FLAT CORDIC: A Unified Architecture for high-speed generation of Trigonometric and Hyperbolic functions

Bimal Gisuthan and TSrikanthan
Center for High Performance Embedded Systems
Nanyang Technological University
Singapore-639798

***Abstract*- The significant advances in VLSI technology provided the impetus for porting algorithms into architectures. The CORDIC algorithm reigned supreme in this regard due to its canny ability to decimate trigonometric and hyperbolic functions with simple shift and add operations. Despite further refinements of the algorithm with the introduction of redundant arithmetic and higher radix CORDIC techniques, in terms of circuit latency and performance, the iterative nature remains to be the major bottleneck for further optimization. Although several techniques have been proposed to minimize this drawback, a technique known as flat CORDIC aims to eliminate it completely. In flat CORDIC, the conventional X and Y recurrences are successively substituted to express the final vectors in terms of the initial vectors. This results in a single equation to compute the complex trigonometric and hyperbolic functions. In this paper, the techniques devised for the VLSI efficient implementation of a 16-bit unified flat CORDIC architecture are presented. The 16-bit architecture has been synthesized using 0.35μ CMOS process library. Finally, a detailed comparison with other major contributions show that the flat CORDIC based sine-cosine generators are, on an average, 30% faster with a significant 30% saving in silicon area.**

I. THE CO-ORDINATE ROTATION DIGITAL COMPUTER (CORDIC)

CORDIC (Coordinate Rotation Digital Computer) has received much attention since it can be used to compute a wide variety of arithmetic functions for various applications using simple hardware components, mainly shifters and adders. The rotation of a vector over a desired angle is carried out by a series of micro-rotations in an iterative manner [1-4]. Flat CORDIC is a total transformation of the conventional CORDIC into a parallelised version. By successive substitution of the CORDIC equations, it is possible to express the final equation in terms of the original inputs. This form of CORDIC achieves high gains in speed.

II. THE GENERALISED FLAT CORDIC EQUATION

In an N-bit Flat CORDIC, the final values of X_N and Y_N are expressed in terms of X_0 and Y_0 for the parallel implementation of the CORDIC algorithm. This is achieved by successive substitutions of X_i and Y_i in the basic CORDIC equations. The final equations of the flat CORDIC can be derived from the basic CORDIC equations as follows:

CORDIC algorithm for the rotation mode in circular co-ordinate system is characterised by the following basic equations:

$$X_{i+1} = X_i - s_i m Y_i 2^{-i}$$

$$Y_{i+1} = Y_i + s_i X_i 2^{-i}$$

where, X_i and Y_i represent the magnitude of X and Y co-ordinates respectively and s_i represents the signed digit which determines the polarity of $(i+1)^{th}$ iteration. The variable 'm' is the co-ordinate parameter, which has a value of '+1' for circular mode and '-1' for hyperbolic mode of operation.

Elaborating the equations for the $(i)^{th}$ iteration

$$X_{i+2} = X_{i+1} - s_{i+1} m Y_{i+1} 2^{-(i+1)}$$

$$Y_{i+2} = Y_{i+1} + s_{i+1} X_{i+1} 2^{-(i+1)}$$

Substituting the values of X_{i+1} and Y_{i+1} into the above equation,

$$X_{i+2} = X_i (1 - m s_i s_{i+1} 2^{-i} 2^{-(i+1)}) - m Y_i (s_i 2^{-i} + s_{i+1} 2^{-(i+1)})$$

$$Y_{i+2} = Y_i (1 - m s_i s_{i+1} 2^{-i} 2^{-(i+1)}) + X_i (s_i 2^{-i} + s_{i+1} 2^{-(i+1)})$$

Proceeding in this manner by expressing the value of X and Y after N iterations in terms of the input values of X and Y co-ordinates, a generalised equation for Flat-CORDIC can be easily derived. Since $X_0=1$ and $Y_0=0$ for sine/cosine computations, the generalised Flat CORDIC equations can be expressed as

$$X_N = \left[1 - m \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} s_i s_j 2^{-i} 2^{-j} + \sum_{i=1}^{N-3} \sum_{j=i+1}^{N-2} \sum_{k=j+1}^{N-1} \sum_{l=k+1}^{N} s_i s_j s_k s_l 2^{-i} 2^{-j} 2^{-k} 2^{-l} \right] \quad (1)$$

$$\left. \ldots\ldots + \left\{ \sum_{i=1}^{1} \sum_{j=i+1}^{2} \sum_{k=j+1}^{3} \ldots\ldots \sum_{r=q+1}^{N-2} \sum_{s=r+1}^{N-1} \sum_{t=s+1}^{N} s_i s_j s_k \ldots s_t 2^{-i} 2^{-j} 2^{-k} \ldots 2^{-t} \right\} \right]$$

$$Y_N = \left(\sum_{i=1}^{N} s_i 2^{-i} - m \times \sum_{i=1}^{N-2} \sum_{j=i+1}^{N-1} \sum_{k=j+1}^{N} s_i s_j s_k 2^{-i} 2^{-j} 2^{-k} \right. \quad (2)$$

$$+ \sum_{i=1}^{N-4} \sum_{j=i+1}^{N-3} \sum_{k=j+1}^{N-2} \sum_{l=k+1}^{N-1} \sum_{m=l+1}^{N} s_i s_j s_k s_l s_m 2^{-i} 2^{-j} 2^{-k} 2^{-l} 2^{-m}$$

$$\left. \ldots\ldots + \left\{ \sum_{i=1}^{2} \sum_{j=i+1}^{3} \sum_{k=j+1}^{4} \ldots \sum_{r=q+1}^{N-1} \sum_{s=r+1}^{N} s_i s_j \ldots s_s 2^{-i} 2^{-j} \ldots 2^{-s} \right\} \right)$$

The equations demonstrate a complete parallelisation of the conventional CORDIC algorithm, which is iterative in nature. The most significant aspect about the generalised

Proc. 43rd IEEE Midwest Symp. on Circuits and Systems, Lansing MI, Aug 8-11, 2000
0-7803-6475-9/00/$10.00 ©IEEE 2000

equations of the flat CORDIC is that the polarity of micro-rotations (also referred to as signed digits in this paper) needs to be precomputed. The N signed digits ($s_1, s_2, s_3, \ldots, s_{N-2}, s_{N-1}, s_N$), for N-bit accuracy, represent the polarity of N micro-rotations required to achieve the target angle. They can be derived from the individual bits of the N-bit binary representation of input angle, in (1, 0) format, by converting it into the signed binary number representation (SBNR) in (1, -1) format. This format for the polarity of micro-rotations is required to keep the final scaling factor constant. Once the signed digits are precomputed, the evaluation of the equation involves the summation of the positional valued products of the signed digits in different combinations. Each term consists of two parts namely the signed digit combination part and the positional value part as shown in Figure 1.

The cumulative index value (also referred to as channel number in this paper) of a term is defined as the sum of all the negative indices associated with it. For example, the term "$s_1s_2s_3\ 2^{-1}2^{-2}2^{-3}$", has a cumulative index value of 6 and a positional value of 2^{-6}. The unravelling process as shown in Equations 1 and 2 produces $2^{(N-1)}$ terms for both X_N and Y_N. But the striking aspect of this equation is that not all terms are required for the computation of the functions. It is noteworthy that all the terms whose cumulative index values are greater than a certain value, E, will not affect the accuracy of computation (where, E is the internal accuracy required for obtaining the N bit external accuracy). This value of E is typically N + log N + 2 in conventional CORDIC [2].

The value of E determines the overall accuracy that can be achieved in a hardware realisation. The value of E is fixed based on exhaustive error analysis of the equations and 2, and it was seen that the desired external accuracy can be achieved when E = N + log N, provided an error compensation factor is considered while computing the value of X_N and Y_N.

So for 16-bit Flat-CORDIC, the generalised Flat CORDIC equation with E = 20 is given as,

$$x_{16} = \begin{bmatrix} 1-\{\ m\times(s_1s_2 2^{-1}2^{-2} - \ldots - s_1s_{23}2^{-1}2^{-23} \\ -\ s_2s_3 2^{-2}2^{-3} - \ldots - s_9s_{10}2^{-9}2^{-10}) \\ +\ (s_1s_2s_3s_4 2^{-1}2^{-2}2^{-3}2^{-4} \\ +\ldots + s_2s_3s_4s_5 2^{-2}2^{-3}2^{-4}2^{-5} \\ \quad +\ldots + s_3s_4s_6s_7 2^{-3}2^{-4}2^{-6}2^{-7}) + E_{C-X}\ \} \end{bmatrix} \quad (3)$$

$$y_{16} = \begin{bmatrix} s_1 2^{-1} + s_2 2^{-2} + \ldots + s_{16}2^{-16} \\ -\ m\times(s_1s_2s_3 2^{-1}2^{-2}2^{-3} - \ldots - s_5s_7s_8 2^{-5}2^{-7}2^{-8}) \\ +\ (s_1s_2s_3s_4s_5 2^{-1}2^{-2}2^{-3}2^{-4}2^{-5} \\ +\ldots + s_2s_3s_4s_5s_6 2^{-2}2^{-3}2^{-4}2^{-5}2^{-6}) + E_{C-Y} \end{bmatrix} \quad (4)$$

where, E_{C-X} and E_{C-Y} are the error compensation factors in X_{16} and Y_{16} respectively.

The equations 3 and 4 can be expressed in a suitable way to make it simpler for hardware implementation. The signed terms (polarity of micro-rotations) are segregated on the basis of the their positional values. For X_{16}, the equation is expressed as the summation of 18 channels (e.g., channel 3, corresponding to a positional value of 2^{-3} to channel 20, corresponding to a positional value of 2^{-20}). Therefore X_{16} can be expressed as

$$X_{16} = [1 - \{\ 2^{-3}(m\times\Sigma\ s_i\ .\ s_j\ |_{i+j=3} - \Sigma\ s_i\ .\ s_j\ .\ s_k\ .\ s_m\ |_{i+j+k+m=3}\ ) + 2^{-4}(m\times\Sigma\ s_i\ .\ s_j\ |_{i+j=4} - \Sigma\ s_i\ .\ s_j\ .\ s_k\ .\ s_m|_{i+j+k+1m=4}\ ) + 2^{-5}(m\times\Sigma\ s_i\ .\ s_{j\,|i+j=5} - \Sigma\ s_i\ .\ s_j\ .\ s_k\ .\ s_m|_{i+j+k+m=5}\ ) + \ldots\ \ldots\ \ldots\ \ldots\ \ldots\ \ldots\ \ldots\ \ldots + 2^{-20}(m\times\Sigma\ s_i\ .\ s_{j\,|i+j=20} - \Sigma\ s_i\ .\ s_j\ .\ s_k\ .\ s_l\ |_{i+j+k+m=20}\ ) + E_{H-X}\}\ ]$$

where i varies from 1 to 16 and j varies from i+1 to 16, k varies from j+1 to 16 and m varies from k+1 to 16

Similarly, Y_6 can be expressed as the summation of 20 positional channels (channel 1, corresponding to a positional value of 2^1, to channel 20, corresponding to a positional value of 2^{-20}.

The signed terms can be reduced to be either '1' or '-1' with the help of a combiner, which comprises of mainly simple XOR gates. Once the signed terms are generated, the evaluation of the Equations 3 and 4 involves simply the summation of '1's and '-1's appropriately. The values of X_N and Y_N have to be scaled with an appropriate constant scaling factor to obtain the values of cosine/sine or cosh/sinh of the input angle respectively. The basic architecture for the evaluation of the generalised Flat-CORDIC equations is shown in Figure 2.

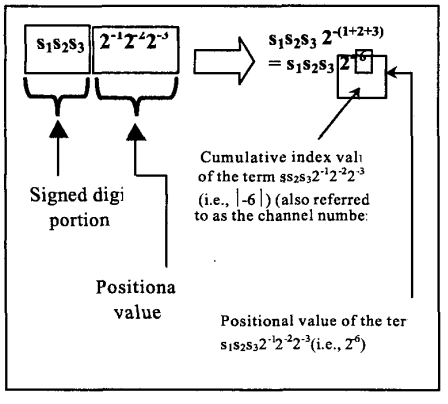

Fig. 1. Partitioning the signed term

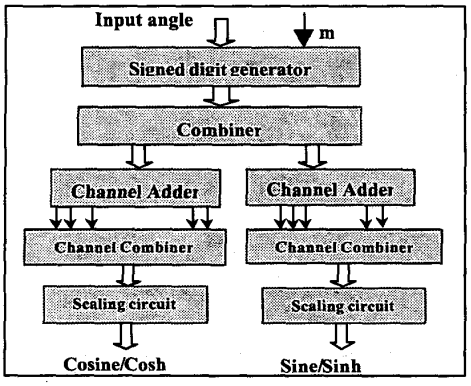

Fig. 2. Basic Flat CORDIC Architecture

The basic architecture of Flat–CORDIC, shown in Figure 2, consists of 5 modules, namely the signed digit generator for pre-computation of the signed digits, a combiner module for generating the signed terms of the final equation, the channel adder arrays for adding up the terms in individual channels, the channel combiner for adding up the individual channel sums and a circuitry to achieve scaling factor compensation. The error compensation factors (i.e., E_X and E_{C-Y}) can be incorporated within either the channel combiner or the scaling factor compensation modules. In subsequent section of this paper, the basic design of a 16-bit sine-cosine generator is discussed.

A. The Precomputation Unit

The Splitdecompositon algorithm presented in Bimal et al is used for precomputing the signed digit or the polarity of micro-rotations. For the hyperbolic mode, the iteration set is different from that used in the rotation mode. In general iterations [4,13,40, 121... k, 3k+1] must be repeated as $\tanh^{-1}2^{-i} > 2^{-i}$ [1]. The iteration set for the 16-bit hyperbolic mode must therefore be chosen as [1,2,3,4,4,5,6,7,8... 13,13... 16]. However, for the precomputation algorithm to be applied, the sixth iteration needs to be repeated as well. The thirteenth iteration need not therefore be repeated as the resulting error will be compensated by the sixth iteration. Hence, the final iteration set for the unified 16-bit flat CORDIC is [1,2,3,4,4,5,6,6,7,8... 16]. More specific details regarding the precomputation of the sign digits are presented in [5]

B. The Combiner Unit

From the generalized equations for the unified flat CORDIC,

X_{16} = (1 - m × (sum of 2's combinations of signed digits) + (sum of 4's combinations of signed digits))= (1 - m [(sum of 2's combinations of signed digits) - m(sum of 4's combinations of signed digits)])

Y_{16} = (sum of the signed digits - m (sum of 3's combinations of signed digits) + ((sum of 5's combinations of signed digits))

Hence, for the X_{16} part, only the 4's combinations of the signed digits are affected by the mode parameter while only 3's combinations of the signed digits are affected for the Y_{16} part. For the 4's combination terms, a three input XOR gate, with 'm' as one of the inputs, can be used instead of an extra XOR gate. For the 3's combination terms an extra XOR gate is necessary as the 3's combination terms are used to generate the 5's combination terms. However, the maximum time taken by the combiner is still one 3-input XOR gate delay + one 2-input XOR gate delay.

C. The Channel Adder Arrays and Channel Combiner Units

The channel adder arrays computes the values of each channel/group output of the combiner (counts the number of '0's and '1's). The channel combiner combines the outputs of the channel adder arrays according to their cumulative index values. The number of channel adder arrays in cos/cosh and sine/sinh parts is 16 and 19 respectively. The channel adder arrays essentially consists of a CSA adder tree. The channel 20 has the maximum number of terms and takes a maximum time of three full adder delays + three half adder delays + two 5-bit CLA + one 6-bit CLA adder times. The channel combiner is implemented using a 7-level CSA tree followed by a 20-bit CLA. The channel combiner for sin/sinh part is implemented in a similar manner. The next stage in the computation is the scaling factor compensation and this is discussed next.

D. Scaling factor Compensation

A common multiplier was designed to perform the scaling factor compensation for both the circular and hyperbolic modes. Since the scaling factors are constant, multiplexers in conjunction with hardwired shifts can be employed to feed the appropriate values into the various arrays of the multiplier. Four levels of full adders are required to realize CSA tree implementation. Figure 3 shows the complete architecture for the 16-bit Unified flat CORDIC architecture.

III. RESULTS

The 16-bit flat CORDIC function generator, shown in Figure 3, was synthesized using Synopsys design Compiler (version 99.05-4). The area-time measures of the synthesized design are shown in Table 1. A 0.35μ CMOS library was used to synthesize the functionally simulated netlist. The area and delay of the main components of the 16-bit flat CORDIC architecture are shown in Table 1. The architecture is fully scalable and can be extended to higher accuracy as well.

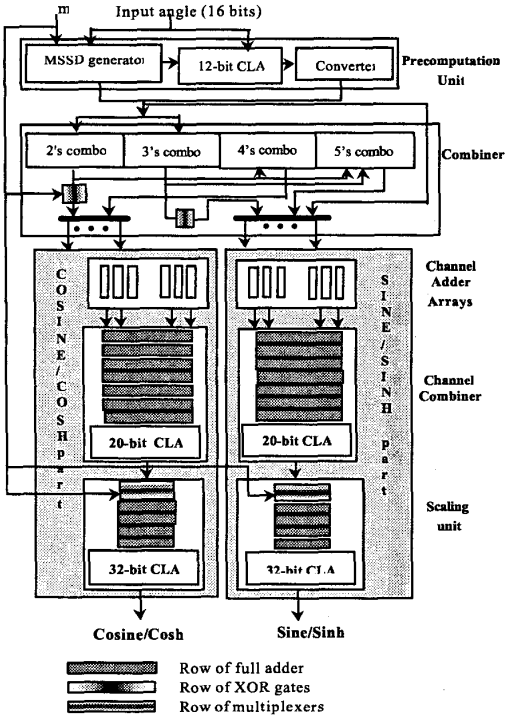

Fig. 3. A Unified16-bit flat CORDIC architecti

Since CSA adders form the predominant component of the flat CORDIC architecture, the overall delay does not rise linearly with accuracy. The relation between the overall delay and the accuracy (in bits) is approximately logarithmic. Table 2 gives the performance comparison of the flat CORDIC architecture with other prominent works in literature. It is evident that flat CORDIC has 30% savings in both area and delay when compared to other CORDIC based sine/cosine generators

III. CONCLUSION

In this paper, a unified flat CORDIC architectyure for generating trigonometric and hyperbolic functions has been presented. The equations for the flat CORDIC have been defined and a generalized architecture has been presented. The salient features of the flat CORDIC architecture have been demonstrated with the help of a 16-bit flat CORDIC. Based on the area-time parameters and the performance comparisons with other CORDIC based sine/cosine generators, it is evident that flat CORDIC sine/cosine generators are on an average 30% faster with a 30% saving on silicon area.

REFERENCES

[1] J. S. Walther, "A Unified Algorithm for Elementary Functions," *Proc. Spring. Joint Computing Conf.*, pp.379-385, 1971.

[2] N. Takagi, T. Asada, and S. Yajima, "Redundant CORDIC Methods with a Constant Scaling Factor for Sine and Cosine Computation,"*IEEE Trans. Computers*, vol.40, no.9, pp.989-995, 1991.

[3] D. Timmermann, H. Hahn, and B. J.Hosticka, "Low Latency Time CORDIC Algorithms,"*IEEE Trans. Computers*, 41(8), pp. 1010-1015, 1992.

[4] H. Dawid and H. Meyr, "The Differential CORDIC Algorithm: Constant Scale Factor Redundant Implementation without Correcting Iterations,"*IEEE Trans. Computers*, 45(3), pp. 307-318, 1996.

[5] Bimal Gisuthan, "A Unified Architecture for Flat CORDIC", M.A.Sc. Thesis, School of Computer Engineering, Nanyang Technological University, Singapore, 2000.

Table 1 Area-time measures for 16-bit Flat-CORDIC

	CELL AREA (1 unit = 52.9 sq. microns)	TIME (ns)
Precomputation Unit	1251	3.25
Combiner Unit	1313	0.41
Channel adder arrays	6322	5.94
Channel Combiner	5687	5.34
Scaling Factor Compensation Unit	2199	2.59

TOTAL CELLAREA	=	16096 units
TOTAL interconnect Area	=	4067 units
TOTALAREA	=	20163 units
CRITICAL PATH TIMING	=	17.53 ns

Table 2 Performance Comparisor

CORDIC Design	Area units in terms of X	Time units (in terms of full adder delay)
Non-redundant CORDIC	13 X	256 $\times T_{FA}$
RedundantCORDIC -Correcting Rotation [4]	33 X	68.5 $\times T_{FA}$
DCORDIC [5]	31 X	57.5 $\times T_{FA}$
Redundant CORDIC-Timmermann[17]	27 X	52 $\times T_{FA}$
Flat CORDIC	18 X	36 $\times T_{FA}$

$X = 5(N + \log_2 N)$ full adder areas
T_{FA} - delay of one full adder