-MINI</td>
<td>gl05rl00z80pe10</td>
<td>Dynamics</td>
<td>3t × 10p</td>
</tr>
<tr>
<td>FFB-MINI</td>
<td>1,048,576要素</td>
<td>MAIN_LOOP</td>
<td>1t × 32p 8t × 4p 1t × 32p</td>
</tr>
<tr>
<td>FFVC-MINI</td>
<td>256 × 256 × 256</td>
<td>Total</td>
<td>4t × 8p 16t × 2p 2t × 16p</td>
</tr>
<tr>
<td>NTChem-MINI</td>
<td>taxol</td>
<td>RIMP2_Driver</td>
<td></td>
</tr>
</tbody>
</table>

■ 測定結果

FF10に対するノードあたり性能向上比

Note: Haswell: 1 node = 2 chips
32 threads&cores
FX100: 1 node = 1 chip
32 threads&cores

FX100は、FX10比平均3.3倍、Haswell比平均1.4倍のノードあたり性能

注) FFVCには開発版コンパイラ、NTChemには開発版数学ライブラリを使用。QCDではセクタキャッシュ利用、FFBではループ“リローリング”のコード変更を実施
Post-K performance evaluation

- Himeno Benchmark (Fortran90)

<table>
<thead>
<tr>
<th>System</th>
<th>Gflops</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel Xeon Platinum 8168 2 CPUs</td>
<td>85</td>
</tr>
<tr>
<td>FX100 1 CPU</td>
<td>103</td>
</tr>
<tr>
<td>Post-K 1 CPU</td>
<td>346</td>
</tr>
<tr>
<td>SX-Aurora 1 VE †</td>
<td>286</td>
</tr>
<tr>
<td>Tesla V100 1 GPU †</td>
<td>305</td>
</tr>
</tbody>
</table>

† “Performance evaluation of a vector supercomputer SX-aurora TSUBASA”, SC18, https://dl.acm.org/citation.cfm?id=3291728
Post-K Chassis, PCB (w/DLC), and CPU Package

A0 Chip Booted in June

Undergoing Tests

CPU Package

W 800mm
D1400mm
H2000mm
384 nodes

CMU
CMU: CPU Memory Unit

- A64FX CPU x2 (Two independent nodes)
- QSFP28 x3 for Active Optical Cables
- Single-side blind mate connectors of signals & water
- ~100% direct water cooling

Water

Water

Electrical signals
TOFU-D 6D Mesh/Torus Network

- Six coordinate axes: X, Y, Z, A, B, C
  - X, Y, Z: the size varies according to the system configuration
  - A, B, C: the size is fixed to $2 \times 3 \times 2$
- Tofu stands for “torus fusion”: $(X, Y, Z) \times (A, B, C)$
A64FX: Tofu interconnect D

- Integrated w/ rich resources
  - Increased TNIs achieves higher injection BW & flexible comm. patterns
  - Increased barrier resources allow flexible collective comm. algorithms
- Memory bypassing achieves low latency
  - Direct descriptor & cache injection

<table>
<thead>
<tr>
<th></th>
<th>TofuD spec</th>
<th>Measured</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Port bandwidth</strong></td>
<td>6.8 GB/s</td>
<td></td>
</tr>
<tr>
<td><strong>Injection bandwidth</strong></td>
<td>40.8 GB/s</td>
<td></td>
</tr>
<tr>
<td><strong>Put throughput</strong></td>
<td>6.35 GB/s</td>
<td></td>
</tr>
<tr>
<td><strong>Ping-pong latency</strong></td>
<td>0.49~0.54 µs</td>
<td></td>
</tr>
</tbody>
</table>
Put Latencies

- 8B Put transfer between nodes on the same board
  - The low-latency features were used

<table>
<thead>
<tr>
<th>Communication settings</th>
<th>Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tofu1 Descriptor on main memory</td>
<td>1.15 μs</td>
</tr>
<tr>
<td>Tofu1 Direct Descriptor</td>
<td>0.91 μs</td>
</tr>
<tr>
<td>Tofu2 Cache injection OFF</td>
<td>0.87 μs</td>
</tr>
<tr>
<td>Tofu2 Cache injection ON</td>
<td>0.71 μs</td>
</tr>
<tr>
<td>TofuD To/From far CMGs</td>
<td>0.54 μs</td>
</tr>
<tr>
<td>TofuD To/From near CMGs</td>
<td>0.49 μs</td>
</tr>
</tbody>
</table>

- Tofu2 reduced the Put latency by 0.20 μs from that of Tofu1
  - The cache injection feature contributed to this reduction
- TofuD reduced the Put latency by 0.22 μs from that of Tofu2
Injection Rates per Node

- Simultaneous Put transfers to multiple nearest-neighbor nodes
  - Tofu1 and Tofu2 used 4 TNIs, and TofuD used 6 TNIs

<table>
<thead>
<tr>
<th></th>
<th>Injection rate</th>
<th>Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tofu1 (K)</td>
<td>15.0 GB/s</td>
<td>77 %</td>
</tr>
<tr>
<td>Tofu1 (FX10)</td>
<td>17.6 GB/s</td>
<td>88 %</td>
</tr>
<tr>
<td>Tofu2</td>
<td>45.8 GB/s</td>
<td>92 %</td>
</tr>
<tr>
<td>TofuD</td>
<td>38.1 GB/s</td>
<td>93 %</td>
</tr>
</tbody>
</table>

- The injection rate of TofuD was approximately 83% that of Tofu2
- The efficiencies of Tofu1 were lower than 90%
  - Because of a bottleneck in the bus that connects CPU and ICC
- The efficiencies of Tofu2 and TofuD exceeded 90 %
  - Integration into the processor chip removed the bottleneck
Packaging – Rack Structure of Post-K

- **Rack**
  - 8 shelves
  - 192 CMUs or 384 CPUs

- **Shelf**
  - 24 CMUs or 48 CPUs
  - \(X \times Y \times Z \times A \times B \times C = 1 \times 1 \times 4 \times 2 \times 3 \times 2\)

- **Top or bottom half of rack**
  - 4 shelves
  - \(X \times Y \times Z \times A \times B \times C = 2 \times 2 \times 4 \times 2 \times 3 \times 2\)
## Post-K system configuration

- **Scalable design**

<table>
<thead>
<tr>
<th>Unit</th>
<th># of nodes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>1</td>
<td>Single socket node with HBM2 &amp; Tofu interconnect D</td>
</tr>
<tr>
<td>CMU</td>
<td>2</td>
<td>CPU Memory Unit: 2x CPU</td>
</tr>
<tr>
<td>BoB</td>
<td>16</td>
<td>Bunch of Blades: 8x CMU</td>
</tr>
<tr>
<td>Shelf</td>
<td>48</td>
<td>3x BoB</td>
</tr>
<tr>
<td>Rack</td>
<td>384</td>
<td>8x Shelf</td>
</tr>
<tr>
<td>System</td>
<td>150k+</td>
<td>As a Post-K system</td>
</tr>
</tbody>
</table>
Overview of Post-K System & Storage

- 3-level hierarchical storage
  - 1st Layer: GFS Cache + Temp FS *(25~30 PB NVMe)*
  - 2nd Layer: Lustre-based GFS (a few hundred PB HDD)
  - 3rd Layer: Off-site Cloud Storage

- Full Machine Spec
  - >150,000 nodes
  - ~8 million High Perf. Arm v8.2 Cores
  - >150PB/s memory BW
  - Tofu-D 10x Global IDC traffic @ 60Pbps
  - ~10,000 I/O fabric endpoints
  - >400 racks
  - ~40 MegaWatts Machine+IDC PUE ~ 1.1 High Pressure DLC
  - NRE pays off: ~ 15~30 million state-of-the-art competing CPU Cores for HPC workloads (both dense and sparse problems)
Prepping the 40+MW Facility (actual photo)
**Post-K system software**

- RIKEN and Fujitsu are developing a software stack for Post-K

### Post-K applications

<table>
<thead>
<tr>
<th>Management software</th>
<th>File system</th>
<th>Programming environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>System management for high availability &amp; power saving operation</td>
<td>FEFS Lustre-based distributed file system</td>
<td>XcalableMP</td>
</tr>
<tr>
<td>Job management for higher system utilization &amp; power efficiency</td>
<td>LLIO NVM-based file I/O accelerator</td>
<td>MPI (Open MPI, MPICH)</td>
</tr>
</tbody>
</table>

- Linux OS / McKernel (Lightweight kernel)

### Post-K system hardware

Under development w/ RIKEN
Post-K Programming Environment

- **Programming Languages and Compilers provided by Fujitsu**
  - Fortran2008 & Fortran2018 subset
  - C11 & GNU and Clang extensions
  - C++14 & C++17 subset and GNU and Clang extensions
  - OpenMP 4.5 & OpenMP 5.0 subset
  - Java

- **Parallel Programming Language & Domain Specific Library provided by RIKEN**
  - XcalableMP
  - FDPS (Framework for Developing Particle Simulator)

- **Process/Thread Library provided by RIKEN**
  - PiP (Process in Process)

- **Script Languages provided by Linux distributor**
  - E.g., Python+NumPy, SciPy

- **Communication Libraries**
  - MPI 3.1 & MPI4.0 subset
    - Open MPI base (Fujitsu), MPICH (RIKEN)
  - Low-level Communication Libraries
    - uTofu (Fujitsu), LLC(RIKEN)

- **File I/O Libraries provided by RIKEN**
  - Lustre
  - pnetCDF, DTF, FTAR

- **Math Libraries**
  - BLAS, LAPACK, ScaLAPACK, SSL II (Fujitsu)
  - EigenEXA, Batched BLAS (RIKEN)

- **Programming Tools provided by Fujitsu**
  - Profiler, Debugger, GUI

- **NEW: Containers (Singularity) and other Cloud APIs**

- **NEW: AI software stacks (w/ARM)**
## OSS Application Porting @ Arm HPC Users Group

(http://arm-hpc.gitlab.io/)

<table>
<thead>
<tr>
<th>Application</th>
<th>Lang.</th>
<th>GCC</th>
<th>LLVM</th>
<th>Arm</th>
<th>Fujitsu</th>
</tr>
</thead>
<tbody>
<tr>
<td>LAMMPS</td>
<td>C++</td>
<td>Modified</td>
<td>Modified</td>
<td>Modified</td>
<td>Modified</td>
</tr>
<tr>
<td>GROMACS</td>
<td>C</td>
<td>Modified</td>
<td>Modified</td>
<td>Modified</td>
<td>Modified</td>
</tr>
<tr>
<td>GAMESS*</td>
<td>Fortran</td>
<td>Modified</td>
<td>Modified</td>
<td>Modified</td>
<td>Modified</td>
</tr>
<tr>
<td>OpenFOAM</td>
<td>C++</td>
<td>Modified</td>
<td>Modified</td>
<td>Modified</td>
<td>Modified</td>
</tr>
<tr>
<td>NAMD</td>
<td>C++</td>
<td>Modified</td>
<td>Modified</td>
<td>Modified</td>
<td>Modified</td>
</tr>
<tr>
<td>WRF</td>
<td>Fortran</td>
<td>Modified</td>
<td>Modified</td>
<td>Modified</td>
<td>Modified</td>
</tr>
<tr>
<td>Quantum ESPRESSO</td>
<td>Fortran</td>
<td>Ok in as is</td>
<td>Ok in as is</td>
<td>Ok in as is</td>
<td>Modified</td>
</tr>
<tr>
<td>NWChem</td>
<td>Fortran</td>
<td>Ok in as is</td>
<td>Modified</td>
<td>Modified</td>
<td>Modified</td>
</tr>
<tr>
<td>ABINIT</td>
<td>Fortran</td>
<td>Modified</td>
<td>Modified</td>
<td>Modified</td>
<td>Modified</td>
</tr>
<tr>
<td>CP2K</td>
<td>Fortran</td>
<td>Ok in as is</td>
<td><strong>Issues found</strong></td>
<td><strong>Issues found</strong></td>
<td>Modified</td>
</tr>
<tr>
<td>NEST*</td>
<td>C++</td>
<td>Ok in as is</td>
<td>Modified</td>
<td>Modified</td>
<td>Modified</td>
</tr>
<tr>
<td>BLAST*</td>
<td>C++</td>
<td>Ok in as is</td>
<td>Modified</td>
<td>Modified</td>
<td>Modified</td>
</tr>
</tbody>
</table>
Twelve primary OSS applications are listed and being tested in the Users Group for each compilers, collaboratively w/ Arm.
Massive Scale Deep Learning on Post-K

**Post-K Processor**
- High perf FP16&Int8
- High mem BW for convolution
- Built-in scalable Tofu network

**Unprecedented DL scalability**
High Performance and Ultra-Scalable Network for massive scaling model & data parallelism

**Low Precision ALU + High Memory Bandwidth + Advanced Combining of Convolution Algorithms (FFT+Winograd+GEMM)**

**Unprecedented Scalability of Data/Model Parallelism**

![Diagram of Post-K Processor](image)
Open “Post-K” Naming until April 8, 2019 5 pm JST

- Foreign submissions welcome

- Requirements for the post-K’s name are:
  - The name should preferably express the idea that RIKEN is a world-class research institute operating a state-of-the-art supercomputer.
  - The name should be attractive not only to Japanese speakers but to people around the world.

https://www.r-ccs.riken.jp/en/topics/naming.html