A Novel SRL Based Sorter Micro Architecture

Alexei Lomakin, Vitaliy Gleyzer, William S. Song, Paul Monticciolo
MIT Lincoln Laboratory
Lexington, MA USA
alexei.lomakin@ll.mit.edu, vgleyzer@ll.mit.edu, song@ll.mit.edu, paul.monticciolo@ll.mit.edu

***Abstract*— Sorting data is a key computational driver in many data processing applications in domains such as bioinformatics, sociology and graph analytics. In this paper, we describe a novel SRL-based compare-and-select (CAS) cell micro-architecture for building a high-performance, area efficient FPGA sorters. A full 32-way merge tree sorter implemented in a Kintex UltraScale FPGA, utilizing this CAS cell, can achieve up to 250MHz operational frequency while sorting 128-bit keys. As compared to other evaluated architectures, the proposed design provides the best operational frequency vs. resource utilization trade-off for a range of analyzed key width.**

I. Introduction

Sorting large datasets is a fundamental operation that enables many data analysis and database applications across domains such as bioinformatics, sociology and graph analytics. As Field Programmable Gate Arrays (FPGAs) are being incorporated into cloud-based data processing platforms, such as Amazon Web Services, utilizing these devices to accelerate the processing of terabytes to petabytes of data can potentially provide significant power and performance improvements as compared to software-only approaches.

Many FPGA-based sorter designs have been proposed in recent years. In this paper, our focus is not on the overall sorter architecture, which is usually optimized for specific system or application, but a fundamental sorter building block: the compare-and-select (CAS) cell. Its main function is to efficiently merge elements from two sorted data streams into one.

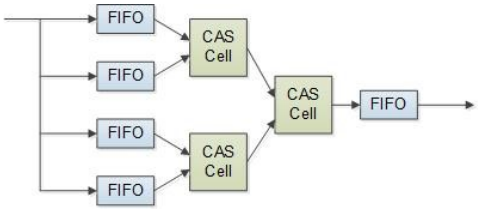

Figure 1. A 4-way conventional merge sort tree architecture

In this paper, we introduce an optimized 2-to-1 CAS cell micro-architecture that utilizes the Xilinx-native SRL primitive. We achieve a significant performance and area improvement as compared to alternative designs for a wide range of data widths, including 32, 64 and 128 bits. Even though we have evaluated the cell in the context of the merge tree sorter, illustrated in Figure 1, the cell design concepts can easily be adapted to support most other architectures.

DISTRIBUTION STATEMENT A. Approved for public release. Distribution is unlimited.

This material is based upon work supported by the Assistant Secretary of Defense for Research and Engineering under Air Force Contract No. FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Assistant Secretary of Defense for Research and Engineering.

II. CAS Cell Micro-Architecture

The CAS block is one of the performance-critical components of the sorter design. In order to output an element every clock cycle, the CAS must be able to select the key and read it from one of the two storage elements within one clock cycle. Thus, only one pipeline stage can be added to the selection logic without significantly increasing the design size. Furthermore, supporting wider keys, such as 128 bits, the comparator carry chain has a significant effect on the operating frequency of the design. We have considered several CAS micro architectures utilizing different Xilinx primitives for comparing a range of key widths. Figure 2 shows some of the evaluated CAS designs.

The two main design parameters are (1) the choice of the primitive to utilize as the cell input storage element, and (2) the location of the pipeline registers.

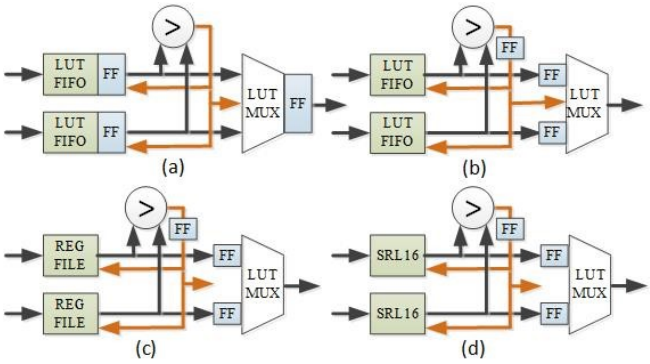

Figure 2. Compare-and-select (CAS) cell micro-architecture (a) BRAM FIFO CAS cell, (b) LUT FIFO CAS cell, (c) REG FIFO CAS cell, (d) SRL FIFO CAS cell.

The first evaluated micro-architecture employs BlockRAM, which is applicable to sorter architectures that require large storage elements in each CAS cell, such as found in [2]. All other CAS cells were implemented with much smaller input storage elements utilizing LUTs, registers and SRLs. After preliminary evaluation of several pipeline configurations, we selected the most promising options. For configurations (c,d), the selected pipeline configuration includes a single bit flip flop after the comparator in addition to a pipeline stage before the output multiplexor. Because of the one clock cycle delay in the BlockRAM read path, the CAS cell (a) utilizes a different pipeline strategy.

III. Evaluation Results

In order to evaluate the candidate designs, we gathered timing and resource information for individual cell

978-1-5386-5989-2/18/$31.00 ©2018 IEEE

performance. Furthermore, to provide representative timing numbers in a fully implemented sorter, we instantiated each cell in a complete 32-way merge sorter tree, similar to the architecture illustrated in Figure 1.

The results for our evaluation are captured in Table 1 and Figure 3. To facilitate comparison to other recently published results, these designs were instantiated on, both, a Kintex UltraScale XCKU115-2 and a Virtex-7 XC7VX690T-2. The operational frequency for the Virtex-7 device was approximately 20% lower than the reported UltraScale performance, and have been excluded for brevity.

TABLE I. SINGLE 128-BIT CAS CELL RESOURCE UTILIZATION ON XCKU115-2

CAS cell	BRAM	FLOP	LUT	DMEM
BRAM FIFO IP (a)	4	407	266	0
LUT FIFO IP (b)	0	935	206	176
REG FIFO (c)	0	1341	450	0
SRL FIFO (d)	0	324	253	136

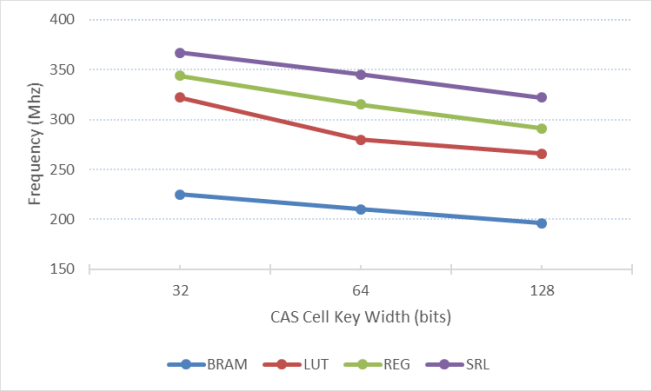

Figure 3. Clock frequency of single CAS cells on XCKU115-2

As seen from the results, the sorter cell illustrated in Figure 2(d), which utilizes the native SRL primitive as the storage element and has single pipeline stage directly after the comparator, delivered the highest operating frequency while requiring the least amount of FPGA resources.

IV. RELATED WORK

In order to get a sense of how the proposed SRL sorter cell may impact other sorters, we surveyed a list of recently published sorter architectures. The results of the survey are summarized in Figure 3.

Based on the reported numbers, we believe utilizing the SRL-based CAS cell could potentially provide significant performance improvement to most, if not all, of the surveyed architectures.

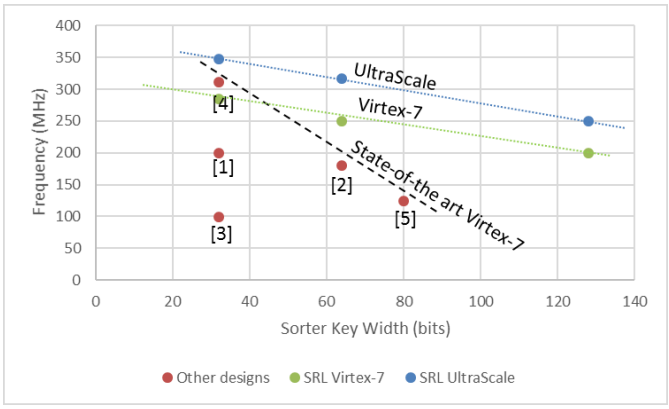

Figure 4. Full 32-way tree sorter clock frequency with proposed SRL CAS cell running on Virtex-7 and UltraScale devices compare to other designs

Notably, the microarchitecture of the CAS cell proposed in [4] was able to achieve a higher operating frequency; however, it had done so at the cost of extra pipeline registers. This trade-off may be acceptable for sorter design with a small number of sorter cells but may be prohibitive for larger sorter networks.

V. CONCLUSION

In this paper, we evaluated several compare-and-select cell micro-architectures capable of sorting 128-bit keys. The best proposed design leveraged the functionality of the SRL primitive. This micro-architecture outperforms all other evaluated sorter cells across a wide range of key widths. It achieves high operating frequency without multi-stage pipelining and utilizing only a single comparator to minimize overall resource utilization. Furthermore, this CAS cell can be easily integrated into most sorter architectures to significantly increase throughput. We successfully utilized the proposed CAS cell as part of a high performance 32-way merge sorter running at 250MHz.

REFERENCES

[1] R. Kobayashi and K. Kise, "FACE: Fast and Customizable. Sorting Accelerator for Heterogeneous Many-core Systems," in *Proceedings of the 9th Annual IEEE International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC)*, 2015, pp. 49–56.

[2] Takuma Usui, Thiem Van Chu and Kenji Kise, "A Cost-Effective and Scalable Merge Sorter Tree on FPGAs," in *Proceedings of the Fourth International Symposium on Computing and Networking,* 2016, pp. 47-56.

[3] Wei Song, Dirk Koch, Mikel Lujan and Jim Garside, "Parallel Hardware Merge Sorter" in *Proceedings of the IEEE 24th Annual International Symposium on Field-Programmable Custom Computin Machines,* 2016, pp. 95-102.

[4] Susumu Mashimo, Thiem Van Chu and Kenji Kise, "High-Performnce Hardware Merge Sorter", in *Proceedings of the IEEE 25th Annual International Symposium on Field-Programmable Custom Computin Machines,* 2017, pp. 1-8.

[5] Sang-Woo Jum, Shuotao Xu, Arvind, "Terabyte Sort on FPGA-Accelerated Flash Storage", in *Proceedings of the IEEE 25th Annual International Symposium on Field-Programmable Custom Computin Machines,* 2017, pp. 17-24.

978-1-5386-5989-2/18/$31.00 ©2018 IEEE