논문 2008-45SP-5-15

CORDIC을 이용한 OFDM 주파수 옵셋 동기부 설계 및 구현

(Design and Implementation of OFDM Frequency Offset Synchronization Block Using CORDIC)

장 영 범*, 한 재 웅**, 홍 대 기*

(Young Beom Jang, Jae Woong Han, and Dae Ki Hong)

요 약

이 논문에서 OFDM(Orthogonal Frequency Division Multiplexing) 시스템의 주파수 옵셋 동기화 블록의 효율적인 구조를 제안한다. 기존의 CORDIC(COordinate Rotation DIgital Computer)을 이용한 주파수 옵셋 동기화 블록들은 위상 추정을 위하여 CORDIC Vector 모드를 사용하고, 보상을 위하여 CORDIC Rotation 모드를 사용하고 있다. 이와 비교하여 제안구조는 Vector 모드만을 사용하고 Rotation 모드는 Divider로 대치하는 알고리즘이다. 제안된 방식을 사용함으로써 Rotation 모드를 사용해야 했던 기존의 방식보다 하드웨어 구현복잡도가 감소함을 구현을 통하여 검증하였다. 검증 Tool로 Design Compiler를 사용하였고 각 비교 구조 마다 동일한 Constraint를 적용하여 검증을 진행하였다. 제안구조에 대한 Front-End 칩 구현을 통하여 기존 구조에 비하여 22.1%의 gate count 감소를 보임으로써 저전력 통신용 칩에서 사용할 수 있음을 보였다.

Abstract

In this paper, an efficient frequency offset synchronization structure for OFDM(Orthogonal Frequency Division Multiplexing) is proposed. Conventional CORDIC(COordinate Rotation DIgital Computer) algorithm for frequency offset synchronization utilizes two CORDIC hardware i.e., one is vector mode for phase estimation, the other is rotation mode for compensation. But, proposed structure utilizes one CORDIC hardware and divider. Through simulation, it is shown that hardware implementation complexity is reduced compared with conventional structures. The Verilog-HDL coding and front-end chip implementation results for the proposed structure show 22.1% gate count reduction comparison with those of the conventional structure.

Keywords: OFDM, CORDIC, Vector Mode, WLAN, Frequency Synchronization

I. 서 론

OFDM(Orthogonal Frequency Division Multiplexing) 방식은 유무선 고속데이터 전송 시스템에 널리 사용되고 있다. OFDM 변조방식은 심볼의 길이가 길고, 좁은 스펙트럼이 서로 겹쳐있기 때문에 주파수 동기에 크게 민감한 특성을 갖는다. 따라서 OFDM 방식의 신호를 정확히 복조하기 위해서는 신호의 시간 동기 및 반송파 주파수 동기가 매우 중요하다. 송신단과 수신단 사이의 오실레이터 차이와 도플러주파수 천이 등에 의해 발생하는 반송파 주파수 옵셋은 전체 부 반송파간의 직교성에 영향을 주게 되어 부 채널 간의 ICI(Inter Channel Interference), 위상회전, 크기 감소 등의 왜곡이 발생하여 시스템의 전체 성능을 저하시킨다. 따라서 ODFM 시스템의 수신 단에서는 FFT를 수행하기 이전에 송수신기간의 주파수 동기가 선행되어야 한다.

디지털 영역에서 프리앰블을 이용하여 반송파 주파수 옵셋을 추정하고 보상하는 블록의 구현에는 CORDIC 알고리즘을 이용하는 방식이 주로 사용되고 있다. 첫 번째 방식은 두 개의 CORDIC 하드웨어와 회전각계산, 회전방향계산 하드웨어를 필요로 한다.[1] 이 방식은 1개의 CORDIC은 Vector 모드로 동작하여 보상

* 정회원, 상명대학교 정보통신공학과
(College of Engineering, Sangmyung University)
** 학생회원, 상명대학교 컴퓨터정보통신공학과
(Graduate School, Sangmyung University)
접수일자: 2007년11월14일, 수정완료일: 2008년8월6일

할 θ값을 추정하며, 다른 1개의 CORDIC은 Rotation 모드로 동작하여 보상할 $cos\theta$와 $sin\theta$를 구한다. 두 번째 방식은 회전각 계산과 회전방향 계산을 제거하고 CORDIC Vector 모드와 CORDIC Rotation 모드를 하나의 제어신호로 동시에 연산하는 방식이다.[2] 이 방식은 두 개의 CORDIC이 동시에 동작하므로 latency가 작고 구현 하드웨어도 작은 장점을 갖는다.

이 논문에서는 한 개의 CORDIC으로 주파수 옵셋을 추정하고 보상하는 저전력 구조를 제안한다. II절에서는 주파수 옵셋 동기화 알고리즘을 기술하고 III절에서는 기존의 방식들을 살펴본다. IV절에서 제안방식의 알고리즘을 기술하고 V절에서 구현을 통해 저전력 구현 방식임을 보인다.

II. OFDM 시스템에서의 주파수 옵셋 동기화기

주파수 옵셋 추정 방식에는 DA(Data Aided) 방식과 NDA(Non-Data Aided) 방식이 있다. DA 방식은 주파수 옵셋 추정을 위하여 특별한 심볼을 보내는 방식이다. 송신기는 수신기와 미리 약속된 파일럿 심볼을 연속으로 실어 보내고 수신기에서는 수신한 이 두 심볼 간의 상관관계를 이용하여 주파수 옵셋을 추정하게 된다.[3]

NDA 방식은 OFDM 심볼 간 보호구간으로 덧붙여지는 CP(cyclic prefix)를 이용한다. 즉, 이 방식은 CP로 복사된 OFDM 심볼의 마지막 부분과 CP간의 상관관계를 이용하여 주파수 옵셋을 추정하게 된다.[4] 이 논문에서는 CP방식이 아닌 ZP(zero prefix) 방식으로서 보호구간에 신호를 모두 0으로 송신하는 DA방식을 사용한다. IEEE 802.11a에 있는 표준에 훈련 심볼인 프리앰블이 정의되어 있으므로 프리앰블을 이용한 DA방식을 적용한다. 프리앰블을 이용한 주파수 옵셋 추정 및 보상 구조는 다음 그림 1과 같다.

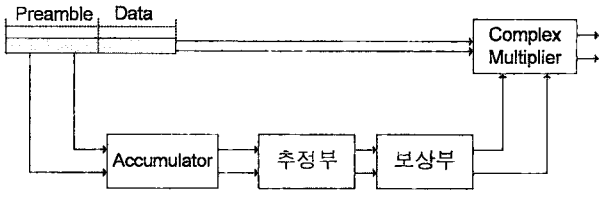

그림 1. 프리앰블을 이용한 주파수 옵셋 추정 및 보상 블록도

Fig. 1. Block diagram of frequency offset estimation and compensation using preamble.

그림 1의 동작원리는 다음과 같다. 먼저 Accumulator 블록에서는 다음의 식을 연산한다.

$$R(n)=\frac{\sum_{n=0}^{L-1} Im(r_{n,k}\times r_{n-1,k}^{*})}{\sum_{n=0}^{L-1} Re(r_{n,k}\times r_{n-1,k}^{*})} \quad (1)$$

그림 1의 추정부에서는 식 (1)을 사용하여 다음의 추정 각도를 연산하게 된다.

$$\hat{\Theta}=\frac{1}{2\pi}tan^{-1}(R(n)) \quad (2)$$

이 값이 보상부로 입력되어 $cos\theta$와 $sin\theta$를 계산하게 된다. 이 결과 값들은 그림 1의 Complex Multiplier 블록에서 위상의 보상을 위해 Multiplier로 사용된다.

우리는 이 논문에서 그림 1의 블록도에서 추정부와 보상부에 대한 저전력 구조를 제안한다. 제안 구조는 한 개의 CORDIC 하드웨어와 나눗셈기를 사용하여 저전력 구조로 구현하였다. 다음 절에서 CORDIC을 이용하여 주파수 옵셋을 추정 및 보상해주는 기존의 두 가지 구조를 먼저 살펴본다.

III. CORDIC을 이용한 기존 주파수 옵셋 추정 방식

1. CORDIC을 이용한 기존의 주파수 옵셋 추정방식

원래 CORDIC 알고리즘은 삼각함수 연산을 위해 제안되었다.[5] 이 알고리즘은 OFDM과 같은 초고속 통신 시스템의 FFT 블록, 주파수 옵셋 추정/보상 블록, IQ 복조 블록 등에서 다양하게 사용되고 있다. CORDIC 알고리즘은 다음 식과 같이 벡터의 회전으로 이루어진다.

$$\begin{aligned} x^{(i+1)} &= x^{(i)} \pm y^{(i)}2^{-i} \\ y^{(i+1)} &= y^{(i)} \mp x^{(i)}2^{-i} \end{aligned} \quad (3)$$

식 (3)에서 $x^{(i)}, y^{(i)}$값은 i번째 반복에서의 복소수의 실수부 값, 허수부 값을 나타낸다. $x^{(i)}, y^{(i)}$의 초기 값과 회전방향에 따라 arctan, cos/sin 연산이 수행된다. arctan 함수연산을 Vector 모드로 정의하고, cos/sin 연산을 Rotation 모드로 정의한다. CORDIC의 Vector 모드와 Rotation 모드를 이용하는 기존의 주파수 옵셋 추정 및 보상 방식은 다음과 같다. 먼저 Vector Mode를 이용하여 arctan값을 추정하고 Rotation Mode를 이용

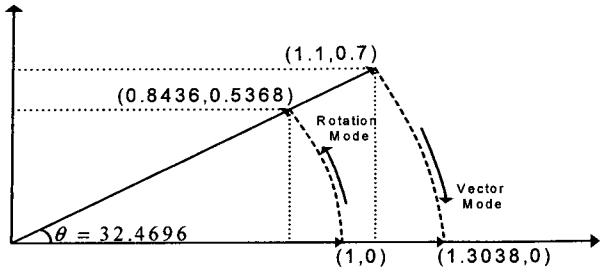

그림 2. CORDIC을 이용한 기존의 추정 및 보상 원리
Fig. 2. Conveontional estimation and compensation principle using CORDIC algorithm.

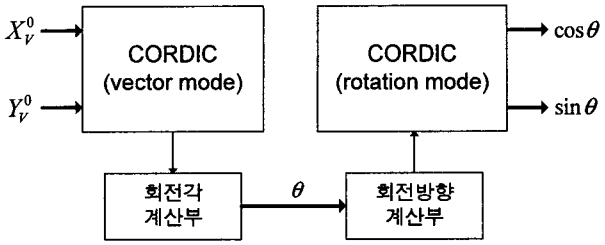

그림 3. CORDIC을 이용한 기존의 주파수 옵셋 추정 및 보상 블록도
Fig. 3. Conventioal frequency offset estimation and compensation block diagram using CORDIC algorithm.

하여 보상할 cos/sin 값을 구하는 방식이다. 즉, Vector 모드에서 구한 θ의 추정 값을 Rotation 모드에서 입력으로 받아서 $\cos\theta$와 $\sin\theta$의 보상을 수행하는 방식이다.[6] Vector 모드와 Rotation 모드를 수행하는 연산은 그림 2와 같다. 초기 벡터 값은 임의로 (1.1, 0.7)을 선택한다.

그림 2에서 보듯이, (1.1, 0.7)의 벡터는 벡터 모드를 사용하여 (1.3038, 0)으로 회전되며 이때 회전각 32.4696°를 얻는다. 이와 같이 얻어진 32.4696°가 Rotation 모드 블록으로 입력되어 이 각도 만큼 회전하게 된다. 이 Rotation 모드의 초기 값은 (1, 0)에서 시작하며 32.4696° 만큼 회전하면 (0.8436, 0.5368)의 cos/sin 값을 구하게 된다. 이와 같은 Vector 모드 블록과 Rotation 모드 블록을 수행하는 CORDIC 하드웨어의 블록도는 그림 3과 같다.

그림 3에서 보듯이 2개의 CORDIC 하드웨어로 구성되며 왼쪽의 CORDIC 하드웨어가 추정을 위한 Vector CORDIC 블록이다. 그림 3에서 16 비트 데이터는 굵은 실선으로 표기하였고 1 비트의 데이터는 가는 실선으로 표기하였다. 이 Vector 모드에서 연산되는 단계별 연산 값은 다음 표 1과 같다.

표 1에서 16단계의 최종 추정 값은 32.4696°이다. 그

표 1. Vector 모드의 결과 값
Table 1. Results of vector mode.

Vector mode			
step	X_V	Y_V	θ
[0]	1.100000000	0.700000000	
[1]	1.800000000	-0.400000000	-45.00000000
[2]	2.000000000	0.500000000	-18.43494881
[3]	2.125000000	0.000000000	-32.47119228
[4]	2.125000000	0.265625000	-25.34617594
[5]	2.141601562	0.132812500	-28.92251031
[6]	2.145751953	0.065887451	-30.71242092
[7]	2.146781444	0.032360076	-31.60759463
[8]	2.147034257	0.015588346	-32.05520880
[9]	2.147095149	0.007201494	-32.27901930
[10]	2.147109215	0.003007949	-32.39092498
[11]	2.147112152	0.000911162	-32.44687787
[12]	2.147112597	-0.000137231	-32.47485432
[13]	2.147112630	0.000386965	-32.46086609
[14]	2.147112678	0.000124866	-32.46786021
[15]	2.147112685	-0.000006182	-32.47135727
[16]	2.147112685	0.000059342	-32.46960874

표 2. Rotation Mode의 결과 값
Table 2. Results of rotation mode.

Rotation mode			
step	X_R	Y_R	θ
[0]	0.6072529351	0.0000000000	0.00000000
[17]	0.6072529351	0.6072529351	45.00000000
[18]	0.9108794026	0.3036264675	18.43494881
[19]	0.8349727857	0.5313463182	32.47119228
[20]	0.9013910755	0.4269747199	25.34617594
[21]	0.8747051555	0.4833116622	28.92251031
[22]	0.8596016660	0.5106461983	30.71242092
[23]	0.8516228192	0.5240774743	31.60759463
[24]	0.8475284639	0.5307307776	32.05520880
[25]	0.8454552968	0.5340414356	32.27901930
[26]	0.8444122471	0.5356927155	32.39092498
[27]	0.8438891097	0.5365173369	32.44687787
[28]	0.8436271384	0.5369293921	32.47485432
[29]	0.8437582247	0.5367234284	32.46086609
[30]	0.8436927067	0.5368264263	32.46786021
[31]	0.8436599414	0.5368779212	32.47135727
[32]	0.8436763256	0.5368521747	32.46960874

림 3에서 오른쪽의 CORDIC 블록은 보상을 위한 Rotation CORDIC 하드웨어이며 초기 값은 (1, 0) 대신에 1.64676으로 스케일된 (0.6072, 0)을 사용한다. 이 Rotation 모드 블록에서 32.4696° 만큼 회전하면 최종 보상 값인 (0.8436, 0.5368)의 cos/sin 값을 얻게 된다. θ를 이용하여 cos/sin을 구하는 단계별 값은 표 2와 같다. 표 2에서 보듯이 X_R과 Y_R의 마지막 32번째 값이 각각 $cos32.4696°$와 $sin32.4696°$의 값이다.

2. Compact CORDIC을 이용한 기존의 주파수 옵셋 추정방식

Compact CORDIC은 Vector 모드와 Rotation 모드를 따로 수행하지 않고 동시에 수행하는 방식이며 블록도는 그림 4와 같다. 그림 4에서의 CORDIC(vector mode) 블록과 CORDIC(rotation mode)블록의 회전방향은 CORDIC(vector mode)의 제어신호인 $Y_V^{(i)}$로 제어하여 동시에 연산을 수행한다.

Compact CORDIC은 Rotation 모드에서의 위상이 Θ로 수렴하도록 부호가 결정되기 때문에 Vector 모드에서 구하는 부호와 반대가 됨을 이용하였다.[2] 이 현상을 이용하면 그림 3의 회전각 계산부와 회전방향 계산부의 하드웨어를 없앨 수 있다. Compact CORDIC을 수식으로 표현 하면 식 (4)와 같다. Vector 모드와 Rotation 모드의 연산이 동시에 수행되며 Vector 모드의 초기 값은 누산부의 출력인 실수부 값과 허수부 값이 입력되고, Rotation 모드의 초기 값은 각각 0.6072529351과 0이 된다.

$$
\begin{aligned}
x_V^{(i+1)} &= x_V^{(i)} - d^{(i)} y_V^{(i)} 2^{-i} \\
y_V^{(i+1)} &= y_V^{(i)} + d^{(i)} x_V^{(i)} 2^{-i} \\
x_V^{(i+1)} &= x_R^{(i)} + d^{(i)} y_R^{(i)} 2^{-i} \\
y_R^{(i+1)} &= x_R^{(i)} - d^{(i)} x_R^{(i)} 2^{-i} \\
&where,\ d^{(i)} = 1,\ \text{if}\ y_V^{(i)} < 0,\ else\ d^{(i)} = -1
\end{aligned}
\tag{4}
$$

첫 번째 기존의 방법과 달리 $Y_V^{(i)}$의 값의 MSB를 참조하여 회전 방향이 결정되기 때문에 회전각 계산부와 회전방향부의 계산이 필요가 없다.

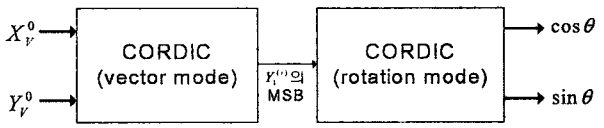

그림 4. Compact CORDIC을 이용한 주파수 옵셋 추정 및 보상 블록도

Fig. 4. Frequency offset estimation and compensation block diagram using Compact CORDIC.

IV. 제안된 CORDIC을 이용한 반송파 주파수 옵셋 및 추정방식 및 구조

제안된 CORDIC 알고리즘은 추정 및 보상 연산에 Vector 모드 한개의 CORDIC 하드웨어만을 사용한다.

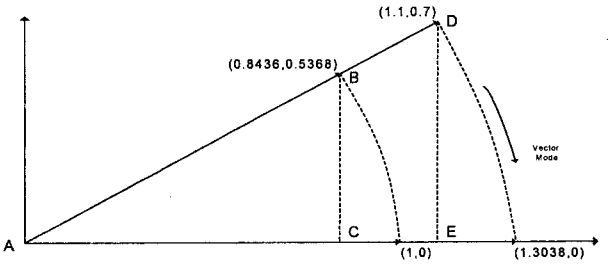

그림 5. 제안된 CORDIC 방식의 원리

Fig. 5. Principle of proposed CORDIC structure.

즉 옵셋 추정을 위한 Vector 모드 CORDIC 하드웨어를 사용하고 보상용 Rotation 모드용 CORDIC은 나눗셈기로 대치한다. 제안된 방식의 원리는 다음 그림 5를 사용하여 설명한다.

먼저 직각삼각형 $\triangle ABC$와 $\triangle ADE$를 보면 $\angle BAC$와 $\angle DAE$와 공통이고 $\angle ACB$와 $\angle AED$는 직각으로 같으므로 두 삼각형은 세 각이 모두 같은 닮은꼴이다. 따라서 세변의 비율이 다음과 같이 같다.

$$
\frac{\overline{AE}}{\overline{AC}} = \frac{\overline{DE}}{\overline{BC}} = \frac{\overline{AD}}{\overline{AB}} \tag{5}
$$

그림 5에서 회전된 벡터의 반지름은 같으므로 다음의 식이 성립한다.

$$
\frac{\overline{AD}}{\overline{AB}} = 1.3038 \tag{6}
$$

따라서 식 (5)와 (6)으로 부터 다음의 관계식을 유도할 수 있다.

$$
\frac{\overline{AE}}{\overline{AC}} = \frac{\overline{DE}}{\overline{BC}} = 1.3038 \tag{7}
$$

따라서 식 (7)로부터 $\overline{AC}$와 $\overline{BC}$를 다음과 같이 유도할 수 있다.

$$
\begin{aligned}
\overline{AC} &= \frac{\overline{AE}}{1.3038} = \frac{1.1}{1.3038} = 0.84369 \\
\overline{BC} &= \frac{\overline{DE}}{1.3038} = \frac{0.7}{1.3038} = 0.53689
\end{aligned}
\tag{8}
$$

식 (8)에서 구한 $\overline{AC}$와 $\overline{BC}$가 그림 1의 보상부에서 구해야 할 $cos\theta$와 $sin\theta$의 값이 된다. 식 (8)에서 보듯이 보상 값 $cos\theta$와 $sin\theta$는 Vector 모드에서 구한 1.3038을 사용하여 나눗셈기를 사용하여 바로 구할 수 있다. 즉, Rotation 모드를 사용하지 않고 세변의 비율에 따라 나눗셈을 사용함으로써 최종의 값을 구한다.

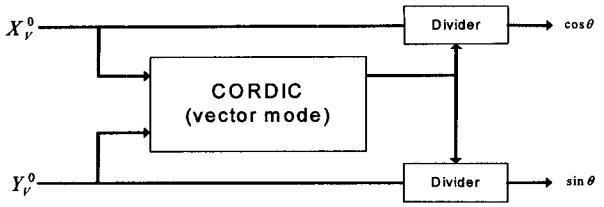

그림 6. 제안 CORDIC 구조의 주파수 옵셋 추정 및 보상 블록도

Fig. 6. Estimation and compensation block diagram of proposed CORDIC structure.

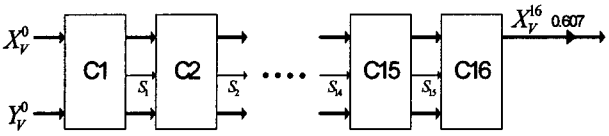

그림 7. CORDIC(vector mode)구조의 세부 블록도

Fig. 7. Detail structure of CORDIC(vector mode).

제안된 알고리즘의 하드웨어는 다음 그림 6과 같다.

그림 6에서 식 (1)의 벡터 $R(n)$값인 (1.1, 0.7)이 각각 X_V^0와 Y_V^0의 입력으로 들어가면 CORDIC(vector mode) 블록은 그림 5의 0도로 회전하여 (1.3038, 0)의 값을 얻도록 설계하였다. 즉, CORDIC 블록의 출력은 1.3038이 된다. 그림 6의 CORDIC 블록의 구현은 2가지 방식이 있다. 첫 번째는 하나의 CORDIC 하드웨어를 반복해서 사용하는 순차 CORDIC 방식으로 반복하는 횟수만큼의 클럭이 필요하며 하드웨어가 작은 대신에 동작시간이 많이 필요하다. 두 번째 방식은 병렬 CORDIC 방식으로 회전하는 횟수만큼의 CORDIC 하드웨어가 필요하나 1 클럭에 연산이 완료되는 고속 구조이다. 제안 구조는 그림 6의 CORDIC 블록을 고속으로 연산하기 위하여 병렬 CORDIC 구조로 설계하였으며 세부구조는 다음과 같이 16개의 CORDIC으로 동작하도록 설계하였다.

제안구조는 그림 7과 같이 16 스텝의 정세도를 갖는 CORDIC으로 구성하였으며 1 클럭에 동작하도록 설계하였다. 즉, (1.1, 0.7)의 벡터는 0도를 향하여 16번 회전하게 된다. 그림 7의 X_V^0와 Y_V^0는 각각의 단에서 식 (3)과 같은 연산을 하고 연산된 Y_V^n의 MSB가 S_n신호로 다음 단으로 넘어 가게 된다. 그림 7의 CORDIC C_n의 세부 구조는 그림 8과 같다.

실제로 CORDIC 하드웨어는 Pseudo rotation 연산을 하기 때문에 X_V^{16}의결과 값은 원래 벡터의 길이보다 1.64676배가 된다. 따라서 1.64676으로 나누어주어야 1.3038을 얻을 수 있다.

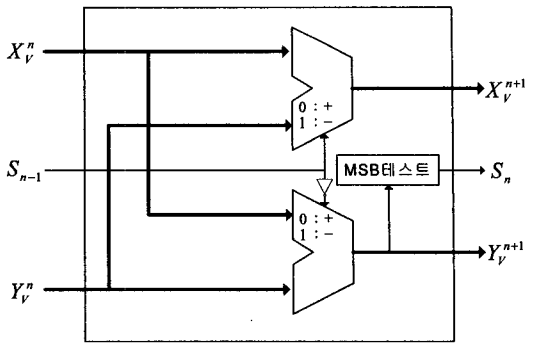

그림 8. C_n의 세부구조

Fig. 8. Detail block diagram of C_n block.

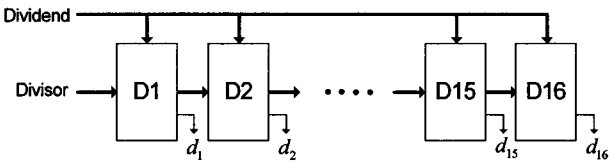

그림 9. 16 by 16 Divider의 세부 구조

Fig. 9. Datail structure of 16 by 16 Divider structure.

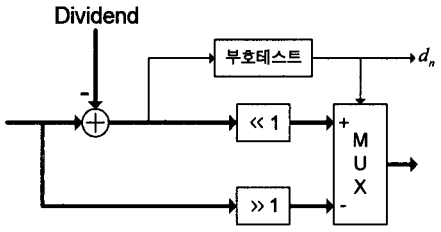

그림 10. D_n의 세부구조

Fig. 10. Detail structure of D_n.

이 1.3038의 값이 그림 6의 나눗셈기의 입력으로 들어가 식 (8)과 같이 X_V^0(=1.1)와 Y_V^0(=0.7)을 나누어 주게 된다. 그림 6의 위의 Divider의 출력이 $cos\theta$가 되며, 아래 Divider 출력이 $sin\theta$가 된다. Divider의 구조는 그림 9와 같이 설계하였다.

그림 9와 같이 Dividend 1.1은 모든 블록에 사용되며 Divisor 1.3038은 첫 번째 블록에서만 사용된다. 각각의 D_n블록에서 출력되는 1 비트의 출력 d_n을 모으면 $cos\theta$ 값인 0.84369를 얻게 된다. 즉, 벡터모드의 출력 1.3038을 Divisor로 입력시키며, 벡터 모드의 입력 값 X_V^0와 Y_V^0를 Dividend로 사용하여 16비트 나누기 연산을 수행한다. D_n의 세부 구조는 그림 10과 같이 설계하였다.

D_n블록에서는 Dividend와 Divisor의 크기를 비교하여 d_n의 값을 결정한다. 이 1 비트 d_n의 값이 몫의 n번째 자리가 되며 이 값이 양수이면 MUX의 위쪽이 선택되며 음수이면 아래쪽이 선택되도록 제어한다. (1.1, 0.7)의 벡터가 입력되었을 때, 제안구조의 단계별 연산

표 3. 제안구조의 결과 값
Table 3. Results of proposed structure.

Vector mode		
step	X_V	Y_V
[0]	1.100000000	0.700000000
[1]	1.800000000	-0.400000000
[2]	2.000000000	0.500000000
[3]	2.125000000	0.000000000
[4]	2.125000000	0.265625000
[5]	2.141601562	0.132812500
[6]	2.145751953	0.065887451
[7]	2.146781444	0.032360076
[8]	2.147034257	0.015588346
[9]	2.147095149	0.007201494
[10]	2.147109215	0.003007949
[11]	2.147112152	0.000911162
[12]	2.147112597	-0.000137231
[13]	2.147112630	0.000386965
[14]	2.147112678	0.000124866
[15]	2.147112685	-0.000006182
[16]	2.147112685	0.000059342
[17]	1.303840480	
[18]	0.834661488	0.5368754924

의 결과 값을 살펴보면 표 3과 같다.

표 3의 연산 단계 중 16 스텝까지의 결과는 Vector 모드의 단계별 연산 값이고 17 스텝이 벡터의 길이를 식 (6)과 같은 방법으로 스케일링한 값 1.3038이다. 마지막 18 스텝이 Divider의 결과 값인 $cos\theta$와 $sin\theta$가 된다. 표 3에서는 18 단계로 계산 결과를 나타내었으나 단계별로 클럭이 필요한 것은 아니다. 즉, 그림 7의 CORDIC 하드웨어는 조합논리회로를 사용하여1 클럭에 동작하도록 설계하였다.

V. 구현 및 고찰

1. Verilog-HDL 시뮬레이션

이 절에서는 제안구조와 기존 구조들을 HDL로 시뮬레이션하여 동작을 검증하기로 한다. 이 절에서 시뮬레이션 할 구조들의 세부 사양은 다음 표 4와 같다.

이 절의 시뮬레이션의 각각의 구조의 입력 값들은 16비트의 정세도를 사용하였으며, 부호 1비트, 정수 2비트, 소수 13비트로 구성하였다. 각각의 구조들은 고정소수점 방식으로 C 코딩을 통하여 구조를 검증하였다.

기존구조 1은 그림 3과 같이 Vector모드와 회전각 계산부를 거쳐 얻어진 θ값을 회전방향 계산부와 Rotation 모드에 입력하여 최종적으로 cos, sin값을 얻도록 하였다. 게이트 딜레이를 고려하여 최종 출력까지 총 2 clock이 걸리도록 Vector 모드와 Rotation 모드를 개별

표 4. 시뮬레이션하는 제안 구조와 기존 구조
Table 4. Proposed and conventional structures for simulation.

세부사양	기존구조 1	기존구조 2	제안구조
블록도	그림 3	그림 4	그림 6
CORDIC 방식	병렬	병렬	병렬
병렬 CORDIC수	2	2	1
clock	2	1	2
기타 하드웨어	회전각계산부 회전방향계산부	-	Divider

모듈로 설계하였다. 그림 3을 구현할 때 각각의 제어신호가 high일 때 동작하도록 제어하여 설계하였다. 또한 기존 구조의 Rotation 모드는 제어신호 rotation_enb 및 Vector모드와 회전각 계산부에서 계산된 출력 값 θ를 입력으로 받아야 동작하게 되므로 Vector 모드 및 회전각 계산부에서 1 클럭, 그리고 회전방향 계산부와 Rotation모드에서 1 클럭을 사용하였다. 회전각계산부에서 사용되는 총 16개의 θ값들은 LUT에 저장하여 사용하였고, Rotation모드에서 입력 값으로 받아들여지는 0.6072529351의 값과 0값은 고정된 값이므로 구현할 때에 입력 값으로 주지 않고 변수 선언하여 사용하였다.

기존구조 2의 Compact CORDIC 구조는 회전각 계산부나 회전방향 계산부가 필요하지 않고 하나의 제어신호로 Vector 모드와 Rotation 모드가 동시에 연산 된다. 최종 출력까지 Vector 모드에 1 클럭, Rotation 모드에 1 클럭을 사용하여 2 클럭에 완료되도록 시뮬레이션하였다.

마지막으로 제안구조는 그림 6의 블록도로 θ의 값을 구하는 Vector 모드 블록과 $cos\theta$와 $sin\theta$를 구하는 나눗셈기 블록으로 나누어 구현하였다. 제안구조의 시뮬레이션에서는 2 클럭 동안에 연산이 되도록 조합논리회로로 구현하였다. 나눗셈기는 한 번에 하나의 출력을 내도록 $cos\theta$와 $sin\theta$를 구하는데 각각 1 클럭을 사용하였다. 지금까지 기존구조 및 제안구조에 대한 시뮬레이션 하드웨어 구조를 살펴보았다. 이와 같은 방법으로 각각을 RTL 코딩하였고 Function Simulation으로 출력 값을 확인하였다. 다음 절에서는 완성된 RTL코드로 Synthesis와 Pre-Simulation을 통하여 Front-end 칩 구현을 진행한다.

2. 칩 구현

이 절에서는 지난 절에서 시뮬레이션한 세 구조들에

대하여 Front-end 칩 구현을 진행하였다. 먼저 Synopsys Design Compiler 합성 Tool을 사용하여 각각의 구조를 합성하여 구조에 대한 면적을 비교해 보기로 한다. 합성을 위한 Frond-end 과정에서는 매그나칩 0.25-Micron 2.5V 공정을 사용하였고, 셀 라이브러리는 hsm222a_bcc를 사용하였다. 각 구조 마다 동일한 제한조건 적용하여 구조검증을 공정하게 진행하였으며, 각각의 구조의 합성결과에 대한 Area_Report를 살펴보도록 한다.

먼저 기존구조 1과 2에 대한 Area_Report는 다음과 같다.

표 5. 기존구조1의 합성결과에 대한 Area_Report
Table 5. Area_Report for synthesis result of conventional structure 1.

구분	Area_Report
Hierarchial Cell Count	46
Hierarchial Port Count	2350
Leaf Cell Count	3473
Combinational Area	177269.764688
Design Area	177269.764688

표 6. 기존구조2의 합성결과에 대한 Area_Report
Table 6. Area_Report for synthesis result of conventional structure 2.

구분	Area_Report
Hierarchial Cell Count	48
Hierarchial Port Count	2448
Leaf Cell Count	1914
Combinational Area	121121.283331
Design Area	121121.283331

마지막으로 제안구조에 대한 Area_Report는 다음 같다.

표 7. 제안구조의 합성결과에 대한 Area_Report
Table 7. Area_Report for synthesis result of proposed structure.

구분	Area_Report
Hierarchial Cell Count	39
Hierarchial Port Count	2019
Leaf Cell Count	1741
Combinational Area	94354.562271
Design Area	94354.562271

표 8. 제안구조의 면적비교
Table 8. Gate count comparison for proposed structure.

구분	기존구조1	기존구조2	제안구조
셀	3473	1914	1741
면적(㎛²)	177269	121121	94354
gate equivalent	10258	7009	5460
면적비율	100%	68.3%	53.2%
면적감소율	-	31.6%	46.8%

Front-end 구현 결과를 정리하면 표 8과 같다.

각 구조의 Pre-Simulation을 위하여 netlist와 SDC, SDF 파일을 write하여 PrimeTime(Synopsys)을 이용하여 각 구조에 대한 타이밍을 검증하였고 Formality (Synopsys)를 사용하여 function을 검증하였다. 표 8에서 보듯이 제안구조는 기존구조 1과 비교하여 46.8%의 면적이 감소되었고 기존구조 2와 비교하면 22.1%의 면적이 감소됨을 알 수 있었다. Front-end 구현을 통하여 제안구조가 저전력으로 구현될 수 있음을 검증하였다.

VI. 결 론

이 논문에서는 프리앰블 신호를 이용한 반송파 주파수 옵셋의 추정 및 보상 블록에 대한 저전력 구조를 제안하였다. 기존 방식들은 Vector CORDIC을 이용하여 위상을 추정한 후에 그 결과 값을 이용하여 다시 Rotation CORDIC을 사용하여 보상 값을 구하는 방식이다. 제안구조는 Vector CORDIC 1개와 나눗셈 연산을 사용하여 저전력 옵셋 추정 및 보상 구조를 얻을 수 있었다. 즉, Vector CORDIC 모드가 수행되면 추정 위상 각 뿐 아니라 보상해야할 비율이 계산되므로 이를 이용하여 바로 원래의 벡터로부터 보상 값을 바로 구하는 방식을 제안하였다. Front-end 구현을 통하여 제안구조는 기존구조 보다 22.1%의 면적이 감소됨을 보였다.

참 고 문 헌

[1] J. Granado, A. Torralba, J. Chavez and V. Baena-Lecuyer "Design of an Efficient CORDIC-Based Architecture for Synchronization in OFDM," IEEE Transactions on Consumer Electronics, Vol. 52, No. 3, pp. 774-782, August

2006.
[2] Kyu In Lee, Jong Han Lee, Jae Kon Lee and Yong Soo Cho, "A Compact CORDIC Algorithm for Synchronization of Carrier Frequency Offset in OFDM Modems," IEICE Transactions on Communications, Vol. E89-B, No .3, pp. 952-954, 2006.
[3] P. H. Moose, "A technique for orthogonal frequency division multiplexing frequency offset correction," IEEE Transactions on Communications, Vol. 42, No. 10, pp. 2908-2914, October 1994.
[4] Jan-Jaap van de Beek, Magnus Sandell and Per Ola Borjesson, "ML estimation of time and frequency offset in OFDM systems," IEEE Transactions on Signal Processing, Vol. 45, No. 7, pp. 1800-1805, July 1997.
[5] Jack E. Volder, "The CORDIC Trigonometric Computing Technique," IRE Transactions on Electronic Computers, Vol. EC-8, No. 3, pp. 330-334, 1959.
[6] Jouko Vankka, "Methods of mapping from phase to sine amplitude in direct digital synthesis," IEEE Transactions on Communications, Vol. 44, No. 2, March 1997.

저 자 소 개

장 영 범(정회원)
1981년 연세대학교 전기공학과 졸업.(공학사)
1990년 Polytechnic University 대학원 졸업.(공학석사)
1994년 Polytechnic University 대학원 졸업.(공학박사)
1981년~1999년 삼성전자 System LSI 사업부 수석연구원.
2000년~2002년 이화여자대학교 정보통신학과 연구교수.
2002년~현재 상명대학교 정보통신공학과 교수.
<주관심분야 : 통신신호처리, 비디오신호처리, SoC 설계>

홍 대 기(정회원)
2003년 연세대학교 전자공학과 대학원 졸업.(공학박사)
2002년~2006년 전자부품연구원 선임연구원.
2006년~현재 상명대학교 정보통신공학과 교수.
<주관심분야 : Digital and Wireless Communication, WPAN>

한 재 웅(학생회원)
2007년 상명대학교 정보통신공학과 졸업.(공학사)
2007년~현재 상명대학교 컴퓨터정보통신공학과 대학원 석사 과정
<주관심분야 : 통신신호처리, SoC 설계>