The phenomenal pace of change making RISC-V more attractive for HPC

Nick Brown
n.brown@epcc.ed.ac.uk

EPCC

RISC-V testbed

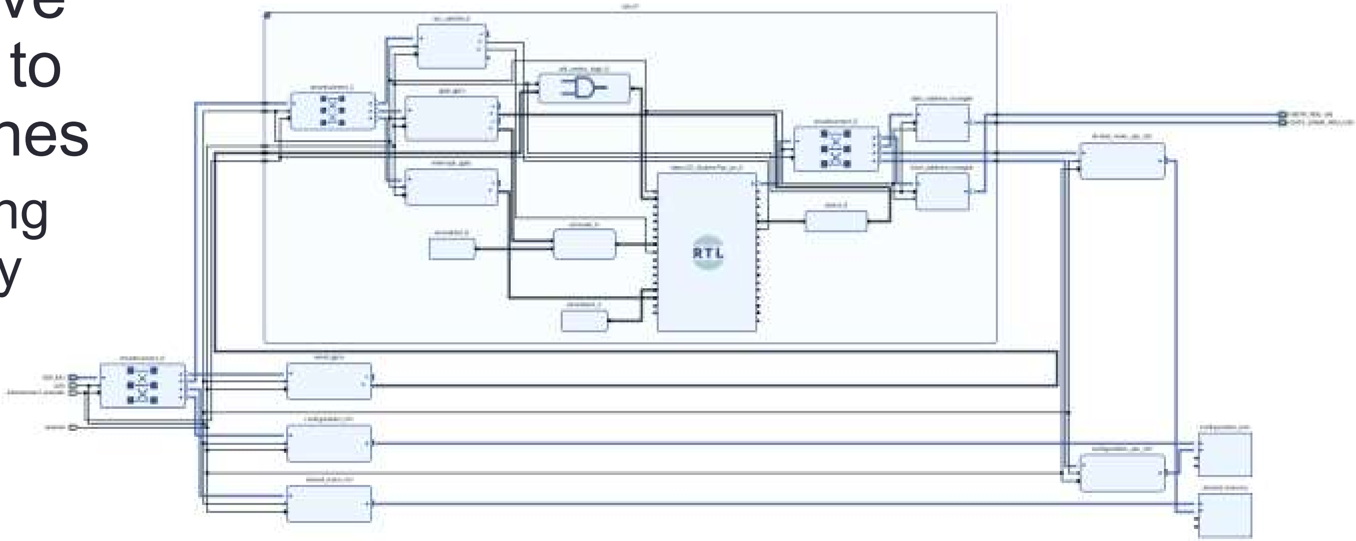
- ExCALIBUR is a UK exascale programme
  - One key question is around what are the next generation hardware technologies that might be in our supercomputers in the next five to ten years
    - And how do we best support these for users?

- RISC-V testbed aims to make this technology available to scientific software developers more widely to experiment with RISC-V
  - Idea is that it should feel like any other HPC style system with login node, compute managed by Slurm and common HPC libraries/compilers as modules, user accounts managed by SAFE
    - riscv.epcc.ed.ac.uk

A mix of soft-cores and physical CPUs

- Soft-cores enabled us to have some RISC-V CPUs available quickly
  - And now we have widened this to a larger range and are including accelerators

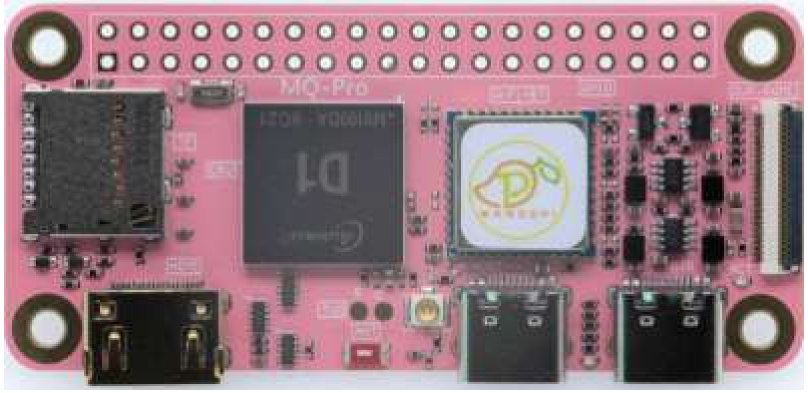
- Host these on VCK1802 and have general structure to easily add new ones
  - Currently integrating with Slurm for easy use



Physical CPUs: Challenging beginnings

- RISC-V hardware availability was initially a major challenge
  - Luckily, we had ordered a HiFive Unmatched board before these were end-of-lifed and-so at-least had this available
  - But the performance was somewhat disappointing here
  - For a time in 2022 that was pretty much it, the only other boards were unavailable due to hardware shortages

More SoCs then became available



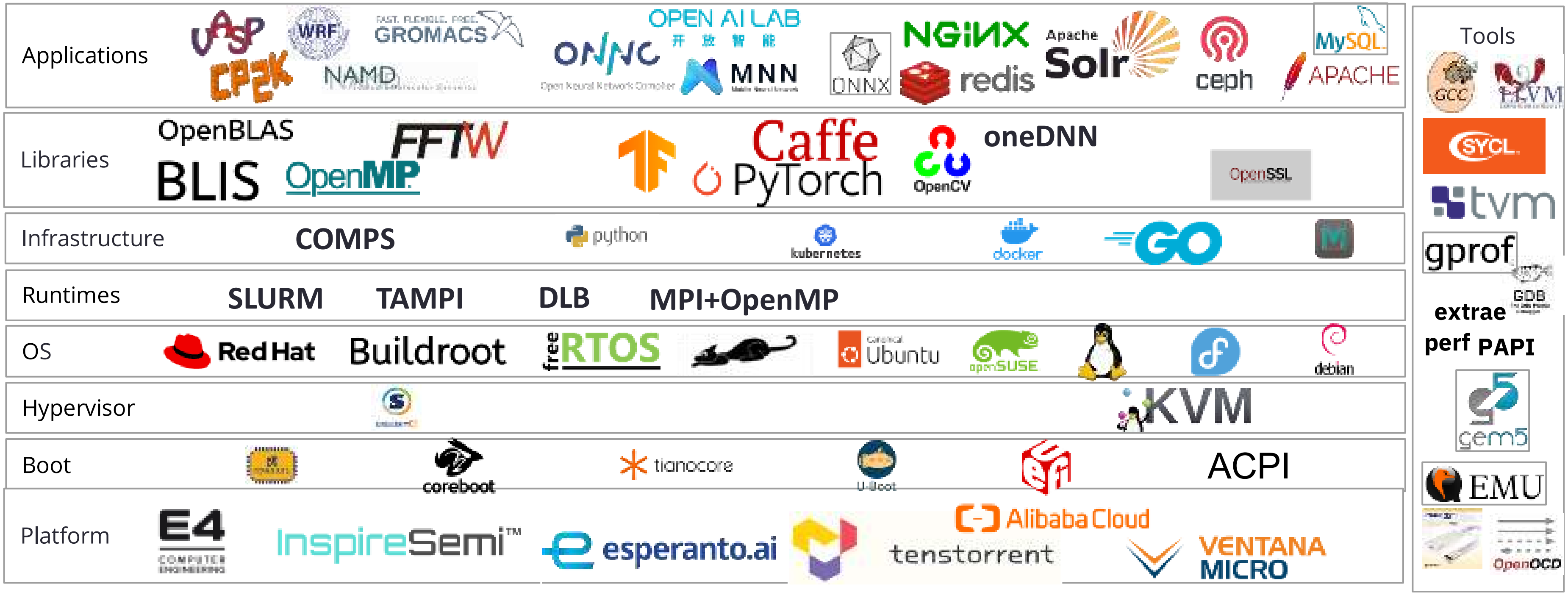
- Single core boards based on AllWinner D1 SoC started to become available (C906 core)
- And then the dual core VisionFive V1 (JH7100 SoC) and quad core VisionFive V2 (JH7110 SoC), both built around U74 core.

Moving to a proper service

- Now we had a whole load of boards we could try and expose as a “proper” service
  - To be acceptable to scientific end users
- Slurm to manage access to the compute nodes
- Shared filesystem using NFS

- Login node with compilers and libraries (installed via modules)
  - MPI so can run distributed memory across the RISC-V compute nodes
  - GCC & LLVM compilers, and common HPC libraries preinstalled and available

RISC-V HPC Landscape



Applications	VASP, CP2K, WRF, GROMACS, NAMD, ONNC, OPEN AI LAB, MNN, ONNX, NGINX, redis, Apache Solr, ceph, MySQL, APACHE
Libraries	OpenBLAS, BLIS, FFTW, OpenMP, Caffe, PyTorch, OpenCV, oneDNN, OpenSSL
Infrastructure	COMPS, python, kubernetes, docker, GO
Runtimes	SLURM, TAMPI, DLB, MPI+OpenMP
OS	Red Hat, Buildroot, freeRTOS, Ubuntu, openSUSE, debian
Hypervisor	KVM
Boot	coreboot, tianocore, U-Boot, efi, ACPI
Platform	E4 Computer Engineering, InspireSemi™, esperanto.ai, Alibaba Cloud, tenstorrent, VENTANA MICRO

Tools: GCC, LLVM, SYCL, tvm, gprof, GDB, extrae, perf, PAPI, gem5, EMU, OpenOCD

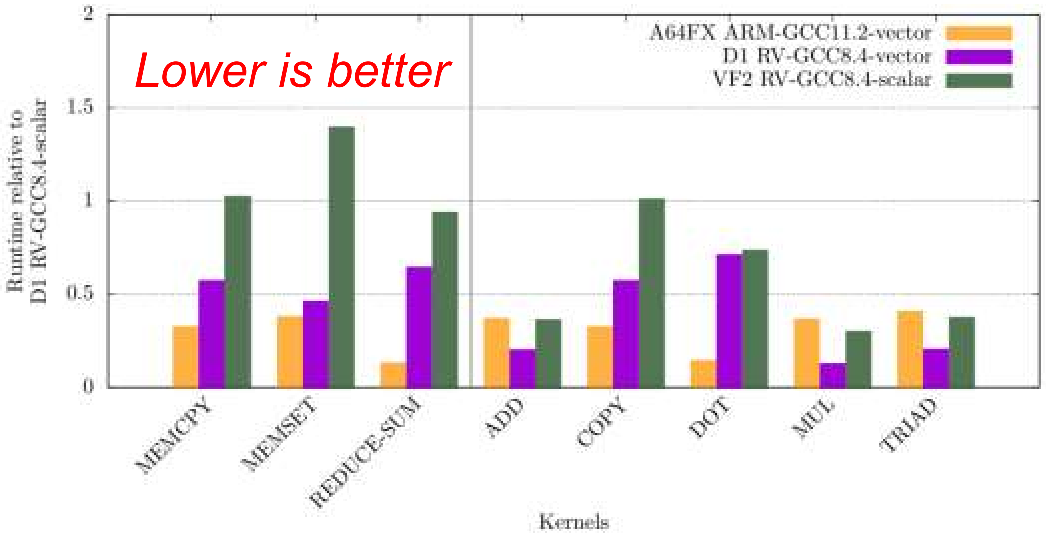
epcc

ExCALIBUR

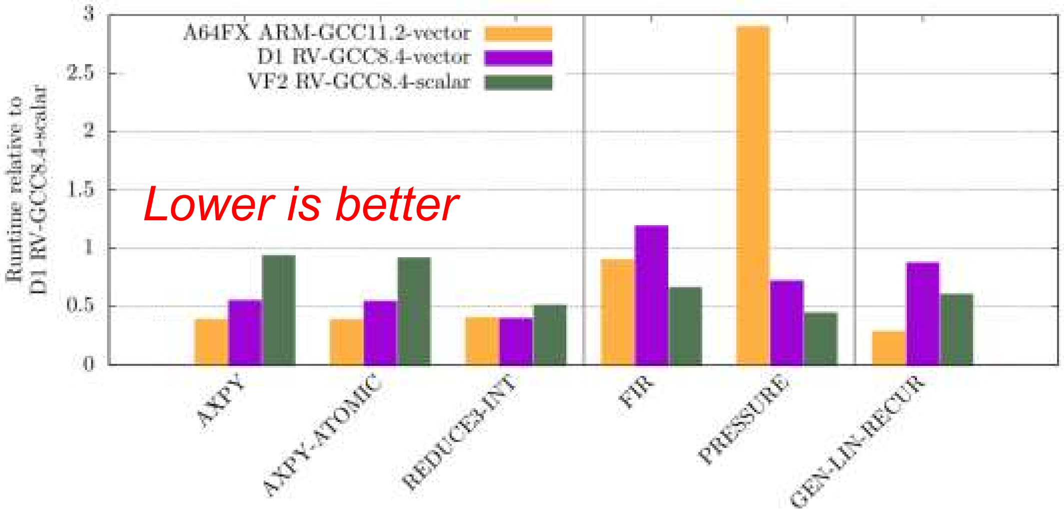
Deceptively easy initial steps

- The Linux image provided with these boards was painless to install and run
  - However we needed to modify this to add in kernel modules for NFS (and also modify the bootloader to enable RVV on the D1)
  - The bootloader was locked and-so the kernel could not be modified
- Needed to rebuild kernel and bootloader externally (cross compile) whenever a new kernel was needed
  - Vendor specific patches need to be applied and for the D1 need to use T-Head specific GCC version
  - This is OK, but documentation and availability of these things was limited!

Interesting performance comparisons



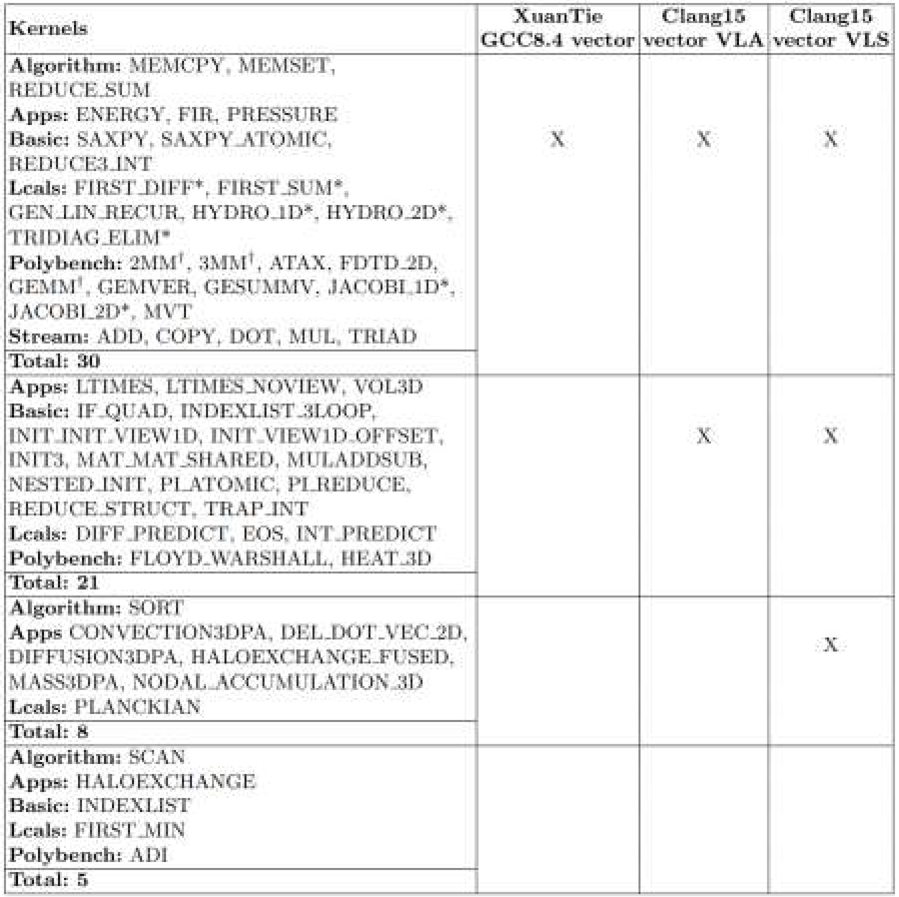
- Running RAJAPerf over D1 (C906 core), VF2 (U74 core), and a single core of A64FX with Neon only (i.e. no SVE)
- Using GCC compiler (8.4 on RISC-V and 11.2 on A64FX).
- This surprised us a bit, D1 based boards are very cheap and actually performs reasonably well against more expensive U74 core in V2



More details at https://arxiv.org/pdf/2304.10319.pdf

RVV programming with RVV rollback

- Lots of the silicon only provides RVV v0.7.1
- Bespoke GCC version by T-Head for RVV v0.7.1
  - LLVM only supports RVV v1.0
- Would be nice to support LLVM compiler generating RVV v0.7.1
  - Developed an RVV rollback tool, which will rewrite the generated assembly and modify RVV v1.0 to backport it to RVV v0.7.1

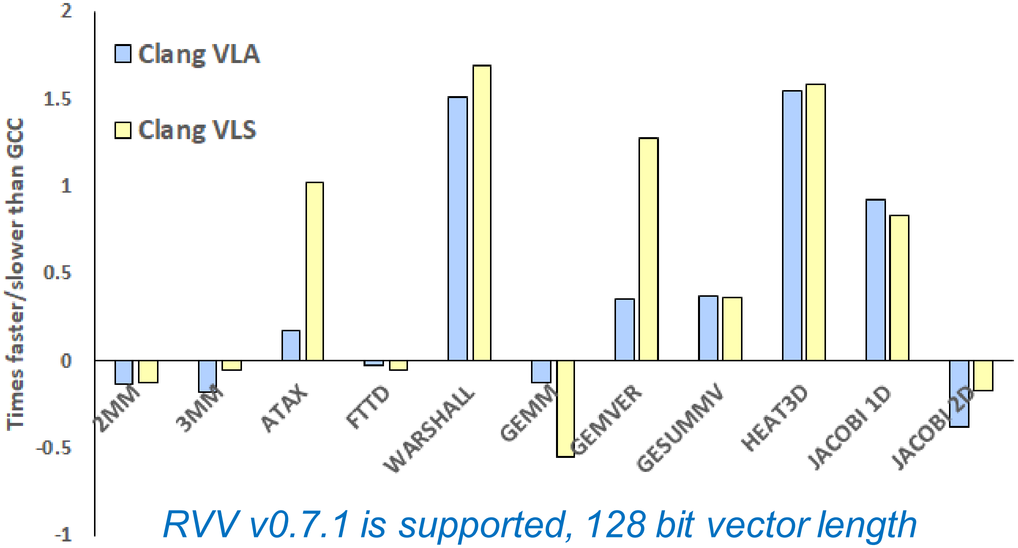


Kernels	XuanTie GCC8.4 vector	Clang15 vector VLA	Clang15 vector VLS
Algorithm: MEMCPY, MEMSET, REDUCE_SUM **Apps:** ENERGY, FIR, PRESSURE **Basic:** SAXPY, SAXPY_ATOMIC, REDUCE3_INT **Lcals:** FIRST_DIFF*, FIRST_SUM*, GEN_LIN_RECUR, HYDRO_1D*, HYDRO_2D*, TRIDIAG_ELIM* **Polybench:** 2MM†, 3MM†, ATAX, FDTD_2D, GEMM†, GEMVER, GESUMMV, JACOBI_1D*, JACOBI_2D*, MVT **Stream:** ADD, COPY, DOT, MUL, TRIAD	X	X	X
Total: 30			
Apps: LTIMES, LTIMES_NOVIEW, VOL3D **Basic:** IF_QUAD, INDEXLIST_3LOOP, INIT_INIT_VIEW1D, INIT_VIEW1D_OFFSET, INIT3, MAT_MAT_SHARED, MULADDSUB, NESTED_INIT, PI_ATOMIC, PI_REDUCE, REDUCE_STRUCT, TRAP_INT **Lcals:** DIFF_PREDICT, EOS, INT_PREDICT **Polybench:** FLOYD_WARSHALL, HEAT_3D		X	X
Total: 21			
Algorithm: SORT **Apps** CONVECTION3DPA, DEL_DOT_VEC_2D, DIFFUSION3DPA, HALOEXCHANGE_FUSED, MASS3DPA, NODAL_ACCUMULATION_3D **Lcals:** PLANCKIAN			X
Total: 8			
Algorithm: SCAN **Apps:** HALOEXCHANGE **Basic:** INDEXLIST **Lcals:** FIRST_MIN **Polybench:** ADI			
Total: 5			

https://github.com/RISCVtestbed/rvv-rollback

SG2042: 64-core RISC-V CPU

- SG2042 has 64 C920 cores
  - First commodity available high-core count RISC-V CPU, so very interesting for us
  - Since November we have two of these machines in the testbed
  - Each with 128GB of RAM

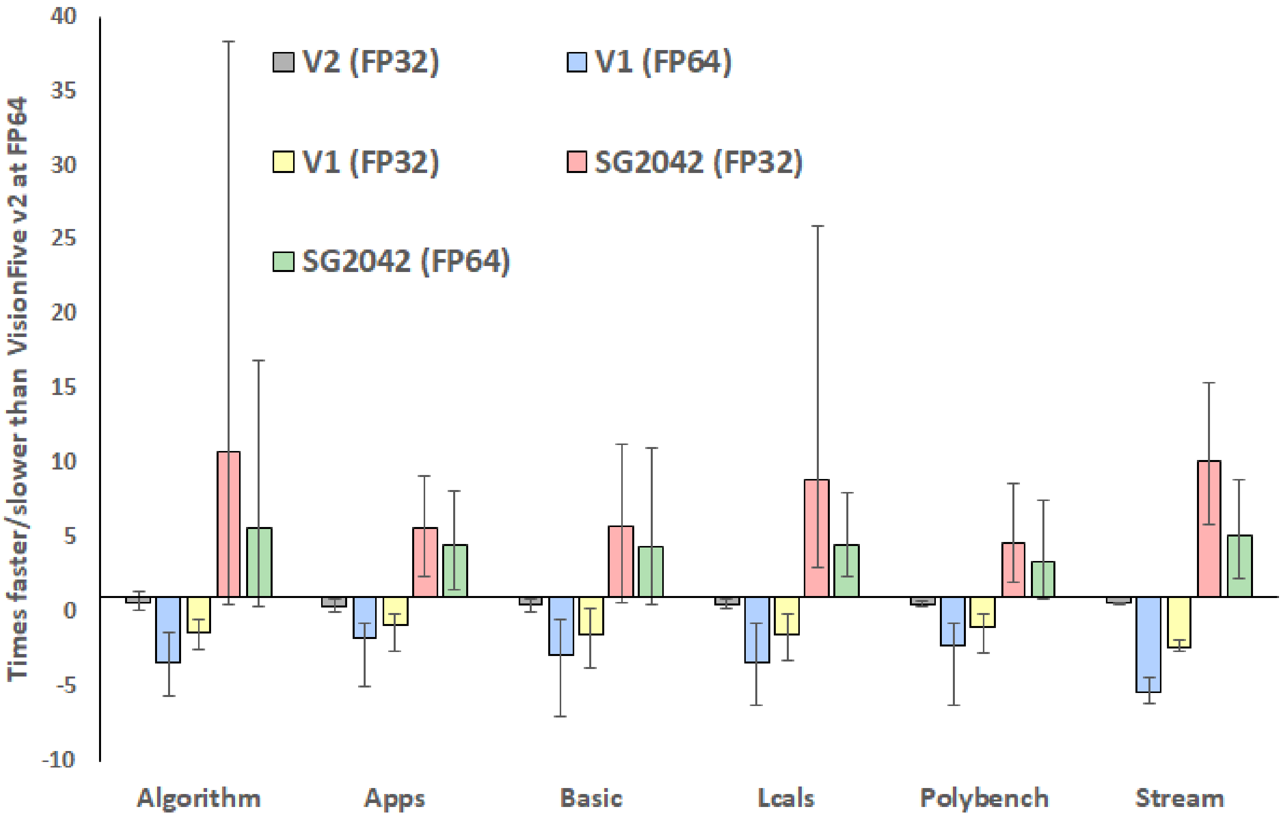


RVV v0.7.1 is supported, 128 bit vector length

Performance: SG2042 vs other RISC-V

- Using RAJAPerf benchmark suite and T-Head GCC 8.4

Higher is better

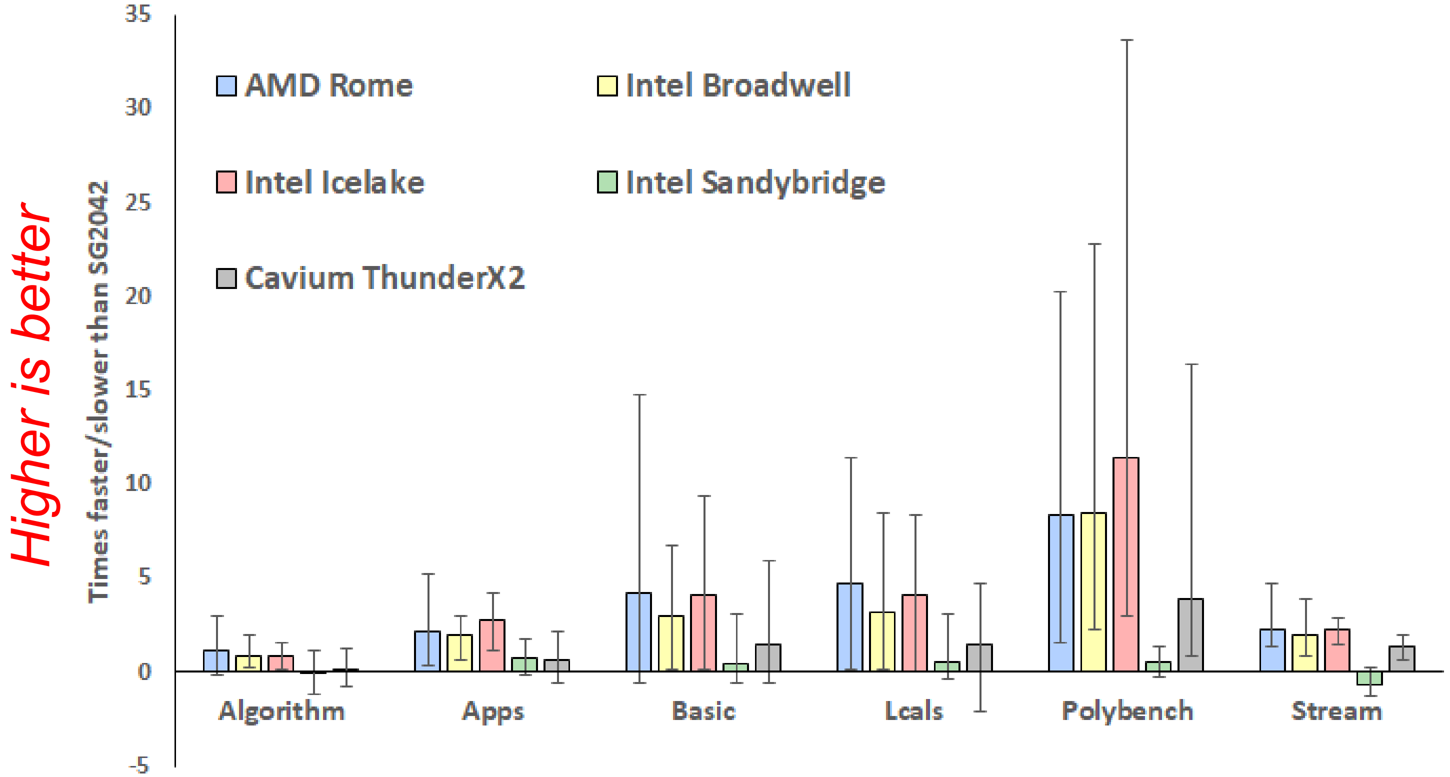


This is comparing the C920 core in the SG2042 against the U74 in the V2 and V1

Single core comparison against the VisionFive V2 at double precision

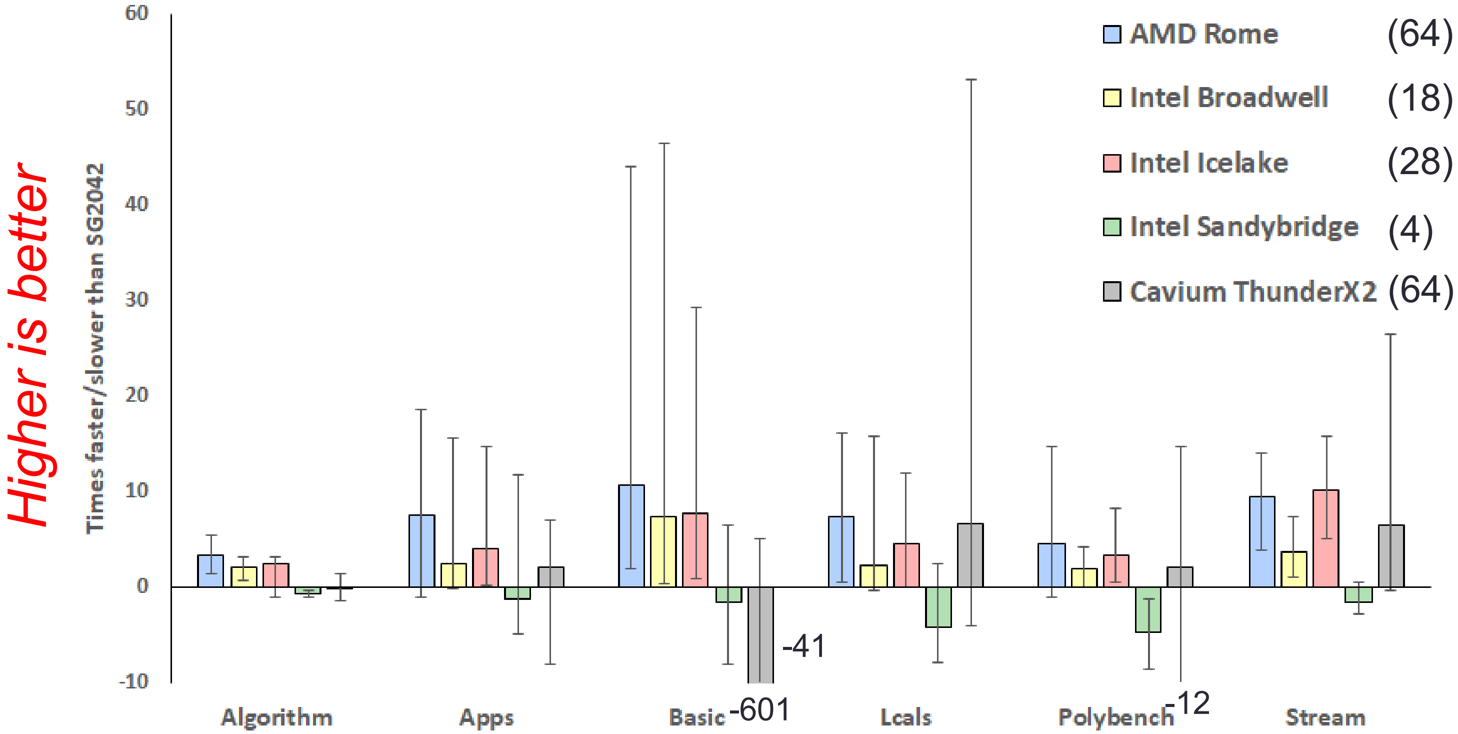
SG2042 performance against others

- Single core performance, running in double precision, GCC version 8.4 (RVV) on SG2042



SG2042 performance against others

- Multi core performance, running in double precision, GCC version 8.4 (RVV) on SG2042



More details at https://browse.arxiv.org/pdf/2309.00381.pdf

Conclusions

- Pace of change here is very exciting with lots of new hardware on the horizon
- Getting HPC codes running on RISC-V is pretty quick
  - But the devil is in the detail around optimisation
    - We are currently using some HPC codes as vehicles to explore optimisations & *kick the tyres*
  - Free access to our testbed riscv.epcc.ed.ac.uk

- I think a key question we need to answer here for is why?
  - Specifically, what is the benefit of RISC-V for HPC workloads compared to other architectures
    - Why buy this for future supercomputers?
  - Potentially RISC-V based accelerators will be the first technologies that start to gain traction in production supercomputers?

RISC-V for HPC workshop at ISC24 (in Hamburg), call for papers currently open with deadline of mid-March. https://riscv.epcc.ed.ac.uk/community/isc24-workshop/