Data Communications in an HPC Hybrid Cluster and Performance Evaluation

Ovidiu GHERMAN, Ioan UNGUREAN, Ştefan Gheorghe PENTIUC, Oana VULTUR
"Stefan cel Mare" University of Suceava
str.Universitatii nr.13, RO-720229 Suceava
ovidiuh@stud.usv.ro,ioanu@eed.us.ro,pentiuc@usv.ro, ovultur@stud.usv.ro

***Abstract* — The need for powerful computing systems is more present than ever in the today scientific (and not only) environment. Cluster systems built with general purpose processors (a trend that caught in the previous years) are superseded today by platforms built around specialized multicore processors, capable of processing massive amounts of arithmetic operations, usually from the field of graphic accelerators (GPU). These units are specifically created to manage intensive operations so they are the logical choice for this purpose. IBM took this step by creating a platform that uses a multicore accelerator CPU (Cell BE) for the HPC operations, managed by nodes with a more traditional architecture, with general purpose CPUs (AMD Opteron). This hybrid approach toward HPC is successful, even if the application deployment and software development can pose a certain degree of difficulty.**

***Index Terms* —data communications, networking, Cell BE, hybrid cluster, k-means, RISC architecture, speedup**

I. INTRODUCTION

More recently, the evolution of HPC systems relied on the development of advanced technologies using microprocessors, such as graphics processing units (GPU used previously in 2D and 3D graphics acceleration), that have the same number of transistors (or higher) with CPU units, but with specialized architectures for arithmetic intensive (like those used in generating 3D graphics). For this reason they are suitable to run applications in the HPC field.

One of the most famous systems in this category is Roadrunner (which held first place in the Top 500 in June 2009). This system was provided by IBM at the Los Alamos National Laboratories (USA). One of the strengths of this platform, which prevail in the face of dedicated high-performance systems, is the architecture – hybrid architecture achieved with conventional systems (AMD Opteron) and units to speed up calculations with RISC architecture (PowerPC).

This type of system is extremely efficient; the occupied floor space was much smaller than its alternatives for general purpose, and - extremely important - with very low power consumption (usually almost half of the consumption of traditional architecture).

The trend of replacing general-purpose processors in computation-intensive systems is growing - IBM and nVidia are also implementing hybrid systems (AMD and ATI also developing its own hybrid platforms).

This paper presents some of the characteristics of a hybrid cluster architecture using dedicated processors with RISC architecture (Cell BE processor type similar to those used in the Roadrunner system). These processors are used for arithmetic operation and they are controlled by management components built around general purpose CPU units. After the description of this system is presented the result obtained with Linpack benchmark, executed in the same conditions and specifications as those used on TOP500.org. It is followed by the presentation of an application for document clustering, which was developed on this platform. The increased performances are indicated next.

II. HYBRID ARCHITECTURE OF THE CLUSTER

A more complicated problem arises when we speak about hybrid architectures. An example of such platform is IBM Roadrunner system, which is based on the x86 (CISC architecture) and Cell BE eDP (RISC architecture, similar with PowerPC) processors. The structure of the system allows a special programming model, where the application written must consider the characteristics and architectural specifications of both processor types. Compiling the application is also subject to these restrictions. The user must manipulate his program so as to use the special computing capabilities of the accelerator processors (PowerXCell 8i).

The dedicated processor is composed of two types of internal cores [1]. Those nine cores, along with the interconnecting bus and DMA controller, form a powerful computing unit controlled by one of the internal core (master core) – PPE (PowerPC Processing Element). This core is based on a 64-bit PowerPC architecture but nevertheless has – in addition to the original instruction set, compliant with general purpose processors with similar architecture – an instruction set designed specifically for computing vector processors (Vector/SIMD Multimedia Extension).

This core is the main processing element, providing interface between the other 8 subordinate cores and the management node. Thus, it deals especially with the process management and problem decomposition in basic problems easier to handle by the arithmetic units.

The rest of the processor is composed of eight Synergistic Processing Elements (SPE), basically SIMD units optimized for data intensive operations (data allocated by the PPE because the SPE cores cannot communicate directly with other CPUs or external devices). Each of these identical elements contains a RISC core with 256 KB of memory to form a local storage (LS) for instruction and data, storage that can be manipulated by software.

Each unit has a unified 128-bit register file with 128

inputs for floating point or integer operations. SPE supports a special set of SIMD instructions (specifically developed to accelerate calculations in floating point arithmetic) and is based in asynchronous DMA transfers to move data and instructions between the main storage system (effectively addressing space that includes the main shared memory) and the LS. Thus the memory architecture is organized on three levels: main storage (shared), local store (LS) and register file.

The SPEs are not made to run an operating system, so it depends on the PPE unit to receive the workload [2].

The advantage of these elements is their performance in implementation of vector calculations (one of the reasons to name them ”accelerators”). As a result, the user application must be optimized for these units to be used at their full potential.

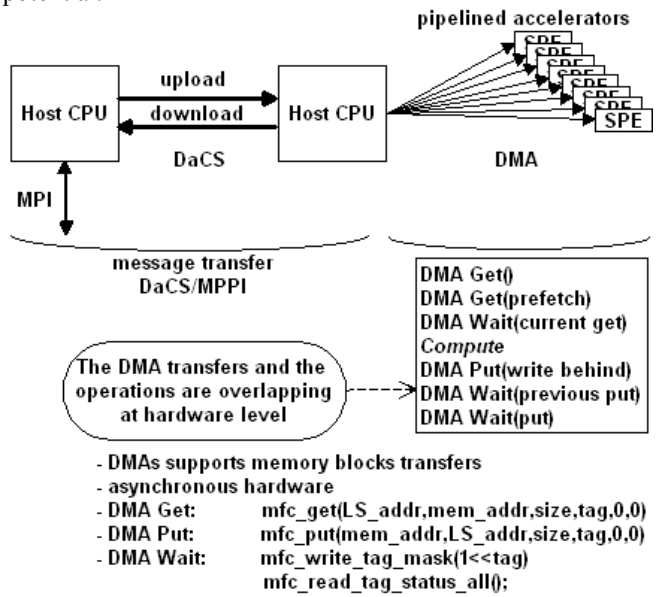

Fig. 1 The internal workflow of a Cell processor [3]

Every SPE element has a modular design, consisting of SPU (Synergistic Processing Unit) and a MFC (Memory Flow Controller) independent unit – a DMA controller that uses a memory management module to attain synchronized operations with other SPEs on the chip or with PPE. SPU uses instructions and data taken from the local storage system [4] and communicate through a dedicated channel, integrated in MFC, with the shared main memory or other local systems (on the same physical chip) – Figure 1.

As said before, once a problem is decomposed in sub-problems that can be solved by the accelerators, those sub-problems are sent to all SPE cores. These vector processors do not have much local memory (256 KB shared between the program code and the data in use) but works at high frequency (3.2 GHz) and can use DMA memory transfers (up to 204 GB/s peak speed). The Element Interconnect Bus (EIB) manages the coherent communications between PPE, SPEs and I/O and it has a ring structure with 4 channels and can process more than 100 concurrent DMA request between cores. Its bandwidth is 96 bits per cycle, with half the speed of the CPU.

Because of this way of working, the Cell BE has outstanding performance when dealing with small data blocks. The most effective way of using the system is when an application running on the parallel PPEs must quickly make a series of small granularity calculations (intensive computations). Moreover, it is recommended that these calculations to involve data structures that can be vector-processed (e.g. two-dimensional arrays will be processed line by line). This “offloading” technique requires using some acceleration libraries specifically created for Cell BE (DaCS, Alf) when dealing with communication between PPE and SPEs, but at the PPE level (or communicating with other CPUs) the MPI protocol can be used – basically we have a two-level parallelization (with different granularity and different working domains) – Figure 2.

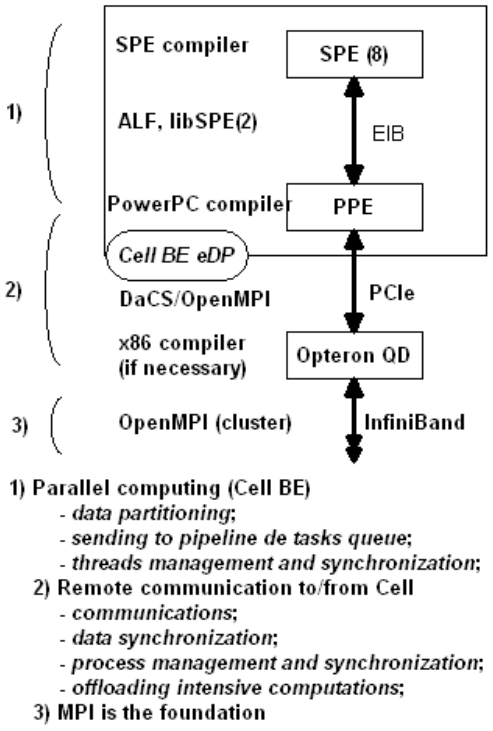

Fig. 2 Programming model in a hybrid architecture [3]

The purpose of the application at the PPE level can be either computational (although probably will not achieve the acceleration performance of a dedicated processor) or problem management (e.g. data formatting, problem decomposition, task assignment to computing cores, communication with other CPU, data I/O, etc.). This complicates the writing of a program that must run on such platform. Besides the need to use code sharing between the two processor types (existing applications requires at least partial rewriting of the source code), the programmer must take into account Cell BE particularities if it is desired high performance (memory access, data formats, working with data structures, DMA, etc.). Usually, a new technique for making programs is required [5].

From a functional point of view, cluster structure is a hybrid one, with management nodes optimized for process and distributed applications management. These nodes (LS22 dual CPU blades with Opteron 2376 quadcore) are not used in performance calculations, but for distribution of task, applications, files and jobs. The environment used for distributed applications is MPI (OpenMPI 1.3.3 installed on Red Hat Linux Enterprise Edition 5 operating system).

These I/O nodes are in communication with the subordinate acceleration nodes (QS22 dual CPU blades), each Opteron processor managing 6 PowerXCell 8i – Figure

3. The integrated accelerators (on the same blade) can not communicate directly with other accelerators, but with the master processor (via a PCI Express bus).

The operating system is either RHEL 5 (if we are to use a variety of software tools for programming hybrid applications – compilers, debuggers, tracers, development environments) or a highly optimized Fedora Core 9 kernel when we wish to achieve high performance (developments tools are poor) – the two SO are interchangeable. Pre-existing programs must be at least partially rewritten and properly compiled to run on the platform.

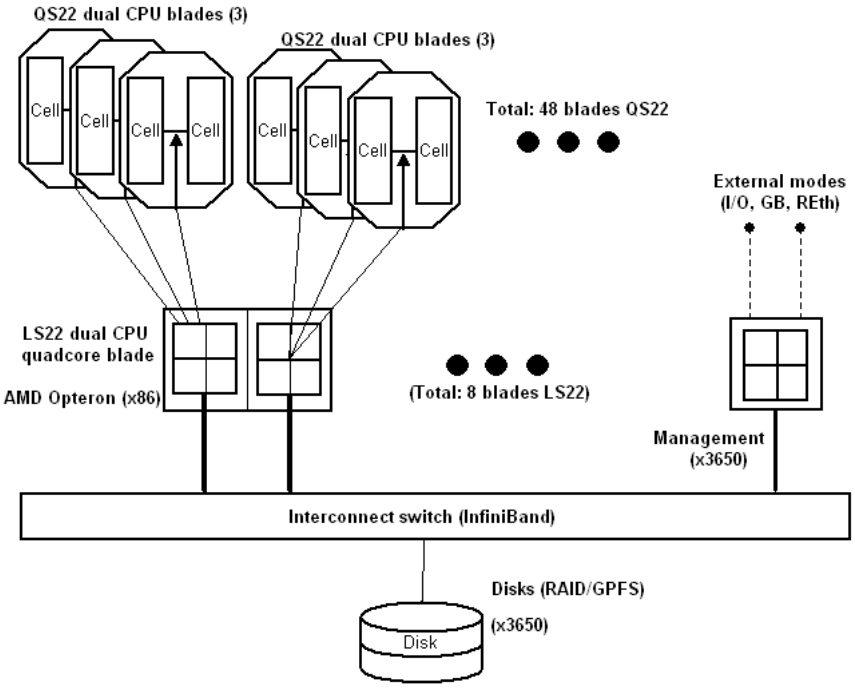

Fig. 3 Hybrid structure of a typical RoadRunner cluster

Since QS22 accelerator units can communicate with the cluster environment only through the host nodes (LS22), they will depend on them to get computing tasks. This technique involves two layer of problem parallelization, and can be viewed either in terms of Cell unit ("Cell-centric") or from the host processor ("host-centric") [3][6]. For developing applications that run on the cluster, we can use either of these two methods, especially since the LS22 management nodes does not contribute to computing, but only to managements of QS22 units. Therefore, an elegant solution that solves this problem is the decomposition method in features depending on the complexity of calculations. However, the performance of such system (e.g. power consumption) is very good (in case of tested system, the performance has been validated with a Linpack test).

Although, this style of programming is at the beginning, the trend to use such units (vector multi-core processors) increases exponentially. Because of this, IBM is trying to expand their hybrid libraries for non-hybrid systems (especially ALF and DACS), precisely to avoid the compatibility problems that may arise. So, most likely, they will compete with distributed programming environments such as MPI [3].

The hybrid cluster requires an efficient internal architecture that can allow fast data transfers between its components so that to minimize or eliminate the bottlenecks that can occur (and such can reduce the overall speedup of the processing platform). The most effective (and proven) solution is to use a high speed interconnect bus (InfiniBand) for data transfers and an additional Ethernet Gigabit interconnect for management and administration.

The high speed bus is composed by four Infiniband commutation matrices (Voltaire IB switches), each one controlling one BladeCenter. All internal equipments (the blades) are connected through it with 4x DDR interfaces that allow a peak speed of 20 GB/s.

The commutation matrices are connected in a full mesh network (every node is connected with the all other nodes) to maximize the system performance (reducing latency by eliminating possible intermediary nodes).

All matrices are connected with the "outside" network using external 4x QDR external ports (max. 40 GB/s / port).

To increase the system performance, GPFS (IBM General Parallel File System) file system is used. This file system allow concurrently access of same the same file by more process, each having exclusive rights to the current block, greatly increasing the performance.

III. HIBRID CLUSTER PERFROMANCE

The performance certified for the hybrid system (using kernels based on Fedora Core 9 to accelerate the arithmetic operation) obtained by the Linpack (hybrid) test is 6.53 TFlops (Figure 4). Cluster was optimized by IBM Germany team in cooperation IBM Romania team.

Also, this system is very efficient in terms of energy consumption, exceeding all platforms in this aspect (composed - in most cases – of general purpose processors). Thus, the energy efficiency of the hybrid system is 437 MFlops/watt, compared with equivalent systems, whose efficiency usually does not exceed 270-378 MFlops/watt (BlueGene - in the latter case).

```
================================================================================
T/V            N   NB    P    Q          Time          Gflops
--------------------------------------------------------------------------------
WR01C2C2    191999  128   16    6        721.94      6.536e+03
--------------------------------------------------------------------------------
||Ax-b||_oo / ( eps * ||A||_1  * N      ) =        0.0237506 ..... PASSED
||Ax-b||_oo / ( eps * ||A||_1  * ||x||_1 ) =        0.0160849 ..... PASSED
||Ax-b||_oo / ( eps * ||A||_oo * ||x||_oo ) =       0.0026390 ..... PASSED
================================================================================

Finished      1 tests with the following results:
              1 tests completed and passed residual checks,
              0 tests completed and failed residual checks,
              0 tests skipped because of illegal input values.
--------------------------------------------------------------------------------

End of Tests.
================================================================================
```

Fig. 4 Linpack benchmark results

IV. EXPERIMENTAL RESULTS

An application that uses a parallel k-means algorithm for document clustering was used in order to test the hybrid system. These blade servers include two PowerXCell 8i processors, the new generation of processors based upon the Cell Broadband Engine (Cell/B.E.) Architecture [7].

In order to test the proposed algorithm, the input data sets offered by UCI Machine Learning Repository [8] were used. Four input sets were helpful: the KOS blog entries that contain 3430 documents with 6906 of dictionary size.

The tests were carried out on a cluster foreseen with 48 QS22 nodes. Regarding these tests, the effective time of computing was measured, but the time necessary for reading the input dataset of the file and the distribution of this dataset towards the computing nodes were not taken into consideration. During tests, two MPI processes run on each node (one for each PowerXCell 8i processor of the QS22 blade server). For instance, if two nodes are used for a test, then the application will execute four MPI processes. The input dataset is shared equally to each MPI process. Since every MPI process corresponds to a CELL processor, the distribution within a MPI process of the input data set and of the calculus will be performed at those eight SPE processors specific to PowerXCell 8i processor, attached to the current MPI process.

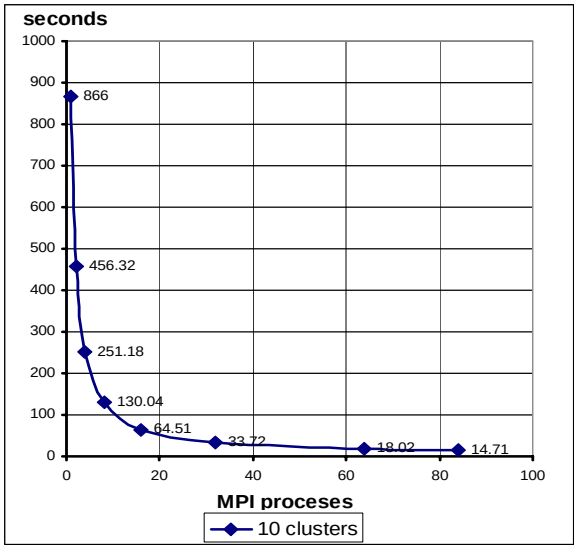

Fig.5 Experimental results of the KOS dataset and 10 clusters

Considering these results, one might analyze that by increasing the number of MPI processes, a significant reduction, specific to execution time, takes place. Subsequently, the execution time will increase if the number of clusters, where documents are grouped, is increased. Fig. 5 shows the reducing of execution time for documents' grouping within the KOS input dataset into 25 clusters, by means of increasing the number of MPI processes (QS22 nodes), where application runs.

V. CONCLUSION

The hybrid cluster system is a powerful system for the execution of the HPC tasks, holding state of the art technology. The special scalability of this type of system (as it was proven on the much bigger system of Los Alamos laboratories - this scaling is almost linear even on a system with over 10.000 processors) makes possible the extension of the number of the computation units in a simple way and with no alterations of the infrastructure (by adding LS22 and QS22 blades in the 1:6 ratio in case of the tested system).

Within the present paper, a parallel algorithm specific to document clustering was approached. This algorithm was implemented and tested on a cluster of RoadRunner hybrid architecture. Analyzing the results of the experimental tests, one might observe that by doubling the number of QS22 nodes, where the application is running, the execution time will be reduced by a percentage highly near to 50% (certain delays might occur, due to the communication amongst QS22 nodes). The next step is to parallelize algorithms from gesture recognition such as are described in [9].

ACKNOWLEDGMENTS

The cluster used in order to carry out the experiments belongs to the University "Stefan cel Mare" of Suceava, and was purchased within a national grant entitled "Grid for Developing Pattern Recognition and Distributed Artificial Intelligence Applications – GRIDNORD", contract no. 80/13.09.2007, PN II, Research Capabilities Programme.

REFERENCES

[1] T. Chen, R. Raghavan, J. Dale, E. Iwata, "Cell Broadband Engine Architecture", IBM Journal of Research and Development, vol. 51, issue 5, 2007.

[2] ****, "Cell Broadband Engine – Programming Handbook v 1.1", IBM Systems and Technology Group, available: https://www-01.ibm.com/chips/techlib/techlib.nsf/techdocs/7A77CCDF14FE70D5852575CA0074E8ED.

[3] K. Koch, "Roadrunner Platform Overview", Roadrunner Technical Seminar Series, Los Alamos National Laboratory, March 2008.

[4] ****, "SPU Application Binary Interface Specification v 1.9", CBEA JSRE Series, Cell Broadband Engine Architecture, Join Software Reference Environment Series, July 2008

[5] C. Kessler, "Programming Techniques for the CELL Processor", Multicore Day seminar 2009, Kista, Sweden.

[6] J.A. Turner, "The Los Alamos Roadrunner Petascale Hybrid Supercomputer – Overview of Applications, Results and Programming", Roadrunner Technical Seminar Series, Los Alamos National Laboratory, March 2008.

[7] ****, "IBM BladeCenter QS22", available: http://www-03.ibm.com/systems/bladecenter/hardware/servers/qs22/

[8] A. Asuncion, D.J. Newman, UCI Machine Learning Repository, Irvine, CA: University of California, School of Inf. and Comp. Sci, 2007, available: http://www.ics.uci.edu/~mlearn/MLRepository.htm

[9] Cristian Andy Tănase, Radu-Daniel Vatavu, Ştefan Gheorghe Pentiuc, Adrian Graur (2008), "Detecting and Tracking Multiple Users in the Proximity of Interactive Tabletops", Advances in Electrical and Computer Engineering, Volume 8 (15), Number 2 (30), 2008, University „Stefan cel Mare" of Suceava, ISSN 1582-7445, pp. 61-64