1999年度 大韓電子工學會 秋季綜合學術大會論文集 第22卷 第2號 99/11

고속 CORDIC에 기반한 직접 디지털 주파수 합성기

박민경, 최성수, 김기선, 이정아
광주과학기술원 정보통신공학과
전화 : (062) 970-2252 / 팩스 : (062) 970-2204

Hight throughput CORDIC-based Direct Digital Frequency Synthesizer

Minkyoung Park, Sungsoo Choi, Kiseon Kim, and Jeong-A Lee
Dept. of Information and Communications, Kwang-Ju Institute of Science and Technology
E-mail : mkpark@kjist.ac.kr

Abstract

This paper describes a direct digital frequency synthesizer using the CORDIC algorithm, which can be implemented efficiently for a digital sinusoid synthesis. To optimize the hardware design parameters, we perform numerical analysis of the quantization effects for the CORDIC-based architecture. A pipelined architecture is employed to obtain a high data throughput,. We estimate and summarize its hardware costs for a variable accuracy, and a CORDIC-based architecture for 9 bit accuracy is emulated in FPGA.

I. Introduction

High performance Direct Digital Frequency Synthesizer(DDFS) plays an extremely important role in modern digital communications. They offer many advantages including fast continuous phase switching response, fine frequency resolution and large bandwidth[1]. A typical architecture used for DDFS, originally introduced by Tinery *et al.*, which generates sine values from a Read Only Memory(ROM) lookup table[1]. However the size of ROM lookup table is approximately doubled for obtaining 1-bit finer frequency resolution.

A different approach of phase-to-amplitude mapping is the COordinate Rotation DIgital Computer(CORDIC) algorithm[2], which is hardware attractive by using a unified shift and add approach. Since the basic aim of the CORDIC algorithm is to compute the trigonometric functions, the CORDIC can be used efficiently for an implementation of the DDFS. In fact, there are many related works to implement the CORDIC algorithm to generate a sine wave. Gielis *et al.* proved the good performance by implementing a 10-bit polar-to-cartesian converter with a maximum sample rate of 540*MHz* in [3]. Recently, Vankka simulated the CORDIC algorithm and several other techniques for an implementation of the DDFS through the computer program in Matlabs[4].

In this paper, we investigate and design efficiently CORDIC-based DDFS implementation for optimized design parameters in Field Programmable Gate Array(FPGA) implementation.

This paper is organized as follows. Both CORDIC-based architectures are briefly described in Section II. In Section III, to optimize the hardware design parameters, the quantization effects are analyzed numerically. The estimated hardware cost and emulation results of CORDIC-based DDFS are shown in Section IV and V. Finally, conclusion is presented in Section VI.

II. CORDIC-based DDFS

The DDFS is mainly composed of a phase accumulator and a sine function generator. The phase accumulator consists of a j-bit frequency register which stores a digital phase increment followed by a j-bit full adder and a phase register. The digital input phase increment is entered in the frequency register. At each clock, this data is added

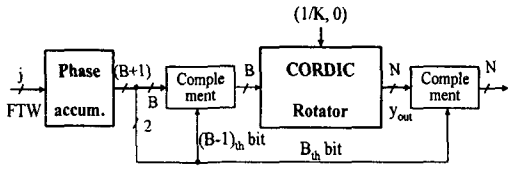

Figure 1. Block Diagram of CORDIC-based DDFS.

to the data previously held in the phase register. The phase accumulator exploits the modulo 2^j overflow property of the j-bit accumulator to generate the phase argument of the sine function generator[1]. Each overflow of the phase accumulator represents one period of a sine wave and the rate of overflows is the output frequency f_{out} :

$$f_{out}=\frac{\Delta\phi f_{clk}}{2^j}, \tag{1}$$

where $\Delta\phi$ is the input phase increment, j is the number of accumulator bits, and f_{clk} is the clock frequency. The input word to the phase accumulator controls the frequency of the generated sine waveform, which is called Frequency Tuning Word(FTW). The minimum frequency resolution corresponds to the case when $\Delta\phi=1$, which is given by

$$\Delta f=\frac{f_{clk}}{2^j}. \tag{2}$$

The DDFS can be implemented efficiently by using the circular rotation mode of the CORDIC algorithm. The relation between the inputs and the outputs of the CORDIC rotation is given by following equations[2] :

$$\begin{aligned} x_o &= K[x_i\cos(\theta-\delta)-y_i\sin(\theta-\delta)] \\ y_o &= K[y_i\cos(\theta-\delta)+x_i\sin(\theta-\delta)] \\ \theta-\delta &= \sum_{i=0}^{n-1}\sigma(i)\arctan(2^{-i}), \end{aligned} \tag{3}$$

where K is a scaling factor, $(\theta-\delta)$ is the total rotated angle after n iterations, $\sigma(i)$ specifies the sign of the target phase, $\theta(i)$, and δ is an approximation error. Thus, if the (x_i, y_i) is given as $(\frac{1}{K}, 0)$, then outputs are given as follows

$$x_o=\cos\theta \qquad y_o=\sin\theta, \tag{4}$$

which corresponds to the pre-scaled CORDIC rotator. As a matter of fact, the post-scaled CORDIC rotator has a better error performance than the pre-scaled, considering that the same input enters into the CORDIC rotator[5]. However, for the DDFS application, the pre-scaling corresponds to a constant input of $\frac{1}{K}$, while the post-scaling is a

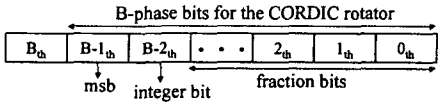

Figure 2. The output word of the phase accumulator and 2's complement representation of the phase for the CORDIC roator.

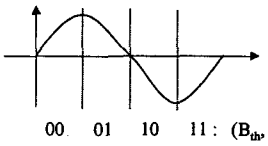

Figure 3. Decoding rule of each quadrant of a sine wave.

division operation of outputs by K. Hence, the pre-scaling shows much simpler architecture at the cost of small degradation of performance[5].

Since the CORDIC algorithm has the restricted Region of Convergence(ROC) as $[-\frac{\pi}{2}, \frac{\pi}{2}]$, two-phase bits are used to generate the full sine wave. The architecture of CORDIC-based DDFS is shown in Figure 1, which is composed of a phase accumulator and the CORDIC rotator with two complementors. The phase accumulator word length j is truncated into $(B+1)$ bits, whose B least significant bits are used as the rotated angle and provide the required phase precision to the CORDIC rotator. Figure 2 shows the output word from the phase accumulator and the number representation used for the phase input of the CORDIC rotator. The output word length of the CORDIC-based DDFS is N bits. The front complementor is to invert the phase and the end complementor is to invert the output of the CORDIC rotator. These complementors are controlled by two-phase bits, B_{th} bit and $(B-1)_{th}$ bit shown in Figure 2, and the decoding method for generating the full sine wave is shown in Figure 3.

Ⅲ. The Quantization Effects

The quantization effects are analyzed the CORDIC-based architecture. For the case of the CORDIC-based architecture, the numerical error bound is basically analyzed in Hu's study[6]. There are two major errors of the CORDIC computation, which is the approximation error and the truncation error. The former is due to the quantized representation of elementary angles and the latter is due to the finite word size.

Table 1. Optimized word lengths.

Accuracy	L	M	n	B
4	5	3	5	8
5	6	3	6	9
6	7	3	7	10
7	8	4	8	11
8	9	4	9	12
9	10	4	10	13
10	11	4	11	14
11	12	4	12	15

Table 2. Number of CLBs to implement 9-bit accuracy in XILINX FPGA 4000x series.

Elementary block	Word length (bits)	Number of CLBs
Adder	N = 10	8
	B = 13	7

Further, considering that the input word is represented by the finite number of bits, thus the intermediate results of the angle rotation part can be designed to reduce the truncation error by adding guard bits[7]. In the angle computation part, the angle is approximated by n elementary rotations, whose direction is calculated in B-bit arithmetic. The CORDIC-based architecture has an input word length, L, and guard bits to reduce an internal truncation error, M. The internal word length, N, is $(1+M+L)$ bits because there is one bit to avoid the overflow in the angle rotation part. The quantization error (ε_C) for the CORDIC-based architecture can be represented as

$$\varepsilon_C \leq 2K\arctan(2^{-n+1}) + 2^{-L-M+1}\left(1+\sum_{i=M+1}^{n-1}\sum_{j=i}^{n-1}\sqrt{1+2^{-2j}}\right). \quad (5)$$

Based on the aforementioned quantization error equation (5), we determined optimum word lengths of the DDFS, corresponding to accuracy bits in a fixed point data representation. Table 1 summarizes the optimized word lengths.

Ⅳ. Performance of CORDIC-based DDFS

The performance is estimated from its hardware size and operation speed. Regarding the speed estimation in FPGA, it is not trivial to estimate its post-layout timing because the combinational logic delay is different with that in ASIC, which is severely tool dependent. To get a desired speed, we have to give some timing constraints. For a speed estimation of DDFS, in general, the phase accumulator has the largest ripple carry adder, because its word length is much larger than the truncated phase word length in the DDFS[1]. So the bottleneck of the DDFS is the phase accumulator and the phase accumulator determines the operation speed of the DDFS.

Table 3. Size estimation results of the CORDIC-based DDFS.

Accuracy	Comp.	Rot.	Accu.	Total
4	52	23	17	92
5	68	30	17	115
6	90	36	17	143
7	108	44	17	169
8	136	51	17	195
9	168	60	17	245
10	190	68	17	275
11	228	78	17	323

The implementation of the CORDIC rotator requires adders, shifters, and registers. In the CORDIC rotator architecture, to obtain the high data throughput, a pipelined architecture is employed. Basically, all shifters are replaced by hardwiring in pipelining implementation to reduce large area that occupied by variable shifters in the CORDIC rotator circuit. In the conventional angle computation block, the word length of n adders is B bits. However, arctangent values that determine the direction of rotation are already set, and the required word length of the i_{th} adder for angle computation to implement the CORDIC rotator is reduced :

$$\begin{pmatrix} B & : \text{ for } & i=0 \\ B-i+1 & : \text{ for } & 1\leq i\leq n-1 \end{pmatrix}$$

where i represents the order of iteration[3].

In FPGA, hardware cost can be evaluated based on the number of Configurable Logic Blocks (CLBs) used. Table 2 shows the number of CLBs corresponding to elementary blocks for 9-bit accuracy. Note that this size estimation is based on XILINX FPGA 4000x series.

We calculated hardware requirements and summarized the size estimation results of CORDIC-based DDFS in Table 3. In this table, the unit is 1 CLB. We ignored the complexity of the complementors because those logics are very simple.

Ⅴ. Emulation Results

We implemented and emulated the pipelined CORDIC-based direct digital frequency synthesizer on a XILINX FPGA. The target device is the *4020XLPQ160 -2*. The used number representation is the two's complement representation of the fixed arithmetic. The phase accumulator word length is 16 bits. To get 9 precision bits, we selected L=10, M=4, n=10 from Table 1. So the output word length

N is 15 bits. In the angle computation circuit, the input word length is 13 bits and the output word length is 4 bits.

Figure 4 shows the generated sine waves from the designed CORDIC-based DDFS. The spectral purity as a performance parameter in DDFS, which means how close the generated waves are to the original sine waves, is good as shown in Figure 4. When the desired precision is 9 bits, its worst error is 0.0044 and its mean error value is 0.0020. Figure 5 shows the error performances of the generated sine waves. The error is calculated from $(\sin\theta - y_o)$, where y_o is the generated value. Based on Section III, the main trend is due to the approximation error that is proportional to the magnitude of the phase input and the fluctuation of error pattern is due to the truncation error of the finite word in a fixed point implementation.

Another important thing is an ability to synthesize the variable output frequency of the DDFS. In Figure 4, there are two sine waves. The solid line corresponding to the case when FTW=0002(*hex*), while the dashed to FTW=0001(*hex*). We can see the frequency of the solid wave is exactly 2 times the frequency of the dashed. To get a higher frequency sine wave, we have to give the higher frequency tuning word. We can control the output frequency by varying the FTW. The operation speed of the synthesized *4020XLPQ 160* is about 62.135*MHz*. Hence, the frequency resolution to be generated by the proposed CORDIC-based DDFS is 948.1*Hz*. The number of CLBs used to implement 9 bit accuracy DDFS is 248, which is added 3 CLBs to the estimated number, 245. These 3 CLBs are consumed for two complementors.

VI. Conclusion

In this paper, we proposed the pipelined CORDIC-based DDFS and estimated its performance with the XILINX FPGA. Through the numerical analysis of the quantization effects, we set the hardware design parameters and estimated its hardware complexity properly for given design parameters. We also implemented a practical CORDIC-based DDFS, assuring 9 bits accuracy, which operating speed is 62.135*MHz*, frequency resolution is 948.1*Hz*, and the number of CLBs consumed is 248.

Reference

[1] H.T.Nicholas III and H.Samueli, "A 150-MHz Direct Digital Frequency Synthesizer in 1.25-μ CMOS with 90-*dBc* Spurious Performance," IEEE J. Solid State Circuits, vol. 26, no. 12, pp. 1959-1969, Dec. 1991.

[2] J.S.Walther, "A Unified Algorithm for Elementary Functions," SJCC, pp. 379-385, 1971.

[3] G.Gielis, R.Plassche, and J.Valburg, "A 540-*MHz* 10-b Polar-to-Cartesian Converter." IEEE J. Solid State Circuits, vol. 26, no. 11, pp. 1645-1650, Mar. 1991.

[4] J.Vankka, "Methods of Mapping from Phase to Sine Amplitude in Direct Digital Synthesis," IEEE Trans. Ultrasonics, Ferroeletrics, and Frequency control, vol. 44, no. 2, pp. 526-534, Mar. 1997.

[5] L.K.Tan and H.Samueli, "A 200 *MHz* Quadrature Digital Synthesizer/Mixer in 0.8-μ CMOS," IEEE J. Solid State Circuits, vol. 30, no. 3, pp. 193-200, Mar. 1995.

[5] M.K.Park, S.S.Choi, S.G.Kim, K.S.Kim, and J.A.Lee, "A Digital Sinusoid Synthesis Based on the Postscaled CORDIC," APCC/OECC'99, Oct. 1999.

[6] Y.H.Hu, "The Quantization Effects of the CORDIC Algorithm," IEEE Trans. on Signal Processing, vol. 40, no. 4, pp. 834-844, Apr. 1992.

[7] S.H.Nahm and W.Y.Sung, "VLSI Implemenation of CORDIC-Based Digital Quadrature Demodulator," The Jornal of Korea Institute of Communications Science, pp. 1718-1731, July 1998.

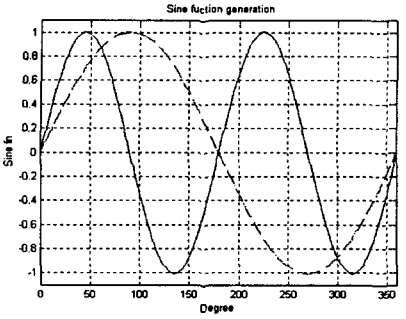

Figure 4. Generated sine waves. FTW corresponding to the dashed line is 0001(*hex*) and that for the solid is 0002(*hex*).

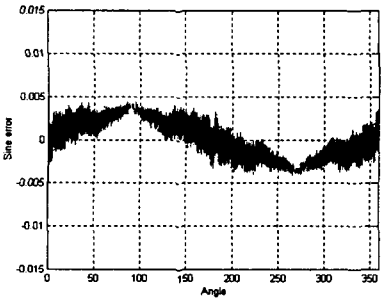

Figure 5. Error of the generated sine wave, corresponding FTW = 0001(*hex*)