Best Practice Guide Anselm

Bull Extreme Computing at IT4Innovations / VSB

Roman Sliva, IT4Innovations / VSB - Technical University of Ostrava

Filip Stanek, IT4Innovations / VSB - Technical University of Ostrava

May 2013

Table of Contents

1. Introduction 3
2. System architecture and configuration 3
    2.1. System configuration 3
    2.2. Memory architecture 5
    2.3. Network infrastructure 6
    2.4. I/O subsystem 6
    2.5. Filesystems 6
3. System Access 7
4. User environment and programming 8
    4.1. Generic x86 environment 8
    4.2. System specific environment 8
        4.2.1. Batch System 8
5. Tuning applications and Performance analysis 9
    5.1. General optimization for x86 architecture 9
    5.2. System specific optimization 9
        5.2.1. Runtime choices and options 9
        5.2.2. Memory optimizations 10
        5.2.3. I/O optimizations 10
        5.2.4. Specific optimized libraries 10
    5.3. Available generic x86 performance analysis tools 10

1. Introduction

This Best Practice Guide is intended to help users to get the best productivity out of the PRACE Tier-1 Anselm system.

Anselm is x86-64 Intel based supercomputer of the Bull Extreme Computing bullx serie located in Czech Republic.

Anselm is the first system operated at IT4Innovations. The aim of IT4Innovations is to establish Czech national supercomputer center and to provide Tier 1 HPC services. Anselm is hosted at VSB - Technical University of Ostrava which is the principal partner of IT4Innovations.

More information about IT4Innovations can be found on www.it4i.cz/en [http://www.it4i.cz/en]

More information about VSB - Technical University of Ostrava can be found on www.vsb.cz/en [http://www.vsb.cz/en]

2. System architecture and configuration

2.1. System configuration

Anselm is cluster of x86-64 Intel based nodes built on Bull Extreme Computing bullx technology with a total peak performance of 94.5 Tflop/s. The cluster contains four types of compute nodes:

- Compute nodes without accelerator:
  - 180 nodes
  - 2880 cores
  - two Intel Sandy Bridge E5-2665, 8-core, 2.4GHz processors per node
  - 64 GB of physical memory per node
  - 55.2 Tflop/s
  - bullx B510 blade servers
- Compute nodes with GPU accelerator:
  - 23 nodes
  - 368 cores
  - two Intel Sandy Bridge E5-2470, 8-core, 2.3GHz processors per node
  - 96 GB of physical memory per node
  - GPU accelerator 1x NVIDIA Tesla Kepler K20 per node
  - 33.6 Tflop/s
  - bullx B515 blade servers
- Compute nodes with MIC accelerator:
  - 4 nodes
  - 64 cores

  - two Intel Sandy Bridge E5-2470, 8-core, 2.3GHz processors per node
  - 96 GB of physical memory per node
  - MIC accelerator 1x Intel Phi 5110P per node
  - 5.1 Tflop/s
  - bullx B515 blade servers
- Fat compute nodes:
  - 2 nodes
  - 32 cores
  - 2 Intel Sandy Bridge E5-2665, 8-core, 2.4GHz processors per node
  - 512 GB of physical memory per node
  - two 128GB SLC SSD per node
  - 0.6 Tflop/s
  - bullx R423-E3 servers

In total, Anselm is composed of 209 compute nodes, 3344 CPU cores and 15.136TB RAM. Compute nodes with MIC accelerator and Fat compute nodes are not currently provided to PRACE users.

Figure 1. Anselm bullx B510 servers

Processor architecture

Anselm is equipped with Intel Sandy Bridge processors Intel Xeon E5-2665 (nodes without accelerator) and Intel Xeon E5-2470 (nodes with accelerator).

Intel Sandy Bridge E5-2665 processor

- eight-core
- speed: 2.4 GHz, up to 3.1 GHz using Turbo Boost Technology
- peak performance: *19.2 Gflop/s* per core
- caches:
  - L2: 256 KB per core
  - L3: 20 MB per processor
- memory bandwidth at the level of the processor: 51.2 GB/s

Intel Sandy Bridge E5-2470 processor

- eight-core
- speed: 2.3 GHz, up to 3.1 GHz using Turbo Boost Technology
- peak performance: *18.4 Gflop/s* per core
- caches:
  - L2: 256 KB per core
  - L3: 20 MB per processor
- memory bandwidth at the level of the processor: 38.4 GB/s

Operating system

The operating system on Anselm is Linux - bullx Linux Server release 6.3.

bullx Linux is based on Red Hat Enterprise Linux. bullx Linux is a Linux distribution provided by Bull and dedicated to HPC applications. In addition to general enterprise Linux functionalities, bullx Linux features an OS jitter reduction function, a scalable OFED InfiniBand stack and a set of optimizations (CPU-Set, GPU-Set, Lazy Page Migration).

2.2. Memory architecture

Compute node without accelerator

Compute node without accelerator is composed of:

- 2 sockets
- Memory Controllers are integrated into processors.
  - 8 DDR3 DIMMS per node
  - 4 DDR3 DIMMS per CPU
  - 1 DDR3 DIMMS per channel
  - Data rate support: up to 1600MT/s
- Populated memory: 8x 8GB DDR3 DIMM 1600Mhz
- One 4x QDR IB HCA provide connection to the InfiniBand interconnect used in the cluster.

Compute node with GPU or MIC accelerator

Compute node with GPU or MIC accelerator is composed of:

- 2 sockets
- Memory Controllers are integrated into processors.
  - 6 DDR3 DIMMS per node
  - 3 DDR3 DIMMS per CPU
  - 1 DDR3 DIMMS per channel

  - Data rate support: up to 1600MT/s
- Populated memory: 6x 16GB DDR3 DIMM 1600Mhz
- One 4x QDR IB HCA provide connection to the InfiniBand interconnect used in the cluster.

Fat compute node

Fat compute node is composed of:

- 2 sockets
- Memory Controllers are integrated into processors.
  - 16 DDR3 DIMMS per node
  - 8 DDR3 DIMMS per CPU
  - 2 DDR3 DIMMS per channel
  - Data rate support: up to 1600MT/s
- Populated memory: 16x 32GB DDR3 DIMM 1600Mhz
- One 4x QDR IB HCA provide connection to the InfiniBand interconnect used in the cluster.

2.3. Network infrastructure

Computing nodes of Anselm are interconnected by a high-bandwidth, low-latency Infiniband QDR network (IB 4x QDR, 40 Gbps). The network topology is a fully non-blocking fat-tree.

2.4. I/O subsystem

All compute nodes are equipped with 4x QDR Infiniband HCA, Gigabit Ethernet Controller and local disk subsystem (SATA or SAS).

2.5. Filesystems

There are two shared file systems on Anselm: `/home` and `/scratch`. All compute nodes have also local (non-shared) filesystem `/lscratch` (local scratch).

Table 1. Best usage of the file systems

Space	Usage	Protocol	Net Capacity	Throughput	Limitations	Access	Services
`/home`	home directory	Lustre	300 TiB	2 GiB/s	Quotas	Compute and login nodes	Partially backed up
`/scratch`	cluster shared jobs' data	Lustre	135 TiB	6 GiB/s	Quotas	Compute and login nodes	none
`/lscratch`	node local jobs' data	local	330 GB	100 MB/s	none	Compute nodes	none

File systems `/home` and `/scratch` are provided by Lustre. Both shared file systems are accessible in Infiniband network.

The general architecture of Lustre is composed of two metadata servers (MDS) and four data/object storage servers (OSS). Two object storage servers are used for file system `/home` and another two object storage servers are used for file system `/scratch`

Configuration of the storages

- *HOME Lustre object storage*
  - One disk array NetApp E5400
  - 227 2TB NL-SAS 7.2krpm disks
  - 22 groups of 10 disks in RAID6 (8+2)
  - 7 hot-spare disks
- *SCRATCH Lustre object storage*
  - Two disk arrays NetApp E5400
  - 106 2TB NL-SAS 7.2krpm disks
  - 10 groups of 10 disks in RAID6 (8+2)
  - 6 hot-spare disks
- *Lustre metadata storage*
  - One disk array NetApp E2600
  - 12 300GB SAS 15krpm disks
  - 2 groups of 5 disks in RAID5
  - 2 hot-spare disks

3. System Access

Application for an account

Academic researchers can apply for computational resources via IT4Innovations' Open Access Competitions.

IT4Innovations' access competitions are aimed at distributing computational resources while taking account of the development and application of supercomputing methods and their benefits and usefulness for society. Open Access Competition is held twice a year. Proposals must undergo a scientific, technical and economic evaluation. More information can be found on www.it4i.cz/en [http://www.it4i.cz/en]

Foreign (mostly European) users can obtain computational resources via the PRACE (DECI) program.

Login to the system

- Login with username and password

  Users with a username and password can login to Anselm by SSH: **ssh username@Anselm.it4i.cz**

- Login with a certificate (PRACE))

  Users with a valid certificate must use gsissh. Use following command if you come from a PRACE door-node or a PRACE site: **gsissh your-prace-username@login-prace.it4i.cz**

  gsissh uses the default port 2222. Alternatively, you can use a Java webstart application called *gsissh-term*.

4. User environment and programming

4.1. Generic x86 environment

- GNU:
  - Compilers : gfortran, gcc, g++
- Intel:
  - Compilers : ifort, icc
  - Libraries : MKL (Math Kernel Library) - includes BLAS, sparse BLAS, LAPACK, ScaLAPACK, PBLAS, sparse solvers, fast Fourier transforms, vector math, and more; Threading Building Blocks, Integrated Performance Primitives
- bullx MPI, OpenMPI
- Perl, Python, Java, Ruby
- Editors: vi, vim, mcedit, gedit and emacs

4.2. System specific environment

There is a wide range of software packages targeted at different scientific domains installed on Anselm. These packages are accessible with modules environment. To see which modules are available, type:

module avail

Modules environment

For more information on using `Modules` please refer to the PRACE Generic x86 Best Practice Guide [http://www.prace-ri.eu/IMG/pdf/Best-Practice-Guide-Generic-x86.pdf].

4.2.1. Batch System

The job scheduler on Anselm is PBS Professional 12 (Portable Batch System).

There are especially these operating queues:

- queue qprod for standard jobs
- queue qexp for testing of short jobs
- queue qprace for PRACE/DECI users' jobs

Jobs can be MPI, OpenMP, or hybrid MPI/OpenMP.

The maximum number of physical cores for a job is 2048.

Use following command for job submitting: **qsub -q queue-name job-script**

Use following command for showing jobs' status: **qstat [-f] [job-identifier]**

Using PBS Batch System

For more information on using Portable Batch System please refer to the PRACE Generic x86 Best Practice Guide [http://www.prace-ri.eu/IMG/pdf/Best-Practice-Guide-Generic-x86.pdf].

Portability and compatibility with other x86 systems

Availability of standard compilers combined with the Unix/Linux operating system provides good portability. Scientific applications need recompilation if ported from another x86 system.

5. Tuning applications and Performance analysis

5.1. General optimization for x86 architecture

For detailed information on code optimization on x86 architecture refer to the PRACE Generic x86 Best Practice Guide [http://www.prace-ri.eu/IMG/pdf/Best-Practice-Guide-Generic-x86.pdf].

5.2. System specific optimization

Anselm uses bullx Supercomputer Suite (SCS) AE3 software stack. bullx SCS provides number of mechanisms and enhancements to improve HPC experience.

To use the processors' AVX (Advanced Vector Extensions) instruction set compile with Intel compiler **-march=core-avx-i** flag.

5.2.1. Runtime choices and options

5.2.1.1. bullx MPI

bullx MPI is based on OpenMPI enhanced with a set of features that improve OpenMPI.

- Fine-grained process affinity using advanced placement technology through integration with workload/resource managers
- Kernel based data mover to insure intra-node performance. It enables a zero-copy technology, optimizing the memory bandwidth for MPI transfers within node.
- Fine-tuned demanding collective operations, using topological information.
- Fine-tuned MPI-IO operations. A tight coupling between MPI-IO with the Lustre infrastructure is provided.
- Multipath failover support.
- Etc.

These enhancements are implemented in a way that protects the compatibility with OpenMPI. Any program that runs with OpenMPI will also run (with at least the same level of performance) with bullx MPI.

Bullx MPI is installed in the /opt/mpi/bullxmpi/<version> directory.

MPI applications should be compiled using bullx MPI wrappers:

- C programs: mpicc
- C++ programs: mpiCC or mpic++
- F77 programs: mpif77
- F90 programs: mpif90

Wrappers to compilers simply add various command line flags and invoke a back-end compiler; they are not compilers in themselves. Bullx MPI currently uses Intel C and FORTRAN compilers to compile MPI applications.

To use Bullx MPI load appropriate module: **module load bullxmpi/bullxmpi-<version>**

For example: **module load bullxmpi/bullxmpi-1.2.4.1**

5.2.2. Memory optimizations

To take advantage of caches use data access patterns in a spatial and time locality.

5.2.3. I/O optimizations

To optimize I/O on a directory `dir`, just after creating the directory, type: **lfs setstripe –c 10 dir**

This command will divide all the files you will put in `dir` into stripes and spread these stripes over 10 OSS. In this manner the stripes can be accessed more quickly in read or write mode.

5.2.4. Specific optimized libraries

The Intel Math Kernel Library is optimized for the Intel Sandy Bridge processors of Anselm.

5.3. Available generic x86 performance analysis tools

- gprof
- PAPI
- HPCToolkit
- Open|Speedshop
- Scalasca

Many of these tools are available in module bullxde, use command: **module load bullxde/2.0**