IEEE 802.11a OFDM System을 위한 파이프라인 구조 IFFT/FFT 모듈의 설계와 비교

이창훈* · 김주현** · 강봉순***

Design and Comparison of the Pipelined IFFT/FFT modules for IEEE 802.11a OFDM System

Chang-Hoon Lee* · Ju-Hyun Kim** · Bongsoon Kang***

요 약

본 논문에서는 고속 무선 LAN에서 사용하는 IEEE 802.11a OFDM(Orthogonal Frequency Division Multiplexing)에서 주요 구성인 IFFT/FFT(Inverse Fast Fourier Transform/Fast Fourier Transform)에 대한 설계에 대해 비교하였다. 설계된 IFFT/FFT는 무선 LAN의 표준에 맞게 64 point의 FFT로 연산을 수행하며, S/P(Serial-to-Parallel)이나 P/S(Parallel-to-Serial)변환기가 필요 없는 Pipelined FFT의 구조로 설계하였다. 그 중 Radix-2 알고리즘을 이용한 R22SDF(Radix-2 Single-path Delay Feedback) 방식, R2SDF(Radix-2 Single-path Delay Feedback) 방식과 Radix-4 알고리즘을 이용한 R4SDF(Radix-4 Single-path Delay Feedback) 방식, R4SDC(Radix-4 Single-path Delay Commutator) 방식을 사용하여 비교하였다. 하드웨어 구현 시 발생하는 오차를 줄이기 위해 Butterfly 연산 후 일부 소수점을 가지고 계산하는 구조로 설계하였다. R22SDF 방식을 이용할 경우 메모리를 제외한 전체 게이트 수가 44,747개로 다른 구조에 비해 적은 하드웨어와 낮은 오차율을 가진다.

ABSTRACT

In this paper, we design the IFFT/FFT (Inverse Fast Fourier Transform/Fast Fourier Transform) modules for IEEE 802.11a-1999, which is a standard of the High-speed Wireless LAN using the OFDM (Orthogonal Frequency Division Multiplexing). The designed IFFT/FFT is the 64-point FFT to be compatible with IEEE 802.11a and the pipelined architecture which needs neither serial-to-parallel nor parallel-to-serial converter. We compare four types of IFFT/FFT modules for the hardware complexity and operation : R22SDF (Radix-2 Single-path Delay Feedback), the R2SDF (Radix-2 Single-path Delay Feedback), R4SDF (Radix-4 Single-path Delay Feedback), and R4SDC (Radix-4 Single-path Delay Commutator). In order to minimize the error, we design the IFFT/FFT module to operate with additional decimal parts after butterfly operation. In case of the R22SDF, the IFFT/FFT module has 44,747 gate counts excluding RAMs and the minimized error rate as compared with other types. And we know that the R22SDF has a small hardware structure as compared with other types.

키워드

IEEE 802.11a, OFDM, IFFT/FFT, Pipelined FFT 구조

*동아대학교 전자공학과 석사과정
**동아대학교 전자공학과 박사과정
***동아대학교 전기전자컴퓨터공학부 조교수
접수일자 : 2004. 3. 11

Ⅰ. 서 론

최근 고속 무선 LAN에 대한 관심이 점차 증대되고 있는 가운데, 그 중 OFDM 방식을 이용한 무선 LAN 시스템이 큰 관심을 받고 있다. OFDM 방식은 BPSK, QPSK, 또는 QAM 등으로 변조된 신호들을 병렬화한 후 여러 개의 직교 부반송파를 이용해서 전송하는 방식으로, 데이터의 병렬화를 통해 수십 Mbps로 고속 전송이 가능하고 단일 반송파를 사용하는 시스템과 비교할 때 다중 경로 페이딩 채널 환경에 강하다는 점과 부반송파 변복조를 IFFT/FFT 알고리즘을 이용하여 효율적으로 구현할 수 있다는 장점이 있다[1-2].

이에 5GHz 대역에서 54Mbps까지의 데이터 전송이 가능한 OFDM 방식의 고속 무선 LAN 표준으로 IEEE 802.11a와 HiperLAN/2가 확정되어있다. IFFT/FFT는 이러한 표준의 시스템에서 변복조를 담당하고, OFDM 시스템의 성능을 좌우하는 핵심요소이다[3]. 그러므로 IFFT/FFT에서 발생한 오차는 전체 무선 LAN 시스템에 아주 큰 영향을 미치게 된다. 따라서 하드웨어로 설계할 때 IFFT/FFT에서 발생하는 오차를 최소화할 필요가 있다.

본 논문에서는 이러한 IFFT/FFT 설계를 위하여, 발생할 수 있는 Scaling, Coefficient, Round-off에 대한 오차와 무선 LAN 시스템의 특성을 고려한 하드웨어의 구성에 초점을 맞추어 R22SDF (Radix-2 Single-path Delay Feedback)방식, R2SDF (Radix-2 Single-path Delay Feedback)방식, R4SDF (Radix-4 Single-path Delay Feedback)방식, R4SDC(Radix-4 Single-path Delay Commutator) 방식의 Pipelined FFT의 구조를 사용하였다[4-6].

이를 위한 본 논문의 구성은 다음과 같다. 2장에 제안된 이론에 대한 알고리즘과, 3장에 알고리즘에 대한 하드웨어 구현, 그리고 4장에 시뮬레이션 및 합성 후의 결과를 비교하여 나타내었고, 마지막 5장에서는 결론을 맺는다.

Ⅱ. 알고리즘

FFT 알고리즘은 크게 두 가지 회전인자의 곱셈이 있는 알고리즘과 회전인자의 곱셈이 없는 알고리즘으로 구분할 수 있다. 본 논문에서는 회전인자가 있는 알고리즘에 대해서 다룰 것이다. 이는 N개 길이의 시퀀스를 보다 작은 길이의 시퀀스로 연속해서 분해함으로서 DFT 연산량을 줄이는 방식이다.

먼저 본 논문의 설계에서 사용된 DFT와 IDFT의 수식을 나타내면 아래의 수식과 같다.

$$\mathrm{DFT}: X(k) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x(n) W_N^{kn} \quad (1)$$

$$\mathrm{IDFT}: X(k) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} X(k) W_N^{-kn} \quad (2)$$

수식 (1)과 (2)에서 $W = e^{-j2\pi/N}$이고, DFT와 IDFT는 연산 후 입출력 신호의 전력 균일화를 위해 각각 $1/\sqrt{N}$의 계수를 가진다.

Pipelined FFT 구조는 내부 구조가 규칙적이고 비교적 제어가 간단하며 일렬(serial) 입력과 일렬(serial) 출력을 할 수 있기 때문에 높은 성능을 요구하는 응용 분야에 가장 많이 사용하는 구조이다. 파이프라인 구조에서 하나의 BF PE (Butterfly Processing Element)는 FFT의 신호 흐름도에서 각각의 Stage에 해당되는 것으로 열(column)을 하나의 BF PE로 공유하는 것이다. 이것은 Radix-r 알고리즘에서 단수는 $\log_r N$으로 병렬성이 있음을 의미한다[7].

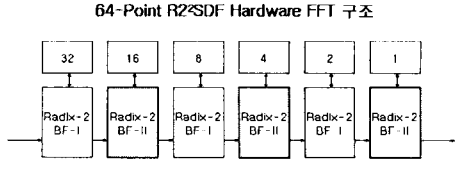


그림 1. 64-Point R22SDF FFT 구조
FIG. 1 the architecture of 64-Point R22SDF FFT

첫 번째 R22SDF는 그림 1과 같이 각 단은 서로 하나의 패스로 연결되어 있으며 새로운 데이터나 중간 연산 결과를 저장하기 위한 궤환 레지스터(feedback register)가 사용된다. 이 구조는 Radix-2 알고리즘의 버터플라이 구조를 가지며 복소 곱셈기의 수는 Radix-4 SDF와 같은 구조로서 두 가지의 버터플라이 구조(Radix-2 BF-I, Radix-2 BF-II)를 가지고 있다. R2SDF와 같은 단의 구조를 사용하고 있으나 복소 곱셈기의 수가 줄어 각 단과 궤환 레지스터의 하드웨어 사이즈가 다른 구조들에 비해 작다.

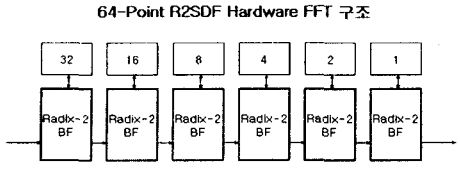


그림 2. 64-Point R2SDF FFT 구조
FIG. 2 the architecture of 64-Point R2SDF FFT

두 번째 R2SDF는 그림 2과 같이 그림 1의 R22SDF와 같은 데이터 흐름을 가진다. 여기에서 사용되는 버터플라이 구조는 Radix-2 버터플라이(Radix-2 BF)와 두개의 멀티플렉서로 구성된다. 입력의 절반은 그대로 궤환 레지스터에 저장하고, 나머지 절반의 입력이 들어오면 궤환 레지스터에 저장된 결과와 같이 버터플라이 연산을 수행하여 다음 단에 전달하는 것과 다시 저장하는 것으로 나뉘어 진다. 이 구조는 각 단마다 복소 곱셈기가 사용되므로 각 단과 궤환 레지스터의 하드웨어 크기가 다른 구조들에 비해 크다.

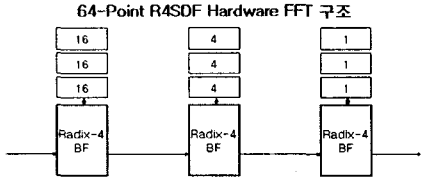


그림 3. 64-Point R4SDF FFT 구조
FIG. 3 the architecture of 64-Point R4SDF FFT

세 번째 R4SDF는 그림 3와 같이 R2SDF의 궤환 레지스터를 사용하여 4개의 입력을 받아 4개의 출력을 발생하는 Radix-4 알고리즘 구조이다. 이 구조에서는 기본적인 동작원리는 R2SDF 와 동일하며 R2SDF의 Radix-4 구조로 생각하면 된다. 이 구조는 각 단에서 한번의 연산으로 4개의 출력을 함으로 복소 곱셈기를 제외한 덧셈기로 이루어진 버터플라이의 종류(4가지)가 모두 필요로 함으로 각 단 버터플라이의 하드웨어 사이즈가 크기다.

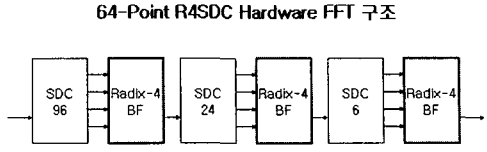


그림 4. 64-Point R4SDC FFT 구조
FIG. 4 the architecture of 64-Point R4SDC FFT

마지막 R4SDC 구조는 그림 4과 같이 4개의 입력을 받아 4개의 출력을 발생하는 Radix-4 알고리즘을 변형한 구조로서 한번에 하나의 FFT 출력을 내보내는 구조이다. 따라서 버터플라이의 덧셈기의 수를 줄일 수 있으며, 버터플라이의 구조도 일반적인 Radix-4 구조보다 단순화한 SBF(Simplified BF)로 간단하게 구현할 수 있다. 이 구조는 R4SDF에 비해 버터플라이를 공유함으로 각 단의 하드웨어 사이즈는 줄일 수 있으나 SDC구조를 사용하므로 지연과 저장을 위한 레지스터의 사이즈가 크다.

III. 하드웨어 구현

본 논문에서 IFFT/FFT는 하드웨어 설계언어인 Verilog HDL을 사용하여 구현하였다. 입력과 출력 각각 실수와 허수를 16비트로 사용하였고, 시작을 알리는 신호와 연산의 수행이 끝났음을 알리는 신호를 두었으며, IFFT/FFT 선택과 scale을 선택할 수 있도록 하였다. 설계된 IFFT/FFT는 64-point로서 Pipelined FFT를 사용하여 OFDM 시스템의 IFFT/FFT에 입출력 단의 S/P Converter나 P/S Converter를 사용하지 않고 직렬 데이터를 입력을 받아 처리하고, 직렬로 데이터를 출력할 수 있도록 하였다.

R22SDF는 Radix-2 DIF FFT 알고리즘의 SFG (Single Flow Graph)에서 Twiddle Factor의 위치를 변형하여 Pipeline화 한 것이다. R2SDF는 일반적인 radix-2 구조의 일반적인 버터플라이를 사용하고 있다. R4SDF의 구조는 일반적인 Radix-4 구조의 일반적인 버터플라이를 사용하고 있다. R4SDC는 SDC라는 쉬프트 레지스터로 구성되어 있는 지연소자를 사용하였고, 버터플라이는 일반적인 Radix-4 구조보다 간단한 SBF (Simplified BF)를 사용하고 있다.

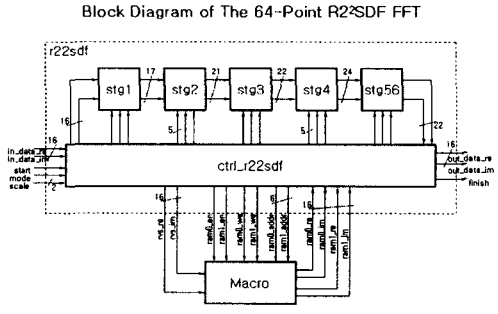


그림 5. R22SDF 구조의 기능별 블록도
Fig. 5 R22SDF Block Diagram

그림 5는 R22SDF 구조 IFFT/FFT의 블록다이어그램이다. 그림 5에서 Macro에 해당하는 부분은 각 Stage의 값을 임시 저장하기 위한ram32, ram16, ram08, ram04와 FFT 연산을 수행 후 Bit-reverse된 순서를 재배열하기 위한 ram_br0, ram_br1인 메모리블록으로 분리하여 설계한 것이다. 그리고 64-point radix-2이므로 6개의 단을 가지고 있으며, ctrl_r22sdf 블록으로 나누어진다. ctrl_r22sdf 블록은 시작신호를 검출하고 IFFT/FFT를 선택하며, 5개의 블록을 제어한다. 그리고 설정된 scale에 따라 Scaling 오차를 제어할 수 있도록 하는 전반적인 기능을 담당하고 있다. 또한 IFFT/FFT 연산 후의 데이터의 순서를 재배열하는 기능을 담당하고 있다. 즉, 데이터를 RAM에 저장시키고, 불러내는 순서를 제어하고, 선택한다[8].

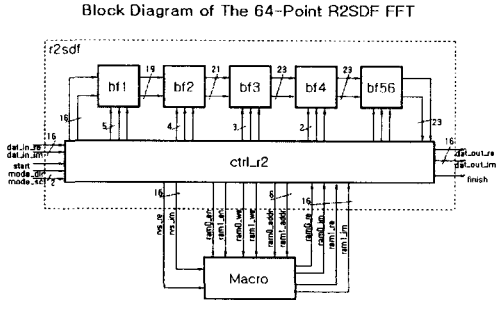


그림 6. R2SDF 구조의 기능별 블록도
Fig. 6 R2SDF Block Diagram

그림 6는 R2SDF 구조 IFFT/FFT의 블록다이어그램이다. 그림 6에서 Macro에 해당하는 부분은 그림 5과 같이 각 Stage의 값을 임시 저장과 FFT 연산을 수행 후 Bit-reverse된 순서를 재배열하기 위한 메모리블록으로 분리하여 설계한 것이다. 그리고 64-point radix-2이므로 6개의 단을 가지고 있으며, ctrl_r2 블록으로 나누어진다[9].

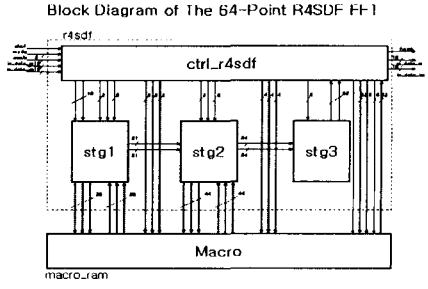


그림 7. R4SDF 구조의 기능별 블록도
Fig. 7 R4SDF Block Diagram

그림 7은 R4SDF 구조 IFFT/FFT의 블록다이어그램이다. 그림 7에서 Macro에 해당하는 부분은 각 Stage의 값을 임시 저장하기 위한 ram11, ram12, ram13, ram21, ram22, ram23과 FFT 연산을 수행 후 Bit-reverse된 순서를 재배열하기 위한 ram_br0, ram_br1인 메모리블록으로 분리하여 설계한 것이다. 그리고 64-point radix-4이므로 3개의 단을 가지고 있으며, ctrl_r4sdf 블록으로 나누어진다.

ctrl_r4sdf 블록은 시작신호를 검출하고 IFFT/FFT를 선택하며, 3개의 블록을 제어한다. 그리고 설정된 scale에 따라 Scaling 오차를 제어할 수 있도록 하는 전반적인 기능을 담당하고 있다. 또한 IFFT/FFT 연산 후의 데이터의 순서를 재배열하는 기능을 담당하고 있다. 즉, 데이터를 RAM에 저장시키고, 불러내는 순서를 제어하고, 선택한다.

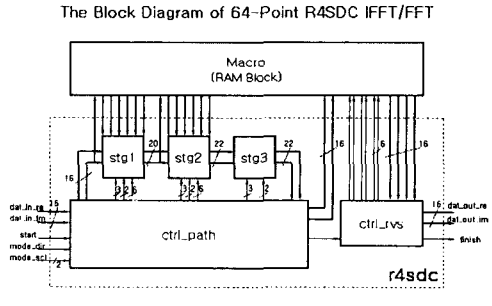


그림 8. R4SDC 구조의 기능별 블록도
Fig. 8 R4SDC Block Diagram

그림 8는 R4SDC 구조 IFFT/FFT의 블록다이어그램이다. 그림 8에서 Macro 블록은 지연소자인 SDC와 순서 재배열을 위한 RAM을 메모리블록으로 분리하여 설계한 것이다. 여기에 사용되는 SDC는 쉬프트레지스터로 32비트×6×16개와 40비트×6×6개이고, 메모리는 FFT 연산을 수행 후 Bit-reverse된 순서를 재배열하기 위한 ram_br0, ram_br1이 쓰이고 있다. 그리고 64-point radix-4이므로 3개의 단을 가지고 있으며, ctrl_path 블록과 ctrl_rvs 블록으로 나누어진다. 이 제어블록은 R2SDF 구조에서의 제어블록을 두 부분으로 분리한 것이다. 하나의 단은 SDC, SBF, 복소곱셈기로 나누어진다. 첫 번째와 두 번째 단의 SDC는 용량이 큰 쉬프트레지스터로 구성되어 별도의 메모리 블록인 Macro 블록에 구성되고, 세 번째 단의 SDC는 용량이 적은 쉬프트레지스터로 구성되어 있어 stg3 블록에 포함되었다[10].

Radix-2 알고리즘을 사용하는 R22SDF와 R2SDF는 64-point의 경우에는 6개의 Stage로 구성되어 있고, Radix-4 알고리즘을 사용하는 R4SDF와 R4SDC는 64-point의 경우에는 3개의 Stage로 구성되어 있다.

IV. 각 구조에 대한 성능 비교 및 합성 결과

SDF 구조로 되어 있는 R22SDF, R2SDF, R4SDF는 동일하게 모든 동작을 수행하는데 142 clock이 소요되며, 이 중 순서 재배열을 제외하면 IFFT/FFT를 수행하기 위해 75 clock이 소요된다. 그리고 SDC 구조로 되어 있는 R4SDC는 모든 동작을 수행하는데 137 clock이 소요되며, 이 중 순서 재배열을 제외하면 IFFT/FFT를 수행하기 위해 71 clock이 소요된다[11].

본 논문에서 설계된 IFFT/FFT는 OFDM에 사용하기 위한 것이므로 OFDM에서의 모듈 시뮬레이션 모델을 그림 9에 나타내었다.

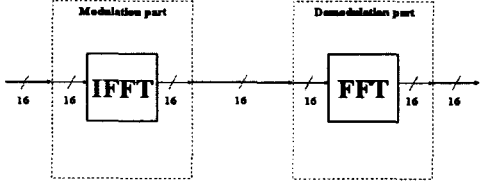


그림 9. IFFT/FFT 모듈의 시뮬레이션 모델
Fig. 9 IFFT/FFT Simulation Model

그림 9에서 IFFT는 송신부에서의 구성을, FFT는 수신부에서의 구성을 맞추어 시뮬레이션 하였다. 입력되는 데이터는 Signed 16-bit Chirp pattern을 사용하였다.

아래 표 1는 그림 9의 모델에서 Chirp pattern 데이터를 사용하여 4가지 모듈의 시뮬레이션 결과이다. 입력된 데이터와 IFFT/FFT 연산 후의 데이터와의 오차를 STD(STandard Deviation) 값을 사용하여 각 모듈의 오차율을 나타내었다. 사용된 STD는 수식 (3)와 같이 계산되어 진다.

$$STD = \left(\frac{1}{N} \sum_{i=1}^{n} (x_i - x_0)^2 \right)^{1/2}$$
$$here) x_0 = \frac{1}{N} \sum_{i=1}^{n} x_i \qquad (3)$$

표 1. 각 Module의 오차에 대한 STD 값
Table. 1 STD on errors of Modules

STD / Modules	Input Data	
	Real	Imaginary
R2^{2}SDF	2.3429	2.3422
R2SDF	2.4421	2.4129
R4SDF	2.3618	2.3582
R4SDC	2.3802	2.3799

설계된 4가지 IFFT/FFT Module은 Synopsys사의 Design Analyzer로 Synthesis을 수행하였다. Synthesis 조건은 다음과 같다. 라이브러리는 Samsung 0.35㎛의 STD90, 동작 조건은 V300WTP 0850(3.0V, Worst case, 85℃), 동작 속도는 20MHz (50ns)로 설정 후 합성하였다. 결과는 표 2와 같다.

표 2. IFFT/FFT Module 4가지 각 Synthesis 결과
Table. 2 Synthesis result of IFFT/FFT modules

Modules	Gate Count	Max Timing[ns]	Memory Size(bits)
R2^{2}SDF	44,747	33.75	6,288
R2SDF	71,796	33.87	6,344
R4SDF	50,377	34.34	6,352
R4SDC	44,298	31.83	7,264

표 1와 2의 결과를 분석하면 4가지 모듈은 같은

FFT 기능을 하지만 R22SDF 구조은 다른 구조들과 비교해서 게이트 카운터와 사용된 메모리 크기가 작게 사용되고 있으며, 오차율도 보다 좋은 결과를 가지고 있다.

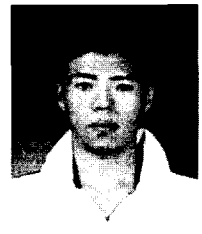
V. 결 론

본 논문에서는 순차적인 입력 데이터 처리방식을 이용하여, IEEE 802.11a OFDM에 사용되는 64-point IFFT/FFT Module을 디자인하여 각 구조들을 비교하였다. 각 구조들에 쓰인 알고리즘들은 같으나 하드웨어적인 측면에서는 각각 다른 특성을 보였다. 각 구조의 주요 특징은 순차적으로 들어오는 데이터를 지연하기 위해서 SDF 구조에서는 RAM을 사용하였고, SDC 구조에서는 Shift-register를 사용하였다. 각 구조들은 오차를 줄이기 위해서 부동 소수점의 일부를 연산에 사용하고, 사용자가 Bit-Scaling을 할 수 있도록 하였다. 본 논문에서 각기 다른 구조를 가지는 FFT 모듈을 이용하여 비교한 결과 R22SDF 방식을 이용할 경우 6,288bits의 메모리 크기와 44,747개의게이트 수로 보다 적은 하드웨어와 낮은 오차율을 가진다.

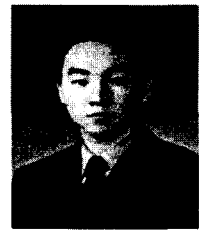
감사의 글
본 논문에서 사용한 Synopsys사의 Design Analyzer와 Xilinx사의 ISE, Active HDL software는 IDEC을 통해 공급 받았음.

참고문헌

[1] IEEE Std 802.11a, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specification: High-speed Physical Layer in the 5Ghz Band, IEEE, Inc., 1999.
[2] J. Oh and M. Lim, "Implementation of FFT processor for IEEE 802.11a Wireless LAN," IDEC Conference 2000, pp. 131-133, June. 2000.
[3] W. Li and L. Wanhammar, "A Pipeline FFT Processor," SiPS, 1999.
[4] W. Li, Y. Ma, and L. Wanhammar, "Word Length Estimation for Memory Efficient Pipeline FFT/IFFT Processors," ICSPAT, Nov. 1999.
[5] L. Wanhammar, DSP Integrated Circuits, Academic Press, 1999.
[6] A. V. Oppenheim and R. W. Schafer, Discrete-Time Signal Processing, Prentice-Hall, 1999.
[7] 김재석, 조용수, 조중휘, 이동통신용 모뎀의 VHDL 설계, 대영사, 2001.
[8] 이창훈, 곽성민, 강봉순, "OFDM을 위한 파이프라인 구조인 R4SDC와 R22SDF IFFT/FFT 모듈의 설계 비교," 대한전자공학회 한국통신학회 부산 경남지부 춘계 종합 학술논문발표 논문집, pp. 63, 2003.6.
[9] 변형수, 곽성민, 강봉순, "OFDM에 적용 가능한 오차를 고려한 파이프라인 구조의 IFFT/FFT의 설계 비교," 2002년도 하계종합학술발표회 논문초록집, 한국통신학회, pp. 516, 2002.7.
[10] 변형수, 이봉근, 강봉순, "IEEE 802.11a-1999를 위한 IFFT/FFT 모듈의 최적화 설계," 한국 신호처리 · 시스템 학회 추계 종합 학술논문집 2권 2호, pp. 549, 2001.11.



이창훈(Chang-Hoon Lee)
2002년 동아대학교 전기전자컴퓨터공학부 전자전공 (공학사)
2002년~현재 동아대학교 대학원 전자공학과 석사과정
※관심분야 : OFDM FFT, DAB FFT



김주현(Ju-Hyun Kim)
2002년 동아대학교 전기전자컴퓨터공학부 전자전공 졸업 (공학사)
2004년 동아대학교 대학원 전자공학과 (공학석사)
2004년~현재 동아대학교 대학원 전자공학과 박사과정
※관심분야 : System IC Design

강봉순(Bongsoon Kang)

1985년 연세대학교 전자공학과 (공학사)
1987년 미국 University of Pennsylvania 전기공학과 (공학석사)
1990년 미국 Drexel University 전기 및 컴퓨터공학과 (공학박사)
1989년~1999년 삼성전자 반도체 수석연구원
1999년~현재 동아대학교 전기전자컴퓨터 공학부 조교수
※관심분야 : VLSI Design, ASIC Design