International Journal of Electronics Signals and Systems

Volume 1 | Issue 1 Article 2

January 2011

Design and Implementation of a Lossless Serial High-Speed Data Compression System

J. Sunil Kumar Mr.
Nalla Malla Reddy Engineering College (JNTU-HYDERABAD), Sunil5718@gmail.com

G. Mahesh Kumar Mr.
Nalla Malla Reddy Engineering College (JNTU-HYDERABAD), maheshkumargubbala@gmail.com

Recommended Citation

Sunil Kumar, J. Mr. and Mahesh Kumar, G. Mr. (2011) "Design and Implementation of a Lossless Serial High-Speed Data Compression System," *International Journal of Electronics Signals and Systems*: Vol. 1 : Iss. 1 , Article 2.
DOI: 10.47893/IJESS.2011.1001
Available at: https://www.interscience.in/ijess/vol1/iss1/2

Available online at www.interscience.in

Design and Implementation of a Lossless Serial High-Speed Data Compression System

J. Sunil Kumar & G. Mahesh Kumar
Nalla Malla Reddy Engineering College (JNTU-HYDERABAD)
Email: Sunil5718@gmail.com, maheshkumargubbala@gmail.com

Abstract **-** The paper presents a novel VLSI architecture for high-speed data compressor designs which implement the X-Match algorithm. This design involves important trade off that affects the compression performance, latency, and throughput. The most promising approach is implemented into FPGA hardware. This device typical compression ratio that halves the original uncompressed data. This device is specifically targeted to enhance the performance of Gbits/s data networks and storage applications where it can double the performance of the original systems. To get high compression rate or to get high data rate of communication not restriction to follow the parallel architecture of data compression. By using existing method the main draw backs are 1. Variation in compression 2. Throughput, 3.Latency, 4.High space, 5. High power. So by using this proposed method we can reduce the variation in the compression, latency and increase through put. And this novel VLSI architecture has a power consumption of 81mwatts power

Keywords: *Data compression, Match logic unit, Lossless, X-MatchPRO, FPGA.*

I. INTRODUCTION

Information has become one of the most important commodities of the 21st century, and there appears to be insatiable demands for ever-greater bandwidth in communication networks and computer buses and for ever-greater storage capacity in computer systems. For example, in communication networks, standards are under development to move from 1Gbit/s Ethernet to 10Gbit/s fast Ethernet and, in computer busses, the latest successor to the PCI bus is the PCI-X standard capable of delivering a bandwidth of 4.3Gbit/s. A more efficient use can be made of available bandwidth or storage if lossless compression is performed on the data involved. However, data compression will probably only be adopted if it can meet the bandwidth requirements of modern systems, otherwise, the compression itself would become the bottleneck in these systems [1].

The remainder of this paper is organized as follows Section 1: Introduction section, Section 2: Establishes the motivation of our work, Section 3: Review of Serial Data compression, Section 4: Describes the basic architecture of the X-MatchPRO algorithm, Section 5: Describes the serial high speed data compression architecture, Section 6: Compares our device with other high performance loss less data compressions methods, Section 7: Concludes this paper

II. MOTIVATION

To overcome some of the drawbacks of existing methods, the authors have investigated parallel data compression approaches. One possible approach is to construct or modify an existing algorithm with the aim of exploiting inherent parallelism. However, existing compression algorithms are not inherently parallel and to adapt them to parallel architectures would need significant simplifications that would adversely affect compression performance.

Another approach would be to share the compression between a number of identical algorithms running concurrently. There would be no performance gain if the algorithm runs on a single CPU, but with recent advances in chip logic densities, it would be possible to integrate a number of compression engines into a single chip, and this is the approach considered in the work described in this paper.

There are two important contributions made by the current serial compression and decompression work, namely, improved compression rates and the inherent scalability. Significant improvements in data compression rates have been achieved by sharing the computational requirement between compressors. The scalability feature permits future bandwidth or storage demands to be met by adding additional compression engines.

The paper briefly reviews the serial hardware compression literature, before describing the high-performance X-MatchProRli[4] processor that is used as the compression engine in the remainder of the work. The serial X-MatchProRli section discusses sequential routing strategies, describes the effect each has on compression performance and presents simulation results. To demonstrate proof of concept, the implementation of an FPGA device containing two compressors is described.

THE BASIC ARCHITECTURE OF THE X-MATCH.

A .Serial compression architecture details.

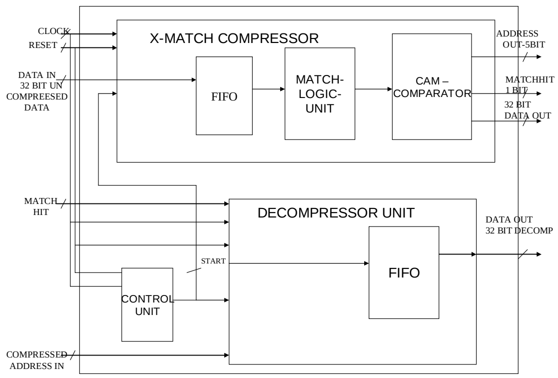

Fig 1. Serial Architecture for data Compression / De-Compression

III. REVIEW OF SERIAL DATA COMPRESSION

The majority of work on hardware approaches to lossless parallel data compression has used an adapted form of the dictionary-based Lempel-Ziv algorithm, [2] in which a large number of simple processing elements are arranged in a systolic array. A second Lempel-Ziv method uses a content addressable memory (CAM), capable of performing a complete dictionary search in one clock cycle. The search for the most common string in the dictionary (normally, the most computationally expensive operation in the Lempel-Ziv algorithm)[2] can be performed by the CAM in a single clock cycle, while the systolic array method uses a much slower deep pipelining technique to implement its dictionary search. However, compared to the CAM solution, the systolic array method has advantages in terms of reduced hardware costs and lower power consumption, which may be more important criteria in some situations than having faster dictionary searching.

IV. X-MATCHPRORLI

Previous research on high-speed lossless compression in the Electronic System Design Group at Loughborough University resulted in the development of the CAM-based X-MatchProRli algorithm optimized for high speed, good compression performance, and low complexity for hard-ware implementations by using VHDL Language [5].

From the Fig 1 it explains as for single-byte CAM architectures, data throughput improvements can only result from the shortening of the cycle time which, in turn, largely result only from silicon technology advancements. To make significant improvements in data throughput, it is necessary to implement a scheme permitting the processing of more than one byte simultaneously. As increasing the data granularity has the consequence of reducing the success of CAM data matches, a larger dictionary is necessary to maintain compression performance. However, the larger the dictionary, the greater the number of address bits needed to identify each memory location, reducing compression performance. Clearly, to maximize throughput, a compromise involving granularity and dictionary size must be made.

This observation led to the development of the X-MatchProRli architecture, which allows partial matching of incoming data with the data stored in the dictionary. This has the effect of increasing the effective length of the dictionary, while at the same time reducing the required number of address lines. Practical investigations revealed that when using 4-byte wide granularity in the data stream, X-MatchProRli was able to apply data width parallelism to its algorithm to improve throughput without compromising compression performance. This feature offers X-MatchProRli processing speed advantages compared with the majority of compression algorithms that are based on a granularity of a single bit or of a single byte. The X-MatchProRli algorithm attempts to match a 4-byte data element with previously seen data entries in a dictionary implemented in a CAM. As each entry is also 4-bytes wide, several types of match are possible. If fewer than two bytes match in the dictionary, the full four bytes are transmitted with an additional miss bit. If all bytes are matched, then both the match location and match type are coded and transmitted, and this match is then moved to the front of the dictionary. If the incoming four bytes are partially matched, then the match location and match type are transmitted along with the bytes that do not match.

Initially, all the entries in the dictionary are empty and 4-bytes are added to the front of the dictionary, while the rest move one position down if a full match has not occurred. However, when dealing with full matches a move-to-front technique is applied. In this case, the data from the first location remains same until the location previous to the matching data move down one location, while the matching data is placed at the front of the dictionary. The number of entries in the dictionary grows dynamically, thus if the input data only contains a few different 4-byte data elements, then the dictionary remains small. Since the number of bits needed to code each location address is a function of the dictionary size greater compression is obtained in comparison to the case where a fixed size dictionary uses fixed address codes for a partially full dictionary.

X-MatchProRli uses a pipelining technique to allow steps in the compression and decompression process to be carried out simultaneously and so to increase throughput. The X-MatchProRli design has been fully implemented and tested in FPGA technology with data independent throughput speeds in excess of 1.1Gbit/s. However, attempts to extract further internal parallelism from the X-MatchProRli algorithm produced diminishing returns and any future substantial improvements are likely to result only from silicon technology advances. This has directed the investigations to increase throughput toward architectures and routing strategies for multiple X-MatchProRli compressors.

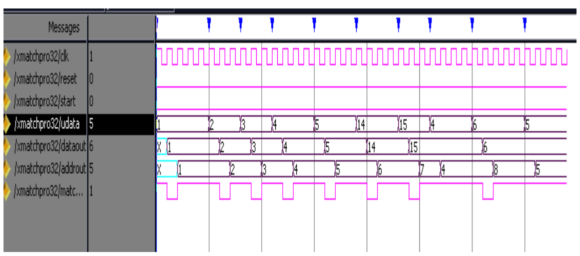

Fig 2. X-MATCHCOMPRESSOR SIMULATION RESULTS.

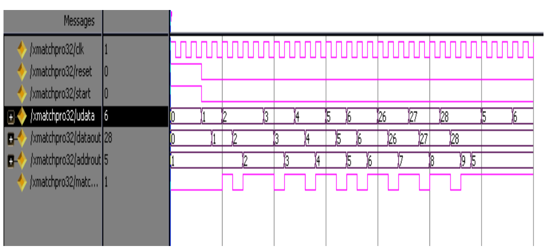

Fig 3. 32-BITCOMPRESSOR / DE-COMPRESSOR SIMULATION RESULTS.

The paper presents a novel VLSI architecture for high-speed data compressor[3] designs which implement the X-Match algorithm. The architecture mainly consists of Five units, namely, FIFO, Matchlogic unit [4], CAM(content addressable memory) comparator [1], X-matchUnit [4], and Output-stage(DE-x-match) unit.

The content-address-memory unit generates a set of hits signals which identify those positions whose symbols in a specified window are the same as the input symbol. These hits signals are then passed to the X-match unit which determines both match length and location to form the kernel of compressed data. These two items are then passed to the output-stage unit for packetisation before being sent out. Logic density increases have made feasible the implementation of multiprocessor systems able to meet the intensive data processing demands of highly concurrent systems.

Compression with previous methods by our method: Fig 2 Shows compressed data simulation results and Fig3 shows top level simulation i.e Compression/Decompression simulation results.by using ModelSim simulator using VHDL[5].

TABLE I.

Experimental Result

	LOGIC BLOCK (PARALLEL)	LOOK UP (PARALLEL)	EQUIVALENT GATE COUNT (PARALLEL)	MY COMPRESSOR (SERIAL)
COMPRESSOR 1	X-Match ProRli	4540	111,000	1888
	Control Logic	346	10,000	No Need
COMPRESSOR 2	X-Match ProRli	4540	111,000	No Need
	Control Logic	346	10,000	No Need
Routing	Input / Output	78	5,000	67
Total		9850 of 38,400	245,000	41204 E-Gates

The above table shows compression details about our compression method. The table itself indicates that our method was good related to the previous methods. The resource allocation for the design is synthesized for SPATRAN 3E FPGA

The logic used by the FIFO depends on the maximum block length selected for the system. The system clock speed of 108.658 MHz and for the compressor system a 4.283ns bit/s throughput is achieved. One of the major benefits of adopting the particular routing strategies used is that they give a scalable solution that maintains the compression performance as the number of compressors in the system is increased. The resource allocation figures demonstrate that with modest FPGA technology multiple compressor architectures with their own dedicated memory and routing mechanisms can easily fit on a single chip.

V. CONCLUSIONS AND FUTURE WORK

The paper has identified a range of techniques for routing data, both to and from parallel compressors each with their own dedicated memory and has shown that important design considerations need to be made that affect both compression and latency. It has also been shown that suitable architectures and routing strategies can be applied to the implementation of scalable high-speed compression systems and that these can be tailored to meet the requirements of different data types. For example, the main priority for backing up data is normally achievable compression rather than latency, while for compression of memory data more emphasis will be laid on the latency due to the time constraints involved.

Further work being considered includes providing the facility to select or dynamically change the routing strategy in the multiple compressor systems depending on data characteristics or system requirements. Similarly, a dynamic system that allocates additional compressors depending on the current throughput is also possible.

REFERENCES

[1] C.Y. Lee and R.Y. Yang, "High-Throughput Data Compressor Designs Using Content Addressable Memory," IEE Proc. Conf. Circuits Devices Systems, vol. 142, pp. 69-73, Feb. 1995

[2]. B.W.Y. Wei, R. Tarver, J.S. Kim, and K. Ng, "A Single Chip Lempel-Ziv Data Compressor," Proc. IEEE Int'l Symp. Circuits and Systems (ISCAS), pp. 1953-1955, May 1993.

[3] S. Henriques and N. Ranganathan, "High Speed VLSI Design for Lempel-Ziv Based Data Compression," IEEE Trans. Circuits and Systems, vol. 40, no. 2, pp. 90-106, Feb. 1993.

[4]. J.L. Nu´n˜ ez, C. Feregrino, S. Jones, and S. Bateman, "X-MatchPRO: A ProASIC-Based 200 Mbytes/s Full-Duplex Lossless Data Compressor," Proc. 11th Int'l Conf. Field-Programmable Logic and Applications, pp. 613-617, Aug. 2001.

[5] A VHDL Primer- 3rd Edition –J.Bhasker

❑❑❑