Performance of the AMS Offline Software on the IBM Blue Gene/Q Architecture

V. Choutko[1], A. Egorov[1], B. Shan[2]

[1]Massachusetts Institute of Technology, Cambridge, MA [2]Beihang University, Beijing, China

Abstract

The Alpha Magnetic Spectrometer (AMS) is the high energy physics experiment installed and operating on board of the International Space Station (ISS) from May 2011 and expected to last through Year 2024 and beyond. The details of porting of the AMS software to the IBM Blue Gene/Q Architecture are discussed. The performance of the AMS reconstruction and simulation software in that architecture is evaluated and compared to the performance obtained on Intel based architecture.

Introduction

The Alpha Magnetic Spectrometer (AMS)[1] is the high energy physics experiment operating on board of the International Space Station (ISS). The detector has large geometrical acceptance of $0.5 m^2 \cdot sr$, and is equipped with permanent magnet, Time of Flight hodoscope, precision nine layers silicon tracker, gaseous Transition Radiation Detector, Ring Image Cherenkov Detector and 3D Electromagnetic Calorimeter. In five years of operation more than 85 billion of cosmic ray events were triggered by, recorded and transferred to the ground.

IBM Blue Gene/Q: architecture and compilers

The IBM Blue Gene/Q architecture is described in details in Ref. [2]. It includes the login nodes, equipped with POWER7 3.55 GHz processors, 128 GB of memory and running Linux 2.6 operation system; and compute nodes, equipped with 16+1 cores PowerPC A2 1.6 GHz processors running light weight proprietary kernel (CNK), with 16 GB of memory. The PowerPC A2 processor features four-way hyper-threading, so up to 64 threads per node can be run. All performance studies were done on compute nodes, while ported software equally works on login nodes.

There are few distinct features of this architecture found to be essential for the software porting:

- 64 bit address space;
- Big Endian addressing scheme;
- Limited support for Linux system calls and in particular no support for *fork()* and *system()* calls on compute nodes;
- Massive parallelization beyond the SMP one. Open MPI[3] is used usually to synchronize threads running in different nodes.

The actual porting of the software were done on JUQUEEN computer of Juelich SuperComputing Center[4], where the minimal job configuration in the batch system includes 32 nodes or 2048 threads, while typical one consists of 128 to 512 nodes.

The IBM compilers xlC 12.1 and xlF 14.1 were used to compile and link all the software. These compilers support OpenMP[5] directives, with the major exception of not supporting the *omp threadprivate* pragma for any STL container (vector,map, etc). xlC 12.1 supports a subset of C++11 directives, and in particular thread local storage (TLS) via *__thread* directive with the same exception for STL containers.

Software porting – ROOT

The ROOT 5.34[6] was not available to this platform (codenamed here as linuxppcbgxlc) due to incompatibility between the ROOT CINT interpreter and 64 Bit addressing space with BigEndian features of PowerPC processors[7]. This was fixed by changing a line in the cint/cint/src/value.h file like:

```
< if (buftype == 'i') return (T) buf->obj.i;
---
> if (buftype == 'i') return (T) buf->obj.in;
```

Another minor issue was xlC compiler internal error during compilation of RooFit dictionary. This was fixed by division the dictionary file by several parts.

After this the successful build of the root executable and all shared and static libraries became possible, see Fig. 1.

```
[vsk1006@juqueen2 install]$ $ROOTSYS/bin/root -b
 ******************************************
 *                                        *
 *        W E L C O M E  to  R O O T      *
 *                                        *
 *   Version   5.34/09      26 June 2013   *
 *                                        *
 *  You are welcome to visit our Web site *
 *          http://root.cern.ch           *
 *                                        *
 ******************************************

ROOT 5.34/09 (v5-34-09@v5-34-09, May 31 2016, 23:21:19 on linuxppcbgxlc)

CINT/ROOT C/C++ Interpreter version 5.18.00, July 2, 2010
Type ? for help. Commands must be C++ statements.
Enclose multiple statements between { }.
root [0]   gSystem->Load("$AMSWD/lib/linuxppcbgxlc5.34/ntuple_slc4_PG.so");
AMSCommonsI-I-HardwareIdentifiedAs Linux 2.6.32-642.3.1.el6.ppc64 ppc64 5
AMSCommonsI-W-AMSDataDir variable is not defined.
AMSCommonsI-W-Default value /afs/cern.ch/exp/ams/Offline/AMSDataDir will be used
.
AMSCommonsI-I-Identified as BigEndian 64 bit machine. 4294967295
AMSCommonsI-I-amsdatabase /afs/cern.ch/exp/ams/Offline/AMSDataDir/DataBase/
AMSCommonsI-I-MipsFromCPUInfo: 3550
AMSCommonsI-I-SystemIdentified as  POWER7 (architected), altivec supported
AMSCommonsI-I-ComputerEvaluatedAsMips 2378
AMS Software version v5.01/1074/15 build Thu Jun 23 16:13:39 2016

AMSaPool::SetLastResort-I-ResortSet 2000000
AMSaPool::SetLastResort-I-ResortSet 2000000
root [1] TH1D *p=new TH1D("1d","1d",100,0,1)
root [2] sizeof(p)
(const int)8
```

Figure 1. The 64bit ROOT starting screen in the JUQUEEN login node.

Software porting – GEANT4

The IBM Blue Gene/Q architecture was not supported by GEANT4.10.1 package[8]. To do that, the following architecture file Linux-ppc-mt.gmk was added:

```
..
CXX := bgxlC_r
CXXFLAGS := -q64 -qmaxmem=-1 -D__PPC64
ifdef G4USE_STD11
  CXXFLAGS += -qlanglvl=extc1x
endif
ifdef G4MULTITHREADED
  CXXFLAGS += -qthreaded -qsmp=omp -qtls
endif
..
```

Few source files need to be changed, namely to add the thread specification for this architecture as well as to overcome the xlC compiler template specialization initialization limitations. Namely the following files were modified:

```
global/management/include/tls.hh
global/management/include/G4TWorkspacePool.hh
particles/management/src/G4ParticlesWorkspace.cc
geometry/navigation/src/G4VIntersectionLocator.cc
```

After the changes, the GEANT4.10.1 libraries were built and test examples successfully ran in multi-threaded mode.

Software porting – CERNLIB

The port of CERNLIB[9] software was needed, as AMS software depends on it, to ensure the FORTRAN local variables being initialized in stack to allow thread safe processing. Also the MINUIT package need to be adapted to thread safe mode using OPENMP technique.

Software porting – AMS software

- Simulation of Linux *system()* calls

Due to absence of system() support on CNK kernel, the following system calls were rewritten using the C++ language I/O constructions: *mkdir, rm, rmdir, cat, grep, ln -s, ...* .

- Memory management

Due to lack of support Linux system routines like *getrlimit()* ..., the proprietary routines were used to estimate the amount of free memory available for jobs execution.

- C++ features

The portions of AMS software contained *threadprivate* STL vector and/or maps were rewritten. In one particular case the *threadprivate* map was replaced by array of maps with explicit thread addressing, using OPENMP and/or GEANT4.10.1 methods.

- Fortran features

The AMS is using DPMJET2.5 FORTRAN code to simulate nuclei-nuclei interactions. To provide thread safe usage of this library OPENMP was used. The incompatibility of thread local storage between IBM xlF 14.1 and xlC 12.1 was found, which prevented to use TLS option of the compiler. The combination of *-qsmp=omp:noostls -g5* FORTRAN compiler flags was found to work correctly.

- ROOT dictionary

Due to platform Table of Content (TOC) 24bit size issue, the ROOT dictionary had to be divided by many (10) separate files.

- Linking

For the AMS software version with Open MPI support, no static linking was possible, because of another TOC size issue. Dynamic linking to those libraries did not show the problem. Finally statically bound executable with no Open MPI support and MPI emulation was used during the software performance tests.

- MPI Emulation

For the massively parallel jobs which can be run on JUQUEEN computer, the inter nodes communication can be done via Open MPI libraries, provided by IBM. As the AMS software parallelization is limited inside one node using the OpenMP (in case of reconstruction) and the mixture of POSIX threads and OpenMP (in case of simulation), see Ref.[10], the only communications needed are the proper ranking of the jobs and the synchronization of jobs finishing phase.

Despite the Open MPI was able to provide desired features, the custom software was written to emulate needed features of Open MPI messaging. It includes:

- MPI-like-Initialization, to create the job ranks and properly reroute the input and output files;
- Special thread to govern the job execution, to calculate CPU limits, to pass to and receive messages from other jobs and to intercept and reinterpret of all Linux signals, including SIGSEGV;
- MPI-like-Termination routine, called during the job static destructor execution, to ensure simultaneous termination of all the jobs.

Results

The efficient use of the JUQUEEN batch system for simulation jobs containing up to 2048 nodes and 131K threads was possible. This allows us to use JUQUEEN for AMS massive simulated data production.

The reconstructed jobs, due to their well defined number of input data, could not be efficiently used in MPI environment, which prevents us from using JUQUEEN for massive AMS data reruns.

- Simulation Software performance

We were able to run up to 62 threads per compute node. This became possible due to our customized GEANT4 Memory manager[10]. Figure 2 shows the performance of IBM Blue Gene/Q versus the number of threads. As seen, no 4-way hyper-threading degradation can be noticed. The multithreaded overhead is limited to about 0.7%.

In terms of absolute values, the Intel(R) Xeon(R) CPU E5-2699 v3 @ 2.30GHz (18 cores, 36 threads, 145 Watt) outperforms the Blue Gene/Q node (16 cores, 64 threads, 60 Watt) by factor of 5.

- Reconstruction Software performance

Up to 64 threads per node were run without problems, as memory requirements are less challenging for the AMS reconstruction software. However due to the fact that AMS reconstruction jobs are somewhat I/O bounded and the relatively low I/O bandwidth for the IBM Blue Gene/Q nodes, the wall clock performance is saturated above 32 threads per job.

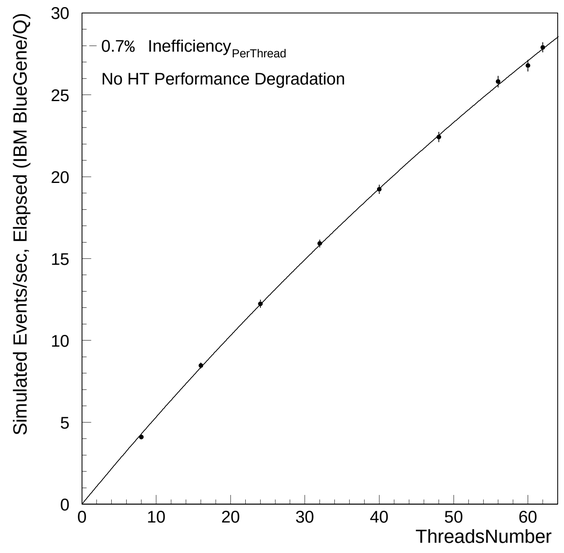

Figure 2. The performance of the AMS simulation software on IBM Blue Gene/Q node vs the number of threads. The line shows the fit of the $\frac{Events_{PerThread}}{Sec} \cdot \frac{1-(1-\xi)^{ThreadsNumber}}{\xi}$ to the measured performance. The value of the ξ parameter, which measures the per-thread inefficiency, was found to be about 0.007.

Conclusions

The AMS and other (ROOT, GEANT, CERNLIB) software was successfully ported to IBM Blue Gene/Q architecture. Massively parallel jobs, up to 2048 nodes and 131K threads successfully ran on JUQUEEN computer for wall clock of 24 hours, which it the maximum amount of time allowed by batch job scheduler. The AMS massive simulation data production is expected to start in the year 2017 to deliver up to 15% of AMS simulated data.

Acknowledgements

The authors thank the Juelich Supercomputer Center for hospitality and for providing the access to JUQUEEN computer.

References

[1] Ting S C 2007 AMS-02 TIM Meeting am CERN
[2] Blue Gene/Q application development URL http://www.redbooks.ibm.com/redbooks/pdfs/sg247948.pdf
[3] Open mpi: Open source high performance computing URL https://www.open-mpi.org/
[4] JuQueen Juelich Blue Gene/Q URLhttp://www.fz-juelich.de/ias/jsc/EN/Expertise/Supercomputers/JUQUEEN/JUQUEENnode.h
[5] Dagum L and Menon R 1998 Computational Science & Engineering, IEEE 5 46–55
[6] Antcheva I, Ballintijn M, Bellenot B, Biskup M, Brun R, Buncic N, Canal P, Casadei D, Couet O, Fine V et al. 2011 Computer Physics Communications 182 1384–1385
[7] URLhttp://savannah.web.cern.ch/savannah/HEPApplications/savroot/bugs/70542.html
[8] Agostinelli S, Allison J, Amako K a, Apostolakis J, Araujo H, Arce P, Asai M, Axen D, Banerjee S, Barrand G et al. 2003 Nuclear instruments and methods in physics research section A: Accelerators, Spectrometers, Detectors and Associated Equipment 506 250–303
[9] Shiers J et al. 1996 CERN Geneva
[10] Choutko V et al. 2015 Journal of Physics: Conference Series vol 664 (IOP Publishing) p 032029