HPC Case Study: CFD Applications on ARM®

In this paper, we examine the readiness and potential of ARM® based platforms for High Performance Computing, and have benchmarked two different computational fluid dynamics (CFD) applications. CFD represents one of the most widely used HPC applications in aerospace, automotive and other engineering areas such as turbine-design. For server hardware, we leverage the ThunderX® platform from Cavium®, with a highly integrated SoC architecture and a range of configurations addressing modern data center and cloud implementations[1].

OpenFOAM[2] is a popular, general purpose solver package with many academic and industrial users. It can solve a wide range of both steady-state and time-dependent problems, in compressible and incompressible fluids, solid mechanics and electromagnetism. OpenFOAM is written in C++ and parallelised with MPI.

Cloverleaf[3] is a shock hydrodynamics code solving the compressible, time-dependent Euler equations using a second-order-accurate staggered-grid method. Cloverleaf is a mini-app, part of the international Mantevo Project, replicating the computational requirements of production codes used at defence laboratories worldwide. Originally written in Fortran, with hybrid parallelization using OpenMP/MPI, it has been ported to a wide range of programming models, allowing application performance to be compared across different parallel architectures.

These applications were chosen so that we can both investigate the effectiveness of the ThunderX platform for scientists and engineers working on computational fluid dynamics problems, and provide some initial comparisons of the performance of ThunderX with other architectures.

Architecture

Cavium ThunderX is a System on Chip (SoC) based on the ARM-v8 architecture. A single socket is comprised of 48 physical, fully out-of-order cores with up to four DDR4 memory controllers. A compute node typically has two sockets for a total of 96 cores and up to 1TB of attached memory. The mini-cluster used in this study was configured with 128GB of attached DDR4 memory and a 40Gb Ethernet interconnect.

[1] cavium.com/ThunderX_ARM_Processors.html
[2] openfoam.com
[3] uk-mac.github.io/CloverLeaf

Software ecosystem

The ARM-v8 architecture is well supported by common Gnu/Linux distributions and the test platform uses Ubuntu 16.04. The Gnu Compiler Collection (gcc) targets the ARM-v8 architecture (with the AArch64 instruction set). The HPC toolchain including recent versions of OpenMPI compiled without any problem on the system.

As the system is running a standard Linux distribution, the UNIX development applications, libraries and tools (i.e. Python, CMake, Autotools, etc) are included. ARM also provides a performance library, including BLAS and LAPACK interfaces, but this was not required for the two applications investigated in this work.

It is key to note the ease with which these legacy x86 applications ported to ARM's AArch64 instruction set. The Cloverleaf application code was compiled with no intervention in the code base or the build system. OpenFOAM required some small (~10 lines added) changes to its custom *wmake* build system to compile on AArch64 but none of the application code was affected. This modification has been previously documented by ARM[4] on their HPC developer portal.

Cloverleaf application performance

The time per time-step of the Cloverleaf bm series test cases from size 1-128 was measured on one socket (48 cores), one dual-socket node (96 cores), two dual-socket nodes and four dual-socket nodes. This gives the required numbers to present both the weak and strong scaling curves.

A range of different configurations for the OpenMP/MPI split of threads/tasks was explored to find the optimal configuration. It was found that filling each socket with 48 OpenMP threads, and using an MPI task on each socket gave good performance. The application was compiled with gfortran version 6.3.0 and linked with OpenMPI version 2.0.2.

[4] developer.arm.com/-/media/developer/developers/hpc/files/OpenFOAM-for-ARM-ubuntu-A57.pdf

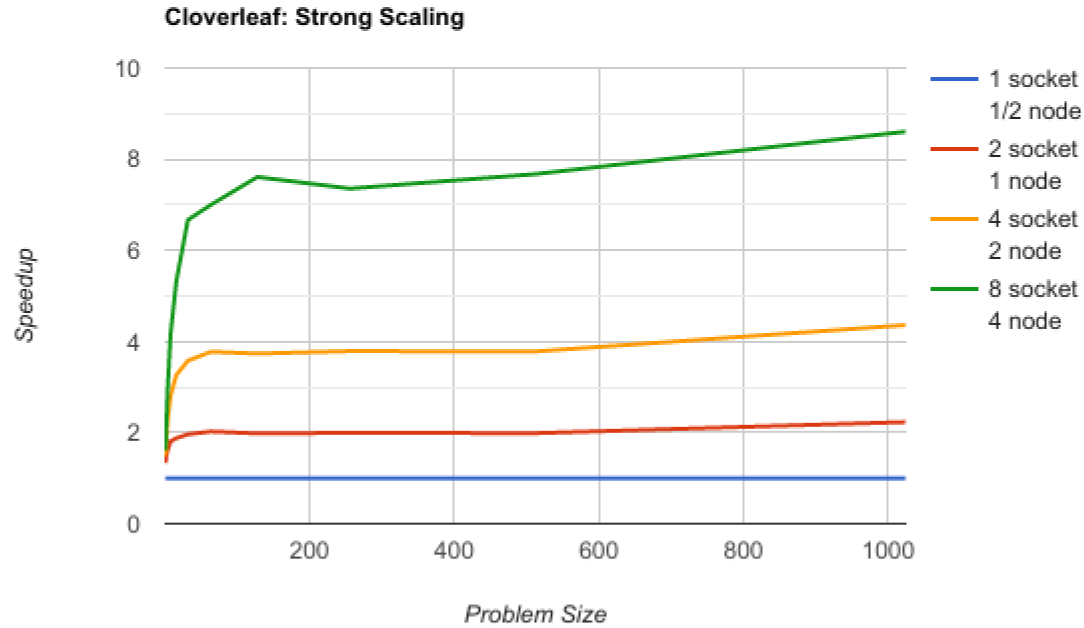

Figure 1. Strong scaling results for the Cloverleaf *bm* series test cases. Curves show the speedup on 2, 4 and 8 ThunderX sockets, normalised to the time of 1 socket.

The Cloverleaf benchmark shows excellent scaling performance, even at small problem sizes. At the largest problem sizes the benchmark even demonstrates some super-scalar speedup, as more with more cores, more data can be stored in cache.

Figure 2. Weak scaling results for the Cloverleaf *bm* series test cases. Curves show the efficiency of runs on 2, 4 and 8 ThunderX sockets, normalised to the time of 1 socket.

Parallel efficiency of the Cloverleaf benchmark is also excellent, close to 95% on two or four nodes across a wide range of problem sizes.

The absolute timings are given in Table 1, with times given in seconds for each of the *bm_short* series. We compare the published runtime results[5] for this test with problem size 16 for a range of different hardware, and that data shows times of approximately 46-50 seconds on a 16 core Intel Xeon Haswell and around 18 seconds on an Nvidia® K40 GPU. Therefore, for this application, the results indicate that two ThunderX sockets give a time-to-solution slightly less than the single Intel Haswell socket and four ThunderX sockets slightly more than a single K40.

Problem size \ Number of Sockets	1	2	4	8
1	8.33	6.26	5.6	5.17
2	12.73	8.63	7.16	6.03
4	21.37	13.23	9.43	7.24
8	40.52	22.41	14.42	9.74
16	77.01	41.02	23.59	14.47
32	157.58	80.26	44.02	23.63
64	309.28	152.79	81.91	44.17
128	623.4	314.64	166.9	81.93
256	1237.23	620.11	326.4	168.19
512	2499.46	1257.66	660.86	325.77
1024	5702.19	2554.26	1307.64	662.77

Table 1. Absolute timings in seconds for the *bm_short* series of Cloverleaf test cases.

OpenFOAM application performance

To investigate the performance of OpenFOAM, we measure the time taken for 1000 time-steps on three different grid sizes for the icoFoam 'cavity' test case.

[5] github.com/UK-MAC/CloverLeaf/wiki/bm16_short

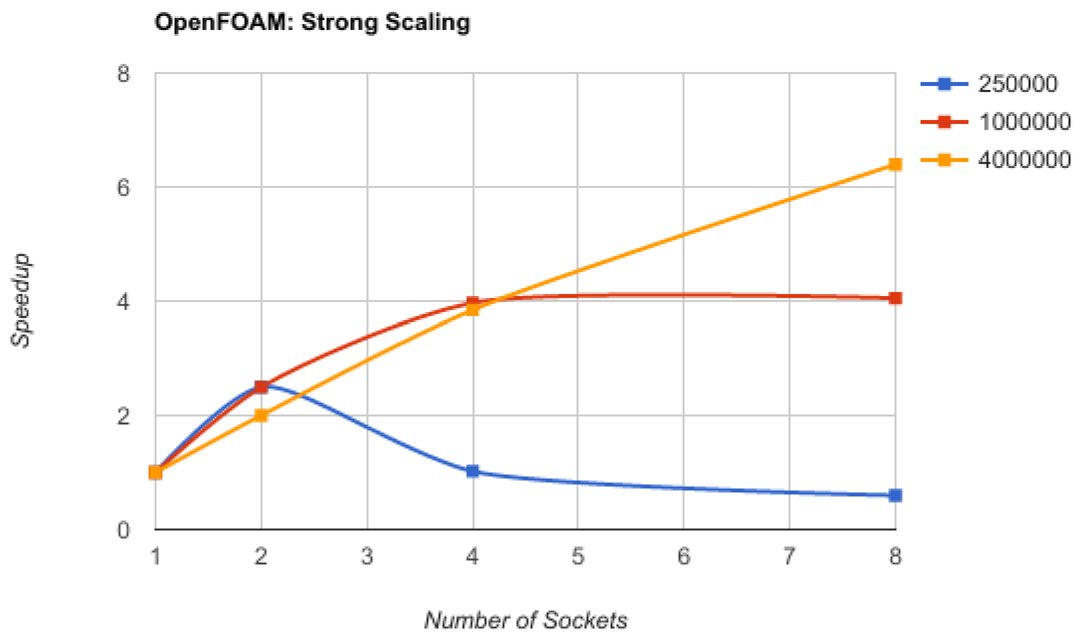

Figure 3: Strong scaling results for the OpenFOAM *cavity* test case. Curves show three different size simulations, with 250 thousand, 1 million and 4 million cells.

The results in Figure 3 show that this code only scales well at the larger problem sizes, and even at the largest problem size the application scales well only up to 4 sockets (2 nodes). Due to the nature of the application, it is not possible to precisely measure weak scaling but the results in Figure 3 indicate that scalability would improve at even larger problem sizes.

The reason for the comparatively poorer performance in this benchmark is that Cloverleaf has a hybrid OpenMP/MPI parallel structure, whereas OpenFOAM is parallelised purely with MPI, and the hybrid parallelization tends to reduce the communication overhead. Furthermore, Cloverleaf is more able to employ asynchronous communication primitives, thereby overlapping communication with useful computation and further reducing communication overhead. We note that implementing hybrid parallelisation and asynchronous communication would be expected to improve parallel performance across all HPC platforms.

Conclusion

Two complex HPC applications have been built and executed on the ThunderX platform to analyse the utility and performance of this platform for computational fluid dynamics science and engineering. The freely available ARM open-source toolchain based on GCC functioned in a predictable manner. Despite the unfamiliar architecture, neither CFD application was at all difficult to get running with no changes required to Cloverleaf and very minor (see page 1) changes to the OpenFOAM build system. Cloverleaf demonstrates excellent scalability and competitive performance with contemporary architectures. The performance of OpenFOAM shows promise, with reasonable scalability at large problem sizes.

For more information on ARM in HPC, go to www.arm.com/hpc