CORDIC 알고리즘을 이용한 DDFS 설계

이민석, 조원경
경희대학교 전자공학과
Tel : (0331)201-2942 / Fax : (0331)202-4941
lms4u@csvlsi.kyunghee.ac.kr

Direct Digital Frequency Synthesizer design using CORDIC algorithm

Minsok Lee , Wonkyung Cho
School of Electronic Engineering Kyunghee University
E-mail : lms4u@csvlsi.kyunghee.ac.kr

Abstract

This paper describes the architecture and the IC implementation of a Direct Digital Frequency Synthesizer (DDFS). That is based on an angle rotation algorithm (CORDIC). It is shown that the architecture can be implemented as a multipliers, feedfoward, and easily pipelineable datapath. A prototype IC has been designed, fabricated in 0.35㎛ SAMSUNG KG90 Library.

Ⅰ. 서 론

최근 급속히 발전하고 있는 이동통신 시스템에 있어서 수신기와 송신기에는 고속 주파수 전환이 가능한 주파수 합성장치가 필요하다. 주파수 합성장치는 기지국과 단말기의 통신에 필요한 반송파 주파수를 발생하는 회로이며, 요구되는 주파수 합성기의 성능은 주파수 해상도, 대역폭, 천이속도 등으로 판단되며 연속적인 파형을 얻기 위해서는 시스템의 향상 및 Chip Size의 소형화가 요구된다.[1]

일반적인 주파수 합성기술의 세 가지 주류는 간접 방식, 직접 방식, 그리고, 직접 디지털 방식이 있다.

기존의 PLL(Phase Lock Loop) 주파수 합성방법은 기준 주파수에 대해 출력 주파수를 고정시키는 기계적인 피드백 방식이다. 이 합성 방식은 출력 주파수 대역이 넓고, 비교적 우수한 신호대 잡음비를 가지고 있어 현재 주파수 합성기의 주종을 이루고 있다. 그러나, PLL합성기는 궤환 루프의 특성상 발생되는 동기 시간의 지연에 의해 빠른 주파수 천이가 필요하는 시스템에는 적당하지 않다.

직접 (Direct Analog)주파수 합성기를 사용한 방식은 디지털 신호 처리(DSP)의 원리를 사용하여 기준 주파수에서 다른 주파수를 디지털 신호로 발생시킨다.

직접 디지털 주파수 합성 방식은 주기적인 샘플링과 디지털-아날로그 변환기술로 출력 파형을 합성하는 방식으로, 주파수 천이 시간이 빠르고, 고해상도의 주파수 특성을 갖고 있으며, 광대역에서도 사용이 가능하며, 발생 주파수의 안정도가 뛰어나다. 발생되는 주파수는 변환시 연속적인 위상변환을 이루므로 연속적인 통신에 사용이 가능하고, 특히, DDFS(Direct Digital Frequency Synthesizer)는 부피가 작고, 소비 전력이 낮아 휴대용 통신기기에 적합하다.

Ⅱ. 직접 디지털 주파수 합성기의 구조

2.1 직접 디지털 주파수 합성기 기본구조

일반적인 DDFS의 구조를 보면 주파수 레지스터에 Frequency Control Word를 입력하면 해당하는 주파수가 출력되며 이 주파수 레지스터에서 출력된 데이터값

은 계수형 발진기를 기본으로 한 위상가산기와 위상레지스터에 입력되어 클럭주파수에 따라 가산되고, 위상레지스터의 출력값이 궤환되어 원래값과 가산되어 출력신호의 위상값을 계산한다. 누산기에서 출력된 주소는 SIN LUT에 입력되어 정현 파형을 나타내는 일련의 데이터 값으로 출력된다. 출력된 12Bit의 데이터는 DAC를 거쳐 양자화된 계단파형이 출력되며 이는 다시 LPF를 통해 정현파를 출력하게 된다.

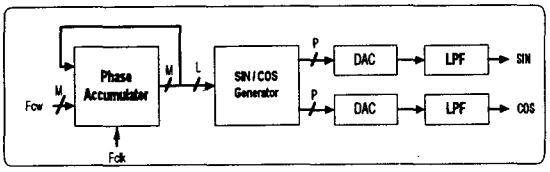

그림 1. 일반적인 DDFS 구조
Fig. Traditional DDFS architecture

2.2 위상 누산기(Phase Accumulator) 구조

입력되는 Frequency Control Ward에 따라 위상 정보를 발생하는 기능을 하는 위상 누산기는 주파수 저장 레지스터, 전가산기, 위상 레지스터로 구성이 된다. 주파수 조정 워드값은 주파수 저장 레지스터로 입력되고, 전가산기에서는 주파수 저장 레지스터의 정보를 입력 받아 디지털 값을 발생시킨다. 매 클럭 마다 발생된 디지털 값, 즉 주파수의 위상은 위상레지스터에 저장된다.

주파수 합성기는 주파수 해상도와 주파수 증가 간격을 고려하여 위상 누산기의 비트수를 결정하는데, 일반적으로 12 - 32비트의 위상 누산기를 사용한다.

2.3 사인함수값의 계산

사인 함수 계산은 사인 파형의 위상값과 진폭값을 롬에 저장하고 입력되는 어드레스에 따라 사인파형의 위상값과 진폭값을 출력시키는 ROM-based Look-Up Table 방식이 일반적으로 사용되고 있다. 이 방식은 사인 파형의 위상값과 진폭값을 저장하는 SIN LUT의 워드수와 워드의 비트수에 따라 각각 위상 양자화와 진폭 양자화 오차가 결정되기 때문에 위상이나 진폭의 잘림없이 SIN LUT을 구현하기 위해서는 많은 양의 데이터를 저장하는 대용량의 ROM이 필요하다. 그러나, ROM이 커짐에 따라 시스템의 크기가 증가하고, ROM의 액세스 시간이 지연 되므로 ROM의 크기와 스펙트럼의 특성을 절충해야 한다. 식(2.1)은 어드레스 비트수에 따른 SIN LUT에 저장되는 샘플수이다.

$$Samples = 2^{A} \qquad (2.1)$$

여기서, Samples는 SIN LUT에 저장되는 샘플들의 수이고, A는 어드레스 비트 수이다.

직접 디지털 주파수 합성기에서는 위상누산기의 출력 중 상위 비트만 SIN LUT의 어드레스 비트로 사용하는데, 이로 인한 위상 잘림(Phase Truncation)과 한정된 SIN LUT의 크기 즉, 한정된 샘플링 데이터로 인한 진폭 잘림(Amplitude Truncation)의 잡음이 발생한다. 이러한 현상을 제거하기 위하여 종래의 직접 디지털 주파수 합성기에서는 위상 누산기의 출력 전부를 SIN LUT의 어드레스 비트로 사용하고, SIN LUT 사이즈를 늘려서 위상잘림을 감소시키는 진폭 양자화(Amplitude Quantization)방법으로 출력 잡음을 개선했지만, 어드레스 비트수의 제곱에 비례하여 지수 함수적으로 증가하는 많은 양의 출력 저장 데이터가 요구되었기 때문에 주파수 합성기의 칩 크기가 커지는 단점이 있었다.[2]

그러므로 본 논문에서는 SIN함수를 LUT를 사용하는 방법이 아닌 직접 CORDIC 알고리즘을 이용하여 원하는 주파수에 따른 SIN 값을 계산하여 빠른 속도와 정확도를 얻는 방법을 사용하였다.[3]

Ⅲ. CORDIC 알고리즘의 기본개념

3.1 CORDIC 알고리즘의 원리

CORDIC 알고리즘은 Volder에 의해 소개되었고 Walther에 의해 확장되었다. CORDIC의 매력은 삼각함수, 지수함수, 로그함수 등의 초월함수들을 아주 간단한 하드웨어로 구현할 수 있다는 것이다. 오직 가산기, 감산기, shift 만이 필요하다.[4][5]

구체적으로 기본 원리와 동작 과정을 설명하면, 어떤 벡터 P=(x, y)를 반시계 방향으로 각 α만큼 회전한 후의 좌표는 다음과 같이 표현할 수 있다.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \cos\alpha \begin{bmatrix} 1 & -\tan\alpha \\ \tan\alpha & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \qquad (3.1)$$

위에서 α가 $\tan^{-1}(2^{-i})$ 라고 가정하면, 식(1.1)은 다음과 같이 표현된다.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \cos\alpha \begin{bmatrix} 1 & -2^{-i} \\ 2^{-i} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \qquad (3.2)$$

위에서 어떤 이진수에 2^{-i} 를 곱하는 것은 그 수를 단순히 i 만큼 오른쪽으로 shift 한다는 것과 같은 의미를 가진다. 따라서 위 식(1.2)를 계산하는데는 한번의 곱셈만으로 간단히 구현될 수 있다. Volder는 이 알고리즘을 더욱 개선하여 한번의 곱셈도 필요 없게 하였다.

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} = \cos\alpha \begin{bmatrix} 1 & -d_i 2^{-i} \\ d_i 2^{-i} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \qquad (3.3)$$

$$Z_{i+1} = z_i - d_i \tan^{-1}(2^{-i}) \qquad (3.4)$$

식 (1.3)과 식(1.4)에서 d_i는 회전의 방향에 따라 1 도는 -1의 값을 갖는 변수이고, 한번 회전할 때마다 $\cos\alpha_i$만큼의 오차가 발생하지만 이 $\cos\alpha_i$의 값은 항상 일정한 값을 가지게 되므로 회전이 끝나고 최종단계에서 보상을 함으로써 간단히 해결될 수 있다. 식 (2.4)는 회전의 방향을 결정짓는 제어식으로써, 반복될 때 변화된 각의 누적된 값을 저장하고 있다.

쌍곡선함수(Hyperbolic Function)와 삼각함수는 서로 밀접한 관계가 있다는 점에 착안하여, Walter는 Volder의 알고리즘을 확장된 각과 반지름의 개념을 이용하여 확장시켰다.[5]

$$R(\text{반경}) = [x^2+my^2]^{1/2} \qquad (3.5)$$

$$A(\text{각도}) = m^{-1/2}\arctan[m^{1/2}\frac{y}{x}] \qquad (3.6)$$

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & -md_i\delta_i \\ d_i\delta_i & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \qquad (3.7)$$

$$z_{i+1} = z_i - d_i\alpha_i \qquad (3.8)$$

$$\alpha_i = m^{-1/2}\tan^{-1}[m^{1/2}\delta_i] \qquad (3.9)$$

여기서 δ_i는 앞에 나온 2^{-i} 을 확장시킨 것으로서 수렴조건을 만족하는 값이다.

3.2 반복형구조와 파이프라인 구조

CORDIC 알고리즘을 구현하는 방법에는 크게 반복형 구조와 파이프라인구조로 나눌 수 있는데, 먼저 반복형 구조는 Haviland에 의해 제안되었으며, 기본 블록은 X_i와 Y_i의 값을 구하기 위한 연산부와 연산부의 회전방향을 제어하는 Z_i를 구하는 제어부로 나누어진다. 연산부는 그림 2에서와 같이 입력값을 저장하는 레지스터와 Barrel Shift, shift된 결과와 X_i와 Y_i를 가산 / 감산하는 누산기로 구성되어 있다. 제어부는 제어부호를 생성하여 연산부의 가산기와 감산기 중에서 하나의 동작을 선택하는 기능을 한다. 제어부는 LUT에 저장된 기준 각도와 입력된 각도에 의한 차이를 연산하여 그 결과를 통해 연산부의 누산기의 가산/감산을 제어하는 출력신호를 만들게 된다. 이때 제어신호는 각도값의 MSB로서 간단히 구할수 있다.

파이프라인 구조는 CORDIC의 단점인 속도문제를 해결하기 위하여 최적의 반복횟수를 찾아서 그 결과에 의해 연산부와 제어부로 이루어진 기본구조를 파이프라인으로 연결하여 속도를 크게 향상시킨 구조이다.

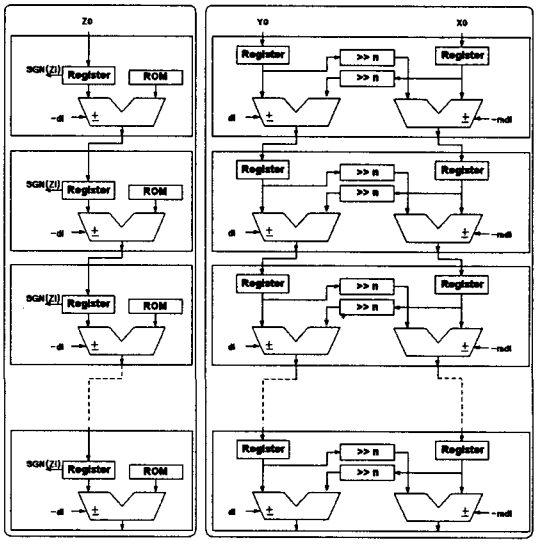

그림 2. CORDIC의 파이프라인 구조
Fig 2. Pipeline CORDIC Block Diagram

Ⅳ. CORDIC을 이용한 DDFS 구조

제안한 직접 디지털 주파수 합성기는 Frequency Control Word가 32비트, 출력이 12비트이며 20MHZ의 동작주파수를 목표로 하였다.

그림 3은 제안된 삼각함수 계산회로를 사용한 32비트 DDFS의 전체 블록도이다. 이는 주파수 저장 레지스터, 위상누산기, 위상제어부, 삼각함수 계산회로부, DAC, LPF로 구성된다.

디지털 주파수 합성기에서의 전체적인 동작속도는 위상가산기의 가산속도에 의존하며, 위상 가산기의 주파수 조정 입력 데이터 비트 수는 주파수 해상도와 위상 증가값을 고려하여 32 Bit를 사용하였다. SIN 파형을 만드는데 있어 $\pi/2$ 부분까지만 위상값만을 계산하면 되므로 시스템의 크기를 축소할 수 있었고 CORDIC 알고리즘 이용시 동시에 SIN함수와 COS함수 구할 수 있으므로 $\pi/4$까지의 값만을 계산하여 SIN함수와 COS 함수값을 선택하여 사용하면 계산과정을 단순화 할 수 있다.

블럭도에서 위상이 90 °보다 넘으면 뺄셈기를 거쳐서 90 °이전의 값의 보수가 출력값이 되고 또한, 180 °과 360 °사이의 값은 이전의 값에 사인 비트를 추가함으로써 0 ~ 2π까지의 데이터를 모두 표현하였다.

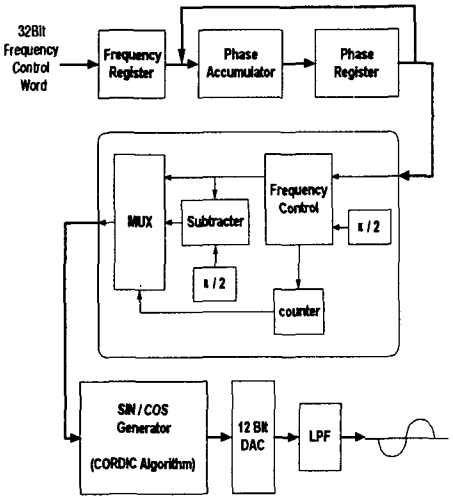

그림 3. CORDIC 알고리즘을 이용한 DDFS 구조
Fig. DDFS architecture using CORDIC algorithm

Ⅳ. 실험 및 고찰

CORDIC에 의한 초월함수 계산에서 발생하는 오차는 반복회수오차와 절단오차로 분류할 수 있다. 실험에 의한 결과 그림4와 같이 16Bit CORDIC 연산기 구현시 오차가 제일 적은 파이프라인의 단수는 10이라는 결과가 얻어졌다. 이것은 0에서 π/2의 각도를 증가하면서 생기는 실제값과 구현값의 오차를 누적하였을 때 반복횟수가 10일 때 최소의 오차가 발생하였다. 이에 근거하여 연산기 구현시 파이프라인은 10단으로 선정하였다.

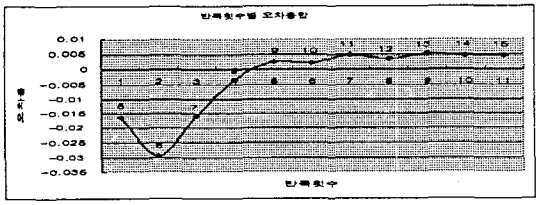

그림 4 반복회수별 발생 오차
Fig 4. Error at Iteration Stage

또한 CORDIC의 특성상 주어지는 각도에 따라서 일정한 오차율이 아닌 각도에 따라 다른 오차율을 보였다. 그림 5는 SIN함수의 입력 각도에 따른 오차를 나타낸다.

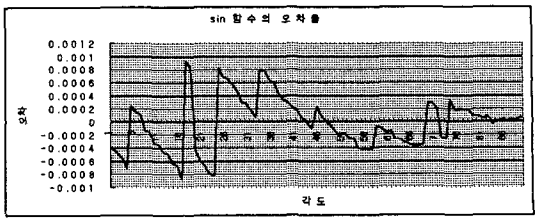

그림 5 SIN 함수의 각도별 오차율
Fig 5. SIN value error at degree

반복형 구조의 CORDIC 연산기를 SAMSUNG KG90 Library로 합성한 결과 Gate Size는 1023 Gate이고, 파이프라인 구조의 경우는 1023×10 Gate 가 필요하다. 설계한 직접 디지털 주파수 합성기의 사양은 Frequency Control Word가 32Bit, Output Bit 12Bit, 20MHz의 동작 주파수를 기본으로 하고 기준 CLOCK은 60Mhz 이다.

Ⅴ. 결 론

본 논문에서는 직접 디지털 주파수 합성기를 설계함에 있어서 기존의 LUT를 이용한 방법대신에 CORDIC 알고리즘을 이용하여 SIN /COS 함수를 계산함으로써 높은 정확도와 빠른 속도를 구현할 수 있도록 하였다. 또한 CORDIC 알고리즘을 이용하여 SIN/COS 함수 연산기를 설계시 오차율을 고려하여 파이프라인의 단수를 결정하였다.

참 고 논 문

[1] J.Tierney, C.M. Rader, and B.Gold, "A digital frequency synthesizer", IEEE Trans Audio Electroacoust., Vol. AU-19, pp.48-56, Mar. 1971

[2] L.K. Tan, E.Roth, G.E,Yee, and Samueli, "800MHz, Quadrature digital synthesizer with ECL-compatible drivers in 0.8㎛ CMOS", ISSCC digest of Technical Papers, pp258-259, Feb, 1995

[3] A. Madisetti, A.Kwentus, and A.N.Willson, "A sin/cosine direct digtal frequency synthesizer using an angle rotation algorithm", ISSCC Digest of Technical Papers, pp.262-263, Feb, 1995

[4] J.E Volder, " The CORDIC trigonometric Computing technique", IRE Trans. Electron. Comput. Vol. EC-8, no.3, pp.335-339, Sept. 1959

[5] J.S Walter. "A Unified algorithm for elementary functions". Spring Joint Computer Conference. Seiten 379-385, Mai.1971.