System-on-Chip Design for Fast Ultrasonic Chirplet Signal Decomposition Algorithm

Tianyang Fang, Austin Fite, Mikhail Gromov and Jafar Saniie
Embedded Computing and Signal Processing (ECASP) Research Laboratory (http://ecasp.ece.iit.edu)
Department of Electrical and Computer Engineering
Illinois Institute of Technology, Chicago, IL, U.S.A.

2023 IEEE International Ultrasonics Symposium (IUS) | 979-8-3503-4645-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/IUS51837.2023.10307989

***Abstract*—Decomposition and characterization of backscattered ultrasonic multiple interfering echoes is a critical step for analyzing the ultrasonic propagation path, propagation modes, coupling condition of the transducers and detecting defects within the propagation path. Chirplet Signal Decomposition (CSD) is an efficient way of analyzing ultrasonic echoes. However, CSD is computationally expensive and time-consuming for real-time signal processing applications. To address this problem, we present a System-on-Chip (SoC) implementation of the CSD algorithm, with the goal of speed optimization and real-time execution without sacrificing the accuracy of the signal decomposition and reconstruction. The implementation was tested on a Zynq Ultrascale+ series FPGA and achieved an echo estimation within two milliseconds.**

Keywords—Signal Processing, Ultrasonic, Optimization, FPGA, System-on-Chip.

I. Introduction

In examining the ultrasonic propagation channel, modes of propagation, transducer coupling state, and defect detection in the propagation path, the decomposition and characterization of backscattered ultrasonic multiple interfering echoes is the essential first step. First introduced in [1], the Chirplet Signal Decomposition algorithm has been proven a reliable and efficient way to analyze ultrasonic echoes. However, CSD is computationally expensive and time-consuming for real-time signal processing applications due to signal convolutions. Using FPGA fabric for some of the computationally heavy processes showed significant speed improvement in later research (see [2-3]). In this research, a System-on-Chip (SoC) implementation of the CSD algorithm is developed with respect to speed optimization and real-time execution without sacrificing the accuracy of the signal decomposition and reconstruction.

This paper overlays as follows: in Section II, a new SoC implementation of the CSD algorithm is described. Then different aspects of the improvements, including an improved searching algorithm for the chirplet parameter estimation, an FPGA implementation of the chirplet generator module, and the FIFO interface, which speeds up the parameter transportation, are discussed in each subsection. In section III, the SoC implementation is tested in various platforms, and the performance improvement compared to previous research is discussed. Section IV concludes the research.

II. System-on-Chip Design for CSD

As shown in Fig. 1, the System-On-Chip Design follows a top-down design where the CPU handles the I/O and higher-level functions like buffer control and signal windowing, and the programmable logic handles the lower-level and most time-consuming processes, namely the chirplet generation and signal cross-correlation.

The major modules instantiated in the top-level diagram are the chirplet transform module and two Xilinx AXI-Stream DMAs, as shown in Fig. 2. One DMA is used to transfer the reference signal from the Programmable System (PS) to the chirplet transform module, and the other DMA is to transfer the chirplet estimation from the chirplet transform to the PS, but note that the chirplet transform module must be in feedback mode for this transfer. It would be advantageous to add an additional DMA, which is used to program the chirplet generator

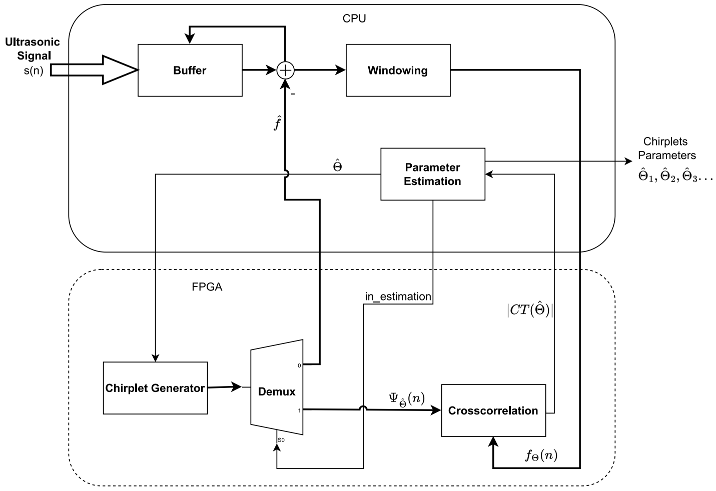

Fig. 1. The System-on-Chip design of CSD.

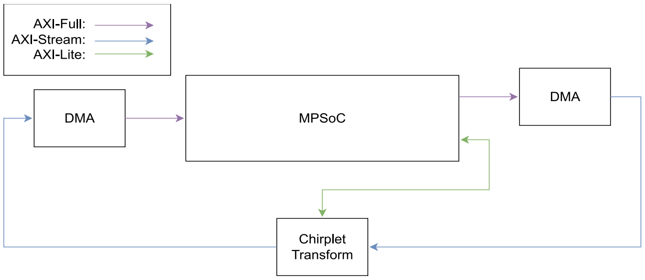

Fig. 2. DMA modules and AXI busses in the SoC design.

sub-module in the chirplet transform module and add a DMA for collecting cross-correlation data from the chirplet transform module. Both of these interfaces are currently handled by the register system, which uses an AXI-Lite bus. This slower bus interface causes the chirplet transform module to slow down due to AXI-Lite transactions without calculations taking place. Adding DMAs to control these interfaces will switch these interfaces to an AXI stream and allow the Xilinx IP to facilitate a fast transaction to the PS [4][5].

A. *Improved Searching Algorithm*

A major improvement made to increase the efficiency of the CSD algorithm is to create an efficient parameter-searching algorithm. The previous method for parameter searching followed the format of: Select a parameter value; Chirplet transform; Store the result. These results are then compared against each other, and the result with the strongest chirplet transform is selected as the parameter used for estimation.

The improved search algorithm separates the parameter search into two components: coarse search and fine search. Each step follows the same basic steps as before. Because there are two stages, the second stage only selects parameters around the estimate with the strongest chirplet transform from the first stage. Meaning the first stage uses coarse parameter spacing, and the second stage uses fine parameter spacing.

Fig. 3 shows an exemplary scenario where seven coarse parameter search estimates are calculated via the chirplet transform (shown in red). Then six fine parameter search estimates are calculated around the coarse search maxima (shown in green). This method results in a parameter search that is accurate and computationally efficient.

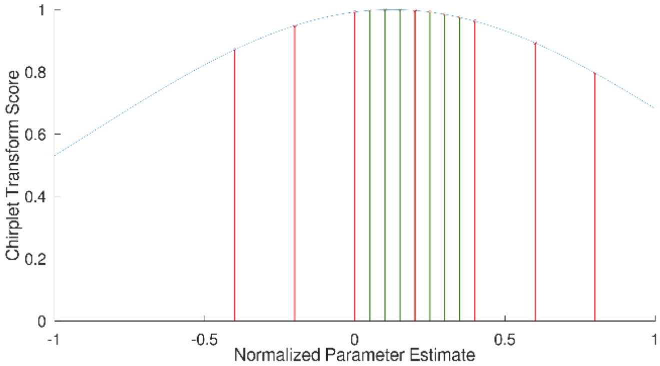

Fig. 3. Demonstration of the Improved Searching Algorithm

B. *FPGA Implementation*

As shown in Fig. 4, the Chirplet Transform Module (CTM) implementation in FPGA follows a top-down design where the CPU handles the I/O and higher-level functions like buffer control and signal windowing, and the programmable logic handles the lower-level and most time-consuming processes, namely the chirplet generation and signal cross-correlation.

The CTM is a programmable hardware block instantiated in an FPGA that performs a fast hardware-accelerated chirplet transform. The main advantage of using hardware to calculate the chirplet transform is that multiple chirplet samples may be calculated per clock cycle by using the FPGA resources to instantiate parallelization. These multiple samples are also correlated against a reference measured signal where the correlation operation also takes advantage of the parallel processing allowed by FPGA instantiation.

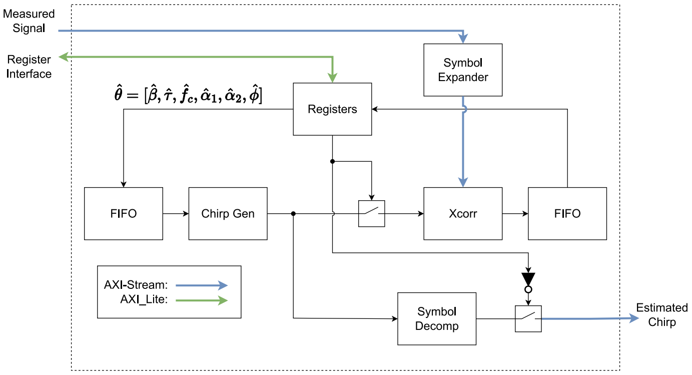

Fig. 4. The Chirplet Transform Module implementation in FPGA.

C. *FIFO System for the Chirplet Transform Module*

One of the features added to the hardware design that had a large impact on the system performance is the inclusion of FIFOs in the CTM. FIFO stands for "first in, first out," meaning that the first valid data written to a FIFO will also be the first valid data out of the FIFO, as shown in Fig. 5. A FIFO, in the context of an FPGA, is a digital hardware component that acts as a way to query data [6][7]. The FIFOs in the CTM are used to simplify a handshaking procedure between the PS and PL. When writing data from the PS to the CTM in the PL, the PS must know that the CTM is ready to accept chirplet parameters. This was originally accomplished via a register interface from the CTM to the PS, meaning that each time the PS wanted to write data to the CTM, first, the PS had to do a register read to determine if the CTM was ready to accept data. Likewise, the PS also had to check with the CTM that data is available at the output before reading new data, which is also accomplished with a register read.

By adding a FIFO to encapsulate the chirplet generator and cross-correlation sub-modules, the PS is guaranteed to have a certain number of data transfers that it may accomplish, knowing that the CTM is ready to accept data. The number of guaranteed transfers is determined by the depth of the FIFO (a depth of 128 is used for this project). The cross-correlation may

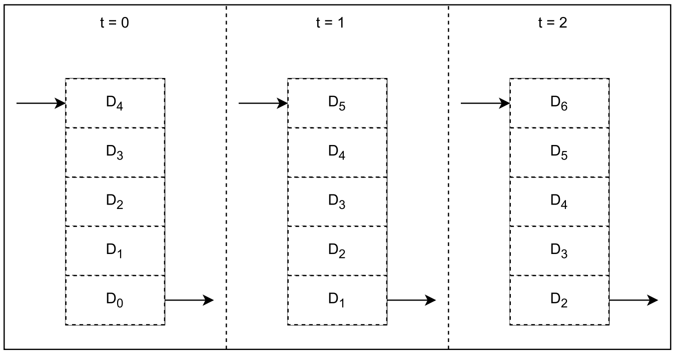

Fig. 5. Demonstration of a FIFO system. The first valid data written to a FIFO will also be the first valid data out of the FIFO.

TABLE I. HARDWARE TEST RESULTS AND COMPARISON

Scheme	SoC System	Architecture	FPGA clk (MHz)	FIFO System	Time to Calc Single CT (µs)	Time To Calc Param (µs)	Number of Clocks per Chirp Estimate	Time per Chirp Estimate (µs)	Performance Improvement by Clock Cycles [b]	Performance Improvement by Time [b]
0[a]	Xilinx Virtex II Pro	Virtex II Pro	100	-	-	-	12786876	127868.76	-	-
1	PYNQ Z2	ZYNQ 7000 Series	50	No	5.75	747.5	448500	8970	28.5x	14.2x
2	PYNQ Z2	ZYNQ 7000 Series	50	Yes	3.62	470.6	282360	5647.2	45.3x	22.6x
4	ZCU102	Zynq Ultrascale+	150	Yes	1.2	156	280800	1872	45.5x	68.3x

[a] The implementation from [2].

[b] Compared to Scheme 0.

then feed a FIFO at its output, storing that data for the PS to read from. The PS must still verify that the data is valid from this output FIFO, but the elimination of the PS needs to verify that the CTM is ready to accept data results in a speed improvement of 37% compared to the same system without FIFOs.

III. HARDWARE TESTS

The SoC implementation of CSD is tested on two different SoC platforms, the PYNQ Z2 and the ZCU 102, for performance evaluation. The number of clocks per chirplet estimation is chosen as the main performance indicator. This is because the clock rate can be changed according to the reliability requirement. Thus, the clock rate is not a deterministic factor for the performance comparison. The results of the hardware test are then compared to that presented in [2] and listed in Table I.

A major improvement to the hardware design compared to [2] and [3] is introducing a FIFO system encapsulating the chirplet transform core sub-modules. This FIFO system allows the PS to guarantee that the chirplet transform core will be ready to accept data for a certain number of inputs, meaning that the PS does much less status polling. This improvement is shown in the table above, with scheme one using a polling system and scheme two and three using a FIFO system. Schemes one, two, and three are also compared against a reference scheme with data taken from [2]. Two improvement metrics are used, clock cycles per chirplet estimation and total time per chirplet transformation. Clock cycles per chirplet estimation are more system agnostic and are mostly referring only to the efficiency of the hardware design. Time per chirplet estimation depends on the system's ability to route hardware with a fast clock and is more affected by the system used to implement hardware.

IV. CONCLUSION

The proposed SoC implementation of the CSD algorithm is tested on multiple Xilinx FPGA SoC architectures, where the top-level algorithm module runs on the Programmable System (PS), and the Chirplet Transform Module (CTM) is implemented in FPGA fabric. For the decomposition of a single chirplet of 512 samples, the proposed implementation takes 1872 µs to execute. Compared to previous implementations of the CSD algorithm, an improvement factor of 23x is made with a ZYNQ 7000 series FPGA, and an improvement factor of 68x is made with a ZYNQ Ultrascale+ series FPGA. Therefore, we can conclude that the new implementation has a significant speed improvement compared to previous implementations.

REFERENCES

[1] Y. Lu, R. Demirli, G. Cardoso and J. Saniie, "A successive parameter estimation algorithm for chirplet signal decomposition," in IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control, vol. 53, no. 11, pp. 2121-2131, November 2006, doi: 10.1109/TUFFC.2006.152.

[2] Y. Lu, E. Oruklu and J. Saniie, "Fast Chirplet Transform With FPGA-Based Implementation," in IEEE Signal Processing Letters, vol. 15, pp. 577-580, 2008, doi: 10.1109/LSP.2008.2001816.

[3] A. Fite, M. Gromov, T. Fang and J. Saniie, "Speed-Optimized Implementation of Fast Chirplet Decomposition Algorithm on FPGA-SoC," 2023 IEEE International Conference on Electro Information Technology (EIT), Romeoville, IL, USA, 2023, pp. 452-457, doi: 10.1109/eIT57321.2023.10187330.

[4] F. Shanehsazzadeh and M. S. Sadri, "Area and performance evaluation of central DMA controller in Xilinx embedded FPGA designs," 2017 Iranian Conference on Electrical Engineering (ICEE), Tehran, Iran, 2017, pp. 546-550, doi: 10.1109/IranianCEE.2017.7985100.

[5] G H. Kavianipour, S. Muschter and C. Bohm, "High Performance FPGA-Based DMA Interface for PCIe," in IEEE Transactions on Nuclear Science, vol. 61, no. 2, pp. 745-749, April 2014, doi: 10.1109/TNS.2014.2304691.

[6] Liu, Bing Qi, Ming Zhe Liu, Gang Yang, Xiao Bo Mao, and Huai Liang Li. "Research and Design of Asynchronous FIFO Based on FPGA." Applied Mechanics and Materials. Trans Tech Publications, Ltd., September 2014. https://doi.org/10.4028/www.scientific.net/amm.644-650.3440.

[7] N. F. Jusoh, A. Ibrahim, M. A. Haron and F. Sulaiman, "An FPGA implementation of shift converter block technique on FIFO for UART," 2011 IEEE International RF & Microwave Conference, Seremban, Malaysia, 2011, pp. 320-324, doi: 10.1109/RFM.2011.6168758.