Low Power CORDIC Implementation Using Redundant Number Representation

Christian V. Schimpfle, Sven Simon and Josef A. Nossek
Institute for Network Theory and Circuit Design
Technical University Munich
Arcisstr. 21, 80333 Munich, Germany
Email: chsc@nws.e-technik.tu-muenchen.de

Abstract

In this paper a methodology for reducing the power consumption of shift-and-add operations in general and especially of CORDIC stages is presented. The proposed method uses the fact of simultaneous carry generation in redundant carry-save and signed digit structures to predict the minimum necessary hardware effort for shift-and-add operations. As a carry once generated in a certain bit position cannot "ripple" through the adder if using redundant number representation, hardware parts can be switched on or off depending on the shift constant. Simulations have shown, that shift dependent hardware utilization of parallel implementations leads to monotonically decreasing power consumption for increasing shift constants. A CORDIC processor element for 16 digit SDNR has been implemented as a layout and simulated with PowerMill in terms of power consumption.

1 Introduction

Low power design methodologies at all levels of the synthesis process have been widely investigated in recent years. Although power optimization oriented high-level synthesis is of increasing interest as design complexity and transistor density requires intensive CAD support, optimizations at gate- and device-level are still of importance in order to use the full potential of power reducing design methodologies.

The switching statistics of a circuit are strongly influenced by the number representation used for the implementation [2]. In [3] it is shown that a reduction of switching activity and hence a reduction of the power consumption can be achieved by redundancy reduction in carry-save and signed digit number representation structures. Carry-save and Avenzienis' signed digit structures are well known for efficient high-speed DSP applications. In [4] Privat figures out the close similarity between radix 2 signed digit and carry-save representation. A methodology is presented to use simple full adders for signed digit arithmetic.

In this paper the power consumption of shift-and-add operations used in CORDIC based approximated rotations [1] is investigated. The use of carry-save or signed digit number representation allows to reduce the operational hardware depending on the shift constant. By doing this, unused hardware parts are simply switched off and bypassed. Thus, the overall load capacitance and the dynamic power dissipation is reduced.

This paper is structured as follows: In Section II power considerations of shift-and-add operations with redundant number representations are made. In Section III the architecture of a 16 digit CORDIC processor element with variable shift constant and shift dependent hardware usage is presented. Section IV shows the same methodology applied for a digit serial architecture. Experimental results are given in Section V. Finally in Section VI some conclusions are made.

1063-6862/97 $10.00 © 1997 IEEE

2 Power Considerations

The power dissipation in a static CMOS circuit can be split up in a dynamic and a static part

$$P = P_{dyn} + P_{static}. \tag{1}$$

Static power dissipation is caused by leakage effects and can be neglected when compared to the dynamic component. The dynamic power will be considered in the following to consist of two components: the switching power consumed for charging and discharging capacitances within gates and a shoot-through component caused by the cross current during transition when p- and n-block are both in the conducting stage.

$$P_{dyn} = \alpha(\frac{1}{2}f_c C_L V_{dd}^2 + P_{trans}) \tag{2}$$

The first term in brackets in (2) is the switching component, the second term is the shoot-through component. Factor α is the switching activity, f_c is the clock frequency, C_L denotes the load capacitance and V_{dd} the supply voltage. The dynamic power dissipated in a two stage adder structure at digit position k with registers placed on each input and load capacitors at each output as shown in Fig. 1 is given by:

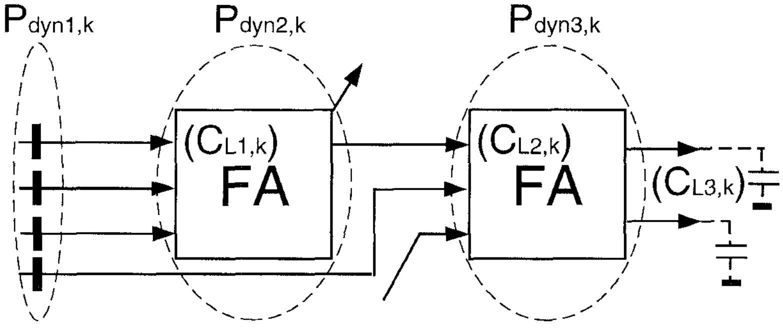

Figure 1: Locations of dynamic power consumption in a two step addition.

$$P_{dynj,k} = \alpha_j(\frac{1}{2}f_c C_{Lj,k} V_{dd}^2 + P_{transj,k}), \tag{3}$$

with $j = 1, 2, 3$.

The dynamic power is splitted up in three stages: the register stage with the dynamic power dissipation $P_{dyn1,k}$, and the first and second adder stage dissipating $P_{dyn2,k}$ and $P_{dyn3,k}$ respectively. If the static power consumption is neglected, the total power consumption at digit position k is the sum of all parts of dynamic power:

$$P_k = \sum_i P_{dyni,k}. \tag{4}$$

Fig. 2 shows a shift-and-add operation with a two stage addition, where word b is shifted two digits to the right:

$$y = a + 2^{-2}b.$$

For carry-save and SDNR structures the free digits due to the shift of b are filled with zeros. This means that the gray shaded hardware parts in the left picture of Fig. 2 are unnecessary and can be left out or switched off and bypassed. Thus, the upper s digits of data word a are directly connected with the outputs of the shift-and-add device. The adders in this positions now cannot cause any dynamic power consumption, only the negligible static power consumption due to leakage currents is not influenced by this method.

The power that will be saved by disabling unnecessary hardware for a right-shift of s digits and a wordlength w is:

$$\begin{aligned} P_{save} = & \sum_{k=w-s}^{w} P_{dyn1,k} + \sum_{k=w-s+1}^{w} P_{dyn2,k} + \\ & \sum_{k=w-s+1}^{w} P_{dyn3,k} - \sum_{k=w-s+1}^{w} \alpha_{1,k}(\frac{1}{2} f_c C_{L3,k} V_{dd}^2 \\ & + P_{trans1,k}) - \alpha_{1,w-s}(\frac{1}{2} f_c C_{L2,w-s} V_{dd}^2 + P_{trans2,w-s}). \end{aligned} \tag{5}$$

In general the proposed method can be applied for every shift-and-add operation with carry-save, SDNR or carry-lookahead arithmetic. One might suggest to apply the method also for two's complement carry-ripple arithmetic when the MSB of the shifted data word b is zero. With some additional control logic it would be possible to detect if a carry at a position i within the upper s bits is zero. For these two conditions ($MSB(b) = 0$ and $c(i) = 0$ with $w - s \leq i < w$) the full adders at positions $> i$ could be switched off. This method could also be extended to array multipliers. But simulations have shown that such an data dependent approach does not lead to power savings. In most cases the control and switch hardware consumes even more power than could be saved by switching off adder hardware. The results show that for a 16 bit parallel carry-ripple shift-and-add operation power savings only could be achieved for shift constants $s > 9$.

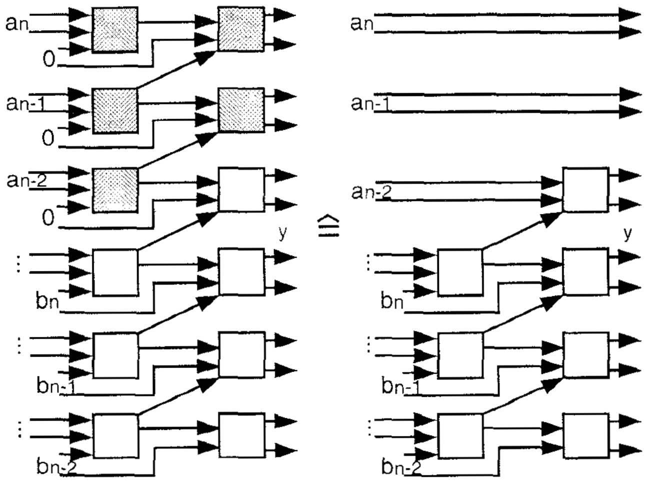

Figure 2: Two stage shift-and-add operation.

The following sections will focus on parallel and digit serial implementations of a 16 digit CORDIC processor element for fixed coefficients using signed digit number representation.

3 Low Power CORDIC Architecture

In [4] Privat shows how a radix 2 SDNR addition can be performed as a two stage addition in the same way as a carry-save addition with simple fulladders. Each digit is represented by the difference of a positive and a negative part, $x = x_+ - x_-$. This is quite similar to the coding scheme of a carry-save structure built up with full adders, where $x = x' + x''$. It becomes clear that the only change that has to be made at the full adders to get a signed digit structure is to insert inverters at all inputs and outputs corresponding to the negative part of a digit. In Fig. 3 an example for both structures, a standard carry-save and the signed digit structure with full adders is given. A 16 bit CORDIC processor element can be built up with a shifter and a subsequent adder section.

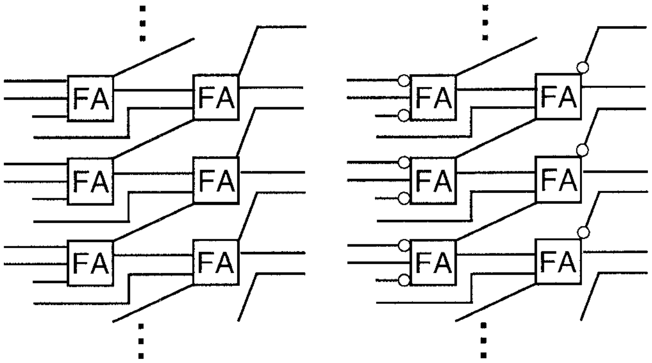

Figure 3: Two step carry-save (left) and signed digit addition.

3.1 Adder Section

Fig. 4 shows the schematic of one complete adder stage with all switches and control signals for SDNR addition. The control signals for the switches are generated in a seperate control unit, which is only active when the shift constant is adjusted. Once this is finished and all switches are in correct position, the control unit is in a static state and cannot consume any dynamic power.

3.2 Shifter Section

The principle of the used shifter is shown in Fig. 5 with an 8-digit example. For simplicity of the diagram, each digit is represented by only one line. The shift constant is BCD-coded. In a 16-digit version shifts can be realized corresponding to multiplications by 2^0 to 2^{-15}. The signals have to pass four switches realized as transmission gates. The propagation delay of the whole shifter loaded with the gate capacitances of the adder section has been simulated and is $0.8ns$ for a $1\mu m$ process. Although in a classical barrel-shifter the signals have to pass only one transmission gate with a delay of only about $0.2ns$ this shifter architecture is a reasonable alternative to the barrel-shifter because of its hardware effectiveness. Where 272 transistors are necessary to built up a barrel-shifter for shifts only in one direction, the proposed alternative only needs 128 transistors. Furthermore, the BCD-coded shift constant has to be decoded into a 1-of-16 signal to control the barrel-shifter.

The signal flow graph (SFG) of the CORDIC processor element is suited for a bit slice architecture as shown in the block diagram of Fig. 6. Again, each digit is represented by only one line in order to simplify the diagram. The shifter is implemented by switch cells which are identical for every bit of the n-bit shift constant and local wire cells [6] which are identical for every bit slice. Global

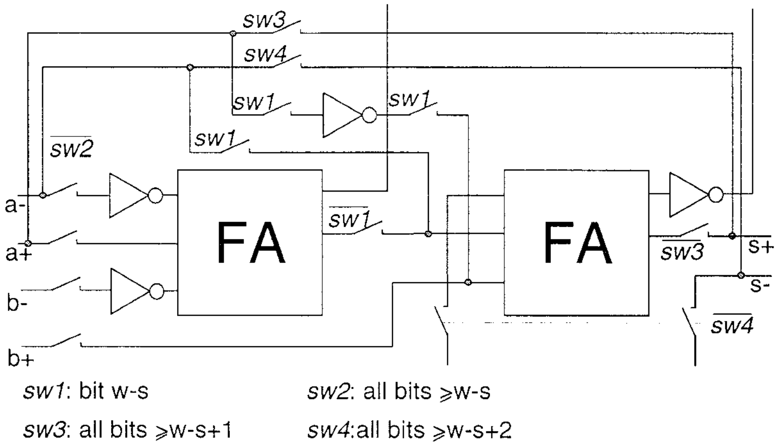

Figure 4: A switchable SDNR adder unit.

interconnections are realized by abutment of the bit slices. Fig. 8 shows the layout of the complete CORDIC processor element.

4 Digit Serial Architecture

Digit serial architectures are achieved by projecting the dependence graph of the parallel architecture along a scheduling vector $\vec{s}$, where nodes of the dependence graph along a straight line parallel to $\vec{s}$ are assigned to one processor element [5]. The whole N-digit SDNR addition is then managed by only one SDNR adder element. Fig. 7 shows the digit serial architecture of a SDNR addition with LSB-first schedule.

5 Experimental Results

The SDNR shift-and-add processor element has been implemented as a layout in $1\mu m$ technology for a 16 digit data format in parallel and digit serial architecture. Both circuits have been simulated with PowerMill in terms of power consumption. Table 1 shows the results for the parallel and the serial architecture both realizing the equation $y = a + 2^{-s}b$ with $s = 1, ..., 7$. Each value given in Tab. 1 is the power consumed during one complete sample, this means during one clock period of the parallel and 16 clock periods of the digit serial architecture. For each simulation 100 equally distributed test samples were used. The column "P unsw." and "P sw." include the results for the implementations where unnecessary hardware parts are not switched off and switched off respectively. The higher total power consumption of the digit serial circuit results from the additionally necessary registers to realize the shifts and to store the carrys of the previous additions, carried out by the registers in the adder of Fig. 7. Note that the power savings of the digit serial variant become smaller for increasing shift constants. This is because the percentage of power consumed in the adder to the total power consumption decreases due to the increasing amount of registers necessary to implement the shift constants. Hence, the percentage of power that can be saved by switching off the adder becomes smaller.

CORDIC impl.	s	P unsw. (mW)	P sw. (mW)	Power savings (%)
parallel	1	1.728	1.709	1.1
	2	1.683	1.616	4.0
	3	1.631	1.522	6.7
	4	1.506	1.371	9.0
	5	1.501	1.270	15.3
	6	1.425	1.129	20.7
	7	1.379	0.920	32.3
serial	1	4.976	4.683	6.8
	2	5.496	5.132	6.6
	3	5.979	5.617	6.1
	4	6.532	6.138	6.0
	5	7.192	6.771	5.8
	6	7.866	7.413	5.7
	7	8.591	8.118	5.5

Table 1: Power consumption in a 16 bit CORDIC prozessor element for parallel and digit serial implementation.

6 Conclusion

Redundant number representations allow to reduce the hardware for shift-and-add operations. This results in lower switching power consumption. In this paper a CORDIC processor element with signed digit number representaion is presented where hardware, unnecessary due to shifting, can be switched off. Simulation results for parallel and digit serial implementations of the CORDIC show, that significant power savings are possible with this method. A 16 digit parallel CORDIC processor element has been realized as a layout using a bit slice architecture.

References

[1] J. Götze and G. J. Hekstra: An Algorithm and Architecture based on Orthonormal μ–Rotations for Computing the EVD. *INTEGRATION. Sp. Iss. on Parallel Algorithms and Architectures, vol. 20, pp. 21-39, 1995.*

[2] A. P. Chandrakasan, R. Allmon, A. Stratakos, R. W. Brodersen: Design of Portable Systems. *Proc. CICC'94, San Diego, pp. 12.1.1.-12.1.8.*

[3] A. E. Schlegel and T. G. Noll: Switching Activity Optimization in Digital CMOS Circuits. *Proc. ISLPD 1996, Monterey.*

[4] G. Privat: A Novel Class of Serial-Parallel Redundant Signed-Digit Multipliers. *Proc. ISCAS 1990, New Orleans, pp. 2116-2119.*

[5] S. Y. Kung: VLSI Array Processing. *Prentice Hall, 1988.*

[6] S. Simon,P. Rieder,C. Schimpfle and J. A. Nossek: CORDIC-based Architectures for the Efficient Implementation of Discrete Wavelet Transforms. *Proc. ISCAS 1996, Atlanta, pp. 77-80.*

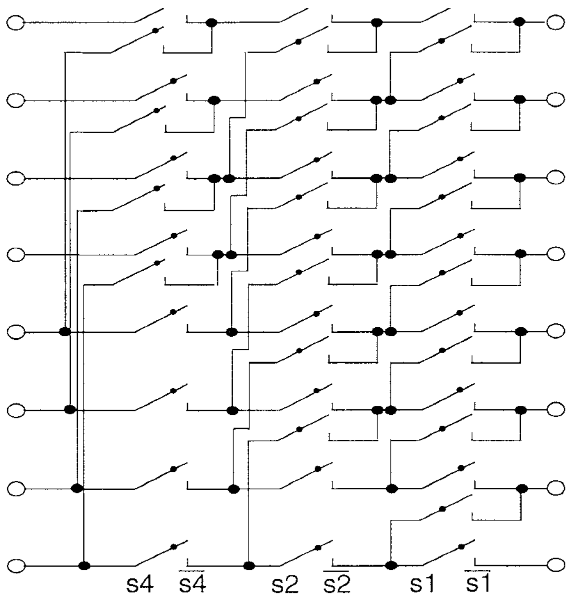

Figure 5: Shifter for shifts $s = 1, 2, ..., 7$.

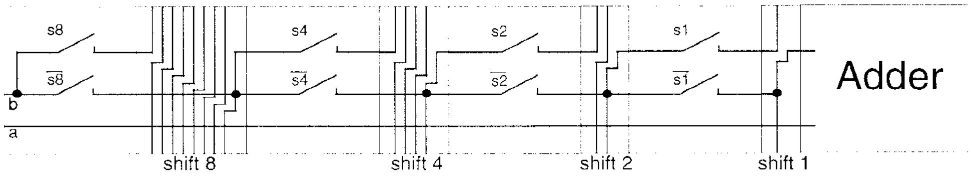

Figure 6: Block diagram for one bit slice of the SDNR shift-and-add PE.

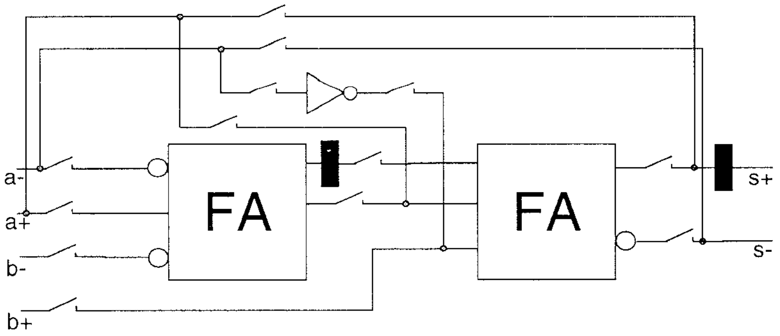

Figure 7: LSB-first digit serial SDNR addition.

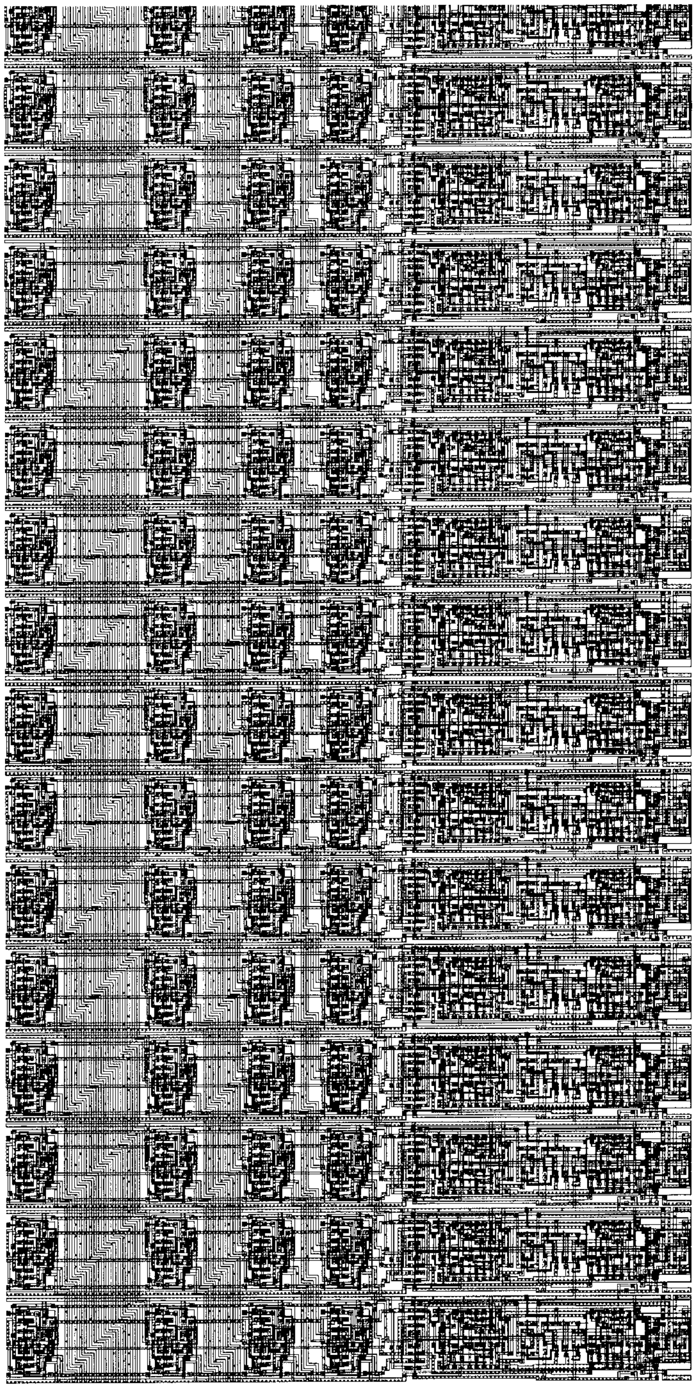

Figure 8: Layout of the 16 digit low power CORDIC processor element on a chip area of $0.37mm^2$.