Journal of The Institute of Electronics and Information Engineers Vol.52, NO.3, March 2015 http://dx.doi.org/10.5573/ieie.2015.52.3.075
ISSN 2287-5026(Print) / ISSN 2288-159X(Online)

논문 2015-52-3-10

연산복잡도 감소를 위한 새로운 8-병렬 MDC FFT 프로세서

(New Parallel MDC FFT Processor for Low Computation Complexity)

김 문 기*, 선 우 명 훈**

(Moon Gi Kim and Myung Hoon Sunwoo©)

요 약

본 논문에서는 고속 데이터 전송을 위한 OFDM 시스템에 적용 가능한 고속 FFT 프로세서를 제안하였다. 8개의 병렬 경로를 가지는 MDC 파이프라인 고속 FFT 프로세서를 제안한다. 제안하는 구조는 연산과 하드웨어의 최적화를 위해 radix-2^6 알고리즘에 기반하고 있다. 하드웨어 복잡도를 감소시키기 위해서 상수 곱셈기와 교환기 구조를 제안하고 새로운 스케즐링 기법을 적용하였다. 제안하는 FFT 프로세서는 새로운 구조를 적용해 지연 소자와 연산 사이클의 증가 없이 복소 곱셈기 및 연산 복잡도를 감소시킬 수 있다. 또한 최적화한 twiddle factor W_{64} 상수 곱셈기는 기존 복소 booth 곱셈기에 비해 65%만의 하드웨어 복잡도를 보였다. 설계한 FFT 프로세서는 Verilog HDL로 모델링하여 IBM 90nm 공정으로 합성하였으며 0.27mm^2의 면적과 388MHz의 주파수에서 2.7 GSample/s를 보이고 있다.

Abstract

This paper proposed the new eight-parallel MDC FFT processor using the eight-parallel MDC architecture and the efficient scheduling scheme. The proposed FFT processor supports the 256-point FFT based on the modified radix-2^6 FFT algorithm. The proposed scheduling scheme can reduce the number of complex multipliers from eight to six without increasing delay buffers and computation cycles. Moreover, the proposed FFT processor can be used in OFDM systems required high throughput and low hardware complexity. The proposed FFT processor has been designed and implemented with a 90nm CMOS technology. The experimental result shows that the area of the proposed FFT processor is 0.27mm^2. Furthermore, the proposed eight-parallel MDC FFT processor can achieve the throughput rate up to 2.7 GSample/s at 388 MHz.

Keywords : FFT, Radix-2^6 algorithm, MDC processor, IEEE 802.11.ac/ad, Eight-parallel

Ⅰ. 서 론

FFT (Fast Fourier Transform) 알고리즘은 DFT (Discrete Fourier Transform)의 연산량을 줄이기 위한 수학적 알고리즘으로 넓게 사용되고 있다. DFT 연산이 많은 양의 연산을 필요로 하지만 효율적인 연산인 FFT 알고리즘은 시간 또는 주파수 기준으로 입력 데이터의 순서를 일정 규칙으로 재구성한다. 이는 길이가 N인 DFT를 보다 작은 길이의 DFT 로 연속해서 분해하여 계산하는 방법이다. 이렇게 함으로써 DFT 연산의 복잡도는 $O(N^2)$에서 $O(NlogN)$으로 줄어든다.

* 학생회원, ** 평생회원, 아주대학교 전자공학과
(Department of Electrical and Computer Engineering, Ajou University)
© Corresponding Author(E-mail: sunwoo@ajou.ac.kr)
※ 본 논문은 미래창조과학부 전자정보디바이스산업원천기술개발사업[10044092, 7Gbps급 무선멀티미디어 통신서비스 제공을 위한 60Ghz대역 무선 LAN/PAN용 OFDM기반 PHY 및 RF 트랜시버 핵심 IP 기술 개발]과 미래창조과학부 중견연구자지원사업[2014R1A2A2A01002952, 미래형 실감미디어 지원을 위한 영상처리 전용 프로세서 연구]의 지원에 의하여 연구되었음.
Received ; December 24, 2014 Revised ; January 21, 2015
Accepted ; February 26, 2015

FFT 알고리즘은 통신 시스템, 바이오 어플리케이션, 센서 신호 처리 및 위성 신호처리 등 많은 분야에서 사용되고 있다. 또한, FFT 프로세서는 OFDM (Orthogonal Frequency Division Multiplexing) 전송 방식을 채택한 IEEE 802.11n/ac/ad[1], IEEE 802.15.3.c[2], IEEE 802.16e[3] 표준에서 가장 큰 복잡도를 가지며 고속 연산이 요구되어 가장 중요한 모듈 중 하나이다. OFDM을 이용한 다양한 시스템이 현재 구현되고 있으나 주파수나 사용 환경 전송속도 등에 맞추어서 최적의 시스템을 구현하는 것이 쉽지 않다. 이를 위해 각각의 시스템이 가지고 있는 채널 특성을 잘 파악하고 하드웨어 자원을 잘 활용하여 성능을 최적화해야 한다. 그러므로 OFDM 전송방식에서 요구하는 높은 처리율을 만족하며 실시간 신호처리를 위해서 다양한 FFT 프로세서[2]-[9] 들이 활발히 연구되고 있다.

FFT 구조는 크게 메모리 기반 구조와 파이프라인 구조로 나뉜다. 메모리 기반 구조는[2] 저면적을 만족하기 위해 제안되었다. 그러나 이 구조들은 많은 연산 사이클이 요구되어 높은 처리 속도를 얻는데 어려움이 있다. 반면에 파이프라인 구조는[3~9] 높은 처리율을 만족하므로 실시간 신호 처리에 적합하다. 파이프라인 구조는 데이터 교환 방식에 따라 SDF (Single-path Delay Feedback), MDF (Multi-path Delay Feedback), SDC (Single-path Delay Commutator), MDC (Multi-path Delay Commutator) 등으로 분류할 수 있다. SDF 구조는[4] 연산을 위한 같은 지연 소자를 사용하여 낮은 하드웨어 복잡도를 보이지만, 단일 경로로 데이터를 보내 처리율이 낮다. 이에 반해 MDC 구조는[5~7] 교환기를 사용하여 데이터를 보내게 되어 처리율은 높아지고 하드웨어 복잡도가 높아지게 된다. 각각의 구조에 따라 전체 구조의 하드웨어 복잡도와 데이터 처리율이 결정된다.

최근 실시간 어플리케이션에서 FFT 프로세서는 수 GSample/s 이상의 높은 처리율을 만족해야만 한다. 이에 따라 병렬 처리 기법을 사용한 파이프라인 구조 연구가 활발히 제안되고 있다. MDF 구조는[8~9] SDF 구조의 병렬 처리를 통해 설계되어 진다. 병렬 처리 기법을 적용하면 메모리 크기는 유지되며 단지 연산기만 증가하게 된다. 이는 높은 처리율을 만족할 수 있으면서, 하드웨어 크기의 증가는 크지 않다는 것을 의미한다. 최근에는 MDC 구조에 병렬 처리 기법을 적용하는 연구가 활발히 진행 되고 있다. 기존 MDC 구조의 장점인 높은 처리율에 병렬 처리 기법을 적용하여 더 높은 처리율을 만족하며, radix-2^n 알고리즘을 적용함으로서 하드웨어 복잡도를 낮추는 방식이 제안되고 있다[5].

이에 본 논문에서는 높은 처리율을 만족하는 8-병렬 MDC 파이프라인 구조를 제안하였다. 기존의 처리율이 높은 MDC 구조에 8-병렬 기법을 적용하여 더욱 높은 처리율을 만족하였다. 또한, 제안하는 구조는 연산과 하드웨어의 최적화를 위해 radix-2^n FFT 알고리즘에 기반하고 있다. 스케즐링 기법과 제안하는 교환기 (Commutator)를 적용해 지연 소자와 연산 사이클의 증가 없이 복소 곱셈기 및 연산 복잡도를 감소시키는 구조를 제안한다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 기존 FFT 구조인 파이프라인 구조의 최근 연구에 대해 설명한다. Ⅲ장에서는 제안하는 FFT 알고리즘, FFT 구조 및 제안하는 교환기와 상수 곱셈기에 대해 기술한다. Ⅳ장에서는 MATLAB simulator와 HDL (Hardware Description Language)을 사용하여 제안하는 FFT 프로세서의 구현 및 성능 평가를 기술한다. 마지막으로 Ⅴ장에서 결론을 맺는다.

Ⅱ. 기존 FFT 프로세서

서론에서 제시한 바와 같이 FFT 구조는 적은 하드웨어 면적을 가지지만 데이터 처리율이 다소 낮은 메모리 기반 구조[2]와 높은 처리율을 만족하면서 하드웨어 복잡도 또한 높은 파이프라인 구조[3~9]로 나뉜다. FFT 구조의 특성상 포인트 수가 늘어나면 블록 크기가 커지며 소요시간도 증가하게 된다. 하드웨어 크기는 무한정하게 증가할 수 없으며 동일한 면적에 구현할 수 있는 자원이 제한된다. 따라서 적용하고자 하는 표준이 요구하는 포인트, 데이터 처리율, 하드웨어 복잡도 등을 고려하여 적합한 구조를 채택하여야 한다.

높은 데이터 처리율을 위해서 제안된 메모리 기반 구조 FFT 프로세서[3]는 WPAN 표준인 IEEE 802. 15. 3c 시스템을 위해서 제안한 구조로 512-포인트 FFT 연산을 지원한다. WPAN 시스템에서는 symbol duration이 $222.22ns$이다. 따라서 FFT 연산은 하나의 OFDM symbol을 $222.22ns$안에 연산을 수행하여야 한다. 즉, FFT 프로세서는 512-포인트 연산을 수행할 때 2.592

GSample/s 이상의 데이터 처리율을 제공하여야 한다.

파이프라인 구조는 크게 MDC, SDC, SDF, MDF 구조로 분류할 수 있다. MDC 구조는[5~7] 다른 구조들보다 데이터 처리율이 높고 제어가 간단하여 파이프라인 구조 중 가장 보편적으로 사용되는 구조이다. 하지만 교환기(Commutator)를 통한 데이터 교환 방식을 사용하기 때문에 지연 버퍼가 추가로 사용되어 하드웨어 복잡도가 크다. 반면, SDF 구조는[4] 버터플라이 연산의 출력 중 일부를 피드백 시프터 레지스터에 저장함으로써 지연 버퍼의 재사용을 통해 레지스터를 효율적으로 사용할 수 있는 구조이다. 하지만 단일 경로이므로 낮은 동작 속도를 가지며 버터플라이 연산부를 제어하는 것이 비교적 복잡하다. SDC 구조는 여러 개의 입력들과 하나의 출력을 가지는 교환기를 사용하기 때문에 SDF 구조와 같은 곱셈기의 수를 필요로 하여 하드웨어 복잡도를 줄일 수 있다. 하지만 이 구조 또한 단일 경로를 기반으로 한 구조로 낮은 속도에서 동작을 하게 된다.

최근에는 SDF 구조에 병렬 처리 기법을 사용하여 더 높은 데이터 처리율을 만족하는 MDF 구조가 제안되었다[8~9]. 이와 유사하게 MDC 구조를 기반으로 병렬 처리 기법을 적용하여 데이터 처리율을 향상시킨 구조들 역시 제안되고 있다[5].

Ⅲ. 제안하는 FFT 프로세서

본 절에서는 제안하는 알고리즘에 기반 하여 새로운 256-포인트 8-병렬 파이프라인 MDC FFT 프로세서를 제안한다. 본 구조는 8-병렬 구조를 적용함으로서 높은 데이터 처리율을 만족하였다. 또한, radix-2^6 알고리즘을 사용하여 버터플라이 연산은 radix-2와 같은 구조를 유지하면서 twiddle factor의 곱셈 연산을 크게 줄일 수 있다. 따라서 하드웨어 복잡도를 크게 감소시키면서 기본 구조자체의 처리율이 높고 제어가 간단한 MDC 구조를 채택하였다. MDC 구조와 함께 8개의 병렬 경로를 사용하여 데이터 처리율을 더욱 향상시킬 수 있다.

제안하는 FFT 프로세서는 MDC 구조를 적용한 파이프라인 구조이며 대략적인 구조를 그림 1에 나타내었다. 전체 8개의 병렬 경로를 사용하여 연속적으로 시퀀스를 연산함으로서 높은 처리율을 만족한다. 제안하는 구조는 복소 덧셈기, 지연 소자, 교환기, 복소 곱셈기, 상수 곱셈기로 구성된다. 제안하는 구조는 연산 구조의 효율을 높이기 위해 Module-1에서는 radix-2^6 알고리즘을 Module-2에서는 radix-2^2 알고리즘을 사용한다. 제안하는 구조는 스테이지 6 연산 후 새로운 교환기(Commutator)를 구성하여 데이터 재구성을 통해 복소 곱셈기 수를 감소시킬 수 있다.

각 스테이지마다 버터플라이(BF) 연산은 덧셈과 뺄

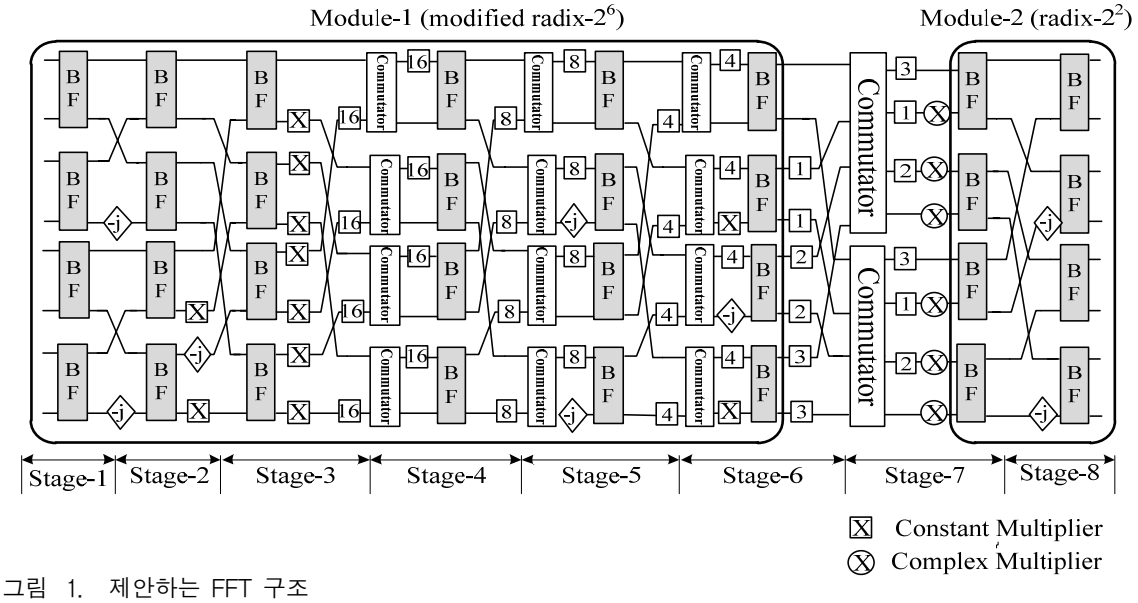


그림 1. 제안하는 FFT 구조
Fig. 1. Proposed FFT architecture.

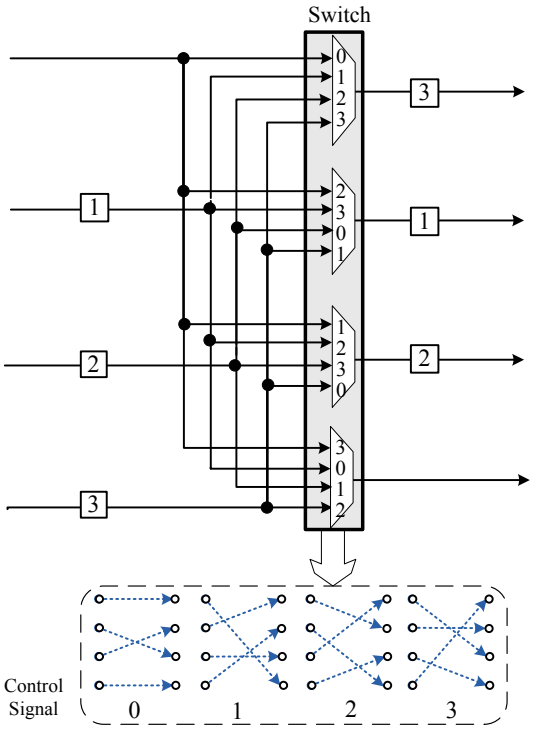


그림 2. 제안하는 교환기 구조
Fig. 2. Proposed switch architecture.

셈 연산을 수행한다. Twiddle factor W_8은 스테이지 2와 스테이지 5에서 연산되고 twiddle factor W_{64}은 스테이지 3에서 연산된다. Twiddle factor W_{64}는 복소 곱셈기가 아닌 덧셈연산과 시프터만을 사용하는 상수 곱셈기를 제안하여 하드웨어 복잡도를 감소시킨다. Twiddle factor -j 는 실수 부분과 허수 부분의 데이터를 바꾸고 허수 부분에 2의 보수형을 통해 연산이 가능하다. 스테이지 6에서 연산되는 복소 곱셈기는 제안하는 교환기를 적용하여 그 수를 줄였다.

그림 2는 제안하는 교환기(Commutator) 구조를 나타낸다. 스테이지 6과 스테이지 7 사이에서 사용하여 데이터를 재정렬함으로서 복소 곱셈기 수를 감소시킨다. 제안하는 교환기는 4-to-1 멀티플렉서, 컨트롤 신호, 입력 지연 버퍼와 출력 지연 버퍼로 구성된다. 그림 2와 같이 각각의 클럭마다 멀티플렉서는 컨트롤 신호에 따라서 4개의 입력 중 하나의 신호를 선택하여 출력 신호를 보내주게 된다. 선택하는 교환기 패턴은 그림 2의 Control Signal 방법을 따르며 256-포인트를 연산하기 위해서 각 클럭마다 컨트롤 신호는 계속 바뀌게 된다.

하드웨어 복잡도를 감소시키기 위해서 스테이지 3에서 필요한 twiddle factor W_{64} 상수 곱셈기를 제안하였

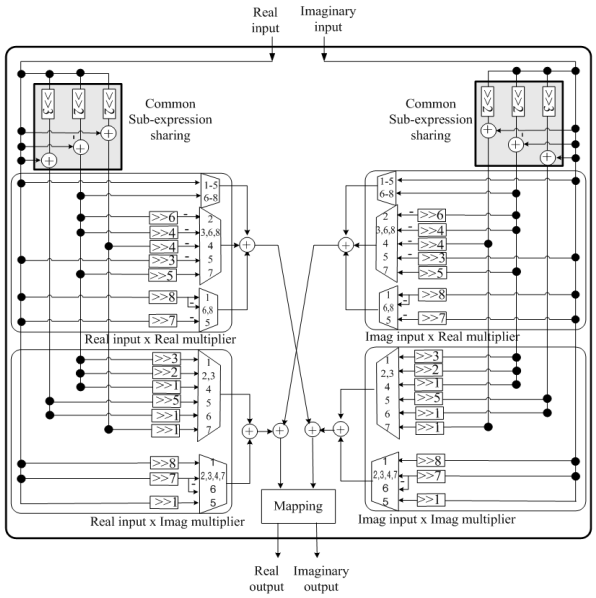


그림 3. 제안하는 상수 곱셈기 구조
Fig. 3. Architecture of the proposed constant multiplier.

다. 그림 3은 멀티플렉서, 덧셈기, 시프터 그리고 Mapping block으로 구성된 제안하는 상수 곱셈기 구조를 설명하고 있다. 스테이지 3에서 데이터 값들은 실수와 허수 값으로 분리되고, 적절한 twiddle factor W_{64}에 따라 데이터 곱셈 연산이 수행된다.

제안하는 상수 곱셈기 구조는 CSD(Canonical Signed Digit) 와 CSS (Common Sub-expression Sharing) 방법을 사용하여 효율적으로 복소 booth 곱셈기를 대체할 수 있다. CSD 방식은 곱셈 연산을 덧셈연산과 시프터를 이용하여 구현하는 방법으로서 공통패턴을 정의하여 서로 공유하는 방식이다. 그리고 CSS 방식은 중복되는 연산을 미리 처리함으로서 연산을 크게 줄일 수 있다. 그러므로 제안하는 상수 곱셈기를 사용하면 시프터와 덧셈 연산만을 사용하여 복소 곱셈 연산을 수행할 수 있다. 게다가 CSD와 CSS 방식의 최적화를 통해서 기존 복소 곱셈기[3]에 비해 하드웨어 복잡도가 크게 감소한다. 최종 실수와 허수의 데이터 값은 Mapping block을 통해서 얻을 수 있다.

CSS 방식을 사용하기 위해서는 먼저 사용되는 상수 계수들을 CSD형으로 나타내어야 한다. CSD형으로 twiddle factor W_{64} 연산을 표현하기 위해서 그림 4와 같이 1/8에 해당하는 8개의 twiddle factor 만을 고려하면 된다. 스테이지 3에서 그림 4의 A 영역에 분포한 twiddle factor가 이용된다. 이 A 영역에 분포한 8개의

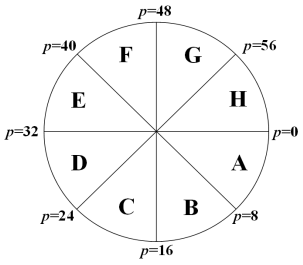
twiddle factor에 대한 실수부 8개, 허수부 8개의 총 16개의 계수들이 사용되는데 45° 부분의 값은 (0.7071, 0.7071)로 같으므로 총 15개의 계수를 사용하게 된다.

스테이지 3에서 사용되는 15개의 계수 중에서 실수에 해당하는 8개의 계수를 10비트 CSD형으로 나타내면 표 1과 같다. 표에서 N은 -1을 나타낸 것이고 이와 같d이 CSD형으로 나타내면 2의 보수형 보다 non-zero 비트의 수가 더 적게 쓰이는 것을 알 수 있다. 덧셈 연산의 수를 더욱 줄이기 위해 CSS 방식을 실수부에 적용하였다. 표 1에서 공통패턴을 먼저 계산하여 연산을 더 줄일 수 있다.

스테이지 3에서 사용되는 15개의 계수 중에서 허수를 10비트 CSD형으로 나타내면 표 2와 같다. 표 2에서 0.7071은 그림 4의 twiddle factor 영역 중 45° 부분의 값은 (0.7071, 0.7071)로 계수가 같으므로 실부의 표 1에서 구현하였기 때문에 허수부에서는 구현하지 않는다.

표 1. Twiddle factor W_{64} 의 실수부 CSD형 계수
Table 1. Real data of CSD for twiddle factor W_{64}.

	Decimal	CSD	CSS
W_{64}^{1}	0.9951	010000000N	$1-2^{-8}$
W_{64}^{2}	0.9808	0100000N0N	$1-(1+2^{-2})2^{-6}$
W_{64}^{3}	0.9569	01000N0101	$1-(1-2^{-2})2^{-4}+2^{-8}$
W_{64}^{4}	0.9239	01000N0N00	$1-(1+2^{-2})2^{-4}$
W_{64}^{5}	0.8819	0100N00010	$1-2^{-3}+2^{-7}$
W_{64}^{6}	0.8315	010N010101	$(1-2^{-2})-(1+2^{-2})2^{-4}+2^{-8}$
W_{64}^{7}	0.773	010N0010N0	$(1-2^{-2})+(1-2^{-2})2^{-5}$
W_{64}^{8}	0.7071	010N0N0101	$(1-2^{-2})-(1-2^{-2})2^{-4}+2^{-8}$



Region	Real	Imag.
A	x_p	y_p
B	$-y_p$	$-x_p$
C	y_p	$-x_p$
D	$-x_p$	y_p
E	$-x_p$	$-y_p$
F	y_p	x_p
G	$-y_p$	x_p
H	x_p	$-y_p$

그림 4. Twiddle factor W_{64} 에 사용되는 8개 영역
Fig. 4. Eight-Region mapping of twiddle factor W_{64}.

표 2. Twiddle factor W_{64} 의 허수부 CSD형 계수
Table 2. Imaginary data of CSD for twiddle factor W_{64}.

	Decimal	CSD	CSS
W_{64}^{1}	0.0980	000010N001	$(1-2^{-2})2^{-3}+2^{-8}$
W_{64}^{2}	0.1951	00010N0010	$(1-2^{-2})2^{-2}+2^{-7}$
W_{64}^{3}	0.2903	0001001010	$(1+2^{-3})2^{-2}+2^{-7}$
W_{64}^{4}	0.3827	0010N00010	$(1-2^{-2})2^{-1}+2^{-7}$
W_{64}^{5}	0.4714	001000N001	$2^{-1}-(1-2^{-3})2^{-5}$
W_{64}^{6}	0.5556	00100100N0	$(1+2^{-3})2^{-1}-2^{-7}$
W_{64}^{7}	0.6344	0010100010	$(1+2^{-2})2^{-1}+2^{-7}$
W_{64}^{8}	0.7071	010N0N0101	$(1-2^{-2})-(1-2^{-2})2^{-4}+2^{-8}$

하드웨어 복잡도 감소를 확인하기 위해서 제안한 구조를 Synopsys Design Compiler의 합성을 통해 비교를 하였다. 기존 복소 booth 곱셈기는 60,095μm^2의 하드웨어 크기를 갖는 반면, 제안하는 상수 곱셈기는 38,151 μm^2의 하드웨어 복잡도를 가지게 되어 약 35%의 하드웨어 감소를 보인다. 결과적으로, 제안하는 상수 곱셈기는 최적화된 CSD와 CSS 방법을 사용하여 시프터와 덧셈 연산만을 사용하게 되고 이는 하드웨어 복잡도 감소를 가진다.

Ⅳ. 실 험

제안하는 FFT 프로세서는 Verilog HDL 언어를 이용하여 하드웨어로 구현하였다. 8-병렬 경로와 MDC 구조를 사용하여 고속 동작이 가능한 구조를 설계하였으며 256-포인트 FFT 연산이 가능한 구조를 설계하였다. 비트길이 길이는 실수부와 허수부를 각각 12비트로 하였다. 같은 입력에 대한 제안한 프로세서의 시뮬레이션 결과와 MATLAB 결과를 비교하면서 동작 검증을 하였고 SQNR(Signal to Quantization Noise Ratio)은 38dB의 성능을 보였다.

본 검증에서는 256-포인트를 사용하는 논문과의 알고리즘, 처리율, 면적을 비교한다. 표 3은 제안하는 구조의 효과를 검증하기 위해 제안하는 구조와 기존의 FFT 구조 [3], [7], [8]과의 성능 비교표를 나타내었다.

표 3. 기존의 FFT 구조와의 성능 비교
Table 3. Performance comparison with other existing architectures.

	제안하는 구조	[3]	[7]	[9]
설계 공정	90nm	180nm	65nm	90nm
구조	MDC	MDF	MDC	MDF
포인트 수	256	256	512	512
병렬 수	8	2	8	8
알고리즘	Radix-2^6	Mixed-radix	Radix-8	Radix-2^5
비트길이	12 bits	15 bits	12 bits	12 bits
SQNR	38 dB	44 dB	-	35 dB
Clock rate	338 MHz	64 MHz	330 MHz	310 MHz
Throughput	2.7 GS/s	0.13 GS/s	2.64 GS/s	2.5 GS/s
면적	0.27 mm^2	0.747 mm^2	0.88 mm^2	0.78 mm^2
일반화한 면적	4.82 mm^2	11.67 mm^2	23.43 mm^2	10.83 mm^2

제안한 구조는 338 MHz의 주파수에서 2.7 GSample/s의 처리율을 만족한다. 이 처리율을 다른 논문 [3], [7], [8]과 비교하여 볼 때 가장 높다. 또한, 일반화한 면적[13]만을 비교하면 다른 논문들 [3], [7], [8]에 비해 큰 하드웨어 면적 감소를 보임을 알 수 있다. 특히, [7]번 논문에 비해 약 21%만의 하드웨어 크기를 가짐을 알 수 있다.

자세하게 각각의 논문과 비교하면 다음과 같다. [3] 논문은 높은 비트길이를 가져 SQNR은 높지만, 제안하는 논문은 처리율이 높으면서 일반화한 면적이 약 41%만을 가진다. 512-포인트를 목표로 하는 [7]번 논문과 비교한다며 같은 8-병렬 MDC 구조를 제안하고 있기 때문에 비슷한 처리율을 같지만, 일반화한 구조에서 약 21%만의 하드웨어 면적을 보이게 된다. [9]번 논문 역시 일반화한 하드웨어 면적이 약 45%이지만 처리율은 제안하는 논문이 더 높다.하드웨어 면적 감소를 보임을 알 수 있다. 특히, [7]번 논문에 비해 약 21%만의 하드웨어 크기를 가짐을 알 수 있다.

V. 결 론

본 논문에서는 초고속 데이터 전송을 위한 OFDM 시스템에 적용 가능한 고속 FFT 프로세서를 제안하였다. 제안하는 8-병렬 MDC FFT 프로세서는 256-포인트 FFT 연산에 적용할 수 있다. 하나의 radix-2^6 알고리즘과 radix-2^2 알고리즘을 사용하여 스테이지 8개로 구성된다. 하드웨어 크기를 감소시키기 위해서 상수 곱셈기와 교환기의 새로운 구조를 제안하고 새로운 스케줄링 기법을 적용하였다. 제안하는 새로운 구조를 적용해 지연 소자와 연산 사이클의 증가 없이 복소 곱셈기 및 연산 복잡도를 감소시키는 구조를 제안한다.

설계한 FFT 프로세서는 Verilog HDL로 모델링하여 Synopsys Design Compiler를 통해 IBM 90nm 공정으로 합성하였으며 동작속도가 338MHz로 256-포인트 연산을 29.88ns에 처리 가능한 구조이다. 제안한 고속 FFT 프로세서는 802.11.a, UWB, WiMAX 등과 같은 OFDM 변복조 방식의 통신 시스템 개발에 활용될 수 있다.

REFERENCES

[1] IEEE P802.11-T, ask Group AD, http://www.ieee802.org/11/
[2] S. J. Huang and S. G. Chen, “A high-throughput radix-16 FFT processor with parallel and normal input/output ordering for IEEE 802.15.3c systems,” IEEE Trans. Circuits Syst. I, vol. 59, no. 8, pp. 1752 - 1765, Aug. 2012.
[3] Fang-Li Yuan, Yi-Hsien Lin, Chih-Feng Wu, Muh-Tian Shiue and Chorng-Kuang Wang, “A 256-Point dataflow scheduling 2×2 MIMO FFT FFT/IFFT processor for ieee 802.16 WMAN,” in Proc. IEEE ASSCC, pp.309-312, Nov. 2008.
[4] C. T. Lin, Y. C. Yu and L. D. Van, “Cost-effective triple-mode reconfigurable pipeline FFT/IFFT/2-D DCT processor,” IEEE Trans. VLSI systems, vol. 16, no.8, pp. 1058-1071, Aug. 2008.
[5] M. Garrido, J. Grajal, M. S´anchez, and O. Gustafsson, “Pipelined radix-2k feedforward FFT architectures,” IEEE Trans. VLSI Syst., vol. 21, no. 1, pp. 23-32, Jan. 2013.
[6] M. Ayinala and K.K. Parhi, “Parallel Pipelined FFT Architectures with Reduced Number of Delays,” in Proc. ACM Great Lakes Symp. on VLSI, pp. 63-66, May 2012.
[7] T. Ahmed, M. Garrido, and O. Gustafsson, “A 512-point 8-parallel pipelined feedforward FFT

for WPAN," in Proc. ASILOMAR, pp. 981 - 984, Nov. 2011.
[8] Y. Chen, Y.-W. Lin, Y.-C. Taso, and C.-Y. Lee, "A 2.4-Gsample/s DVFS FFT processor for MIMO OFDM communication systems," IEEE J. Solid-State Circuits, vol. 43, no. 5, pp. 1260 - 1273, May 2008.
[9] T. Cho and H. Lee, "A High-Speed Low-Complexity Modified Radix-2^5 FFT Processor for High Rate WPAN Applications," IEEE Trans. VLSI systems, vol. 21, pp. 187-191, Jan. 2012.
[10] K. Yang, S. Tsai, and G. C. Chuang, "MDC FFT/IFFT processor with variable length for MIMO-OFDM systems," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 21, no. 4, pp. 720-731, Mar. 2013.

저 자 소 개

김 문 기(학생회원)
2013년 아주대학교 전자공학과 학사 졸업.
2015년 아주대학교 전자공학과 석사 졸업.
2015년～현재 (주)한화/방산 연구원
<주관심분야 : DSP 칩 설계, 통신알고리즘>

선우 명 훈(평생회원)
1980년 서강대학교 전자공학 학사 졸업.
1982년 한국과학기술원 전기전자 공학 석사 졸업.
1982년～1985년 한국전자통신 연구소(ETRI)연구원
1985년～1990년 Univ. of Texas at Austin 전자공학 박사 졸업
1990년～1992년 Motorola, DSP Chip Operation (미국)
1992년～현재 아주대학교 전자공학과 교수
2011년～현재 IEEE CASS Board of Governor
2011년～현재 IEEE Fellow Member
2012년～2014년 대한전자공학회 반도체 소사이어티 회장
2014년～현재 대한전자공학회 반도체 소사이어티 자문위원
<주관심분야 : SoC 설계, VLSI Architecture, 통신 및 멀티미디어 ASIP 설계, 저전력 설계, 의료영상 플랫폼>