PIPELINED CORDIC ARCHITECTURES FOR FAST VLSI FILTERING AND ARRAY PROCESSING

Ed.F. Deprettere, P. Dewilde and R. Udo
Delft University of Technology
2628 CD Delft, The Netherlands

ABSTRACT

The paper presents a revised functional description of Volder's Coordinate Rotation Digital Computer algorithm (CORDIC), as well as allied VLSI implementable processor architectures. Both pipelined and sequential structures are considered. In the general purpose or multi-function case, pipeline length (number of cycles), function evaluation time and accuracy are all independent of the various executable functions. High regularity and minimality of data-paths, simplicity of control circuits and enhancement of function evaluation speed are ensured, partly by mapping a unified set of micro-operations, and partly by invoking a natural encoding of the angle parameters. The approach benefits the execution speed in array configurations, since it will allow pipelining at the bit level, thereby providing fast VLSI implementations of certain algorithms exhibiting substantial structural pipelining or parallelism.

THE CORDIC ALGORITHM

Although Coordinate Rotation Arithmetic [1],[2] cannot be seen as a generally applicable processing technique, it nonetheless provides a powerful functionality that out performs the conventional multiplier/accumulator approach in many important signal processing applications and closely related matrix equation solving algorithms [3]. Examples are certain orthogonalizing matrix factorizations, [3],[4],[5], orthogonal digital filters [6],[7], and all sorts of square root lattice recursion algorithms [8], to mention but a few.

It is well known, that the CORDIC algorithm is a realization of certain norm preserving plane vector transformations $R_m(\alpha)$, (1), called 'rotations' in circular (m=1), hyperbolic (m=-1) and linear (m=0) coordinate systems:

$$R_m(\alpha) = \begin{bmatrix} \cos\sqrt{m}\,\alpha & \sqrt{m}\sin\sqrt{m}\,\alpha \\ -\frac{1}{\sqrt{m}}\sin\sqrt{m}\,\alpha & \cos\sqrt{m}\,\alpha \end{bmatrix} \qquad (1)$$

through which the transform $u_{p(m)}(x,y)$ of a vector $u_0(x,y)$ is evaluated bit recursively in a sequence of shift and add cycles as follows, for $i=0,1,..,p(m)-1$:

$$u_{i+1}(x,y)=(1-m\epsilon_{m,i}2^{-S_{m,i}})\begin{bmatrix} 1 & m\sigma_i 2^{-S_{m,i}} \\ -\sigma_i 2^{-S_{m,i}} & 1 \end{bmatrix} u_i(x,y), \qquad (2)$$

where:

denoting,

$$\frac{1}{\sqrt{m}}\tan^{-1}\sqrt{m}\,2^{-S_{m,i}} = \alpha_{m,i}, \qquad (3)$$

$$(1)\quad \frac{1}{2}\alpha_{m,i} \le \alpha_{m,i+1} \le \alpha_{m,i}, \qquad (4a)$$

$$(2)\quad \left|\alpha - \sum_{i=0}^{p(m)} \sigma_i \alpha_{m,i}\right| \le \alpha_{m,p(m)}, \qquad (4b)$$

$$(3)\quad \prod_{i=0}^{p(m)} (1-m\epsilon_{m,i}2^{-S_{m,i}}) \simeq (1+\tan^2\sqrt{m}\alpha)^{-1/2}, \qquad (4c)$$

with either $\epsilon_{m,i} = 1$ or $\epsilon_{m,i} = 0$, and either $\sigma_i=1$ or $\sigma_i = -1$. The signature string $<\sigma_i>$ is either computed by resolving the given angle parameter $\alpha : \sigma_i = \text{sign}(\alpha - \sum_{j=0}^{i-1} \sigma_j \alpha_{m,j})$, or is identified with the string $<\text{sign } y_i>$ to build up the argument of $u_0(x,y) : \phi_{u_0} = \alpha = \sum_{j=0}^{p(m)} \sigma_j \alpha_{m,j}$.

Although this algorithm is commonly considered a unified routine embedding various elementary operations, any attempt to translate it into a VLSI architecture almost immediately reveals that it is not realy tailored to an optimal mapping to silicon. Indeed, one is faced with several problems. For example, for the range condition (4b) to be equally in force when m=1 and m=-1 on a given interval, $(-\pi,+\pi)$ say, and with equal accuracy, the function evaluation time will be depending on the system parater m, since the convergence condition (4a) will, then, result in an unequal number of basis angles for both coordinate systems. Conversely, when execution time is required to be function independent, angle reachability and/or accuracy will have to be impaired in at least the hyperbolic system. In either case, however, algorithm control will be cumbersome, difficult to design and area consuming. These shortcomings, as well as others such as the area consuming and speed restraining auxiliary angle accumulator, also rule out almost evidently pipelined

CORDIC architectures that would otherwise provide powerful alternative processing elements, especially in high-through put applications, where conventional PU's tend to be inefficacious, even when implemented in full parallel form.

To overcome these drawbacks, we propose to impose the following algorithmic constraints that will considaribly facilitate straight-forward (automated) designs and lay-outs of both sequential and pipelined multi-function CORDIC architectures.

1. The number of cells or cycles N and the overall execution time T is constant and independent of the various functions that are considered for evaluation in any of the 3 coordinate systems.
2. The coordinate systems m=1 and m=-1 have a common function domain, and angle parameters are reachable with equal accuracy.
3. The norm sealing factors $K_m = (1+\tan^2\sqrt{m}\alpha)^{-1/2}$ are single radix 2 shifts: $K_m = 2^{-S(m)}$.

The first of the constraints implies that neither feed-back nor by-pass is allowed and that the signal propagation time (cycle time) in all pipeline cells (for all AU passes) is T/N and independent of cell (cycle) parameters and indices. Conditions (2) and (3) can be satisfied together by observing that there always exists an ordered set of basis angle parameters $\{\alpha_{m,i} | i=0,1,..,p\}$ obeying

$$\frac{1}{\sqrt{m}} \tan\sqrt{m}\,\alpha_{m,i} = 2^{-S_{m,i}} - \eta_{m,i}\, 2^{-S'_{m,i}}, \qquad (5)$$

where $\eta_{m,i}$ is either 0 or 1 (possibly -1), and $S_{m,i}$ and $S'_{m,i}$ are non-negative integers such that:

$$\text{(i)} \quad \tfrac{1}{2}\alpha_{m,i} \le \alpha_{m,i+1} \le \alpha_{m,i}, \qquad (6a)$$

$$\text{(ii)} \quad \prod_{i=0}^{p} (1+\tan^2\sqrt{m}\alpha_{m,i})^{-1/2} = K_m = 2^{-S(m)}\{1-m\mathcal{O}(\alpha_{m,p})\}, \qquad (6b)$$

(iii) for $m = \pm 1$ and all $\alpha \,|\, |\alpha| \le \pi$:

$$\left|\alpha - \sum_{i=0}^{p} \sigma_i \alpha_{m,i}\right| \le \alpha_{m,p} = \alpha_p \; ; \; \sigma_i = \pm 1. \qquad (6c)$$

Although the scaling condition (6b) is not strictly necessary, any other choice, such as the distributed scaling (4c), must be paid for in terms of hardware and processing time. Indeed, the condition (6b) results in a net saving of adders and pipeline cells (AU cycles), notwithstanding the increase incurred via (5). The resulting sequence of micro operations is summarized in table I for the multi function case. In this option, there are 12 cells (cycles), 5 of which implement (execute) 2 single-shift micro operations. The others are double-shift operations ($\eta_{m,i}$ = 1 for at least one m). The scale factors K_m are 1, 1/2 and 4 (16 bit accuracy) for m=0, m=1 and m=-1 respectively.

From (4b), it is obvious that the signature string $<\sigma_i>$ is a valuable equivalent representation of the angle parameter α. Since in most applications, the numerical value of α is not realy of interest, we propose to encode α by this string. This way, we can omit the angle accumulator that otherwise would be necessary for resolving or building up the angle α, as mentioned before. Moreover, this natural angle

cell (cycle) index	micro operation	$2^{-S_{m,i}} - \eta_{m,i} 2^{-S'_{m,i}}$		
		m=1	m=0	m=-1
1	0	*	2^0	2^0-2^{-3}
2	1	2^0-2^{-5}	2^0	2^0-2^{-2}
3	2	$2^0-2^{-2},(2^{-1}+2^{-2})$	2^0	$2^{-1}-2^{-6}$
4	3	$2^{-1}-2^{-5}$	2^{-1}	$2^{-1}-2^{-3}$
5	4	$2^{-2}-2^{-8}$	2^{-2}	$2^{-2}-2^{-6}$
6	5	2^{-3}	2^{-3}	$2^{-3}-2^{-8}$
7	6	2^{-4}	2^{-4}	$2^{-4}-2^{-9}$
8	7	2^{-5}	2^{-5}	2^{-5}
	8	2^{-6}	2^{-6}	2^{-6}
9	9	2^{-7}	2^{-7}	2^{-7}
	10	2^{-8}	2^{-8}	2^{-8}
10	11	2^{-9}	2^{-9}	2^{-9}
	12	2^{-10}	2^{-10}	2^{-10}
11	13	2^{-11}	2^{-11}	2^{-11}
	14	2^{-12}	2^{-12}	2^{-12}
12	15	2^{-13}	2^{-13}	2^{-13}
	16	2^{-14}	2^{-14}	2^{-14}

Table I Sequence of micro operations and cell (cycle) embeddings. The 1st cell includes a rotation through an angle π/2 when m=1(*).

encoding makes pipelining at the bit level feasible, as is examplified by the array processor configuration depicted in fig. 1.

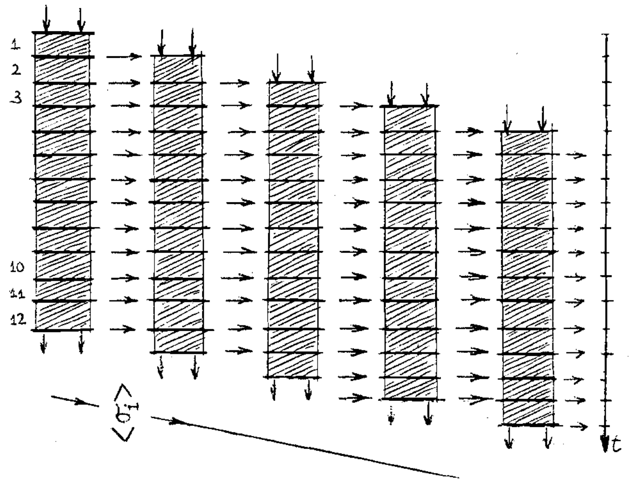

Fig. 1 Array parocessor configuration implementing angle code pipelining at the bit-level.

VLSI IMPLEMENTABLE CORDIC CELLS

Sticking to multi-function pipeline cells and referring to table I, we have to consider 2 types of cells. The x-path portions of these cells are schematically configured in fig. 2a and fig. 2b respectively. The y-path portions are similar mirror images. For the type-2 cell, an alternative configuration can be given that realizes

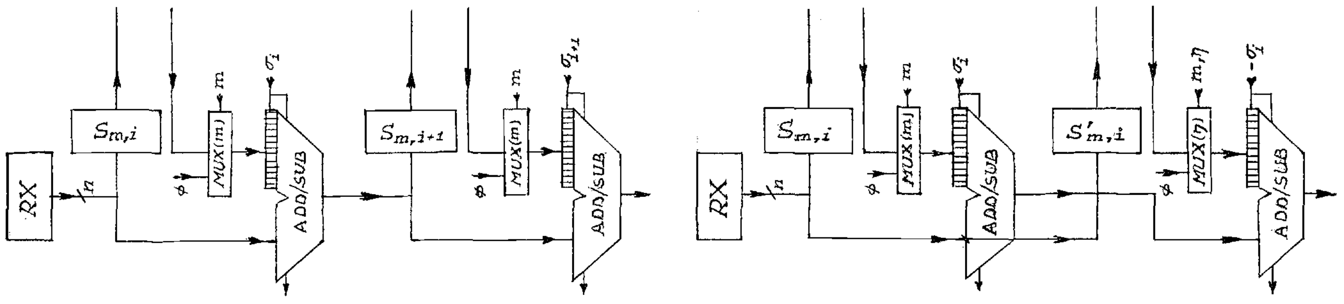

Fig. 2 Pipeline cells, (a): type-1 cell and (b): type 2-cell (x-path only).

$2^{-S_{m,i}} - \eta_{m,i}2^{-S'_{m,i}}$ as $2^{-S_{m,i}}(1-\eta_{m,i}2^{S''_{m,i}})$. Type-1 cell consists of 2 single-shift half cells. Type-2 cell is a double shift full-cell. In the diagrams of fig. 2, RX will be a latch holding the n-bit vector component x in two's complement notation. Boxes labeled $S_{m,i}$ and $S'_{m,i}$ will be wired radix-2 right-shifts.

The ADD/SUB units will be adders with one programmable inverting/non-inverting input. Multiplexers MUX(m) and MUX(η) will pass a 'zero' whenever m=0 and η=0 respectively. When using standard n-bit carry ripple parallel adders composed of full-adder cells (FA) and complementing XOR gates the addition time T_a will be $(n+1)\tau_c$, with τ_c the carry-bit delay in a single FA. The carry in bit depends either on $-\sigma_i(\sigma_i)$ or on $m\sigma_i(-m\sigma_i)$ according to the equations:

$$x:=x+m\sigma_i \text{SHIFT}_i(y) \qquad (7a)$$

$$y:=-\sigma_i \text{ SHIFT}_i(x) + y \qquad (7b)$$

where $\text{SHIFT}_i(*) = (2^{-S_{m,i}} - \eta_{m,i}2^{-S'_{m,i}})(*)$, and for z=x,y , z: denotes 'new z' wihtout loss of z. σ_i is either known or is the sign of y.

Recalling that the signal propagation time in the cells is T/N, where T and N are the (constant) function evaluation time and number of cells respectively, it will be clear that the operation rate is maximized if T_a = T/N. However, there are 2 additions within a signle cell, and an additional delay of $(S_{m,i+1}+2)\tau_c$ in the type-1 cell and of $(S_{m,i}+2)\tau_c$ in the alternative type-2 cell (not shown) is incurred due to the intermediate right-shift. Several measures can be taken to eliminate this cell dependency. One of these is represented in fig. 3 for the type-1 cell, where the expression

$$x[0:n-1]:=x[0:n-1]\pm 2^{-S_i}y[0:n-1] \qquad (8)$$

is decomposed into the following concurrent expressions:

$$(1)\ x[0:n-2-S_{i+1}]:= x[0:n-2-S_{i+1}]\pm y[S_{i+1}:n-2], \qquad (9a)$$

$$(2)\ x[n-2-S_{i+1}:n-1]: = x[n-2-S_{i+1}+1:n-1]+ c \qquad (9b)$$

where c is 0,1 and-1.

The 3 outcomes (9b) are all evaluated in parallel with the regular addition (9a). The bit level complexity is roughly 1 FA(actually 2 independent

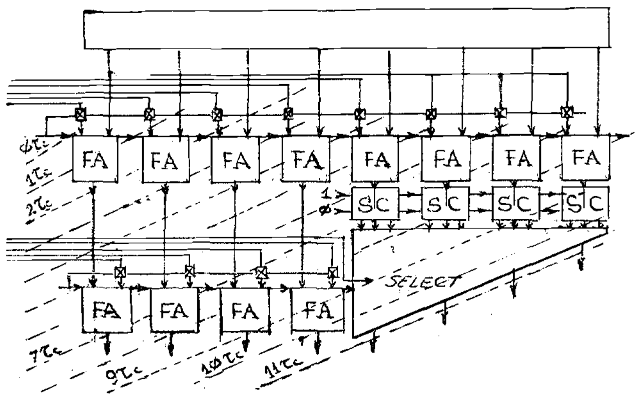

Fig. 3 Logic diagram for the type-1 cell, (n=8 and S_i=2).

half-adder cells denoted 'SC' in fig. 3). The gating circuit selects the actual exact outcome from (9b) when y[n-1] and the carry out $c_{n-2-S_{i+1}+1}$ become availabel in good time. By doing so, the signal propagation time is independent of the cell parameter and indices, and equals $(n+3)\tau_c$. Although the upper most adder in fig. 3 does not exhibit a similar decompostion, we could of course have done so to provide stringent similarity of half-cells, thereby facilitating eventually automated parametrized designs and lay-outs. The proposed solution to overcome cell-dependent propagation time is generally applicable in type-2 cells (the alternative configuration), but not so in type-1 cells. The reason is that in these cells the carry in bits of the 2nd adders may be depending on the sign bit y[n-1] of the output of the upper most cell-adder in the y-path. This sign bit is generallly not available when the lower most adders should have stable carry in bits, i.e., at time $(S_{m,i+1}+2)\tau_c$. A straight forward way to tackle this problem is to implement the expression $x[0:n-2-S_{i+1}]:=x[0:n-2-S_{i+1}]+\sigma_{i+1}y[S_{i+1}:n-2]$, $\sigma_{i+1}=\pm 1$, (and similarly for the y-path) as 2 concurrent expressions $x[0:n-2-S_{i+1}]:=\pm y[S_{i+1}:n-2]+x[0:n-2-S_{i+1}]$ and gating the correct outcome when the sign of the

of the y-entry becomes available. The bit level complexity of the logic realization is roughly 2 FA cells. However, it can be shown [9] that the duplication of these adder fractions can also be advantageously exploited for dynamic quantization control (magnitude truncation), to ensure a numerically stable, i.e., contractive, behaviour of in particular orthogonal algorithms [6],[7] that are subjected to quantization feedback. To close this section we want te make a remark on non-pipelined architectures. Most of the ideas presented in the paper can also be employed in sequential CORDIC architectures. This is certainly of interest since for certain applications it might be preferable to choose for realization structures in the form of parallel configurations of sequential CORDIC PE's rather than in the form of a collection of pipelined PE's.
A possible semi-parallel architecture is shown in fig. 4, where 2 AU passes are required for both a single double-shift micro operations ($\eta_{m,i}=1$) and 2 single-shift micro operations ($S_{m,i}$ and $S_{m,i+1}$, when $\eta_{m,i} = 0 = \eta_{m,i+1}$, see table I). Magnitude truncation control of the very last cycle and the terminating scaling (K_m) can be jointly performed in an additional single pass through the arithmetic unit.

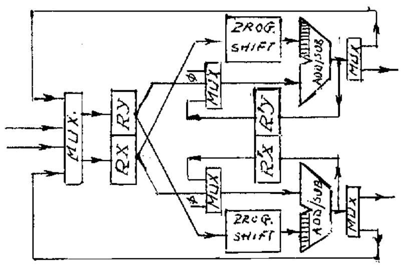

Fig. 4 Semi-parallel non-pipelined CORDIC architecture.

SPECIAL PURPOSE ARCHITECTURES

Table I specifies the micro operations in a CORDIC processor wiht 16-bit word length angle parameters in $[-\pi, +\pi]$. Parameter accuracy can, of course, be increased, but it can also be reduced. Thus in specific applications involving algorithms that are implemented in rotation arithmetic and that exhibit low sensitivity with respect to parameter changes, the rotor angles can be quantized, i.e., the CORDIC pipeline length (number of cycles) can be reduced. Table II shows 3 special designs with reduced number of cells (cycles).
The corresponding regular sequence of micro operations is represented in column 1 of the table. In all 4 cases, only circular rotations can be evaluated. Denoting $\mathrm{SHIFT}_i = 2^{-S_i} - \eta_i s^{-S_i'}$, it turns out that the micro operations embedded in cells 7-10, except for $\mathrm{SHIFT}_i = 2^{-7}$, do not have any influence on the value of the scaling constant $K_1 = \prod_i(1-\epsilon_i \mathrm{SHIFT}_i)$ which equals 2^{-1} with 15 bit accuracy. Thus a simple straight forward reduction amounts to the omission of 1 or more cells (cycles) beyond the 7-th in column 1 of table II. However, if one wishes to further reduce the set of reachable angle parameter values, then the parameters S_i, S_i' and η_i associated with the various micro operations will generally be a function

cell index	$2^{-S_i} - \eta_i 2^{-S_i'}$			
1	*	*	*	*
	2^0-2^{-5}	2^0-2^{-5}	2^0-2^{-5}	2^0-2^{-5}
2	2^0-2^{-2}	2^0-2^{-1}	2^0-2^{-2}	2^0-2^{-2}
3	$2^{-1}-2^{-5}$	$2^{-1}-2^{-9}$	$2^{-1}-2^{-5}$	$2^{-1}-2^{-5}$
4	$2^{-2}-2^{-8}$	$2^{-2}-2^{-9}$	$2^{-2}-2^{-9}$	$2^{-2}-2^{-10}$
5	2^{-3}	$2^{-3}-2^{-9}$	$2^{-3}-2^{-9}$	2^{-3}
	2^{-4}			2^{-4}
6	2^{-5}	$2^{-4}-2^{-9}$	$2^{-4}-2^{-10}$	
	2^{-6}			
7	2^{-7}	$2^{-5}-2^{-9}$	2^{-5}	
	2^{-8}			
8	2^{-9}	2^{-6}		
	2^{-10}			
9/10	$2^{-11}/2^{-14}$			

Table II. Special purpose micro operation sequences. The first cell includes a rotation through an angle $\pi/2$ (*).

of the actual number of micro operations, hence of cells or cycles. See the examples in table II. Notice that for all 4 sequences the scaling factor is 2^{-1} with full 15-bit accuracy. For an automated parametrized lay-out generator, the 3 sorts of PE's, namely 'regular', 'reduced' and 'short' are built with the same ease and optimally tunable to any particular application specification.
'Short' pipelines may for example be used to realize digital orthogonal filters [6]. Thus if $z=e^{j\theta}$ and if $H(z;\Theta)$ is the z-transform domain transfer function of such a filter, whereby Θ is the angle parameter vector, then one can determine a substitute angle vector $\hat{\Theta}$ that minimizes, for a suitably chosen norm $||.||$, the distance $||H(z;\hat{\Theta})-H(z;\Theta)||$, subjected to the condition that all parameters in $\hat{\Theta}$ should be reachable in a reduced-length CORDIC sequence.

REFERENCES

1. Volder, J.E., IRE Trans. Electr. Comp., EC-8/3, 330-334 (1956).
2. Walter, J.S., Spring Joint Computer Conf., AFIPS Conf. Proc., 38, 379-385 (1971).
3. Ahmed, H.M. et.al., Computer, 15/1, 65-82 (1982).
4. Kung, S.Y. and Y.H. Hu, IEEE Trans. Acoustics, Speech and Signal Processing, ASSP-31/1, 66-75 (1983).
5. Lev Ari, H., Information Systems Lab., Dept. of EE., Stanford Univ., Stanford, CA 94305.
6. Deprettere, E., Proc. ICASSP'83, 1, 217-220 (1983).
7. Rao, S.K. and T. Kailath, Information Systems Lab. Dept. of EE, Stanford Univ., Stanford, CA 94305.
8. Friedlander, B., IEEE Trans. on Acoustics, Speech and Signal Processing, ASSP-30/6, 920-930 (1982).
9. Udo, R., and E. Deprettere, Network Theory Section, Dept. of EE, Delft Univ. of Techn., 2628 CD Delft, The Netherlands.