Zero-delay FPGA-based odd-even sorting network

Conference Paper · April 2013

DOI: 10.1109/ISCI.2013.6612389

CITATIONS
17

READS
2,390

4 authors:

Amirshahram Hematian
Towson University
18 PUBLICATIONS **287** CITATIONS
SEE PROFILE

Suriayati Chuprat
Universiti Teknologi Malaysia
90 PUBLICATIONS **1,037** CITATIONS
SEE PROFILE

Azizah Abdul Manaf
University of Jeddah
164 PUBLICATIONS **3,921** CITATIONS
SEE PROFILE

Nadia Parsazadeh
Tamkang University
9 PUBLICATIONS **291** CITATIONS
SEE PROFILE

Zero-Delay FPGA-Based Odd-Even Sorting Network

Amirshahram Hematian, Suriayati Chuprat, Azizah Abdul Manaf, Nadia Parsazadeh
Advanced Informatics School
Universiti Teknologi Malaysia
hamirshahram2@live.utm.my, suria@ic.utm.my, azizah07@ic.utm.my, pnadia4@live.utm.my

Abstract— **Sorting is one of the most well-known problems in computer science and is frequently used for benchmarking computer systems. It can contribute significantly to the overall execution time of a process in a computer system. Dedicated sorting architectures can be used to accelerate applications and/or to reduce energy consumption. In this paper, we propose an efficient sorting network aiming at accelerating the sorting operation in FPGA-based embedded systems. The proposed sorting network is implemented based on an Optimized Odd-even sorting method (O^2) using fully pipelined combinational logic architecture and ring shape processing. Consequently, O^2 generates the sorted array of numbers in parallel when the input array of numbers is given, without any delay or lag. Unlike conventional sorting networks, O^2 sorting network does not need memory to hold data and information about sorting, and neither need input clock to perform the sorting operations sequentially. We conclude that by using O^2 in FPGA-based image processing, we can optimize the performance of filters such as median filter which demands high performance sorting operations for real-time applications.**

Keywords: Combinational Logic, FPGA, Odd-Even Sort, Parallel Computing, Pipeline Architecture, Real-time, Sorting Networks.

I. INTRODUCTION

In data and signal processing, one of the essential challenges is ordering a list of items in a very short time. Most times, the performance of the sorting algorithm is crucial for the overall system performance, as is the case in database management systems [1]. Efficient data sorting is important for searching and optimization algorithms in high time demanding fields such as hard real-time image processing.

In general, odd-even transposition sort compares the adjacent pair of data in an array to be sorted and, if a pair is found to be in the wrong order then those elements are swapped. In the first step, odd index and the adjacent even index elements are compared and are swapped, if found in wrong order. In the next step, even index and the adjacent odd index elements are compared and are swapped, if found in wrong order. This process continues with alternating (odd, even) and (even, odd) phases, until no swapping of data elements are required. Thus the resultant array is a sorted one. This network comprises of the same number of comparator stages as the number of inputs. In each stage either odd or even index positions are compared with their respective neighbors. Each stage alternates between even and odd.

On the other hand, field-programmable gate array (FPGA) can provide high performance, configurable and complex combinational functions or simple logic gates like AND and XOR that can be used in hamming distance calculations, as an example. FPGAs propound the hardware implementation of functions to deliver dedicated hardware for each part of the process. Hardware dedicated processes can function in parallel or sequential operations, depending on the purpose of use.

We should highlight that we considered gate delays in FPGA are equal to zero. Consequently, in comparison with conventional FPGA-based sorting algorithms, we achieved zero-delay sorting by the proposed design. Several high speed comparators are used in order to operate in parallel without using any input clock. The number of available comparators for sorting is depended on the available logical resources within the FPGA.

In simple words, when a process is implemented on an FPGA based on sequential operations, the speed of the calculations directly depends on the frequency of the input clock, memory bandwidth and other bottlenecks. Accordingly, parallel operations which not necessarily need input clocks and memories can be fully implemented on FPGAs based on combinational logics. O^2 is the name of the proposed Optimized Odd-even sorting network which is fully implemented on FPGA in parallel based on combinational logics that does not need input clock and memory. Odd-even sorting network is shown in Fig.1.

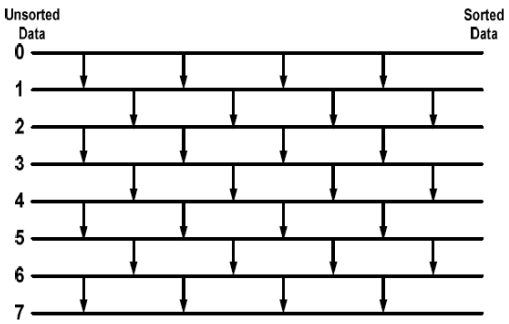

Figure 1. Odd-even sort network diagram, n=8.

978-1-4799-0209-5/13/$31.00 ©2013 IEEE

II. Related Works

Regarding application-specific based architectures, there are two different approaches that have been considered for accelerating the sorting operations, one is focusing on variations of the sorting networks [2-4], and the other exploring systolic linear arrays [5, 6]. Whilst mentioned approaches may attain high performance sorting operations, both depend on the concurrent availability of data to feed the sorting network. For most cases, this ties up their practical use with current technology or at least in systems where data to be sorted are located in the same limited port memory device. Ratnayke and Amer [7] proposed an FPGA implementation variation of the counting sort algorithm [8]. The occurrence histogram is implemented in on-chip FPGA memories (BRAMs). This approach is efficient for relative small datasets. Sorting large data-sets with this approach would require large memories. Note that it is with large data-sets where the execution time for sorting might be more critical. Due to the importance of sorting operations, parallel sorting solutions have also been proposed [9-11]. They make use of parallel sorting algorithms, suitable for multiprocessor implementation, but they still need input clock and memories to hold data and information about sorting.

III. Network Overview

In this section, an Optimized Odd-even sorting network (O^2) based on FPGA is presented and optimized for high performance sorting operation. Benchmark results of sorting on Altera Cyclone-II EP2C20 are also included in this paper. Following is the overview of the proposed sorting network in details as described below:

A. Parallel Processing

O^2 network is designed to sort an array in parallel without using sequential operations for sorting operation. Moreover, O^2 does not use any memory to hold data and information about sorting operation. O^2 is designed based on pipeline processing of array items in order to put items in order by using comparators and multiplexers in parallel. Fig.2 shows the combinational logic block diagram of O^2 network. The number of logic blocks depends on the number of items within the array that needs to be sorted. In worst case, unlike conventional Odd-Even sorting method that needs to do n^2 comparisons to sort an array of n items, O^2 network only needs to do half of the comparisons ($n^2/2$) to sort the given array completely.

The reason that O^2 is able to do this is the ring shape of the sorting operation within the given array. In the conventional Odd-Even sorting, the first and last item will never be compared. In O^2 it is considered that the number of items in the given array is always even. Consequently, the index number of the first item is always an odd number and the last item is always an even number. Thus, the first and last items can be compared when the comparison operation reaches to the end of the given array and it continues with the first and second items until the whole array is sorted. When O^2 sorts the given array, it sorts the items in parallel with ring shape processing. Items of the array can move in two directions within the array in order to get to the right place, rapidly. Accordingly, total number of comparisons between pair of items within the given array is reduced dramatically.

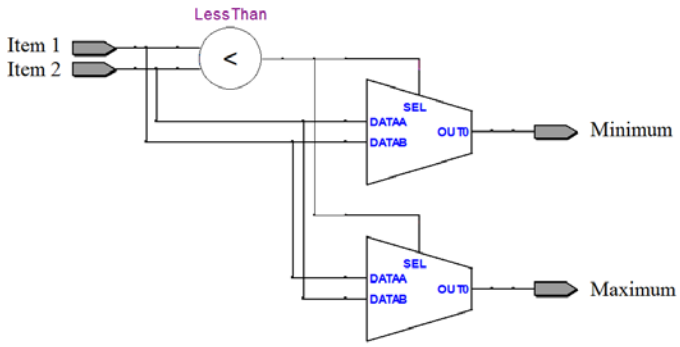

Figure 2. O^2 sorting logic block diagram, using one comparator and two multiplexers per pair of items of an array.

B. Pipeline Network

O^2 network consists many sort logic blocks which are connected in pipeline architecture, as shown in Fig.3. Each pair of items of an array is compared independently in order to generate desired output at each block by swapping items. Output results of each block of the current stage are connected to the related input ports of sort logic blocks in the next stage.

For n number of items of an array as an input to the O^2 network, ($n^2/2$) number of O^2 sorting logic blocks are required in order to sort the input array even in worst case condition. As shown in table 1, O^2 network uses ($n^2/2$) number of comparators to sort n items of an array, unlike conventional odd-even sorting method which uses n^2 comparisons to sort n items of an array [12].

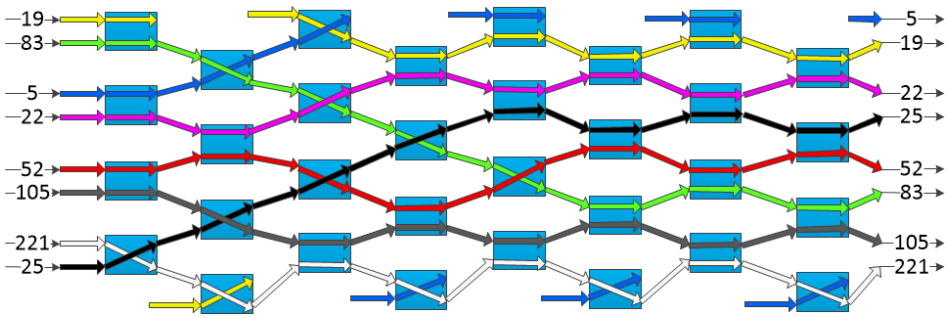

Figure 3. Zero-delay O^2 sorting network diagram, n = 8.

Table 1. Summary of complexities with pipeline stages [12].

Sort Network	*Comparator Stages*	*Comparators*	*LUT*	*Frequency (MHz)*	*Pipeline Stages*
Odd-even transposition sort	O(n)	O(n^2)	632	353	7
Bubble Sort	O(n)	O(n^2)	650	363	12
Bitonic Sort	O($log(n)^2$)	O($n \cdot log(n)^2$)	540	353	5
Odd-Even mergesort	O($log(n)^2$)	O($n \cdot log(n)^2$)	425	368	5
Shell sort	O($log(n)^2$)	O($n \cdot log(n)^2$)	537	360	10
O^2 (proposed)	O(n)	O($n^2/2$)	26	No input clock is required	4

Where n is the number of inputs (n = 8)

IV. EXPERIMENTAL RESULTS

In order to test the proposed sorting network, we used an FPGA development kit for Cyclone-II family (DE-1) as the main part of our test equipment. O^2 network was developed in VHDL and test results are demonstrated in table 1. We achieved to implement optimized odd-even sorting network with complexity O($n^2/2$) on FPGA EP2C20 with 8 parallel inputs (8-bit) and 8 parallel outputs (8-bit) without using any clock and memory.

Based on our experiments with different size of inputs, and comparing our work with existing results of similar works, O^2 sorting network is one of the smallest and fastest sorting networks that have been implemented.

On the other hand, we used O^2 to implement real-time median filter on FPGA, as shown in Fig.4, to find the median value in an array of pixels. We used barrel shifters to provide the parallel input to O^2 network for median filter. This way of implementation of median filter with O^2 network helped us to achieve zero-delay median filter in order to process video frames rapidly, one after another with no delay. Simulation results of O^2 network implemented in VHDL is sown in Fig.5.

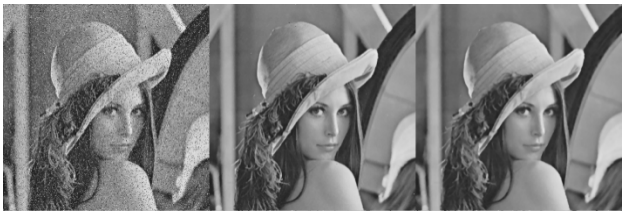

Figure 4. (Left) Noisy image 512x512, (Middle) filtered by median filter 3x3, (Right) filtered by median filter 5x5 using zero-delay O^2 sorting network.

V. CONCLUSION

In this paper, we have proposed a new sorting network (O^2) based on pipeline architecture to be used in low-cost field-programmable gate arrays (FPGA) in order to satisfy all timing considerations by process parallelization. Design and operating principle including experimental results for each section have been demonstrated. It has been proved that using dedicated hardware resources for time consuming operations can end in high performance computing in order to meet timing considerations. Performance comparison of existing methods [12] for sorting network has been conducted on n = 8 number of inputs to O^2 in order to compare with results of previous works, fairly.

Such experiments prove that fully-combinational dedicated hardware architectures can gigantically increase the performance of existing sorting networks. Moreover, Cyclone FPGAs provide a low-cost alternative for parallel computing systems like real-time embedded image processing units that may need to sort millions of arrays in a very short time. Proposed network has been tested on median filter to prove the positive impact of using such sorting network on the overall performance of the system.

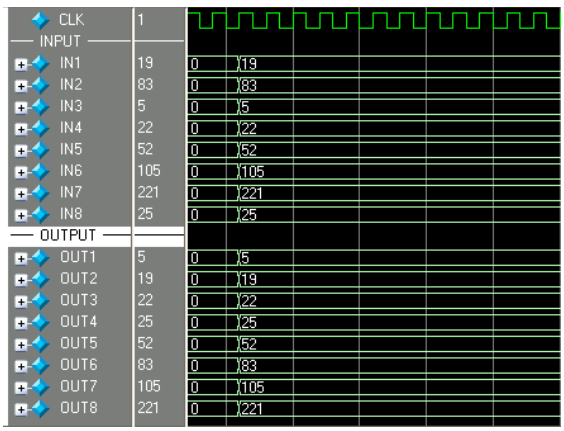

Figure 5. Simulation results of zero-delay O^2 sorting network by Altera ModelSim for n = 8. Inputs are provided in the third rising edge of reference clock and outputs are generated at the same time.

REFERENCES

[1] G. Graefe, "Implementing sorting in database systems," *ACM Comput. Surv.,* vol. 38, p. 10, 2006.

[2] K. E. Batcher, "Sorting networks and their applications," in *Proceedings of the April 30--May 2, 1968, spring joint computer conference*, 1968, pp. 307-314.

[3] J. Martinez, R. Cumplido, and C. Feregrino, "An FPGA-based parallel sorting architecture for the Burrows Wheeler transform," in *Reconfigurable Computing and FPGAs, 2005. ReConFig 2005. International Conference on*, 2005, p. 7 pp.

[4] Y. Zhang and S. Zheng, "An efficient parallel VLSI sorting architecture," *VLSI Design,* vol. 11, pp. 137-147, 2000.

[5] L. Chi-Sheng and L. Bin-Da, "Design of a pipelined and expandable sorting architecture with simple control scheme," in *Circuits and Systems, 2002. ISCAS 2002. IEEE International Symposium on*, 2002, pp. IV-217-IV-220 vol.4.

[6] B. Parhami and K. Ding-Ming, "Data-driven control scheme for linear arrays: application to a stable insertion sorter," *Parallel and Distributed Systems, IEEE Transactions on,* vol. 10, pp. 23-28, 1999.

[7] K. Ratnayake and A. Amer, "An FPGA Architecture of Stable-Sorting on a Large Data Volume : Application to Video Signals," in *Information Sciences and Systems, 2007. CISS '07. 41st Annual Conference on*, 2007, pp. 431-436.

[8] H. H. Seward, "Information sorting in the application of electronic digital computers to business operations," Tech. rep., MIT Digital Computer Laboratory Report R-232, Cambridge, Mass,1954.

[9] M. Edahiro, "Parallelizing fundamental algorithms such as sorting on multi-core processors for EDA acceleration," in *Design Automation Conference, 2009. ASP-DAC 2009. Asia and South Pacific*, 2009, pp. 230-233.

[10] E. Herruzo, G. Ruiz, J. I. Benavides, and O. Plata, "A New Parallel Sorting Algorithm based on Odd-Even Mergesort," presented at the Proceedings of the 15th Euromicro International Conference on Parallel, Distributed and Network-Based Processing, 2007.

[11] H. Inoue, T. Moriyama, H. Komatsu, and T. Nakatani, "AA-Sort: A New Parallel Sorting Algorithm for Multi-Core SIMD Processors," in *Parallel Architecture and Compilation Techniques, 2007. PACT 2007. 16th International Conference on*, 2007, pp. 189-198.

[12] D. Prasad, M. Y. M. Yusof, S. S. Palai, and A. H. Nawi, "Sorting networks on FPGA," presented at the Proceedings of the 10th WSEAS international conference on Telecommunications and informatics and microelectronics, nanoelectronics, optoelectronics, and WSEAS international conference on Signal processing, Canary Islands, Spain, 2011.