논문 2009-46SD-5-5

R2SDF FFT의 메모리 감소를 위한 회전인자 인덱스 생성방법

(Twiddle Factor Index Generate Method for Memory Reduction in R2SDF FFT)

양 승 원*, 김 용 은*, 이 종 열**

(Seung-Won Yang, Yong-Eun Kim, and Jong-Yeol Lee)

요 약

FFT(Fast Fourier Transform) 프로세서는 OFDM(Orthogonal Frequency Division Multiplexing) 시스템에서 사용된다. 근래에는 광대역과 이동성에 대한 요구가 높아짐에 따라 큰 포인트를 가지는 FFT 프로세서의 연구가 필요하다. FFT 포인트 수가 증가할수록 회전인자가 저장된 메모리가 차지하는 면적은 증가한다. 본 논문에서는 Radix-2, 2^2, 2^3, 2^4 알고리즘의 회전인자 인덱스 생성 방법을 제안한다. 제안한 회전인자 인덱스 생성기(Twiddle Factor Index Generator : TFIG)는 간단하게 카운터와 양수곱셈기로만 구성된다. 각각의 R2SDF(Radix-2 Single-Path Delay Feedback), R2^2SDF, R2^3SDF, R2^4SDF 1024포인트 FFT 프로세서에 ROM 크기를 1/8N로 줄인 회전인자 계수 생성기(Twiddle Factor Coefficient Generator : TFCG)를 설계하여 제안한 알고리즘을 검증하였다. R2^4SDF의 TFCG 경우 면적, 전력에서 각 57.9%, 57.5%정도의 이득을 얻었다.

Abstract

FFT(Fast Fourier Transform) processor is widely used in OFDM(Orthogonal Frequency Division Multiplesing) system. Because of the increased requirement of mobility and bandwidth in the OFDM system, they need large point FFT processor. Since the size of memory which stores the twiddle factor coefficients are proportional to the N of FFT size, we propose a new method by which we can reduce the size of the coefficient memory. In the proposed method, we exploit a counter and unsigned multiplier to generate the twiddle factor indices. To verify the proposed algorithm, we design TFCGs(Twiddle Factor Coefficient Generator) for 1024point FFTs with R2SDF(Radix-2 Single-Path Delay Feedback), R2^2SDF, R2^3SDF, R2^4SDF architectures. The size of ROM is reduced to 1/8N. In the case of R2^4SDF architecture, the area and the power are reduced by 57.9%, 57.5% respectively.

Keywords : FFT, R2SDF, twiddle factor coefficient, memory, ROM

I. 서 론

FFT(Fast Fourier Transform) 프로세서의 구현에 대한 중요도가 점차 증가하고 있다. 여러 응용 분야의 OFDM(Orthogonal Frequency Division Multiplexing) 시스템에서는 높은 성능과 다양한 포인트뿐 아니라 저전력 저면적 FFT 프로세서가 필요하다. FFT의 포인트가 증가할수록 FFT 프로세서의 메모리는 증가한다. 파이프라인 FFT 프로세서의 메모리는 버터플라이 연산과 복소곱셈연산 후 중간 데이터를 저장하는 레지스터와 회전인자 계수가 저장된 메모리로 구성된다. 파이프라인 구조의 R2SDF(Radix-2 Single-Path Delay Feedback) FFT 프로세서는 고속처리가 가능하고, 면적대비 처리속도 관점에서 우수하고 비교적 구조가 간단하여 큰 포인트 FFT 프로세서를 요구하는 OFDM 시스템에 적합하다.

* 학생회원, ** 정회원, 전북대학교 전자정보공학부
(Division of Electronic & Information Engineering Chonbuk National University)

※ This work was supported by MOCIE(Ministry Commerce Industry and Energy), Korea under the IDEC support program(MPW, CAD) and samsung 0.35㎛

접수일자: 2009년2월16일, 수정완료일: 2009년4월10일

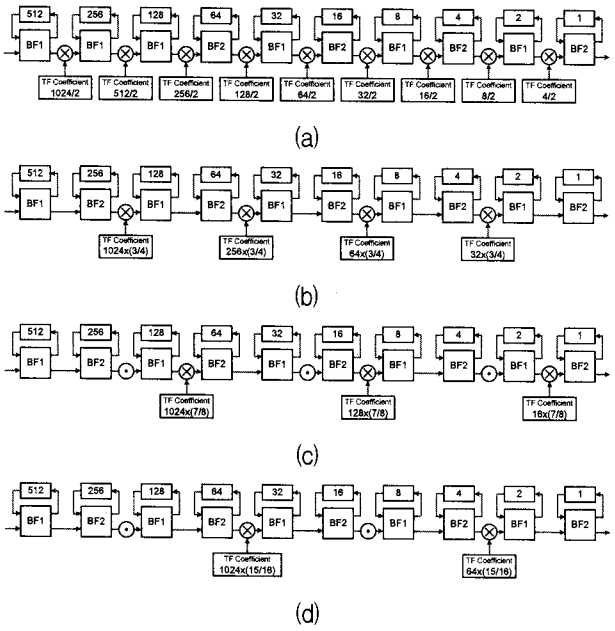


그림 1. 1024point R2^{m}SDF FFT 아키텍처, (a) R2SDF, (b) R2^{2}SDF, (c) R2^{2}SDF, (d) R2^{4}SDF

Fig. 1. 1024point R2^{m}SDF FFT Architecture, (a) R2SDF, (b) R2^{2}SDF, (c) R2^{2}SDF, (d) R2^{4}SDF.

또한 좀 더 복소곱셈을 줄여 효율을 높인 R2^{2}SDF, R2^{3}SDF, R2^{4}SDF FFT 아키텍처 등이 제안되었다[1~4].

일반적으로 R2SDF FFT 프로세서의 복소곱셈 연산은 회전인자 계수를 ROM에 곱해질 순서대로 저장 후 카운터로 저장된 데이터를 불러서 복소곱셈 연산을 수행한다. 그림 1은 1024 포인트 R2^{m}SDF FFT 프로세서에서 요구하는 회전인자 계수의 수를 보여준다. 일반적으로 R2^{2}SDF는 $3/4N$, R2^{3}SDF는 $7/8N$, R2^{4}SDF는 $15/16N$ 크기의 메모리를 요구한다. R2SDF와 R2^{2}SDF FFT 아키텍처에는 정현파 성질을 이용하여 ROM에 저장된 회전인자 계수를 줄여 ROM의 크기를 $1/8N+1$로 줄이는 방법이 제안되었다[5~6]. 기존에는 회전인자 인덱스를 생성하는 표준화된 방법이 없어서 R2^{2}SDF, R2^{3}SDF, R2^{4}SDF FFT 아키텍처에는 적용하기 어려웠다.

본 논문에서는 Radix-2, 2^2, 2^3, 2^4 FFT 알고리즘의 회전인자 W^{nk}를 카운터 신호를 n과 k로 정의하고 회전인자 인덱스를 생성하는 알고리즘을 제안한다. 제안한 알고리즘을 R2^{2}SDF, R2^{3}SDF, R2^{4}SDF FFT 구조에 적용하고 정현파의 성질을 이용하여 ROM 크기를 $1/8N+1$로 줄인 회전인자 계수 생성기를 구현하여 제안한 알고리즘을 검증하였다. Ⅱ장에서는 회전인자 인덱스를 생성하는 방법과 제안한 회전인자 인덱스 생성기(Twiddle Factor Index Generator : TFIG)와 회전인자 계수 생성기(Twiddle Factor Coefficient Generator : TFCG)에 대해 설명한다. Ⅲ장에서는 제안한 TFCG 합성 결과와 복소곱셈연산 블록에 적용할 때 TFCG 최적화 방법을 설명하고 Ⅳ장에서 결론을 맺는다.

Ⅱ. 제안한 회전인자 인덱스 생성 알고리즘

1. Radix-2^m FFT 회전인자

DFT 알고리즘 식(1)과 같다.

$$X[k]=\sum_{n=0}^{N-1}x[n]W_N^{nk}$$

$$W_N^{nk}=e^{-j\frac{2\pi nk}{N}} \qquad (n,k=0,\ 1,\ \cdots,\ N-1) \tag{1}$$

식(1)에 각 2, 3, 4, 5 차원 인덱스 식(2)를 적용하고 간략화 하면,

$$\begin{aligned}&(k=k_1+2k_2)\quad (k_1=0,1,\ k_2=0,1,\cdots,\frac{N}{2}-1)\\&(n=\frac{N}{2}n_1+n_2)\quad (n_1=0,1,\ n_2=0,1,\cdots,\frac{N}{2}-1)\end{aligned} \tag{2}$$

$$\begin{aligned}&(k=k_1+2k_2+4k_3)\quad (k_1,k_2=0,1,\ k_3=0,1,\cdots,\frac{N}{4}-1)\\&(n=\frac{N}{2}n_1+\frac{N}{4}n_2+n_3)\quad (n_1,n_2=0,1,\ n_3=0,1,\cdots,\frac{N}{4}-1)\end{aligned}$$

$$\begin{aligned}&(k=k_1+2k_2+4k_3+8k_4)\quad (k_1,k_2,k_3=0,1,\ k_4=0,1,\cdots,\frac{N}{8}-1)\\&(n=\frac{N}{2}n_1+\frac{N}{4}n_2+\frac{N}{8}n_3+n_4)\quad (n_1,n_2,n_3=0,1,\ n_4=0,1,\cdots,\frac{N}{8}-1)\end{aligned}$$

$$\begin{aligned}&(k=k_1+2k_2+4k_3+8k_4+16k_5)\quad (k_1,k_2,k_3,k_4=0,1,\ k_5=0,1,\cdots,\frac{N}{16}-1)\\&(n=\frac{N}{2}n_1+\frac{N}{4}n_2+\frac{N}{8}n_3+\frac{N}{16}n_4+n_5)\quad (n_1,n_2,n_3,n_4=0,1,\ n_5=0,1,\cdots,\frac{N}{16}-1)\end{aligned}$$

각 Radix-2, 2^2, 2^3, 2^4 FFT 알고리즘은 식(3)과 같다[2~3].

$$X[k]=\sum_{n_2=0}^{N/2-1}\sum_{n_1=0}^{1}x[n]\underbrace{(-1)^{n_1k_1}}_{\text{1st BF}}\underbrace{W_N^{n_2k_1}}_{\text{1st TF}}W_{N/2}^{n_2k_2}$$

$$X[k]=\sum_{n_3=0}^{N/4-1}\sum_{n_2=0}^{1}\sum_{n_1=0}^{1}x[n]\underbrace{(-1)^{n_1k_1}}_{\text{1st BF}}\underbrace{(-j)^{n_2k_1}}_{\text{1st TF}}\underbrace{(-1)^{n_2k_2}}_{\text{2nd BF}}\underbrace{W_N^{n_3(k_1+2k_2)}}_{\text{2nd TF}}W_{N/4}^{n_3k_3}$$

$$X[k]=\sum_{n_4=0}^{N/8-1}\sum_{n_3=0}^{1}\sum_{n_2=0}^{1}\sum_{n_1=0}^{1}x[n]\overbrace{(-1)^{n_1k_1}}^{\text{1st BF}}\overbrace{(-j)^{n_2k_1}}^{\text{1st TF}}$$
$$\bullet\underbrace{(-1)^{n_2k_2}}_{\text{2nd BF}}\underbrace{W_8^{n_3(k_1+k_2)}}_{\text{2nd TF}}\underbrace{(-1)^{n_3k_3}}_{\text{3th BF}}\underbrace{W_N^{n_4(k_1+2k_2+4k_3)}}_{\text{3th TF}}W_{N/8}^{n_4k_4}$$

$$X[k]=\sum_{n_5=0}^{N/16-1}\sum_{n_4=0}^{1}\sum_{n_3=0}^{1}\sum_{n_2=0}^{1}\sum_{n_1=0}^{1}x[n]\overbrace{(-1)^{n_1k_1}}^{\text{1st BF}}\overbrace{(-j)^{n_2k_1}}^{\text{1st TF}}$$
$$\bullet\underbrace{(-1)^{n_2k_2}}_{\text{2nd BF}}\underbrace{W_{16}^{(2n_3+n_4)(k_1+2k_2)}}_{\text{2nd TF}}\underbrace{(-1)^{n_3k_3}}_{\text{3th BF}}\underbrace{(-j)^{n_4k_3}}_{\text{3th TF}} \quad (3)$$
$$\bullet\underbrace{(-1)^{n_4k_4}}_{\text{4th BF}}\underbrace{W_N^{n_5(k_1+2k_2+4k_3+8k_4)}}_{\text{4th TF}}W_{N/16}^{n_5k_5}$$

모든 회전인자 W^{nk}식은 nk곱셈으로 나타난다. 각 Radix-2^m FFT 알고리즘에서 n,k의 범위는 식(2)의 인덱스의 범위와 같다.

표 1과 같이 복소곱셈의 회전인자 k는 각 Radix-2의 1st TF에서 0, 1, Radix-2^2의 2nd TF에서 0, 2, 1, 3로 Radix-2^3의 3th TF에서 0, 4, 2, 6, 1, 5, 3, 7로 Radix-2^4의 4th TF에서 0, 8, 4, 12, 2, 10, 6, 14, 1, 9, 5, 13, 3, 11, 7, 15순으로 변한다. Radix-2^4의 2nd TF에서 nk는 0, 0, 0, 0, 0, 2, 4, 6, 0, 1, 2, 3, 0, 3, 6, 9순으로 변한다. 각 알고리즘의 차수 m이 증가할수록 k는 가변적으로 변한다.

표 1. 회전인자 W^{nk}의 계수 형태
Table 1. Coefficient of twiddle factor W^{nk}.

Radix-2		Radix-2^2		Radix-2^3		Radix-2^4				
	1st TF		2nd TF		3th TF		4th TF	2nd TF		
k_1	$W_N^{n_2k_1}$	k_2	$W_N^{n_3(k_1+2k_2)}$	k_3	$W_N^{n_4(k_1+2k_2+4k_3)}$	k_4	$W_N^{n_5(k_1+2k_2+4k_3+8k_4)}$	$W_{16}^{(2n_3+n_4)(k_1+2k_2)}$	n_4	n_3
0	$W_N^{0n_2}$	0	$W_N^{0n_3}$	0	$W_N^{0n_4}$	0	$W_N^{0n_5}$	W_{16}^{0}	0	0
						1	$W_N^{8n_5}$	W_{16}^{0}	1	
				1	$W_N^{4n_4}$	0	$W_N^{4n_5}$	W_{16}^{0}	0	1
						1	$W_N^{12n_5}$	W_{16}^{0}	1	
		1	$W_N^{2n_3}$	0	$W_N^{2n_4}$	0	$W_N^{2n_5}$	W_{16}^{0}	0	0
						1	$W_N^{10n_5}$	W_{16}^{2}	1	
				1	$W_N^{6n_4}$	0	$W_N^{6n_5}$	W_{16}^{4}	0	1
						1	$W_N^{14n_5}$	W_{16}^{6}	1	
1	$W_N^{1n_2}$	0	$W_N^{1n_3}$	0	$W_N^{1n_4}$	0	$W_N^{1n_5}$	W_{16}^{0}	0	0
						1	$W_N^{9n_5}$	W_{16}^{1}	1	
				1	$W_N^{5n_4}$	0	$W_N^{5n_5}$	W_{16}^{2}	0	1
						1	$W_N^{13n_5}$	W_{16}^{3}	1	
		1	$W_N^{3n_3}$	0	$W_N^{3n_4}$	0	$W_N^{3n_5}$	W_{16}^{0}	0	0
						1	$W_N^{11n_5}$	W_{16}^{3}	1	
				1	$W_N^{7n_4}$	0	$W_N^{7n_5}$	W_{16}^{6}	0	1
						1	$W_N^{15n_5}$	W_{16}^{9}	1	

2. 회전인자 생성 알고리즘

회전인자 W^{nk}의 값은 nk, 즉 인덱스에 따라 회전인자 계수 값이 결정된다.

가. n, k와 카운터 신호의 상관관계

표 1과 같이 회전인자 nk의 곱으로 회전인자 계수가 결정된다. 2^rpoint FFT에서 r은 회전인자 인덱스를 생성하는 카운터의 비트수를 결정하고 Radix-2^m FFT 알고리즘에서 m은 k의 비트수를 결정하며 n의 비트수는 r-m으로 결정한다. 각 알고리즘의 k의 시퀀스 k_{sqen}는 할당된 카운터 신호 MSB(Most Significant Bit)를 비트 리버스 하여 생성하고 n의 시퀀스 n_{sig}는 k_{sqen}의 비트를 제외한 비트이다. 표 2는 카운터 신호와 k의 상관관계를 나타낸다.

표 2. 카운터 신호 이용한 k_{sqen}생성
Table 2. k_{sqen} generation using counter signal.

Radix-2

$C_{sig}[r-1]$		$C_{sig}[r-1]$	
decimal	binary	binary	decimal
0	0	0	0
1	1	1	1

Radix-2^2

$C_{sig}[r-1:r-2]$		$C_{sig}[r-2:r-1]$	
decimal	binary	binary	decimal
0	00	00	0
1	01	10	2
2	10	01	1
3	11	11	3

Radix-2^3

$C_{sig}[r-1:r-3]$		$C_{sig}[r-3:r-1]$	
decimal	binary	binary	decimal
0	000	000	0
1	001	100	4
2	010	010	2
3	011	110	6
4	100	001	1
5	101	101	5
6	110	011	3
7	111	111	7

Radix-2^4

$C_{sig}[r-1:r-4]$		$C_{sig}[r-4:r-1]$	
decimal	binary	binary	decimal
0	0000	0000	0
1	0001	1000	8
2	0010	0100	4
3	0011	1100	12
4	0100	0010	2
5	0101	1010	10
6	0110	0110	6
7	0111	1110	14
8	1000	0001	1
9	1001	1001	9
10	1010	0101	5
11	1011	1101	13
12	1100	0011	3
13	1101	1011	11
14	1110	0111	7
15	1111	1111	15

k_{sqen}는 카운터 신호 $C_{sig}[r-1:r-m]$를 비트리버스한 신호 $C_{sig}[r-m:r-1]$와 같고, n_{sig}는 k_{sqen}를 제외한 $C_{sig}[r-m-1:0]$와 같다. k_{sqen}, n_{sig}은 식(4)과 같다.

$$k_{sqen} = C_{sig}[r-1:r-m]$$
$$n_{sig} = C_{sig}[r-m-1:0] \quad (4)$$

예를 들어 R2^{4}SDF 1024point FFT의 경우, 1024는 2^{10} 이므로 r=10이므로 10비트 카운터가 결정된다. m=4 이므로 k_{sqen}는 MSB 4비트 $C_{sig}[6:9]$이고, n_{sig}는 LSB(Least Significant Bit) 6비트 $C_{sig}[5:0]$가 할당된다.

나. 회전인자 인덱스 생성기 (TFIG)

회전인자 인덱스 생성은 k_{sqen}와 n_{sig}의 곱으로 구할 수 있다. 그림 2는 회전인자 인덱스 생성기(TFIG) 구성도이다. 2^{r}point FFT일 때, (a)와 같이 카운터와 양수곱셈기로 구성한다. 예를 들어 R2SDF는 1×(r-1), R2^{2}SDF는 2×(r-2), 2^{3}SDF는 3×(r-3), 2^{4}SDF는 4×(r-4)의 양수곱셈기가 필요하다. 표 3와 같이 R2SDF는 k_{sqen}는 1비트이므로 and 게이트로 구성되고, R2^{2}SDF는 2비트이므로 덧셈기로 TFIG를 구성된다. k_{sqen}와 n_{sig}의 곱은 k_{sqen}를 n_{sig}번 카운트한 것과 같다. (b)와 같이 카운터와 레지스터 그리고 2^{r-n}번 마다 레지스터를 리셋 하는 로직으로 구성한다.

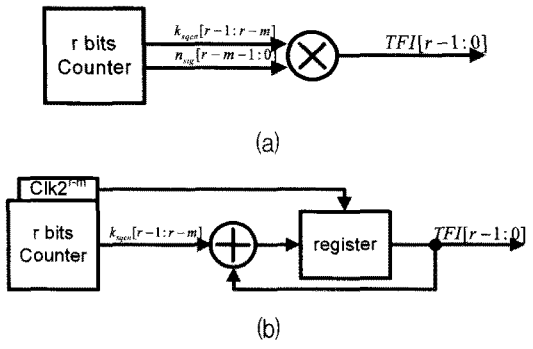


그림 2. 회전인자 인덱스 생성기 (TFIG)
(a) 양수곱셈기 타입, (b) 레지스터 타입
Fig 2. Twiddle factor index generator (TFIG).
(a) unsigned multiplier type (b) register type

표 3. TFIG 구성요소
Table 3. TFIG element.

		counter	calculator	register
unsigned multiplier type	R2SDF	rbits	and gate	×
	R2^{2}SDF	rbits	adder	×
	R2^{3}SDF	rbits	3x(r-3)	×
	R2^{4}SDF	rbits	4x(r-4)	×
register type		rbits	adder	○

다. 정현파 성질을 이용한 제안한 알고리즘 적용

FFT의 회전 인자는 사인과 코사인 값으로 이루어져 있고, 정현파는 일정한 2π 주기로 반복되는 특징이 있다. 그림 3은 $0 \sim 2\pi$구간의 회전인자 $\cos(2\pi nk/N)$과 $-\sin(2\pi nk/N)$의 파형을 보여준다. 위의 함수를 샘플링 하면 MSB 3비트는 구간을 나타낸다. 그림 3에서 [000]구간의 cos함수 값은 [001]구간의 -sin함수와 보수, 대칭, [010]구간의 -sin함수와 보수, [011]구간의 cos함수와 보수, 대칭. [100]구간의 cos 함수와 보수, [101] 구간의 -sin함수와 대칭, [110]구간의 -sin함수와 같고, [111]구간의 cos함수와 대칭관계이다. [000]구간의 -sin함수 값은 [001]구간의 cos함수와 보수, 대칭, [010]구간의 cos함수와 같고, [011]구간의 -sin함수와 대칭. [100]구간의 -sin함수와 보수, [101]구간의 cos함수와 대칭, [110]구간의 -sin 함수와 보수, [111]구간의 -sin함수와 보수, 대칭관계이다. 정현파의 성질을 이용하면 1/8구간의 값으로 모든 회전인자 계수를 알 수 있다. 회전인자 인덱스 MSB 3비트 $TFI[r-1:r-3]$는 구간을 의미한다.

그림 4는 ROM을 $1/8N$로 줄인 제안한 회전인자 계수 생성기(TFCG)이다. 회전인자 인덱스 $TFI[r-1:0]$의 $TFI[r-3]$는 대칭신호로 정의 한다. 이것은 ROM의 주소 ROM_{addr}가 역 으로 증가함을 의미한다. 예를 들어 $TFI[r-3]$이 0일 때 $TFI[r-4:0]$가 ROM_{addr}이고, $TFI[r-3]$이 1일 때는 $TFI[r-3]$에 나머지 $TFI[r-4:0]$

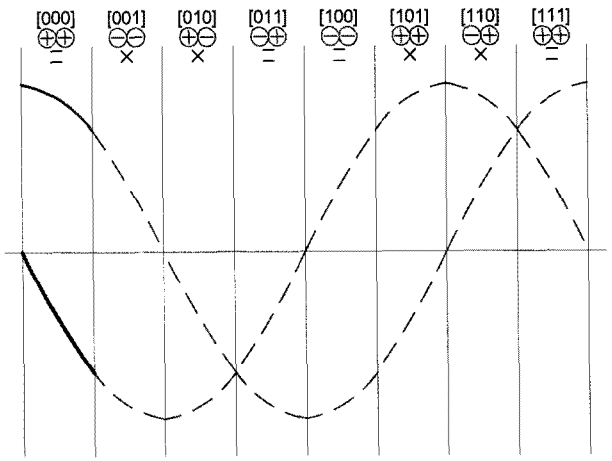


그림 3. 회전인자 W^{nk} 파형
Fig 3. Twiddle factor W^{nk} wave.

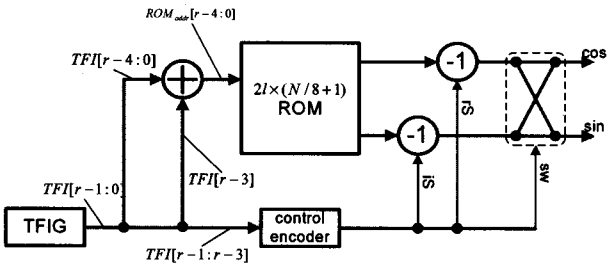


그림 4. 제안한 회전인자 계수 생성기 (TFCG)
Fig 4. Proposed twiddle factor coefficient generator (TFCG).

표 4. 제어 인코더 진리표
Table 4. Control encoder truth table.

$TFI[r-1:r-3]$	control encoder			symmtry $TFI[r-3]$
	rS	iS	sw	
000	0	0	0	0
001	1	1	1	1
010	1	0	1	0
011	1	0	0	1
100	1	1	0	0
101	0	0	1	1
110	0	1	1	0
111	0	1	0	1

표 5. TFCG 구성요소
Table 5. TFCG element.

TFIG	ROM	control encoder	adder	switching circuit
1	$1/8N+1$	1	3	1

인덱스 값을 빼준 값이 ROM_{addr}이다. 제어 인코더는 표 4와 같이 $TFI[r-1:r-3]$를 입력으로 받아 보수, 스위칭 신호를 출력한다. 제안한 TFCG는 표 5와 같이 $0\sim\pi/4$ 구간까지의 회전인자 계수의 값 cos, -sin 각 l비트씩 ROM에 저장한 $2l\times(1/8N+1)$의 크기의 ROM, 2의 보수기 2개, 가감산기, 스위칭 회로와 제어 인코더로 구성한다.

III. 구 현

1. 회전인자 계수 생성기(TFCG) 구현 결과

N이 1024이고 회전인자 계수의 실수, 허수를 각 12bit씩으로 하여 $1/8N$의 크기의 ROM을 가지는 제안한 TFCG와 각 $R2^2$SDF는 $3/4N$, $R2^3$SDF는 $7/8N$, $R2^4$SDF는 $15/16N$ 크기의 ROM 가지는 일반적인 TFCG를 VerilogHDL 코딩하여 구현하였다. 표 6은 삼성 0.35㎛ 공정 라이브러리와 클럭 주기를 10ns로 하여 Synopsys사의 Design Compiler을 이용하여 합성한 결과이다. 표 7은

표 6. 삼성 0.35㎛ 공정 합성 결과
Table 6. Samsung 0.35㎛ process synthesis result.

		기존 방법	제안 방법	이득(%)
$R2^2$SDF	면적(ca)	2942.2	1203.2	59.1
	전력(mW)	3.0032	2.5480	15.1
	속도(ns)	4.87	8.71	-78,9
$R2^3$SDF	면적(ca)	3162.9	1334.7	57.8
	전력(mW)	4.5222	2.9555	34.6
	속도(ns)	6.86	8.73	-27.3
$R2^4$SDF	면적(ca)	3288.8	1382.2	57.9
	전력(mW)	6.3982	2.7154	57.5
	속도(ns)	7.60	8.74	-15

표 7. FPGA 합성 결과
Table 7. FPGA synthesis result.

		기존 방법	제안 방법	이득(%)
$R2^2$SDF	logic element (비율)	76 (< 1%)	128 (< 1%)	-68.5
	memory bit (비율)	17,664 (8%)	2,688 (1%)	84.7
	속도(ns)	9.239	27.188	-194.3
$R2^3$SDF	logic element (비율)	146 (< 1%)	140 (< 1%)	4.2
	memory bit (비율)	20,352 (11%)	2,688 (1%)	86.8
	속도(ns)	10.345	28.853	-179.0
$R2^4$SDF	logic element (비율)	260 (< 1%)	149 (< 1%)	87.7
	memory bit (비율)	21,824 (11%)	2,688 (1%)	90.9
	속도(ns)	12.102	31.516	-160.5

Altera사의 EXCALIBUR_ARM 계열의 EPXA4F672C3 디바이스를 선택하여 Quartus툴을 사용하여 합성한 결과이다. 제안한 TFCG는 기존의 TFCG보다 전력, 면적에서 이득이 있지만 지연시간이 긴 단점이 있다.

2. 회전인자 계수 생성기(TFCG) 최적화

그림 5는 WiBro용 1024point $R2^4$SDF FFT에서 스테이지4의 복소곱셈 연산 블록에 적용한 TFCG이다. 복소곱셈 연산 블록은 실수부, 허수부 입력을 각 16비트, 회전인자 계수를 각 14비트 연산으로 구현하였다. 회전인자 계수의 스위칭 회로를 실수부와 허수부 입력으로 이동시키고 bypass(bpS) 모드를 추가함으로 $\pi/even$을 표현할 수 있다. 그 결과 ROM을 $1/8N$으로 줄일 수 있고, $\pi/even$ 연산이 스위칭으로 구현함으로 SQNR은 약 2dB 이득을 얻었다. ROM에 저장된 cos은 MSB 2비트가 01, -sin은 MSB 1비트가 1로 변함이 없어 ROM의 워드의 3bit를 더 줄일 수 있다. TFIG의 회전인자 계수의 부호

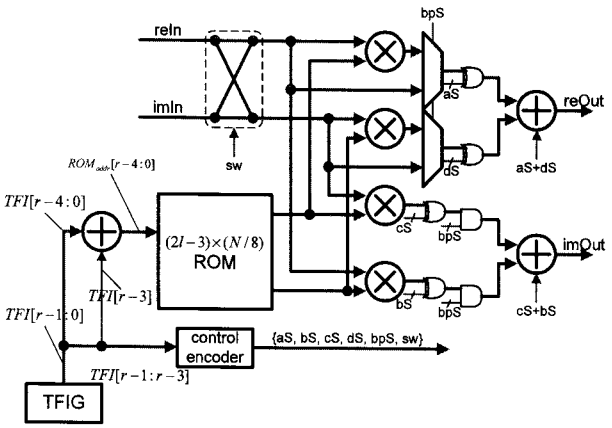


그림 5. 제안한 회전인자 계수 생성기 최적화
Fig 5. Proposed twiddle factor coefficient generator optimization.

를 결정하는 2개의 보수기는 복소곱셈기의 가산기와 감산기를 가감산기로 변경함으로 복소 연산에 포함하여 제거하여 지연시간을 줄일 수 있다.

VI. 결 론

본 논문에서는 R2SDF FFT에 카운터 신호를 이용하여 회전인자 인덱스 생성 방법을 제안하였다. 제안한 방법과 정현파 성질을 이용하여 FFT 프로세서의 ROM을 $1/8N+1$로 줄인 TFCG를 구현하여 검증하였다. 또한 WiBro용 1024포인트 $R2^4SDF$ FFT 프로세서 적용 응용하여 ROM을 $1/8N$로 줄이고 ROM의 워드의 3bit를 더 줄 일수 있음을 보였다. 제안한 TFCG는 기존의 TFCG보다 지연시간이 긴 단점이 있지만 회전 인자 계수의 부호를 결정하는 2의보수기 2개를 복소곱셈단의 덧셈연산으로 치완 시킴으로서 지연시간을 줄일 수 있음을 보였다. 제안한 방법은 회전인자 인덱스와 제어신호가 간단하게 생성됨을 보였다. 이 알고리즘은 큰 포인트 가지는 파이프라인 FFT 프로세서에서 유용하다.

참 고 문 헌

[1] E. H. Wold, A.M. Despain, "Pipeline and parallel-pipeline FFT processors for VLSI implementation". *IEEE Trans. Comput.*, C-33(5): 414-426, May 1984.

[2] Shousheng He, Torkelson, M, "Designing pipeline FFT processor for OFDM (de)modulation", Signals, Systems, and Electronics, 1998. ISSSE 98. 1998 URSI International Symposium on, pp. 257 - 262, 29 Sept.-2 Oct. 1998.

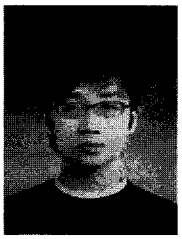
[3] J.Y. Oh, M.S. Lim, "Area and power efficient pipeline FFT algorithm", Signal Processing Systems Design and Implementation, 2005. IEEE Workshop on, pp. 520-525, 2-4 Nov. 2005.

[4] S. Y. Lee, C. C. Chen, C. C. Lee, C. J. Cheng, "A low-power VLSI architecture for a shared-memory FFT processor with a mixed-radix algorithm and a simple memory control scheme", Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on, pp. -160, Sept. 2006.

[5] 민종균, 조중휘, "DVB용. 2K/8K FFT의. Stratix EP1S25F672C6 FPGA 구현", 대한전자공학회 논문지 제44권 SD 제 8호 pp. 60-64, 2007.

[6] Huirae Cho, Myung-Soon Kim, Duk-Bai Kim and Jin-Up Kim, "R22 SDF FFT Implementation with Coefficient Memory Reduction Scheme", Vehicular Technology Conference 2006 pp. 1-4, Sept. 2006.

저 자 소 개



양 승 원(학생회원)

2008년 군산대학교 전자정보 공학부 학사 졸업

2008년~현재 전북대학교 전자 정보공학부 석사과정

<주관심분야 : 신호처리, VLSI 설계>

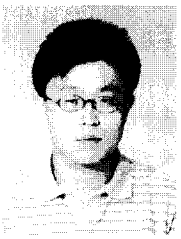
김 용 은(학생회원)

2005년 전북대학교 전자정보 공학부 학사 졸업

2007년 전북대학교 전자정보 공학부 석사 졸업

2007년~현재 전북대학교 전자정보공학부 박사과정

<주관심분야 : 통신, 신호처리, 반도체>



이 종 열(정회원)

1993년 한국과학기술원 전자전산 학과 학사 졸업 (B.S.).

1996년 한국과학기술원 전자전산 학과 석사 졸업 (M.S.).

2002년 한국과학기술원 전자전산 학과 박사 졸업 (Ph.D.).

2002년 9월~ 2003년 9월 하이닉스 반도체 선임 연구원

2003년 10월~ 2004년 2월 한국과학기술원 BK21 초빙 교수

2004년 3월~2006년 3월 전북대학교 전자정보 공학부 전임강사

2006년 4월~현재 전북대학교 전자정보공학부 조교수

<주관심분야 : SoC 설계, 내장형프로세서설계, 내장형 소프트웨어 최적화