직접 디지털 주파수합성기의 ROM 크기를 줄이는 새로운 방식

강형주(姜亨周), 박인철(朴仁哲)
한국과학기술원 전기 및 전자공학과 VLSI Systems Lab.
T.042-869-4402 Fax. 042-869-4410

A New Method to Reduce the Size of the ROM in Direct Digital Frequency Synthesizers

Hyoung-Ju Kang and In-Cheol Park
Dept. of Electrical Engineering Korea Advanced Institute of Science and Technology
E-mail : dk@vslab.kaist.ac.kr

Abstract

In this paper, a new method to reduce the size of ROM in the direct digital frequency synthesizer (DDFS) is proposed. In the case that ROM is used for sinusoidal value calculation, reducing the size of ROM is significant. So the power consumption is affected mostly by its bit width. In the proposed method, the ROM bit width is reduced by 1 bit using the phase subtraction and the approximation. The spurious level is better than 80dBc and the power consumption estimated is 510㎼/㎒.

1. 서론

기존에 존재하던 많은 아날로그 소자들을 디지털로 대체하고 있는 것이 현대의 추세이다. 디지털로 구현할 경우, 설계가 용이하고 기술의 발전에 따라 전력이나 크기 면에서 장점을 가지기 때문이다. 이 추세를 따라 통신 장비에 많이 들어가는 Phase Locked Loop (PLL)을 디지털로 대체하고 있는데 이 PLL을 대체할 소자가 바로 Direct Digital Frequency Synthesizer (DDFS)이다. DDFS는 PLL에 비해 앞에서 말한 디지털 방식이기에 가지고 있는 장점과 더불어 빠른 주파수 천이(frequency switching) 속도와 높은 주파수 해상도(frequency resolution)를 가지는 이점이 있다. [1-8] 그리고 주변소자들이 디지털화 되어 가고 있기에 DDFS는 더욱 중요해지고 있다.

DDFS는 그림 1과 같이 크게 두 부분으로 나뉜다. 앞부분의 Phase Generation Part는 외부에서 받아들인

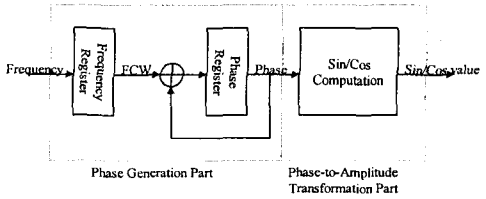

그림 1 DDFS의 기본 구조

frequency 정보(그림에서 Frequency Control Word)에 따라 매 clock마다 적절한 phase를 출력한다. 그 뒤의 Phase-to-Amplitude Transformation Part에서는 앞단에서 건너 온 phase를 이용하여 그에 해당하는 Sin 및 Cos 함수의 amplitude를 출력한다.

Phase-to-Amplitude Transformation Part에서 행해지는 연산은 phase에 따른 Sin과 Cos 값을 계산하는 것이며 이를 행하는 방식은 반복적인 연산을 이용한 CORDIC 방식과 미리 함수 값을 저장해 놓는 ROM방식이 있다.[7] ROM방식에서 가장 큰 문제는 ROM에서의 전력 소모이며 이를 줄이기 위해 많은 연구가 진행되어왔다.[1,4-8] 이 논문에서는 ROM의 전력 소모를 줄이기 위해 ROM의 bit width를 줄이는 방법을 제안하고 그에 따라 synthesis한 결과를 제시할 것이다.

2장에서는 기존에 연구되어 왔던 방식들을 간단히 살펴보고, 3장에서는 새로이 제안된 방식을 설명한 뒤, 4장에서 새로이 구성된 DDFS에 대한 결과를 제시하고, 5장에서 결론을 맺을 것이다.

2 . 기존 DDFS의 구조

DDFS를 가장 간단하게 나타낸 것이 그림 1이다. 각

Part의 역할은, 앞에서 설명하였듯이, Phase Generation Part에서 phase를 발생시키고 그 phase를 이용하여 Phase-to-Amplitude Transformation Part에서 Sin 및 Cos함수를 연산하는 것이다.

Phase Generation Part에서 phase를 생성하는 방식은 그림 1에서와 같이 매 clock마다 frequency를 더해 나가는 것이다. 이 때 bit width가 긴 Carry Propagation Adder가 필요해서 과거에는 이 adder의 delay를 줄이기 위해 pipelining같은 기법이 개발되었으나[4] 요즘의 기술에서는 충분히 빠른 adder가 구현 가능하므로 이 부분의 중요성은 감소하는 추세이다.

Phase Generation Part에서 나오는 값은 진정한 phase값이 아닌 phase에 해당하는 적당한 digital number이므로 argument라고 부른다. 이 argument $\phi(n)$와 phase $\theta(n)$의 관계는 다음과 같이 표현할 수 있다.

$$\theta(n) = 2\pi\frac{\phi(n)}{2^L} \qquad \text{(식 1)}$$

여기서 L은 argument의 bit width이다. argument가 매 clock마다 Frequency Control Word(FCW) 만큼 더해진다는 사실과 (식 1)에서 주어진 관계를 이용하면 FCW와 출력 Sin 함수 주파수의 관계를 다음과 같이 유도할 수 있다.

$$f_{out} = \frac{\theta(n)-\theta(n-1)}{2\pi\Delta t_{clk}} = \frac{f_{clk}\cdot FCW}{2^L} \qquad \text{(식 2)}$$

이 때 출력 주파수 해상도는 다음과 같다.

$$\Delta f_{out} = \frac{f_{clk}}{2^L} \qquad \text{식(3)}$$

ROM방식의 DDFS의 경우 ROM의 크기 및 전력 소모가 결정적이므로 이를 줄이기 위한 노력이 있어왔다.[1-8] 첫 번째의 방법은 Phase Generation Part에서 넘겨져 오는 argument의 하위 bit들을 truncation시키는 것이다.[1-8] 주파수 해상도를 높이기 위해 argument의 bit width는 보통 20bit 이상이다. 이를 그대로 ROM으로 구현하게 되면 그 크기는 구현하기 어려울 정도가 된다. 따라서 10bit 내외로 적당히 truncation시키게 된다. truncation시키는 bit의 위치는 최종 출력 bit width에 따라 달라지며 [2,3]에서 제시된 spur.식에 따라 출력에 영향을 주지 않을 정도로 truncation 시킨다.

두 번째 방법은 Sin 및 Cos함수의 대칭성을 이용한 것이다.[1,4-8] Sin 함수를 예로 들면, $0\sim\pi$구간의 값들과 $\pi\sim2\pi$구간의 값들을 비교해 보면 그 절대값은 반복되지만 부호만 반대인 것을 알 수 있다. 따라서 어느 한 구간의 값들만 ROM에 저장하고 나머지 구간은 ROM에서 읽은 값에 -1을 곱함으로써 얻을 수 있다. 그리고 $0\sim\pi$구간에서의 값들을 보게 되면 역시 대칭성이 존재한다. $0\sim\pi/2$구간과 $\pi/2\sim\pi$구간의 값들이 서로 반대의 순서로 배치되어 있는 것이다. 이 성질을 이용하면 ROM에는 $0\sim\pi/2$구간의 값들만을 저장하고, $\pi/2\sim\pi$구간의 값들을 원할 때에는 $0\sim\pi/2$ 구간의 값들을 access하되 그 순서가 반대가 되도록 즉 ROM을 access하는 address를 반전시켜서 access하면 된다. 이와 같이 대칭성을 이용하게 되면 ROM의 크기를 처음보다 1/4로 줄일 수 있다.

Sin함수의 값과 함께 Cos함수의 값도 같이 얻으려고 할 경우에는 Cos함수가 Sin함수와 90° 의 위상차만 있을 뿐 서로 같음을 이용하여 Sin함수의 ROM만으로 Cos함수의 값을 계산할 수 있다. 다만, Sin함수와 Cos함수가 동시에 만들어져야 하므로 ROM을 $0\sim\pi/4$구간과 $\pi/4\sim\pi/2$구간으로 나누어서 두 ROM을 동시에 access한 뒤 ROM의 데이터는 MUX를 통해 Sin함수의 값 혹은 Cos함수의 값을 연산하는 회로로 전달한다.

세 번째 방법은 ROM에 저장할 때 Sin값을 그대로 저장하는 것이 아니라 phase를 한 번 뺀 뒤 저장시키는 것이다.[1,4-8] phase를 그대로 뺄 경우 음수가 될 수도 있으므로 normalize시킨 phase를 빼어서 다음과 같은 값을 저장시킨다.

$$\sin\theta - \frac{2}{\pi}\theta \qquad \text{(식 4)}$$

ROM에서 값을 읽은 뒤 Sin값을 연산하는 과정에서는 $(2/\pi)\theta$를 다시 더해주어야 된다. 이 때 $(2/\pi)\theta$값은, (식 1)에서 알 수 있듯이, argument값을 적당히 shift시킨 값이므로 adder 1개에 의해 간단히 Sin값을 복구할 수 있다.

이와 같은 방식을 사용할 때 ROM에 저장하게 되는 값의 크기는 다음과 같이 된다.

$$\sin\theta - \frac{2}{\pi}\theta \le 0.21 \qquad \text{(식 5)}$$

즉 ROM에 저장될 값의 bit width가 기존에 비해 2bit 줄어들게 되는 것이다. ROM의 전력 소모가 bit width에 의해 크게 영향받음을 고려하면 이 방식에 의해 ROM의 전력 소모가 많이 줄어들 수 있음을 알 수 있다.

Yamagishi등은 이러한 phase를 빼는 방식을 확장시켜서 double trigonometric approximation을 발표하였다.[6] 앞에서 설명한 phase subtraction 방식을 Sin함수에 대한 linear approximation으로 보고 좀 더 가깝게 approximation시킨 것이 double trigonometric approximation방식이다. 이를 그래프로 표현한 것이 그림 2이다. 그림 2에서 곡선 A가 원하는 Sin함수이고 직선 B가 phase subtraction 방식에서 빼어지는 값을

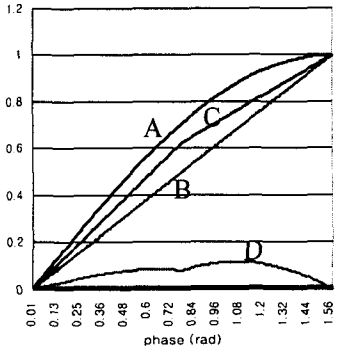

그림 2 Yamagishi 등의 Double Trigonometric Approximation

나타낸다. 그리고 한 번 꺾여진 직선 C가 double trigonometric approximation에서 빼어지는 값이다. ROM에 저장되는 값은 A에서 C를 뺀 곡선 D이다. 그림 2에서 알 수 있듯이 그 곡선 D는 0.125를 넘지 않으며 따라서 ROM의 bit width가 3bit 줄어드는, phase subtraction에 비해 1bit 더 줄어들게 되는 효과를 가져온다. 이를 식으로 표현하면 다음과 같다.

$$\sin\theta - \frac{2}{\pi}\theta - 0.25\frac{2}{\pi}\theta \ , \ 0 \le \theta < \frac{\pi}{4}$$

$$\sin\theta - \frac{2}{\pi}\theta - 0.25(1-\frac{2}{\pi}\theta) \ , \ \frac{\pi}{4} \le \theta < \frac{\pi}{2} \quad \text{(식 6)}$$

위 식을 계산해 보면 ≤0.1163임을 알 수 있다. 위 식을 보면 알 수 있듯이 double trigonometric approximation방식에서는 phase subtraction에 비해 Sin값을 복구하기 위해 adder가 1개 더 필요하게 된다.

3 . Quadrant Approximation

이 논문에서 제안하고자 하는 방식은 double trigonometric approximation을 확장한 것이다. 그림 2를 보게 되면 Sin함수를 나타내는 곡선 A와 double trigonometric approximation을 나타내는 직선 C사이에 아직 틈이 많다는 것을 알 수 있다. 이 틈을 줄이기 위해 한 번 더 근사할 수 있다. 이를 그림으로 표현하면 그림 3과 같이 된다. 그림 2에 비해 approximation 직선을 $\pi/8$과 $3\pi/8$에서 한 번씩 더 꺾음으로 해서 Sin곡선에 훨씬 더 가깝게 근사할 수 있음을 알 수 있다. 따라서 ROM에 저장할 값을 나타내고 있는 곡선 C도 훨씬 작아져 있다. 이 그림을 식을 표현하면 다음과 같다.

$$\sin\theta - \frac{2}{\pi}\theta - 0.5\frac{2}{\pi}\theta \ , \ 0 \le \theta < \frac{\pi}{8}$$

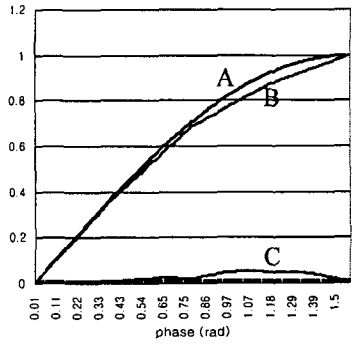

그림 3 Quadrant Approximation

$$\sin\theta - \frac{2}{\pi}\theta - 0.25\frac{2}{\pi}\theta - 0.0625 \ , \ \frac{\pi}{8} \le \theta < \frac{\pi}{4}$$

$$\sin\theta - \frac{2}{\pi}\theta - 0.25(1-\frac{2}{\pi}\theta) - 0.0625 \ , \ \frac{\pi}{8} \le \theta < \frac{\pi}{4}$$

$$\sin\theta - \frac{2}{\pi}\theta - 0.5(1-\frac{2}{\pi}\theta) \ , \ \frac{\pi}{8} \le \theta < \frac{\pi}{4} \quad \text{(식 7)}$$

위 식들의 값을 계산하면 ≤0.0538임을 알 수 있다. 즉 4bit를 줄일 수 있고 double trigonometric approximation과 비교하면 1bit를 더 줄일 수 있다.

phase subtraction에서 double trigonometric approximation으로 옮겨가면서 ROM의 bit width가 1bit 줄어든 반면에 Sin값을 복구하기 위해 더해야 하는 항이 늘어남에 따라 adder가 1개 더 늘었다. double trigonometric approximation에서 quadrant approximation으로 넘어오면서 같은 방식으로 생각하면 ROM의 bit width가 1bit 줄어든 대신 Sin값을 복구하기 위해 더해야 하는 항이 역시 1개 더 늘어나서 -(식 7)의 두 번째와 세 번째 식에서의 0.0625- adder도 1개 더 늘여야 하는 것으로 보인다. 하지만 이 경우에 있어서는 더해야 하는 항이 상수항이며 그 크기가 ROM에 저장되는 값들의 크기보다 더 크다는 점에 유의해야 한다. 즉, ROM에 저장되는 값들은 0.0538이하로써 이진수로 표현할 경우 소수점 아래 넷째 자리까지는 0이 된다. 0.0625는 소수점 아래 넷째 자리이므로 0.0625를 더한다는 것은 ROM에서 읽은 값의 앞에 단순히 1을 concatenation시키는 것이 된다. 따라서 adder가 1개 더 증가할 필요가 없게 된다. 증가하는 logic으로는 argument에 0.5를 곱한 값을 더할 지 argument에 0.25를 곱한 값을 더할 지를 결정할 MUX가 더 필요할 뿐이다. 즉 Sin값을 복구할 때 필요한 회로가 거의 늘어나지 않으면서 ROM의 크기를 줄일 수 있다.

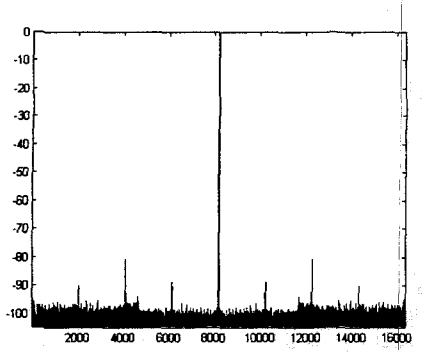

그림 4 FFT결과

4 . 결과

그림 4가 제안한 방식으로 DDFS를 verilog로 기술한 뒤 그 출력을 FFT하여 본 것이다. output bit width가 12bit일 때의 결과이며 spur. level이 -81.4dBc로 나왔다. 즉 이 방식을 사용함으로써 spur. level이 나빠지는 것은 없음을 나타낸다. 이 DDFS에서 ROM을 제외한 logic들을 Synopsys로 합성하면 면적은 1375 μm^2이 되고 전력은 51.53$\mu W/MHz$가 된다. 사용한 ROM의 크기는 4096×7bit로써 Yamagishi의 방법을 사용했을 경우 4096×8bit의 ROM이 필요한 것에 비해 10%이상 ROM의 크기를 줄였다. estimation에 의하면 ROM의 면적은 0.229㎟이고 전력은 459$\mu W/MHz$이다. 사용한 standard cell은 Samsung의 3.3V, 0.35μm이다.

5 . 결론

이 논문에서는 DDFS의 ROM을 구성함에 있어 주변 logic을 크게 늘리지 않으면서 ROM의 크기를 줄일 수 있는 방법을 제시하였다. 이 방식은 Yamagishi등이 제안한 방식을 개선한 것으로써 output이 12bit인 DDFS의 경우 필요한 ROM의 bit width가 8bit에서 7bit으로 감소한다. DDFS에서 면적 및 전력의 많은 부분을 ROM이 차지한다는 사실을 고려할 때 이 효과는 크다고 할 수 있다.

참고 문헌

[1] D.A.Sutherland, R.A.Strauch, S.S.Wharfield, H.T.Peterson and C.R.Cole, "CMOS/SOS frequency synthesizer LSI circuit for spread spectrum communications," IEEE J. Solid-State Circuits, vol.SC-19, pp497-505, Aug. 1984

[2] H.T.Nicholas and H.Samueli, "An analysis of the output spectrum of digital frequency synthesizer in the presence of phase-accumulator truncation," Proc. 41st Annu. Frequency Control Symp., 1987, pp.495-502

[3] H.T.Nicholas, H.Samueli, and B.Kim, "The optimization of direct digital frequency synthesizer performance in the presence of finite word length effects," Proc. 42nd Annu. Frequency Control Symp., 1988, pp.357-363

[4] H.T.Nicholas and H.Samueli, "A 150MHz direct digital frequency synthesizer in 1.25μm CMOS with -90dB spurious performance," IEEE J. Solid-State Circuits, vol.26, no.12, Dec. 1991, pp.1959-1969

[5] L.K.Tan and H.Samueli, "A 200MHz quadrature digital synthesizer/mixer in 0.8μm CMOS," IEEE J. Solid-State Circuits, vol.30, no.3, Mar. 1995, pp.193-200

[6] A.Yamagishi, M.Ishikawa, T.Tsukahara, and S.Date, "A 2V, 2GHz low-power direct digital frequency synthesizer chip set for wireless communication," IEEE Custom Integrated Circuits Conf., 1995, pp.14.4.1-14.4.4

[7] J.Vankka, "Methods of mapping from phase to sine amplitude in direct digital synthesis," IEEE Trans. Ultrasonics, Ferroelectrics, and Frequency Control, vol.44, no.2, Mar. 1997, pp.526-534

[8] J.Vankka, M.Waltari, M.Kosunen, and K.A.I.Halonen, "A direct digital synthesizer with an on-chip D/A-converter," IEEE J. Solid-State Circuits, vol.33, no.2, Feb. 1998, pp.218-227