Design of an IDM-Based Determinant computing unit for a 130nm low power CMOS ASIC acoustic localization processor

Roberto Cerdas-Robles*, Agustín Rodríguez†, Alfonso Chacon-Rodríguez*, and Pedro Julián †
*Escuela de Ingeniería Electrónica, Tecnológico de Costa Rica
Email: {rcerdas, alchacon}@itcr.ac.cr
†Instituto de Investigaciones en Ingeniería Eléctrica, IIIE (UNS-CONICET)
Departamento de Ingeniería Eléctrica y de Computadoras
Universidad Nacional del Sur, Bahía Blanca, Argentina
Email: rodrijuana@gmail.com, pjulian@uns.edu.ar

***Abstract*—A determinant computing circuit in floating point format has been designed and tested for use in a CMOS ASIC acoustic localization processor. The Internal Division Method (IDM) was used to implement the operation, employing a modified SRT radix-4 circuit for division operations. The unit was designed for VLSI implementation in a commercial 130nm low-power CMOS process, with an operation frequency of 100MHz. The algorithm employed is parallelizable for future prototypes, should a higher operation frequency be required.**

***Index Terms*—Acoustic Localization, Multichannel Cross Correlation Coefficient, TDOA, Low Power VLSI, FPGA, UVM.**

I. Introduction

The use of an acoustic localization system has been proposed as part of a design of a redundant sensor network for localization of moving vehicles [1]. It has been demonstrated that the phase difference between separately captured signals in an array formed by two or more microphones can be used to estimate the angle of a sound source with respect to the array [2], [3]. Three algorithms for *Time Difference of Arrival* (TDOA) estimation were evaluated and compared in a previous work [4]. The algorithm chosen to solve the problem stated requires the computation of determinants as part of the estimation process. This document presents a proposal for the implementation of this determinant computing unit in a commercial 130nm low-power CMOS process, to evaluate the suitability of the chosen arithmetic architectures in regards to area, power, and maximum operation frequency as part of an acoustic localization processor. Section II of this document briefly explains the fundamentals of the algorithm the determinant unit will be used in, to provide the reader with a frame of reference for the selected constraints. Section III details the elaboration of the floating-point division unit, and the custom architecture used for implementation of the SRT algorithm. Section IV describes the determinant computation unit's design. Section V summarizes the results obtained when the design was verified using a UVM approach on a XC7A100T FPGA Device, as well as Synopsys Design Compiler's synthesis results on the chosen technology. Finally, Section VI briefly adds concluding remarks and observations, as well as recommendations for further improvements of the unit under study.

II. Description of the MCCC Algorithm

The Multichannel Cross Correlation Coefficient is an algorithm that allows estimation of delay-of-arrival or time-difference-of-arrival (DOA or TDOA, respectively) of a source signal to a microphone array. Unlike either regular or generalized cross correlation methods, MCCC considers data from more than two channels at a time, resulting in a more robust approximation of DOA [5]. Assume a microphone array whose microphones are equidistant from each other, and the sound source is located in the far plane; $\tau_{n1} = F_n(\tau)$ is the delay between the first and nth microphones at a given source angle and, for an equidistant linear array, delay to each microphone is given by:

$$F_n(\tau) = (n-1)\tau_{12} \tag{1}$$

For an array with N microphones, the input signal vector is:

$$y_a(k,p) = [y_1(k)y_2(k+F_2(p)\cdots y_N(k+F_N(p)] \tag{2}$$

with p a hypothetical TDOA. The cross correlation function between any two signals in the array is:

$$r^{CC}_{yiyj}(p) = E[y_i(k)y_j(k+F_j(p)] \tag{3}$$

The spatial correlation matrix for a microphone array is defined as:

$$\mathbf{R_a(p)} = \begin{bmatrix} \sigma_1^2 & r_{y1y2}(p) & \cdots & r_{y1yn}(p) \\ r_{y2y1}(p) & \sigma_2^2 & \cdots & r_{y2yn}(p) \\ \vdots & \vdots & \ddots & \vdots \\ r_{yny1}(p) & r_{yny2}(p) & \cdots & \sigma_n^2 \end{bmatrix} \tag{4}$$

It can be demonstrated, as in [5], that the hypothetical TDOA p that satisfies $p = \tau_{12}$ is the same p that satisfies $argmin\{\mathbf{R_a(p)}\}$. Once the TDOA is found, the angle of arrival θ may be found as

$$\theta = \cos^{-1}(\frac{d}{c\tau_{12}}) \tag{5}$$

with c the speed of sound in m/s, and d the distance between microphones in the array [6].

III. Description of division computing unit

Floating point computing of divisions is required as part of the determinant estimation process, as the reciprocal function $f(x)^{-1}$ is needed. The SRT algorithm is employed. The maximum redundancy Radix-4 variant was chosen and modified to calculate eight quotient bits per clock cycle, by replicating the quotient computation adders four times. The equation that describes the result of each adder is:

$$R_{i+1} = 4R_i + q_i * D \tag{6}$$

in which q_i is the quotient bit generated by each adder stage. When maximum redundancy is used, seven possible quotients can be generated, $q_i \in [-3, -2, -1, 0, 1, 2, 3]$. The term *quotient bit* refers to a bit in redundant notation, as opposed to binary notation. Each quotient bit requires three actual bits to represent its value in binary [7]. The block accepts any input value x in the range $x \in]0, 2[$

The block, coded in Verilog, was designed to accept 16 input bits, in fixed-point notation, with 15 fractional bits. The output range is therefore in the range $y \in]0.5; 2^{15}[$

This input range was chosen to use the block with a half-precision floating point number's mantissa. Due to the restrictions of the SRT algorithm, which state that the partial result may never exceed four times the value of the divisor, the operation is physically implemented as $y = \frac{0.5}{x} * 2$. This multiplication can easily be implemented by performing a shift operation after the division has been computed. Figure 1 presents the block diagram of the designed divisor's datapath.

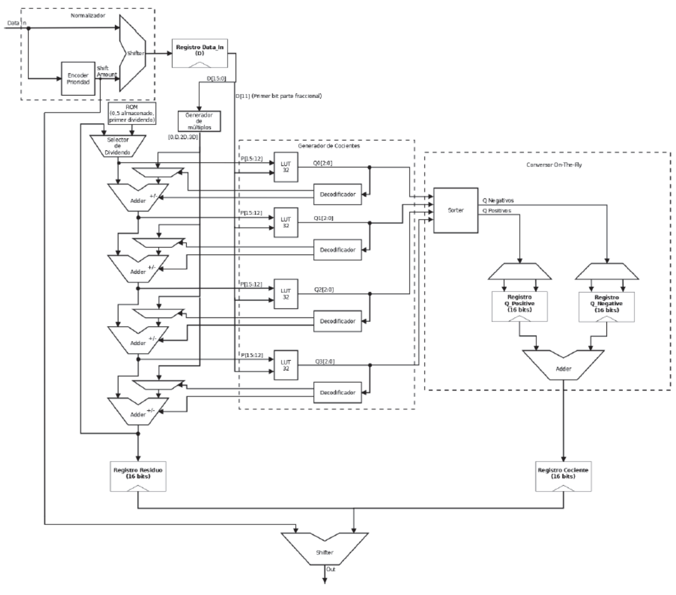

Figure 1. Block diagram of the designed division unit's datapath, using a radix-4 SRT structure replicated four times to estimate eight quotient bits per clock cycle.

IV. Description of the determinant computing unit

The system was implemented with the capacity to evaluate fourth-order matrixes, which correspond to spatial cross correlation matrixes generated by arrays of four microphones.

A. Algorithm used

The algorithm employed is the internal division method (IDM) described in [8], and described in Verilog. It is a parallelizable methodology that can be used to estimate the determinant of any square matrix of order n, as long as the matrix's first element is not zero. The algorithm is a generalization of Chio's rule. The determinant is estimated by breaking up the matrix into second order sub-determinants, which are then used to find the determinant. The algorithm was implemented sequentially to reduce area use and minimize power consumption. The algorithm can be summarized as follows:

1) Isolate the first row and first column of the matrix.
2) Form $(n-1)^2$ second order submatrixes, combining each element of the matrix with the first element of the matrix, the first element of its row, and the first element of its column. Ignore the elements present in either the first row or the first column.
3) Calculate the determinant of each second order submatrix.
4) Divide each calculated subdeterminant by the matrix's first element to normalize them.
5) Replace each element that produced a submatrix by the result obtained from the last step.
6) Repeat the process, ignoring one additional row and one additional column, until $n = 2$.
7) Multiply all elements belonging to the diagonal of the resulting matrix. This value is the determinant of the matrix.

B. Data representation

Due to the precision required by determinant computations to accurately predict the angle of arrival, floating point representation of data is required to minimize the system's word width. A modified IEEE 754 half-precision floating point (16 bit) format was employed, using an explicit hidden bit (which is implicitly set to one in the standard implementation) to facilitate the FPU unit's design. Five bits were used for exponent representation; as in the standard format, the exponent is represented with an offset to avoid the need for two's complement representation. Ten bits were employed to represent the mantissa. The last bit is reserved for to represent the number's sign. Using this format allows for representation of numbers between -2^{15} and 2^{15}, with a maximum resolution of 2^{-9} bits in the mantissa.

C. Datapath's architecture

Figure 2 presents a simplified block diagram of the determinant computing unit's datapath. A normalizer is inserted in the FPU unit along with the SRT algorithm detailed in the previous section, to automatically truncate results to the ten most significant bits. A simple circuit implementing the SPI protocol is used to interface both the input and output shown with any outside master unit.

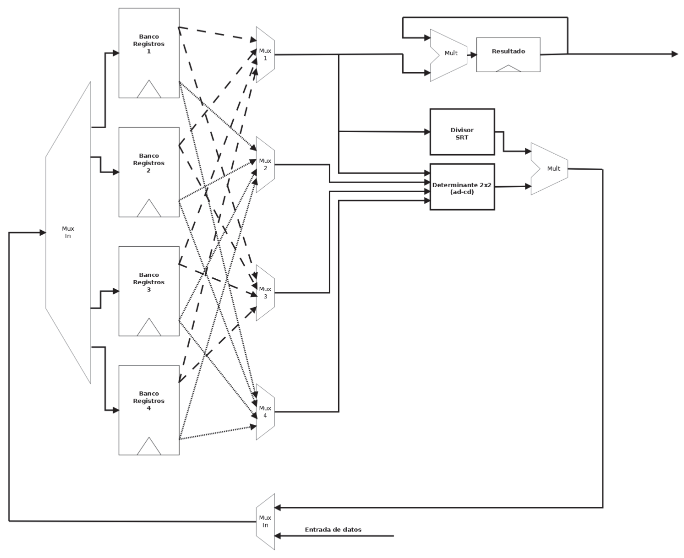

Figure 2. Simplified block diagram of determinant computing unit datapath. Each register bank has two reading address inputs to access data on two separate channels. The reading address pattern is generated using two counters.

V. Results

The design was validated using an UVM verification environment written in System Verilog [9]. Random data sequences were generated and driven into the determinant calculator, and responses were compared via a scoreboard to the reference's results. Determinants were found to be within 5% of the expected value for a 10-bit mantissa. Figure 3 shows the verification environment used. To speed up the validation, the design was synthesized on a XC7A100T FPGA. Test vectors were applied to the DUT on the FPGA via an SPI interface. Table I shows synthesis results on the FPGA.

Table II presents the post-synthesis results obtained using Synopsys Design Compiler, after porting the Verilog code to a low power standard cell library for a 130nm CMOS commercial process. In both cases, the critical delay path is the data route through the normalizer unit in the FPU after encountering a division. Power consumption is significantly reduced when using the low-power standard cells library.

To evaluate the circuit's performance within the context of the desired application, a linear array of four microphones was prepared and deployed in a woodworking workshop. A wood-polishing machine positioned 5m away from the

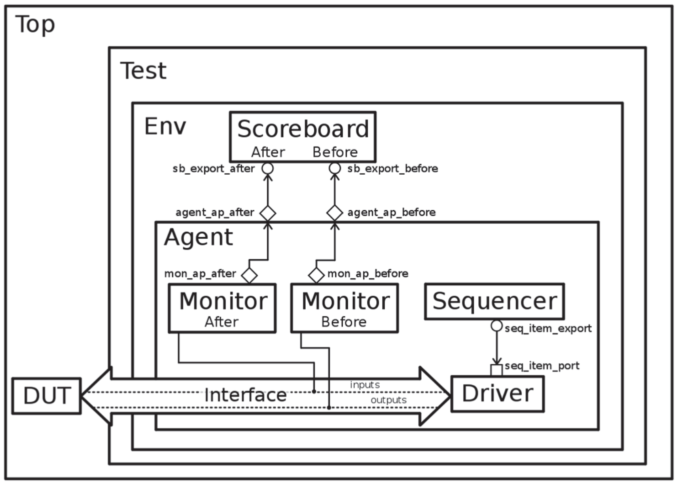

Figure 3. UVM environment used for verification of the determinant computing unit [9].

Table I
SUMMARY OF SYNTHESIS RESULTS FOR XC7A100T FPGA.

Timing	
Critical delay time (nS)	$12,231$
Maximum operation frequency (MHz)	$81,759$
Power	
Total power (mW)	$65,13$
Dynamic power (mW)	$22,69$
Static power (mW)	$42,44$
Area	
Registers	483
LUTs	1742
I/O Blocks	7
Percentage of available resources used	4%

Table II
SUMMARY OF POST-SYNTHESIS RESULTS AFTER PORTING THE CODE TO A LOW POWER STANDARD CELL LIBRARY FOR A 130NM CMOS COMMERCIAL PROCESS. DATA REPORTED BY SYNOPSYS DESIGN COMPILER.

Timing	
Critical delay time (nS)	9.67
Maximum operation frequency (MHz)	103.41
Power	
Total power (mW)	0.8439
Dynamic power (mW)	0.8438
Static power (mW)	$81.53x10^{-6}$
Area (Values given in terms of a NOR gate area)	
Combinational	57372.48
Sequential	18106.56
I/O Pads	7
Total area	70128.00

array was used as the source signal, sampled at 32 kHz for 30 second intervals. The four microphones were spaced linearly, with a spacing of 1m between each device in the array. The angle of arrival of the signal to the array was varied by rotating the array along its horizontal axis seven times. 200 hypothetical delay values were evaluated in each case, to account for sources located between $0°$ and $180°$. Results were calculated using the MCCC method described in previous sections; theoretical data was derived using MATLAB to calculate the determinant of the cross-correlation matrix, and compared to the results of utilizing the proposed circuit to estimate the determinant instead. Table III summarizes the results obtained for each of the seven measurements. The same data is presented graphically in figure 4.

Table III
ANGLE OF ARRIVAL ESTIMATION FOR THEORETICAL AND EXPERIMENTAL CASES.

Theoretical angle	**Experimental angle**
29.31°	47.11°
41.9°	168°
46.27°	60°
64.83°	78.34°
78.34°	66.83°
82°	85.73°
88.78°	90°

Discrepancies between theoretical and experimental data could arise as a result of lower precision in the circuit's arithmetic operations; the determinant calculator's response accuracy is bound by the amount of bits in the representation's mantissa, while MATLAB has no such constraint. More significantly, however, is the variation in TDOA between the peak angle value and the minimum value: approximately $500\mu s$. At lower sample rates, an error of one sample becomes more significant, with each sample representing a greater time displacement. Complicating this issue, the relationship between the angle of arrival and TDOA is non-linear; for angles below $60°$, decreasing the angle results in a fast non-linear decline in the expected TDOA as a result of the cosine term present in Eq. (5). This explains the increasing accuracy as the angle approached $90°$. Therefore, as proven in [6], the sampling frequency must be increased to at least 200 kHz in order to get accurate results to within one degree for the entire angle range. One can thus safely assume that the error in these tests was merely the low data sampling used, and not a problem within the determinant computing units. Tests using a 200kHz sampling speed were being performed as of the writing of this paper, to settle the issue.

VI. CONCLUSIONS

This document presents the implementation of a functional prototype of a determinant computing unit for fourth-order matrixes, which allows for evaluation of the selected arithmetic architectures in the desired application. Modifying the zero counter architecture is recommended as a way to reduce the critical delay path. Alternative architectures for division computing should also be explored.

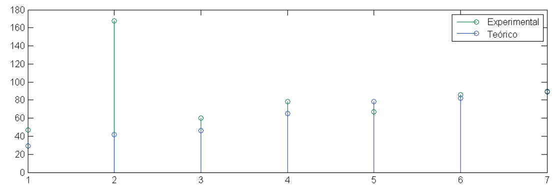

Figure 4. Stem graph comparing the theoretical (blue) versus experimental (green) results.

The MCCC algorithm used to test the circuit presents vulnerability to both low sampling rates and precise localization of sources positioned outside the range $[60°,120°]$. Increasing the distance between microphones to increase the TDOA range beyond $500\mu s$ and reduce the required resolution is not desirable; as distance between microphones is increased, one must also increase the distance of the entire array to the source, or degradation in the algorithm's performance might occur as a result of deviation from the ideal far-plane model assumed for the MCCC's derivation. For an application with moving sources, this is not recommended. Alternatively, additional computing precision can help mitigate the angle issue, and can be achieved by increasing the sample rate; should this prove to be insufficient, the determinant calculator's precision should be increased by adding more mantissa bits to the base word length, possibly by using the 32-bit standard IEEE format instead.

REFERENCES

[1] P. Julián, M. Favio, P. Mandolesi, A. Andreou, S. Shamma, E. Nebot, A. Chacón. 3D Gigascale Integrated Circuits for Nonlinear Computation, Filter and Fusion with Applications in Industrial Field Robotics.

[2] A. Chacon-Rodriguez, F. Martin-Pirchio, S. Sañudo, and P. Julián. A low power integrated circuit for interaural time delay estimation without delay lines. *Circuits and Systems Part II: Express Briefs, IEEE Transactions on*, vol. 56, 2009, pp. 575-579.

[3] A. Chacon-Rodriguez, P. Julian, and F. Masson. Fast and low power integrated circuit for impulsive sound localization using Kalman fillter approach. *Electronics Letters*, 46:533–534, April 2010.

[4] R. Cerdas. *Selección de un algoritmo de localización acústica para su uso en aplicaciones de robótica industrial*. Lic. thesis, Dept. Electron. Eng., Instituto Tecnológico de Costa Rica, Cartago, Costa Rica, 2012.

[5] J. Benesty, J. Chen and Y. Huang. Microphone Array Signal Processing, chapter 9: DOA and TDOA estimation, pages 191–200. Springer, 1 edition, 2008.

[6] P. Julian, A. Andreou, L. Riddle, S. Shamma, D. Goldberg and G. Cauwenberghs. A Comparative Study of Sound Localization Algorithms for Energy Aware Sensor Network Nodes, in IEEE Trans. on Circuits and Systems I: Regular Papers, vol.51, no.4, pp.640,648, April 2004, doi: 10.1109/TCSI.2004.826205

[7] C.-L.Wey and C.-P.Wang. Design of a fast radix-4 SRT divider and its VLSI implementation. In IEEE Proc.-Comput. Digit. Tech., IEEE. 1999.

[8] M. P. Menezes, C. E. M. Pereira and L. M. Sato. IDM – A New Parallel Methodology to Calculate the Determinant of Matrices of the Order n, with Computational Complexity O(n). In IEEE Latin America Transactions, vol. 10:1. 2012.

[9] UVM Introduction (2014, Oct. 6) [Online]. VLSI Encyclopedia. Available: http://www.vlsiencyclopedia.com/2014/10/uvm-introduction.html