APPLICATIONS OF MASSIVELY PARALLEL COMPUTERS IN TELEMETRY PROCESSING

Tarek A. El-Ghazawi*, Jim Pritchard**, and Gordon Knoble

*Department of Electrical Engineering and Computer Science
The George Washington University
Washington, D.C. 20052

**Information Processing Division, Code 560
Mission Operations and Data Systems Directorate
Goddard Space Flight Center
Greenbelt, MD 20771

ABSTRACT

Telemetry processing refers to the reconstruction of full resolution raw instrumentation data with artifacts, of space and ground recording and transmission, removed. Being the first processing phase of satellite data, this process is also referred to as level-zero processing.

This study is aimed at investigating the use of massively parallel computing technology in providing level-zero processing to spaceflights that adhere to the recommendations of the consultative committee on space data systems (CCSDS). The workload characteristics, of level-zero processing, are used to identify processing requirements in high-performance computing systems. An example of level-zero functions on a SIMD MPP, such as the MasPar, is discussed. The requirements in this paper are based in part on the Earth Observing System (EOS) Data and Operation System (EDOS).

1. Introduction

Telemetry processing refers to end-to-end delivery for satellite systems adhering to the Consultative Committee for Space Data Systems (CCSDS) recommendations. This involves link processing as well as production data processing. Return link processing includes: Data Capture, real-time processing, playback processing, and rate buffering. On the other hand, production data handling includes production data processing, quick-look data processing, and level-0 backup archiving.

In return link processing: data capture refers to receiving all unprocessed telemetry data, including fill data, and storing it for a predetermined period of time for use in recovery processing. Real-time processing entails receiving and processing return link data of urgent nature, such as data pertaining to health, and delivering it to its sinks (earth ground system units) with minimal delay, as required. Playback processing, on the other hand, restores the "as recorded order" as data is originally stored on magnetic tape recording devices. This playback processing starts after the completion of one communication session during which telemetry data is received, a TDRSS session in EDOS. Finally, rate-buffering in which data from a spacecraft is received at one rate and transmitted to its final ground destination at another rate.

In production data handling, production data processing of CCSDS packets is the process in which packets from one or more communications sessions with the spacecraft, TSS in EDOS, are sorted by application process identifier (APID), quality checked, and forward ordered by sequence counter. Further, redundant and previously processed packets are deleted and a production data set is formed. Quick-look data processing is similar to production data processing except it is limited to packets from one communication session, one EDOS TSS. It could include all packets of that session or only those that have the same APID. Level-0 data archiving is for storing the production data sets created by the above processes.

Recent advances in high-performance computing have demonstrated that supercomputers based on massively parallel scalable architectures have the potential to offer a much higher performance/cost than traditional vector supercomputers. In fact machines with performance approaching 10 GFLOPS can be now purchased at less that $1M. Combining all this with the flexibility offered by off-the-self general-purpose computers creates a great potential for the use of such technology in telemetry processing.

2. Telemetry Processing and Massively Parallel Computers

Some of the concerns that arise from suggesting a massively parallel high-performance computing architectures for telemetry processing are (1) can the massive hardware parallelism in such architectures be exploited in the processing, (2) can the I/O cope up with the tremendous data rates and adequately balance storage/retrieval and processing activities?, (3) what are the classes of parallel computers that best suite such kind of processing?, (4) is the cost reasonable?.

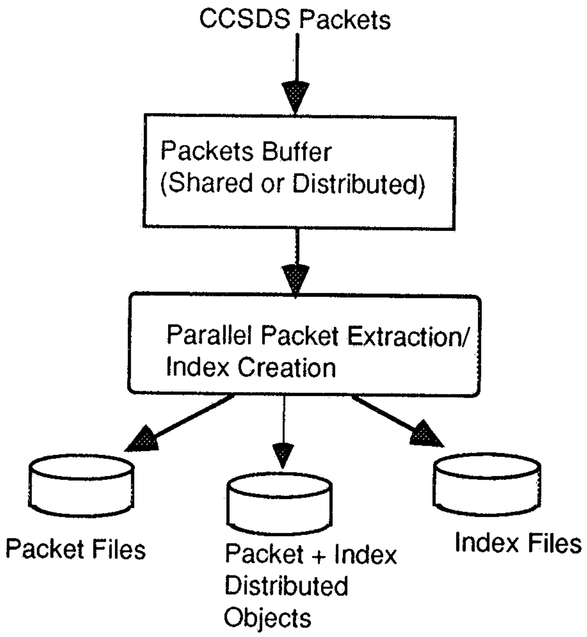

CCSDS Packets
Packets Buffer (Shared or Distributed)
Parallel Packet Extraction/ Index Creation
Packet Files
Packet + Index Distributed Objects
Index Files

Figure 1

CCSDS Parallel Packet Extraction and Generation

Telemetry processing has inherent parallelism which is proportional to the rate of packet arrival. This is because such packets need to be indexed and the indices have to be kept sorted. Sorting the indices can be performed efficiently on such parallel machines when large volumes of data are involved, due to the large degree of data parallelism. This is exactly the case in near future telemetry processing systems. The rate of packet telemetry in such near future systems as the Earth Observing Systems (EOS) Data and Information Systems (EDOS) is projected at a range of 70K to a 120K packets per second [EDOS92C].

Modern massively parallel systems are designed with I/O scalability in mind. This basically means that as more processors participate in a parallel I/O operation, more I/O bandwidth become available. With the real-time nature and the massive data parallelism present in telemetry processing, parallel I/O can be coordinated to take full advantage of the scalability of the I/O systems. Sorting and storing indices are examples of scenarios that give rise to utilizing the power of the massive hardware parallelism and the scalability of the I/O systems. In such processing, similar operations needs to be performed on the data items in these large data sets. This massive data parallelism could be exploited efficiently with the hardware parallelism of massively parallel processing (MPP) computers. Since such data sets could be orders of magnitude larger than the available number of processors and their memory in an MPP, parallel I/O could be used to distribute data onto the processors. Processing and I/O have to be coordinated in order to minimize I/O and amortize I/O overhead using largest possible data block sizes.

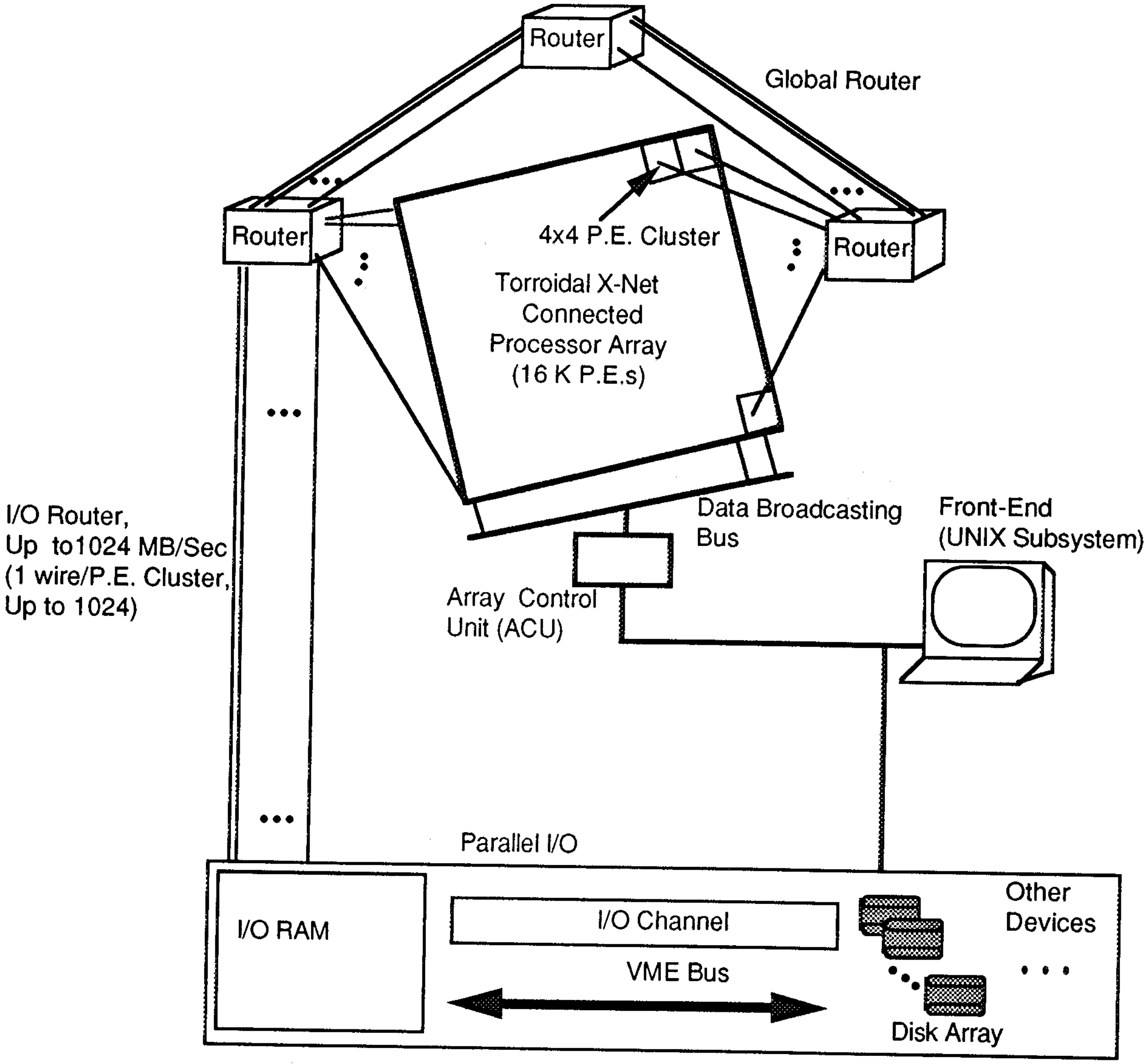
Figure 1 outlines some to of the processing to be initially performed on arriving CCSDS packets. As a one load arrives, it will be partitioned among the systems processors through a parallel read. The exact size of a load depends on the specifics of the MPP system and the telemetry system parameters. All processors scan their data, searching for the first packet boundary in their respective data block. Thereafter, processing could jump over data extracting only index information from packet headers. Index information could be then added into index files, or used to design standardized objects which can contain both indices and packet data. Such object can be distributed in non RAID systems [Katz89], where the mass storage is distributed. The packet data could be gathered and stored at that point into packet files. In preparation for data production, indices must be kept sorted by APID and packet sequence numbers. This means that indices from one load need to be sorted, then merged with the previously sorted indices. Sorting and merging can be done efficiently with MPPs and many parallel algorithms for these operations already exist, [Akl85] and [Akl89]. It must be noted, however, that index files could be very large and hard to accommodate in processors memory. External sorting and/or caching of portions of such Redundant packets can be eliminated during indices sorting or at data sets productions. Doing so at sorting time could cost additional memory accesses or interprocessor communication steps for compacting the data.

3. A Case Study

3.1 A Case for SIMD Architectures

SIMD (Single Instruction Multiple Data) architectures are designed mainly to exploit data parallelism. In such class of architectures, all processors (also called processing elements or PEs) execute the same instruction synchronously and under the guidance of a centralized control unit. Due to the synchronous central control, such machines are very cost effective. With the real-time nature of telemetry processing and the massive data parallelism in this domain SIMD machines have the potential for delivering high performance/cost in telemetry processing. Thus, we are currently developing a telemetry processing architecture based on that technology and creating a scaled-down benchmark for such architecture using the MasPar SIMD MPP.

3.2 MasPar System Overview



The MasPar Architecture

Figure 2

MasPar Computer Corporation currently produces two families of massively parallel-processor computers, namely the MP-1 and the MP-2. Both systems are essentially similar, except that the second generation (MP-2) uses 32-bit RISC processors instead of the 4-bit processors used in MP-1. The MasPar MP-1 (MP-2) is a fine-grained, massively parallel computer with Single Instruction Multiple Data (SIMD) architecture. The MasPar has up to 16,384 parallel processing elements (PEs) arranged in a 128x128 array, operating under the control of a central array control unit (ACU), see figure 3. The processors are interconnected via the X-net into a 2-D mesh with diagonal and toroidal connections. In addition, a multistage interconnection network called the global router (GR) uses circuit switching for fast point-to-point and permutation transactions between distant processors. A data broadcasting facility is also provided between the ACU and the PEs. Every 4x4 grid of PEs constitutes a cluster which shares a serial connection into the global router. Using these shared wires, array I/O is performed via the global router, which is directly connected to the I/O RAM as shown in figure 2. The number of these wires, thus, grows as the number of PEs to provide for scalable I/O bandwidth. Data is striped across the MasPar disk array (MPDA), which uses a RAID-3 configuration. For more information on the MasPar, the reader can consult the MasPar references cited at the end of this study [Bla90], [Mas92], [Nic90].

4.3 MPP Telemetry Processing on the MasPar

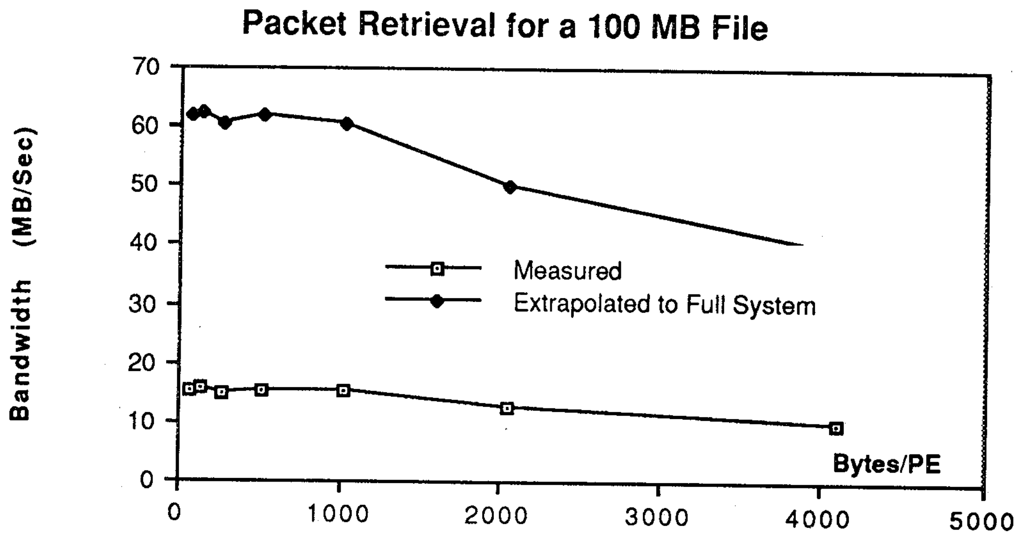


Figure 3

Retrieval Rates for Different Packet Block Sizes at a 100 MB Load

On the MasPar, blocks of packet data streams can be given to each processor. All processors can proceed simultaneously searching for the packet headers and forming the corresponding indices. Retrieving and storage of such packets data need to be performed at a sustained rate at least equal to the incoming data rate. At 70 packets per second and using the EDOS average of 819 bytes per packet [EDOS 92C], data can typically arrive at a 55MB/Sec rate. In EDOS, however, data speed is bottlenecked by ground communications and no more than 150 Mb/Sec arrival rate will be needed. The MasPar seems to be able of handling the I/O rates required by EDOS, but it could have some problems handling the 55

MB/Sec data rate. Measurements were collected on a MasPar system whose parallel disk I/O has a published sustained performance rate of 16 MB/Sec. A full-blown MasPar I/O system, however, has a published sustained performance of about 64 MB/Sec. Our measurements indicate that these published rates are achievable when I/O is amortized over sufficiently large files, more 100 MB, see figure 3. Using the same 819 B average packet and the 70K Packets/Sec rate, a 100 MB is accumulated in a little less than 2 sec.

5. Conclusions and Future Directions

This study supports the belief that SIMD MPPs could provide cost efficient solutions for today's telemetry processing. The initial results demonstrate that although at certain points such systems could not be completely adequate some customization could be done to satisfy the requirements, such as adding more disks or clustering more than one of these systems. We are currently proceeding with a benchmark that will demonstrate the know how and the performance constraints of using these SIMD architectures. Further, our future work will also include novel ways of indexing telemetry packets and applying temporal database concepts on parallel systems in general. Our work will also include performance comparisons of using such a SIMD machine versus the other classes of high-performance computer architectures.

References

[Akl89] S. Akl, The design and Analysis of Parallel Algorithms. Prentice-Hall, 1989.

[Akl85] S. Akl, Parallel Sorting Algorithms. Academic Press, 1985.

[Biswas93] P. Biswas, K.K. Ramakrishnan, and D. Towsley. Trace Analysis of Write Caching Policies for Disks. Proc. of 1993 ACM SIGMETRICS, pages 13-23, 1993.

[Brau89] C.K. Baru and O. Frieder, "Database Operations in a Cube -Connected Multicomputer System, "IEEE Transactions on Computers, 38(6), pp 920-927, June 1989.

[EDOS92A] Earth Observing System (EOS) Data and Operations System (EDOS) Operations Concept. 560-EDOS-0106.0002. IPD, Code 560, NASA GSFC. Greenbelt, December 1992.

[EDOS92B] Earth Observing System (EOS) Data and Operations System (EDOS) Functional and Performance Specifications. 560-EDOS-0202.0004. IPD, Code 560, NASA GSFC. Greenbelt, November 1992.

[EDOS92C] Earth Observing System (EOS) Data and Operations System (EDOS) Traffic Model. 560-EDOS-0106.0002. IPD, Code 560, NASA GSFC. Greenbelt, Dec. 1992.

[El-Ghazawi94B] Tarek A. El-Ghazawi. "I/O Performance of the MasPar MP-1 Testbed". TR-94-111. CESDIS, Code 930.5, NASA-GSFC, January 1994.

[Frieder90] O. Frieder, Multiprocessor Algorithms for Relational Databases Operators on Hypercube Systems, "IEEE Computer, 23(11), November 1990.

[Hwang93] K. Hwang. Advanced Computer Architecture: Parallelism, Scalability, and Programmability. McGraw-Hill, 1993.

[Johnson 93] Theodore Johnson, "Supporting Insertions and Deletions in Striped Parallel File Systems,"7th IPPS, April 1993.

[Katz89] R. H. Katz, G. A. Gibson, and D. A. Patterson, "Disk System Architecture for High Performance Computing", Proc. of the IEEE, Dec.1989, 1842-1858.

[Kotz91] D. Kotz and C. Eillis, "Practical Prefetching Techniques for Parallel File Systems". Proceedings of the IEEE DCS'91.

[Miller 91] E. L. Miller and R. H. Katz, "Input/Output Behavior of Supercomputing Applications", Proc. of Supercomputing'91, 567-576.

[Patterson92] R. Patterson, G. Gibson, M. Stayanarayanan, "Using Transparent Informed Prefetching (TIP) to reduce File Read Latency", Goddard Conference on Mass Storage Technologies, September 1992.

[Staelin88] C. Staelin. File Access Patterns. Tech. Rep. CS-TR-179-88, Department of Computer Science, Princeton University, Sept. 1988.

[Tansel93] Tansel, Clifford, Gadia, Jajodia, and Seguv. Temporal Databases: Theory, Design, and Implementation. Benjamin/Cummings, 1993.