bit ($A0$) with all other bits held low. The SKIP feature of the DEMUX is then used to rotate this active channel to output $O0$. Upon detection and successful reception of the PRBS, the system is returned to the normal data shifting mode.

The DEMUX receives this data and translates it to a parallel format. A switch matrix is used to select a channel and route it to the receive module of the BERT. The output data available DA signal is used for the receive clock. It samples the data and drives the BERT's internal PRBS generator on the receive side. Error counts are accumulated for a second for each channel, all under PC control. Any nonzero error count is a basis for part rejection.

For 8-bit mode, system modifications are required since the BERT cannot operate at the corresponding word rate (250 MHz). To achieve this, the output data of the BERT are translated into a return-to-zero (RZ) format. By translating the data into a RZ format, the maximum bandwidth of the input data cells of the MUX is tested. This RZ pattern is shifted across the MUX inputs at the word rate, transmitted to the DEMUX, and reassembled.

The system can be extended for testing higher frequency parts or lower bit-width parts by the use of additional GaAs MUX/DEMUX pairs to interface from the shift register to the MUX under test and between the DEMUX under test and the BERT.

VI. Results and Conclusions

The devices presented perform parallel/serial and serial/parallel conversion continuously (time-division multiplexing/demultiplexing) at rates up to 2 Gbits/s. A rigorous testing technique using a long PRBS sequence (2^{23}-1) and exercising all channels simultaneously is employed. Fig. 4 is of the "data eye" pattern out of the MUX at this data rate and PRBS, and Fig. 5 shows the two phase indication signals, PH_{REF} and PH_{ERR}, under the same conditions. ECL compatibility is verified both at low-speed test by direct measurement and at high-speed test by interfacing to ECL components, power supplies, and test equipment. Power dissipation is approximately 2 W per device.

The devices are an example of the state of the art in high-speed standard cell design capability. They are simultaneously among the fastest 8- or 16-bit MUX/DEMUX pairs in production regardless of design methodology. The product development cycle has been completed with the assembly of a computer-controlled high-speed test station. Menu-driven controlling software facilitates a high-volume throughput.

Acknowledgment

The authors would like to acknowledge the contributions of G. Inzer, L. Irlandez, and J. Maritz to this project.

References

[1] M. McDonald *et al.*, "A 12:1 multiplexer and demultiplexer chip set for use in a fiber optic communications system," in *GaAs IC Symp. Tech. Dig.*, Oct. 1986, p. 229.

[2] A. Rode *et al.* "A high yield GaAs MSI digital IC process," in *IEDM Tech. Dig.*, Dec. 1982, Paper 7.1.

[3] D. Larson *et al.* "A GaAs counter family designed using standard cells," in *CICC Tech. Dig.*, May 1986, p. 504.

[4] D. Smith *et al.* "New approaches to packaging for high speed GaAs IC applications," in *GaAs IC Symp. Tech. Dig.*, Nov. 1985, p. 151.

A Systolic VLSI Chip for Implementing Orthogonal Transforms

WAYNE P. BURLESON, STUDENT MEMBER, IEEE, LOUIS L. SCHARF, FELLOW, IEEE, ARTHUR R. GABRIEL, AND NEIL H. ENDSLEY, SENIOR MEMBER, IEEE

***Abstract* —This paper describes the design of a systolic VLSI chip for the implementation of signal processing algorithms that may be decomposed into a product of simple real rotations. These include orthogonal transformations. Applications of this chip include projections, discrete Fourier and cosine transforms, and geometrical transformations. Large transforms may be computed by "tiling" together many chips for increased throughput. A CMOS VLSI chip containing 138 000 transistors in a 5×3 array of rotators has been designed, fabricated, and tested. The chip has a 32-MHz clock and performs real rotations at a rate of 30 MHz. The systolic nature of the chip makes use of fully synchronous bit-serial interconnect and a very regular structure at the rotator and bit levels. A distributed arithmetic scheme is used to implement the matrix-vector multiplication of the rotation.**

I. The Problem

The problem addressed by this paper arose from the need to perform real-time computation of projections in image processing without resorting to expensive array processors. Givens rotations provide a useful method for decomposing the projection operator into a regular and numerically stable algorithm. The design specification is derived from the data rates of 1K×1K images at 30 frames/s. The Givens decomposition results in a computational structure that is essentially single-instruction, multiple-data (SIMD), with a simple primitive operation and a regular cell layout and interconnect.

The amount of parallelism and programmability needed in a given algorithm determines what kind of DSP VLSI solution is best suited for its implementation. The trade-offs of performance versus versatility in different applications accommodate a wide range of VLSI designs, shown in Fig. 1. We chose to design a VLSI building block that could be used to configure large arrays of rotation operators that handle the high data throughput rates associated with real-time signal processing. There are several advantages in a building block design. Primarily, design cost is minimized by optimizing the single building block chip and making it fit a number of algorithms. In addition, the building block allows for extensibility of problem size, and reconfiguration for different applications. Applications for this chip include projection operators, real discrete Fourier and cosine transforms, and virtually any orthogonal transform that can be decomposed into a product of Givens rotations. An important consideration is that the transform, although programmable, must not change frequently. The time required to compute and load the Givens rotation coefficients is orders of magnitude greater than the processing rate of the chip.

Manuscript received September 2, 1988; revised December 1, 1988. This work was supported by Ball Aerospace, Boulder, CO, and by the Office of Naval Research, Electronics Branch, Arlington, VA, under Contract ONR 85-K-0693.

W. P. Burleson and L. L. Scharf are with the Department of Electrical and Computer Engineering, University of Colorado, Boulder, CO 80309.

A. R. Gabriel and N. H. Endsley are with Ball Aerospace, Boulder, CO 80306.

IEEE Log Number 8826138.

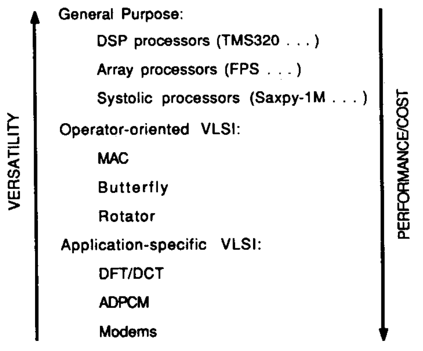
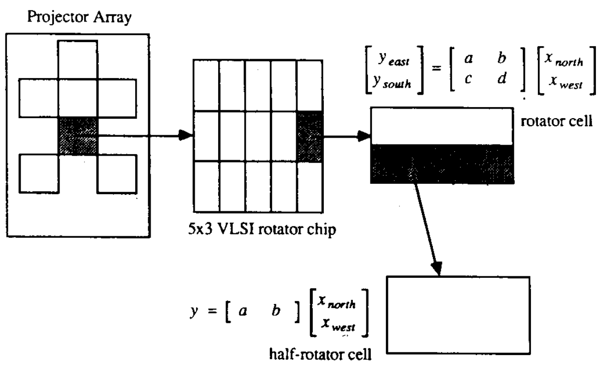

VERSATILITY
General Purpose:
DSP processors (TMS320 . . .)
Array processors (FPS . . .)
Systolic processors (Saxpy-1M . . .)
Operator-oriented VLSI:
MAC
Butterfly
Rotator
Application-specific VLSI:
DFT/DCT
ADPCM
Modems
PERFORMANCE/COST

Fig. 1. Spectrum of DSP VLSI solutions.



$$\begin{bmatrix} y_{east} \\ y_{south} \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x_{north} \\ x_{west} \end{bmatrix}$$

$$y = [\,a \quad b\,] \begin{bmatrix} x_{north} \\ x_{west} \end{bmatrix}$$

Fig. 2. VLSI hierarchy of projector array.

II. Background

Rialan and Scharf [1] propose a cellular architecture in which the projection operator is decomposed into a set of Givens rotations. First, the projection operator is decomposed into a product of orthogonal transformations by QR factoring the subspace basis A:

$$P_A = A(A^TA)^{-1}A^T = QR(R^TR)^{-1}R^TQ^T = QI_0Q^T$$

where

$$Q^TQ = I$$

and I_0 is an identity matrix. The Q matrices are then factored into a product of Givens rotations Q_i

$$Q = \prod_{i=1}^{n} Q_i$$

where Q_i is the Givens rotation matrix

$$Q_i = \begin{bmatrix} I & & \\ & R_i & \\ & & I \end{bmatrix}$$

$$R_i = \begin{bmatrix} \cos\theta_i & \sin\theta_i \\ -\sin\theta_i & \cos\theta_i \end{bmatrix}.$$

This decomposition maps into the orthogonally connected array of regular cells illustrated in Fig. 2, where each cell performs a 2×2 matrix-vector multiply for each Givens rotation. By pipelining the computation of the Givens rotations, a highly parallel solution is achieved. One important factor in the selection of this architecture was the attention paid to design methodology and VLSI implementation. Although the Givens decomposition does not result in a "fast" algorithm that actually reduces the number of computations, it does give a numerically stable implementation which maps into a regular parallel VLSI design. Previous reports of systolic architectures for Givens rotations include those by Gentleman and Kung [8] and Heller and Ipsen [9].

III. The Architecture

The architecture of the rotator array lies somewhere between a general-purpose design and an application-specific design. Due to the general applicability of the rotation array, we refer to this as an operator-oriented architecture, analogous to a multiply–accumulate operator or an FFT butterfly operator. We will discuss the architecture of the rotator array and its VLSI implementation from two different perspectives. First, the design procedure will be presented and the design trade-offs justified. Second, the functionality of the chip will be described in terms of the four modes of operation.

A. Design Trade-offs

The design of a high-performance VLSI implementation consisted of a number of trade-offs of computational method, data format, intercell communication, programmability, testing, and design methodology. The cellular architecture was an obvious candidate for systolic implementation, with local data transfer and a minimum of global signals. Similar to the systolic arrays proposed by Kung [2], the primitive cell consists of a word-level multiply and add function and a minimum of control logic.

1. Design Methodology: By following rules of regularity and interconnect through abutment, a very simple design was achieved. The performance and layout constraints required a full custom design methodology; however, the inherent simplicity of the design resulted in a very rapid design cycle. Despite the availability of standard cells, cell compilers, and data-path compilers, a full-custom design was chosen to maximize performance, minimize area, and produce a regular interconnect scheme. The rotation cell was designed so that all interconnect is by abutment, maintaining the regularity of the layout, minimizing design errors at a high level, and allowing simple extensibility of the array size. The only layout at the chip level is the interconnect between the array and the pad frame.

2. Data Format and Arithmetic: The function of the rotator cell is

$$\begin{bmatrix} y_{\text{east}} \\ y_{\text{south}} \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_{\text{north}} \\ x_{\text{west}} \end{bmatrix}.$$

We actually generalized this to

$$\begin{bmatrix} y_{\text{east}} \\ y_{\text{south}} \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x_{\text{north}} \\ x_{\text{west}} \end{bmatrix}.$$

This operation is partitioned in hardware into two half-rotator cells, each performing one row–column product:

$$y = [\,a \quad b\,] \begin{bmatrix} x_{\text{north}} \\ x_{\text{west}} \end{bmatrix}.$$

This computation is just a sum of two products, for which many hardware implementations exist. It should be noted that the sum of two products, while a special case of a multiply–accumulate function (MAC), can be optimized further because a MAC is designed to compute arbitrary length sums of products.

An area–time trade-off study of arithmetic methods resulted in a choice of a distributed arithmetic [3] solution over CORDIC [4], bit-serial multipliers [5], or fully parallel multipliers. The basic arithmetic function of the rotator cell is a sum of products:

$$\sum_{k=1}^{L} a_k x_k .$$

Break x_k into its fixed-point binary representation:

$$\sum_{k=1}^{L} a_k \sum_{i=0}^{B-1} 2^{-i} x_{ik} .$$

Reverse the order of summation to obtain the kernel of the distributed arithmetic algorithm:

$$\sum_{i=0}^{B-1} 2^{-i} \sum_{k=1}^{L} a_k x_{ik} .$$

This is implemented by precomputing the 2^L values that $\Sigma_{k=1}^{L} a_k x_{ik}$ can take on and selecting the appropriate one by looking up a partial product that depends on the decoded value of the x_{ik}. The partial products are then accumulated with a shift left of each more-significant partial product. Since the data flow of the x_{ik} into the rotator cell is bit serial, no buffering is required. This is an example of an I/O and arithmetic scheme which are well-matched, minimizing hardware and latency through the systolic cell. An example of a poorly matched scheme would be a bit-serial multiplier whose operands are fetched and stored in a parallel register file. Another would be a parallel multiplier with bit-serial interconnect.

Residue number systems or logarithmic number systems were dismissed as being overly complex for the performance gain they would achieve. In addition they would require a new analysis of the precision of the algorithm.

B. Modes of Operation

The rotator chip has four modes of operation, one of which is selected by the two mode pins L and T. The four modes, COMPUTE, LOAD, TEST, and IDLE, just modify the loading of registers on the chip. In the COMPUTE mode, the accumulator and output shift register are loaded once per word time in accordance with the distributed arithmetic algorithm. LOAD mode connects the three partial-product registers as a long shift register to facilitate the serial loading of the coefficients. TEST mode resembles LOAD mode except that the accumulator, output shift register, and control logic latches are added to the scan path. During IDLE mode, no register contents are modified. IDLE mode can be used as a NOP instruction for debugging and synchronizing between LOAD and COMPUTE modes.

1. Computation: The computation mode implements the actual Givens rotations via a simple distributed arithmetic algorithm. This algorithm consists of reordering the summations in a multibit sum of products. This is well-suited to the bit-serial data movement on the chip, and specifies the order of bits as least significant bit (LSB) first. Sixteen clock cycles are needed to perform the two-vector by two-vector product. On the last cycle, the adder output is loaded into the output register and the accumulator is cleared. During the following computation, the result is shifted out bit serially to the next (east or south) cell. Variable input word lengths up to a maximum of 16 bits can be accommodated with a corresponding performance improvement for shorter words. The MSB pin is merely strobed to indicate the end of a data word. Supporting two's complement data words requires that the last accumulation be a subtract. Two's complement partial products and partial sums require sign extension and an extra bit in the accumulator.

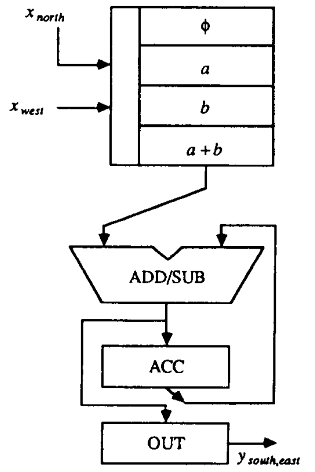


Fig. 3. Half-rotator cell block diagram.

The hardware for implementation of a half-rotator cell is shown in Fig. 3 and consists of a 16-bit data path with three partial product registers ($a, b, a+b$) (zero is in ROM), an accumulator, an output shift register, and an adder/subtracter. The only critical timing path involves the 16-bit adder/subtracter, so a 4-bit carry lookahead scheme was used, sacrificing some regularity to meet a performance constraint. In addition to the data path, there are about 30 gates of control logic.

2. Programming: The programmability of the chip is controlled solely by the values of the stored partial products ($a, b, a+b$), which are loaded serially on the same pins as the incoming data. To implement the distributed arithmetic algorithm, the partial-product registers actually have to be 1 bit longer than the coefficients to accommodate the value $a+b$. We compromised the extra bit for regularity. Therefore the coefficients are only 15 bits.

3. Testing: Testing of the chip is accomplished by connecting all latches onto a common scan path. Since the scan path is so trivial (just a long shift register), the chip is an obvious candidate for on-chip self-test. A pseudorandom sequence generator could drive the scan path and a signature analysis circuit could check the outcoming bit stream, allowing full testability of all on-chip states. A similar apparatus could be put in each cell to provide a method for localizing faulty cells and reconfiguring around them. The testing of the combinatorial logic on the chip is slightly more complex due to the arithmetic functions of the chip. A number of simple functional tests are possible. For example, a signal which is completely contained in the signal subspace or its orthogonal complement would produce either an identical output or an output of zero, respectively.

IV. Performance and Specifications

The VLSI implementation of the cellular rotation array is a CMOS chip containing a 5×3 array of rotator cells, each of which performs rotations at 2 MHz, resulting in an overall computational rate of 180 million fractional operations per second (Mfrops). The chip was designed using 2-μm design rules scalable to 1.6, resulting in a die size of 9652×9900 μm at 2-μm technology and 7722×7920 μm at 1.6 μm. After a ten man-week logic design effort, the full-custom design and verification required one designer eight weeks using the design tools of VLSI Technology Inc. on an Apollo DN660 workstation.

V. Conclusions

The original work of this project was the mapping of a signal processing application (the projection operator) through algorithm decomposition (Givens rotation) to architecture (cellular array) to implementation (fabricated and tested custom VLSI chip). As pointed out earlier, the Givens decomposition does not reduce the number of computations, so a VLSI complexity model would not reflect the advantages of this design; however, a cost model that included design time, design reliability, and versatility would more closely match our implementation goals. The novel concepts in this design are the distributed arithmetic of the rotation cell, programmable coefficients, simple methods of loading and testing the contents of latches on the chip, and the very regular, although full-custom, design.

VI. Extensions

Further optimization may include pipelining at the bit level [6], alternate partitioning of the distributed arithmetic [3], and exploitation of the symmetry in the rotation matrix [7]. Future versions of the design may include the ability to perform complex rotations. Clearly, this will require extra partial-product registers and twice as many operations per rotation. Architectural methods using wavefront processing may allow much larger arrays of chips to be assembled without concern for clock skew. In addition, reconfigurability for fault tolerance would be possible without extra buffering because of the asynchronous data-flow nature of wavefront processing.

References

[1] C. P. Rialan and L. L. Scharf, "Cellar architectures for implementing projection operators," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 1619–1627, Nov. 1987.
[2] H. T. Kung, "Why systolic architecture," *Computer*, vol. 15, no. 1, pp. 37–46, Jan. 1982.
[3] A. Peled and B. Liu, "A new hardware realization of digital filters," *IEEE Trans. Acoust. Speech, Signal Processing*, vol. ASSP-22, no. 6, pp. 456–462, Dec. 1974.
[4] G. L. Haviland and H. A. Tuszynski, "A CORDIC arithmetic processor chip," *IEEE Trans. Computers*, vol. C-29, no. 4, pp. 68–79, Feb. 1980.
[5] R. F. Lyon, "Two's complement pipeline multipliers," *IEEE Trans. Commun*, vol. COM-25, pp. 418–425, Apr. 1976.
[6] H. T. Kung and M. S. Lam, "Wafer-scale integration and two-level pipelined implementations of systolic arrays," *J. Parallel and Distributed Computing*, vol. 1, pp. 32–63, 1984.
[7] S. G. Smith and P. B. Denyer, "Efficient bit-serial complex multiplication and sum-of-products computation using distributed arithmetic," in *Proc. IEEE IECEJ-ASJ ICASSP'86* (Tokyo, Japan), Apr. 1986, pp. 2203–2206.
[8] W. M. Gentleman and H. T. Kung, "Matrix triangularization by systolic array," Comp. Sci. Dept., Carnegie-Mellon Univ., Pittsburgh, PA, Aug. 1981.
[9] D. E. Heller and I. C. F. Ipsen, "Systolic networks for orthogonal equivalence transformations and their applications," in *Proc. Conf. Advanced Res. VLSI*, 1982, pp. 113–122.

Response Surface Methodology: A Modeling Tool for Integrated Circuit Designers

JAMES McDONALD, RAJNISH MAINI, MEMBER, IEEE, LOU SPANGLER, AND HARRISON WEED

Abstract —**The following paper discusses an approach to design which allows the designer to statistically analyze a circuit's output characteristics. The results of this analysis give the designer the ability to reduce the sensitivity and to optimize these output characteristics with respect to process, package, and environmental variations. In addition, the paper demonstrates the means by which to establish output specifications before time and money are invested in fabricating devices in silicon. Finally, examples of the successful application of this technique to the design and development of an ECL family are discussed. The methods for applying the results from this statistical technique to manufacturing and testing are also shown.**

I. Introduction

A. Previous Simulation Methodologies

One of the most common approaches to the simulation of circuits accounting for process, package, and environmental variables is the "worst case" approach. Examples of worst-case runs for bipolar integrated circuits include the following:

1) hot temperature, power supply +10 percent, sheet rho—15 percent,
2) low temperature, power supply +10 percent, sheet rho—15 percent,
3) hot temperature, beta = 150.

The previous conditions are simulated to investigate the possibility of current source transistor saturation, input transistor saturation, changes in V_{be}, and power constraints. Through the worst-case simulations the designer is able to observe any degraded performance in delays, edge rates, and high and low output levels. Consequently, the ability of the device to meet specifications under various input conditions may be determined.

There are several weaknesses, however, to the worst-case approach. It cannot provide the information necessary for a reduction in the sensitivity and for an optimization of critical output responses. The simulations mentioned above only analyze discrete points of an output surface; the behavior in the localized areas around these points is unknown. Thus, the resulting design may fail to meet specifications with the slightest change in any of the input parameters.

Similarly to the worst-case approach, designers often experiment by changing only one variable at a time although several significant variables may actually be changing in the process. If these input variables are not acting independently, several *interactions* which determine the behavior of the output response may exist. Consequently, the conclusions based on the "one variable at a time" approach may often be erroneous when used to predict a circuit's response to simultaneous changes of input parameters.

The inadequacies of the worst-case and one-at-a-time simulation methods led to the implementation of a statistical approach to circuit simulation. This approach is called "response surface methodology" (RSM) [1]. It allows one to analyze several depen-

Manuscript received August 26, 1988; revised December 2, 1988.
The authors are with the Semiconductor Product Sector, Motorola Inc., Mesa, AZ 85202.
IEEE Log Number 8826134.