Low Power Multidimensional Sorters using Clock Gating and Index Sorting

Samahith S A
Department of Electronics and Communication Engineering, PES University
Bangalore, India
sa.samahith@gmail.com

Sai Govardhan
Department of Electronics and Communication Engineering, PES University
Bangalore, India
saigov14@gmail.com

Manogna R
Department of Electronics and Communication Engineering, PES University
Bangalore, India
treja.manu@gmail.com

Hitesh D
Department of Electronics and Communication Engineering, PES University
Bangalore, India
hiteshramaswamy@gmail.com

Sudeendra Kumar K
Department of Electronics and Communication Engineering, PES University
Bangalore, India
kumar.sudeendra@gmail.com

2023 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT) | 979-8-3503-3439-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/CONECCT57959.2023.10234758

***Abstract*- Sorting is an essential function widely used in several applications like data centers, and is implemented either on CPU or GPU, which takes several cycles to finish. Improvement in the performance of sorting is achievable through hardware acceleration either in FPGA or ASIC. The performance improvement and reduction in power consumption are the goals of researchers implementing sorting algorithms on hardware. The Multi-Dimensional Sorting Algorithm (MDSA) improves the basic Bitonic sorting implementation to find the largest and smallest records at any time throughout the sorting process. In this work, we extend the MDSA for Hybrid and Odd-Even sorters while we also optimize the MDSA sorter architectures for low power using clock gating and index sorting techniques. We present the different MDSA sorter topologies and their low power versions with comparative results. The saed90nm standard cell library was used for implementation and analysis. The bitonic sorter architecture with index sorting and clock gating techniques will reduce the dynamic power by 68.56% in comparison with the conventional MDSA bitonic sorter.**

***Keywords:* Parallel Hardware Sorting Algorithm (PHSA), Multi-Dimensional Sorting Algorithm (MDSA), Clock Gating, Index Sorting**

I. INTRODUCTION

Data Centres are a core infrastructure utility for most enterprises, performing computations, data storage, network, and business applications. Their energy consumption depends mainly on their workloads, hardware specifications, applications, and cooling requirements which are all closely coupled based on the design of the data centre and the efficiency of the equipment [1]. Efficient and low power sorting of data sorting is essential in high-performance computing systems such as data centres [2].

Apart from the present software implementations on CPUs and GPUs, Hardware implementations on FPGAs provide an easier implementation of massively parallel architectures. Along with the aid of HLS tools in recent times, enabling easy modeling of high-level synthesis [2] reduces the time-to-market of FPGA implementations. Dedicated FPGA or ASIC-based hardware accelerators using techniques such as iterative networks, reusable comparators, and partial sorting can help achieve high throughput, but the limit on the number of resources available for hardware implementation of sorting is a serious limitation and must be utilized optimally. Current hardware implementations are several hundreds of times faster than running sorting on a general-purpose CPU [3] in terms of execution time and much more efficient in terms of power consumption [3]. Hence the development of fast, low-power and resource-efficient hardware sorters is an active area of research.

Although it has been shown that these GPU-based sorting algorithms are 1.5 to 6.2 times faster than running them on a CPU [3], the major concerns regarding software parallel sorting are the possibility of deadlocks due to limited buffer space in the data-exchange mechanism [4]. Hardware-based sorters are usually obtained by mapping software-based algorithms onto hardware, several algorithms have been mapped with varying degrees of success. The most prominent parallel hardware sorting architectures are obtained from the mapping of parallel merge sorting algorithms, such as odd-even merge sort and bitonic merge sort onto hardware [5]. In this work, we primarily focus on low power implementation of different sorter architectures. FPGA based systems for sorting are attractive and has got several advantages, easy implementation of parallelism for sorting etc, but FPGA based systems have limitations towards designing low power systems because they are pre-fabricated devices and hence the structured low power techniques like power gating and Dynamic Voltage Frequency Scaling(DVFS) are difficult to implement [6]. Hence in this work we focus on ASIC implementation as we have more freedom in exploiting the low power techniques in sorter implementations.

Multi-dimensional sorting algorithm (MDSA) presented in [7], has been implemented for the bitonic sorter to support the requirement and present the results and analysis for FPGA implementation. In this work, we propose a low power sorting techniques using clock gating [8] and index sorting [9] for different sorter topologies such as bitonic , odd-even [8] and hybrid [10] by incorporating all MDSA features mentioned in [7]. The comparative analysis for dynamic power consumption across twelve different sorter topologies has been presented in this work.

This paper is structured as follows: next section presents the literature survey on existing different sorting techniques and short description on taxonomy of sorting. It also includes the brief literature review on low power implementation of sorting techniques. Section III presents the proposed adoptions of different sorters for to achieve low power MDSA. Section IV discusses the implementation and results. Finally, Section V concludes the paper.

979-8-3503-3439-5/23/$31.00 ©2023 IEEE

TABLE I. COMPARISON OF HARDWARE SORTER IMPLEMENTATIONS

Paper	Architecture	Advantages	Limitations	Conclusions
RTHS: A Low-Cost High-Performance Real-Time Hardware Sorter, Using a Multidimensional Sorting Algorithm, 2019. [7]	The authors in [24] have implemented two-dimensional sorting algorithm-MDSA using a control FSM, a group of sorting units and a transpose network, designing a matrix based PHSA.	RTHS achieves a good trade-off between delay and area, through the iterative architecture.	RTHS has only been implemented with the bitonic sorter as the basic sorting unit.	Different sorting units can be explored to achieve power / area optimized sorters.
Hardware Design of Low-Power High-Throughput Sorting Unit, 2017. [14]	The authors in [23] have implemented a low power sorter using a pointer-based design in the compare and exchange units. This technique results in the reduction of dynamic power due to reduction in the data width.	The proposed design was able to reduce the power consumption by ½ to 1/3 times the conventional methods.	This low power design is implemented only for partial sorting.	The low power design can be extended from partial sorting unit to a complete sorting unit and could also function as a low power alternative to bitonic sorter in iterative architectures.
Low Power Sorters Using Clock Gating, 2021. [8]	The authors in [27] have explored the effects of structured low power technique, specifically clock gating on basic sorting units. The Clock gating technique allows the sequential elements in any design to be triggered only when required and not at the usual periodic intervals.	Clock gating resulted in dynamic power reduction of the sorting units by 50% on average.	The clock gating technique was implemented only on basic PHSAs: bitonic, hybrid and odd-even.	Clock gating technique can be used for dynamic power reduction of more complex low power sorters as well as iterative sorters.

II. BACKGROUND

A. Taxonomy of Hardware Sorters:

Hardware sorters can broadly be classified into comparison-based and comparison-free sorters. Comparison-free sorters are built using various architectures with the intention of avoiding the dynamic power consumption caused due to swap operations in comparison-based sorters [11]. There are various architectures for comparison-free sorting architectures based on different techniques such as parallel radix sorting [12] or bit manipulation [13].

Comparison-based sorters are built using a compare and swap (or exchange) block often referred to as CAS or CAE block. These CAE blocks are used to build parallel hardware sorting architectures. The most prominent among these are odd-even merge sort and bitonic merge sort [8]. Most comparison-based hardware sorters are built using the merge sorters.

Further at the architectural level hardware sorters can be classified into partial sorters, PHSA (Parallel Hardware Sorting Architectures), and iterative sorters. Partial sorters as the name suggests are used when we require only either a set of maximum or minimum values instead of a complete set of sorted numbers [7]. PHSAs are sorting networks that receive a set of numbers and based on the architecture give the complete sorted output after a certain number of clock cycles, this architecture is used when throughput is given more importance than the area. Iterative sorters use basic merge sorters in an area and energy-efficient manner to achieve a good trade-off between throughput, area, and power consumption [15].

B. Low power implementation of Sorters: -

Table I presents the following significant low power implementations in the recent past: -

By applying the pointer like technique in which the comparing the modules which move the indexes of samples instead of moving the data is discussed in [14]. This achieves the significant reduction in dynamic power consumption.

Conventional sorters architectures like Bitonic, Odd-Even and Hybrid sort are modified to incorporate clock gating technique to achieve reduction in dynamic power consumption [8].

It is evident from the literature survey that, there is a need for implementation of low power sorting algorithms which support Max/Min queues and finding largest and smallest entry in the data during sorting. The work presented in [7] describes the MDSA which support the Max/Min Queues and finding largest and smallest entry in the data during sorting. There is a need to re-design the MDSA to achieve low power consumption.

The major contributions of this paper are as following: -

- Our work focuses on implementing efficient PHSA by realizing multiple sorting algorithms of different architectures [7] for the Multi Dimension Sorting Algorithm (MDSA) implementation, the RTHS [7] structure, while implementing low power techniques (clock gating [7], index sorting [14]).
- The authors of [15] discuss the effects of pipelining on various hardware sorters and propose a hybrid sorter which reduces the number of compare and exchange (CAE) blocks significantly relatively to bitonic sorting network. Authors in [15] also prove that a reduction in the critical path of pipelined sorters over non-pipelined versions of basic sorting networks (Bitonic and Odd-Even). In this work, we adopt the technique which is used in hybrid sorter presented in [10] to reduce the number of CAE in the sorting architectures implemented in this work.
- The MDSA technique presented in [7] is verified with bitonic sorter only and in this paper, we design the low power MDSA with the following topologies and discuss the results and analysis: -
  1. MDSA Bitonic sorter with low power techniques: index sorting and clock gating,
  2. MDSA Hybrid sorter with low power techniques: index sorting and clock gating,
  3. MDSA Odd-even sorter with low power techniques: index sorting and clock gating.

III. PROPOSED HARWARE ARCHITECTURE

A. *Compare and Exchange Units*

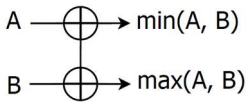

Fig. 1. Figure depicts the 2 input CAE Block.

The compare and exchange (CAE) units presented in figure 1 are primary building blocks of hardware sorters. The two input values are compared using a comparator and the result is applied to the multiplexers, which select the values to generate the output in an ascending order or descending order. The CAE structures are used to implement the different types of hardware sorters like, bitonic, odd-even and hybrid sorters to handle large amount of data. CAE structures provide the required modularity in designing the hardware sorters, which is necessary in handling large data widths.

B. *MDSA Algorithm*

The Multi-Dimensional Sorting Algorithm (MDSA) technique proposed in [7] is applied to different sorter topologies: Odd-Even sorter and Hybrid sorter. The proposed MDSA odd-even sorter and MDSA hybrid sorter is further modified to achieve low power using index sorting and clock gating.

MDSA Algorithm: - MDSA is used to sort data represented in the form of an n-dimensional matrix. The higher the dimension, the smaller is the matrix. For a two-dimensional $P \times P$ matrix where $P = \sqrt{N}$ and for a three-dimensional $P \times P \times P$ matrix where $P = \sqrt[3]{N}$ where N is the number of records. MDSA consists of alternating row and column sorting with varying directions in each phase. The choice of dimension of matrix for hardware implementation depends on the trade-off between resource consumption in terms of area and delay.

TABLE II. PHASES OF THE TWO DIMENSIONAL MDSA SORTING ALGORITHM

Phase	Row/ Column Sorting	Ascending sorting	Descending sorting
1	Column	1,2,3,4,5,6,7,8	-
2	Row	1,3,5,7	2,4,6,8
3	Column	1,2,3,4,5,6,7,8	-
4	Row	2,4,6,8	1,3,5,7
5	Column	1,2,3,4,5,6,7,8	-
6	Row	1,2,3,4,5,6,7,8	-

For our implementation, we have used a two-dimensional sorting algorithm for 64 records, hence six phases Table II shows the phases, the type of sorting that occurs at each phase and which rows or columns of numbers are sorted in which direction. At the end of phase six, we will have 64 sorted records. The three widely discussed 8-input hardware sorter configurations are mentioned below:

1) *Bitonic sort:*

One of the most used sorting networks is the Bitonic sorting circuit as proposed in [16] which is a merge sort based PHSA and is shown in figure 2 using CAE units to form a network.

To achieve a 2^n input bitonic sorter, we need 2^{n-1} elements in n levels each adding up to $n \times 2^{n-1}$ number of elements.

The principle of bitonic sorter is to take a bitonic sequence and rearrange them in monotonic order. Therefore, to perform sorting in an 8-input Bitonic Sorting network, we have 3 units labeled as the BM-2, BM-4, and BM-8 units.

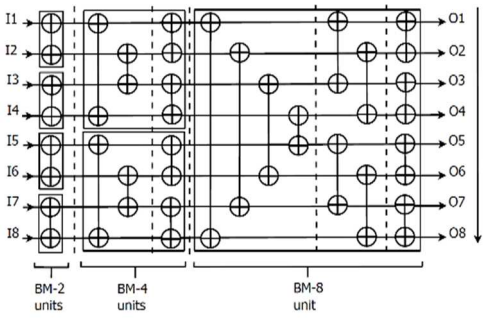

Fig. 2. Figure depicts the 8-input Bitonic Sorting Circuit used in the MDSA implementation. The vertical dotted lines represent the pipeline stages

2) *Hybrid sort:*

The Hybrid sorting network as proposed in [10] is a PHSA implementation with one lower comparator relative to bitonic sorter (23) and two more pipeline stages (8) as shown in figure 3. This hybrid sorting architecture proposes lesser dynamic power consumption considering lesser switching activity in the network due to lesser comparators.

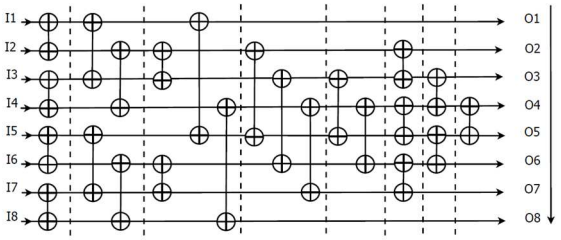

Fig. 3. Figure depicts the 8 input Hybrid Sorting Circuit used in the MDSA implementation. The vertical dotted lines represent the pipeline stages.

3) *Odd-even sort:*

The Odd Even sorting network as proposed in [16] is a PHSA that offers pipelining and is shown in figure 4. The sorter consists of CAE blocks forming merge units namely OE-2s, OE-4s, OE-8s up to OE-n (where n is in the powers of 2). The input is given in the form of odd and even sequences each $(2^m)/2$ (where 2^m is the number of inputs). For a given eight input Odd Even sorting network with six pipeline stages consists of 19 comparators, 5 less than an eight input bitonic sorting network. Hence, the Odd Even merge sorter is proposed to occupy the least area and consume lesser dynamic power.

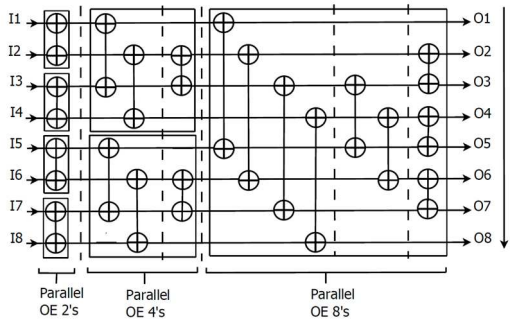

Fig. 4. Figure depicts the 8 input Odd Even Sorting Circuit used in the MDSA implementation. The vertical dotted lines represent the pipeline stages.

C. MDSA Sorter

The input data initially passes through a multiplexor, which chooses the data to be loaded into the register file. Whenever the start pulse is triggered, new data is loaded into the register file. In other cases, the feedback data is loaded into the register file. Register file can hold up to 64 - 32 bit numbers. The output of the multiplexor is sampled by the register file when there is a state transition. The data from the register file is split into 8 parts and fed into separate sorting units. Sorting units consists of 8 input pipelined sorters. The DELAY of the sorting unit is determined by the number of pipeline stages. The state transitions in the FSM occur after completing the set number of clock cycles (DELAY). There are eight sorters working in parallel, sorting eight sets of eight numbers at a time as shown in figure 5. After the completion of one sorting phase, the output of the sorting unit is fed to the implicit switch which exchanges the data between the rows and columns. The output of the implicit switch is fed back to the multiplexor, and this process repeats for six phases.

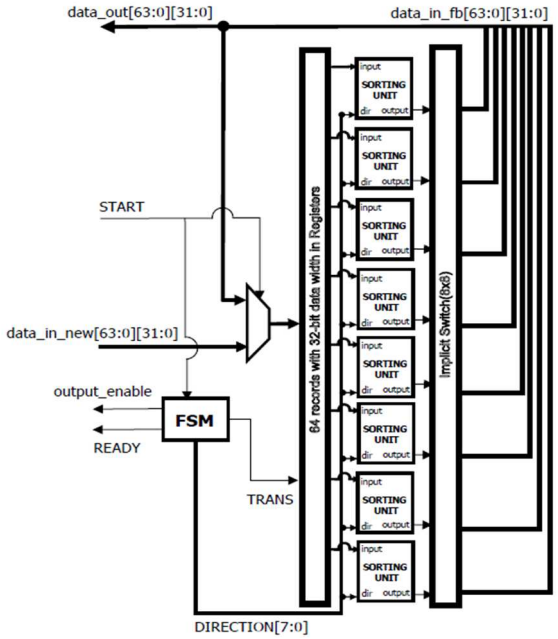

Fig. 5. This architecture represents the MDSA sorter for sorting 64 records of 32 bit width each.

D. FSM

The control unit of the MDSA implementation is an FSM that coordinates all the elements of the architecture in (figure 6) to deliver a sorted output at the end of six phases. The FSM begins or is initialized to the 'WAIT' state when the reset 'rst' signal is triggered. The 'READY' signal indicates that the sorter is ready to receive the next input records and the signal remains high until the 'START' pulse is triggered.

When the 'START' pulse is triggered, and when the enable 'en' signal is high and reset 'rst' signal is low, the FSM enters phases 1 to 6 where the sorting operation is performed. Each phase is indicated by a transition signal 'transes encountered after counting up to the DELAY parameter, which is the number of pipeline stages in the sorting network.

The FSM provides the 'DIRECTION' signal which is a sequence of 0s and 1s to indicate the ascending and descending sorting modes in each sorting network during the alternate column and row sorting phases as mentioned in Table II. At the end of the six phases of sorting, the FSM produces an output enable signal which indicates that the sorted sequence of 64 records can be sampled at the positive edge of the 'output_enable' pulse. The FSM finally activates the 'READY' signal and enters the 'WAIT' state until the next 'START' pulse initiates sorting for the next input sequence.

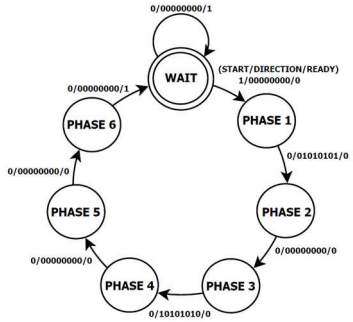

Fig. 6. FSM of the MDSA implementation

E. Index Sorting (Low Power Feature)

Index sorting is a low-power feature proposed by [9] which reduces dynamic power consumption in each PHSA. Since dynamic power consumption in a PHSA increases with increase in the bit-width of the inputs, a new CAE block is proposed where the records are used only to compare, and the indices of the records are used to recover the corresponding records as a sorted output as shown in figure 7. Since the indices (of size $log_2 N$ bits where N is the number of inputs) are swapped instead of the entire number in each CAE unit

The CAE block of the 8-input index sorter (CAE index) uses only 3-bit indices for exchange operation in the multiplexors, instead of swapping 32-bit inputs.

For the MDSA bitonic index implementation, as presented in figure 7, each Bitonic Merge (BM) stage is followed by a recovery unit consisting of multiplexors to swap the numbers based on the indexes obtained after comparing.

Similarly, the MDSA hybrid index and MDSA Odd-Even index implementations consist of recovery stages at the end of each pipeline stage.

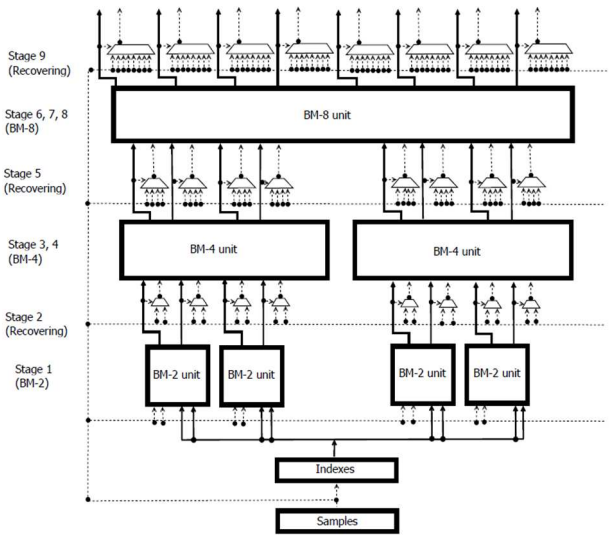

Fig. 7. Index sorting implemented on the Bitonic Sorter.

TABLE III. DYNAMIC AND LEAKAGE POWER OF DIFFERENT SORTER ARCHITECTURES.

Design Name (Data Size: 64, Bit Width: -32 bit inputs)	Low Power Feature	Leakage Power in mW	Dynamic Power in mW	Total Power in mW	Dynamic Power Reduction compared to its basic MDSA sorter
MDSA bitonic	None	4.165638746	40.39866955	44.5643487	-
	Index Sorting	4.033174504	29.12449432	32.95448313	27.90%
	Clock Gating	3.337094761	17.88552704	21.22263969	52.38%
	Index Sorting + Clock Gating	3.710979619	12.699277	16.41026932	68.56%
MDSA hybrid	None	3.903486577	36.2943689	40.19789177	-
	Index Sorting	3.955966276	29.63217326	33.58816917	18.356%
	Clock Gating	3.391287001	19.47521707	22.86652355	46.341%
	Index Sorting + Clock Gating	3.824417711	17.48704878	21.31148398	51.819%
MDSA odd-even	None	2.856259223	25.0794052	27.93568951	-
	Index Sorting	3.329356135	23.10669736	26.4360766	7.866%
	Clock Gating	2.769526724	15.25794563	18.02748761	39.161%
	Index Sorting + Clock Gating	3.290312738	13.59778072	16.88810705	45.781%

IV. IMPLEMENTATION AND RESULTS

A. Implementation

The sorters designs are implemented in Verilog. The MDSA algorithm takes 64 inputs (each 32-bit) and output the sorted sequence of in the state "PHASE-6" of the FSM. gating is also implemented on the index sorting variants and non-index variants of sorters with MDSA algorithms. These sorters differ in the number of pipeline stages and hence delay for each phase in the FSM varies among the sorters. Implementation comprises of various sorting blocks, namely hybrid, OE, bitonic and their index sorting variants. Clock Different sorting topologies are simulated using Cadence NCSim tools and successfully verified for functionality. Cadence Genus synthesis tool was used to synthesize all the sorter topologies with saed90nm library.

The improvements to sorter topologies proposed in this work focuses on dynamic power consumption. This work proposes different topologies of sorters for low power consumption by incorporating clock gating and index sorting techniques in MDSA implementations. Fundamentally, the dynamic power consumption of the CMOS circuits directly depends upon the switching activity of the circuit, which directly depends upon the frequency of operation and activity on the inputs of the circuit. To conduct the accurate dynamic power analysis of the sorters, some standard input patterns or benchmarks may be necessary. The existing sorter benchmarks [17] are useful in assessment of computational complexity and other parameters for software implementation of sorters. The standard sorter benchmarks for hardware implementation of sorters are generally not available and, in this work, we devised a stimulus to conduct dynamic power analysis. The power analysis is conducted by simulating the design using a devised stimulus and VCD file (Value Change Dump) is created. The VCD file is fed into the power analysis tools to calculate the accurate dynamic power. In this work, we make use of the same design constraints for synthesis and same stimulus for power analysis of all proposed low power sorter topologies.

The devised stimulus (testbench) is constructed use the randomization feature available in the system Verilog. Since the different sorter topologies have different architectures and affect the switching activity, the randomization is essential for dynamic power analysis. The randomization runs for 100 iterations. This seed value for randomization is chosen to achieve a balanced trade-off between the amount of switching activity and simulation time. The layered testbench was run on the gate level netlist synthesized using the saed90nm library and constrained for a frequency of 50MHz. The testbench generates a VCD file for each of the simulation runs.

B. Results and Analysis

The area, power and timing results are presented for all proposed sorter topologies and compared with the existing sorter topologies and analysis is presented.

1) Power

Dynamic Power Analysis: - The Table III presents the detailed power analysis of all proposed sorters and values are compared with the MDSA based bitonic sorter presented in [7]. The above power reports prove that the designs with sorting blocks having lesser number of comparators consume lesser power (dynamic and total power), when the designs are unmodified. The low power features implemented, Index sorting and Clock gating show that for a given sorter, the dynamic power is incrementally lower as desired, with the least power consumed when both the low power features were combined (Index sorting + Clock Gating). The basic MDSA implementation with bitonic sorting blocks, when modified with index and clock gating consumed 68.56% lesser power. A similar trend was followed for MDSA hybrid dynamic power when modified with both low power features. It can be observed that the MDSA implementation on sorters with a greater number of CAE blocks have higher reduction in dynamic power, for the same low power feature such as index sorting or clock gating. It is evident from the data that basic Bitonic implementation of MDSA has the highest dynamic power consumption, while the Bitonic implementation with all the low power techniques implemented has the least dynamic power. This is due to Bitonic sorters having a uniform structure and hence the index implementations are done at the BM level. We can also observe that the hybrid sorter and odd-even with 51.81% and 45.78% reduction in dynamic power. Hybrid despite having one less CAE block, is worse than other sorters with the low power techniques implemented, this is due to hybrid sorter having an extra stage of pipeline necessitating extra clock cycles. The odd-even implementation has the least power consumption on almost all the techniques owing to least number of CAE blocks and same number of pipeline stages as Bitonic sorter, it is only marginally worse than the Bitonic sorter with all the low power techniques implemented owing to Bitonic sorter's uniform structure. We can also observe from the Table III that, clock gating technique deliver the maximum reduction of dynamic power in comparison with index sorting. The clock gating performs the structural

changes to the design, that is, replacing the flipflops with gating enabled flipflops and corresponding gating control circuit. The index sorting implementation need significant architectural modifications to basic sorter topology.

2) Timing

The Table IV presents the detailed timing analysis of all proposed sorters and values are compared with the MDSA based bitonic sorter presented in [7]. The MDSA implementations with bitonic, hybrid and odd-even sorters show reduction in slack with low power modifications, which can be attributed to the increased combinational logic being included in the designs due to index sorting and with clock gating. From the Table IV, it is evident that, clock gated architectures has got maximum slack when compared to index sorting architecture and the difference in slack is not very significant. All sorter topologies have the best slack when none of the optimizations are applied.

TABLE IV SLACK OF DIFFERENT SORTER ARCHITECTURES.

Design Name (64 - 32 bit inputs)	Low Power Feature	Slack (in nS)
MDSA_bitonic	None	15820
	Index Sorting	14308
	Clock Gating	9460
	Index Sorting + Clock Gating	9384
MDSA_hybrid	None	15281
	Index Sorting	12844
	Clock Gating	9460
	Index Sorting + Clock Gating	9423
MDSA_odd_even	None	15441
	Index Sorting	12344
	Clock Gating	9460
	Index Sorting + Clock Gating	9392

3) Area

The area results for all sorter topologies are presented in Table V. It is evident from the results that, when no low power feature is implemented, our MDSA implementations of sorting units having lesser number of comparators (in Table V) have lesser cell count and area, hence, the area and cell counts are in order of MDSA bitonic having the highest, followed by MDSA hybrid and further the MDSA odd-even having the least area. The Index sorting feature increases the number of cells in the design due to the increased combinational logic in the recovery stages of index sorting, whereas the smaller comparators in the CAE index block consume lesser area than the regular CAE block. The clock gating feature reduces the number of cells due to the inclusion of the integrated clock gating cell reducing the combinational logic in the design.

TABLE V. AREA RESULTS OF DIFFERENT SORTER ARCHITECTURES FOR SAED90NM LIBRARY

Design Name (64 – 32 bit inputs)	Low Power Feature	Cell Count	Cell Area (in μm^2)
MDSA_bitonic	None	54201	909054.271
	Index Sorting	60891	858519.252
	Clock Gating	39490	731063.796
	Index Sorting + Clock Gating	58593	808250.571
MDSA_hybrid	None	52874	868658.7
	Index Sorting	72833	871090.791
	Clock Gating	40008	725334.209
	Index Sorting + Clock Gating	72853	843230.823
MDSA_odd_even	None	36547	651300.259
	Index Sorting	60049	726624.461
	Clock Gating	32895	603413.904
	Index Sorting + Clock Gating	58466	700651.93

V. CONCLUSION

In this work, we design and analyze the bitonic, odd-even and hybrid sorters with multi-dimensional sorting algorithm for low power consumption. It is evident from literature survey that, there is a need for low power sorters in data centre applications and researchers both in industry and academia are building novel low power and high performance versions of sorters. In total, we present the power consumption, timing, and area results for twelve different implementations of MDSA sorter. It can be concluded from the results that, the low power version of Bitonic sorter (with index sorting and clock gating) will have a 68.56% less dynamic power consumption in comparison with conventional MDSA sorter. Low Power MDSA Bitonic topology consumes less dynamic power in comparison with all other topologies.

REFERENCES

[1] M. Dayarathna, Y. Wen and R. Fan, "Data Center Energy Consumption Modeling: A Survey," in IEEE Communications Surveys & Tutorials, vol. 18, no. 1, pp. 732-794, First quarter 2016.

[2] Valery Sklyarov, Iouliia Skliarova, “High-performance implementation of regular and easily scalable sorting networks on an FPGA”, Elsevier Journal of Microprocessors and Microsystems Volume 38, Issue 5, Pages 470-484, 2014.

[3] B. Falsafi, et al, "FPGAs versus GPUs in Data centers," in IEEE Micro, vol. 37, no. 1, pp. 60-72, Jan.-Feb. 2017.

[4] H. Najafi, et al, "Low-Cost Sorting Network Circuits Using Unary Processing," in IEEE Transactions on Very Large-Scale Integration (VLSI) Systems, vol. 26, no. 8, pp. 1471-1480, Aug. 2018.

[5] Nozar Tabrizia, et al, “An ASIC design and formal analysis of a novel pipelined and parallel sorting accelerator”, Elsevier VLSI Integration Journal, Volume 41, Issue 1, Pages 65- 75, January 2008.

[6] John M. Rabaey, “Low Power Design Essentials”, Springer, 2009. ISBN: 978-0-387-71712-8

[7] A. Norollah, et al, "RTHS: A Low-Cost High-Performance Real-Time Hardware Sorter, Using a Multidimensional Sorting Algorithm," in IEEE Transactions on Very Large-Scale Integration (VLSI) Systems, vol. 27, no. 7, pp. 1601-1613, July 2019.

[8] P. Preethi, et al, “Low Power Sorters Using Clock Gating," 2021 IEEE International Symposium on Smart Electronic Systems (iSES), Jaipur, India, 2021, pp. 6-11.

[9] M. Abdelrasoul, et al, "Index and Sort Algorithm Based on FPGA for Sorting Data," 2021 9th International Japan-Africa Conference on Electronics, Communications, and Computations (JAC-ECC), Alexandria, Egypt, 2021, pp. 61-64.

[10] V. S. Harshini et al, "Design of Hybrid Sorting Unit," 2019 International Conference on Smart Structures and Systems (ICSSS), Chennai, India, 2019, pp. 1-6

[11] S. Ghosh, S. Dasgupta and S. Saha Ray, "A Comparison-free Hardware Sorting Engine," 2019 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), Miami, FL, USA, 2019, pp. 586-591

[12] B. Romanous et al., "High-Performance Parallel Radix Sort on FPGA," 2020 IEEE 28th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), Fayetteville, AR, USA, 2020, pp. 224-224.

[13] Y. Du and S. Li, "A Binary Counter Based on Stacking and Sorting," 2020 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 2020.

[14] S. -H. Lin, et al, “Hardware Design of Low-Power High-Throughput Sorting Unit," in IEEE Transactions on Computers, vol. 66, no. 8, pp. 1383-1395, 1 Aug. 2017.

[15] J.Zhou, et al, “Pipelined Implementation of serial comparison based iterative sort on FPGA”, 2nd International Conference on Artificial Intelligence and Advanced Manufacture October 2020.

[16] K.E. Batcher, “Sorting Networks and Their Applications,” Proc. AFIPS Proc. Spring Joint Computer Conf., pp. 307-314, 1968.

[17] http://sortbenchmark.org