OFDM 기반 통신 시스템용 단일 메모리 구조의 64~8,192점 FFT/IFFT 코어 생성기

임창완* · 전흥우**· 신경욱***

A Generator of 64~8,192-point FFT/IFFT Cores with Single-memory Architecture for OFDM-based Communication Systems

Chang-Wan Yeem* · Heung-Woo Jeon** · Kyung-Wook Shin***

본 연구는 금오공과대학교 학술연구비에 의하여 연구된 논문임

요 약

본 논문에서는 OFDM 기반의 통신 시스템용 FFT/IFFT 코어 생성기(FCore_Gen)를 구현하였다. FCore_Gen은 FFT 길이, 입력 비트수, 내부 중간 결과 값의 비트수, 격자계수 비트수 등의 선택에 따라 총 640가지의 FFT/IFFT 코어를 Verilog-HDL 코드로 생성한다. 생성되는 FFT/IFFT 코어는 in-place 방식의 단일 메모리 구조를 기반으로 하며, FFT 길이에 따라 radix-4, radix-2 알고리듬의 혼합 구조가 적용된다. 또한, 메모리 감소와 연산 정밀도 향상을 위하여 중간 결과 값의 크기에 따른 조건적 스케일링이 연산 stage 단위로 적용되도록 하였다. 생성되는 코어를 0.35-㎛ CMOS 표준 셀로 합성한 결과 75-MHz@3.3-V의 속도로 동작 가능하여 64점 FFT 연산에 2.55-㎲가 소요되고, 8192점 FFT 연산에 762.7-㎲가 소요되어 OFDM 기반의 무선 랜, DMB, DVB 시스템의 요구조건을 만족한다.

ABSTRACT

This paper describes a core generator (FCore_Gen) which generates Verilog-HDL models of 640 different FFT/IFFT cores with selected parameter value for OFDM-based communication systems. The generated FFT/IFFT cores are based on in-place single memory architecture, and use a hybrid structure of radix-4 and radix-2 DIF algorithm to accommodate various FFT lengths. To achieve both memory reduction and the improved SQNR, a conditional scaling technique is adopted, which conditionally scales the intermediate results of each computational stage. The cores synthesized with a 0.35-㎛ CMOS standard cell library can operate with 75-MHz@3.3-V, and a 8,192-point FFT can be computed in 762.7-㎲, thus the cores satisfy the specifications of wireless LAN, DMB, and DVB systems.

키워드

FFT, OFDM, core generator, Wireless LAN, DMB, DVB

Key word

FFT, OFDM, core generator, Wireless LAN, DMB, DVB

* 금오공과대학교 대학원 전자공학과 석사과정

** 금오공과대학교 전자공학부 교수 (교신저자)

*** 금오공과대학교 전자공학부 교수

접수일자 : 2009. 07. 29

심사완료일자 : 2009. 08. 24

Ⅰ. 서 론

직교 주파수 분할 다중화 (Orthogonal Freque- ncy Division Multiplexing; OFDM) 방식은 고속의 데이터 전송을 위한 기술로 유럽, 일본 및 호주의 디지털 TV 표준으로 채택되었으며, 무선 랜, 휴대 인터넷 (Wi-Bro), 지상파 DMB 및 UWB(Ultra Wide-Band) 등 디지털 무선 통신 및 방송 시스템에 폭넓게 이용되고 있다. OFDM의 원리는 고속의 데이터 스트림을 여러 개의 저속의 데이터 스트림으로 분할하여 이들을 다수개의 반송파에 실어 동시에 전송하는 것이다[1]. FFT/IFFT는 다수 반송파의 변·복조를 수행하는 OFDM의 핵심 블록이다. OFDM 기반 변·복조기는 응용 시스템에 따라 64점에서 8192점 범위의 다양한 FFT 코어가 사용되며, FFT 길이에 따른 FFT 코어를 생성하는 IP 개발이 필요하다[2][3].

FFT 구조에는 단일 메모리 구조, 이중 메모리 구조, 파이프라인 구조, 어레이 구조 등이 있으며, 일반적으로 파이프라인 구조와 단일 메모리 구조가 사용된다[4]. 파이프라인 구조의 FFT 코어 생성기가 문헌에 발표된 바 있으며[3], 파이프라인 구조는 높은 처리율을 얻을 수 있지만 다수의 복소수 덧셈기와 곱셈기로 인해 큰 면적을 차지하는 단점을 가진다. 반면에 단일 메모리 구조는 하나의 나비 연산기와 중간 결과 값을 저장하기 위해 하나의 메모리를 사용하기 때문에 다른 구조와 비교하여 처리율이 낮으나 작은 면적으로 구현이 가능한 장점이 있다[5][6][7].

나비 연산기 구조로는 radix-2와 radix-4가 일반적으로 사용되는데, radix-2의 경우 구조가 간단하고 2의 누승 길이의 FFT 연산을 처리하는데 용이하다. 그러나 $\log_2 N$만큼의 연산단계가 필요하여 단일 메모리 구조에 적용할 경우 한 OFDM 심볼에 대한 FFT 연산 시간이 증가하는 단점이 있다. radix-4의 경우는 radix-2와 비교하여 2배의 처리율을 보이나 FFT 길이가 2의 누승이 아닌 형태에 대해 적용하기 힘들기 때문에 효과적인 구조에 대한 고려가 필요하다[8].

본 논문에서는 저면적으로 FFT/IFFT 코어를 구현하기 위하여 단일 메모리 구조를 선택하였으며, radix-2와 radix-4를 FFT/IFFT 길이에 따라 선택적으로 수행이 가능하도록 설계되어 OFDM 통신 시스템에서 사용되는 $N=64\times 2^k$ $(0\le k\le 7)$의 8가지 FFT 길이에 대해 FFT/IFFT 연산이 가능하도록 하였다. 구현된 FFT/IFFT 코어 생성기는 파라메터의 선택에 따라 총 640가지의 FFT/IFFT 코어를 Verilog- HDL 코드로 생성한다.

Ⅱ. 단일 메모리 구조의 FFT/IFFT 코어 생성기

FCore_Gen은 FFT 길이, 입력 비트수, 내부 중간 결과 값의 비트수, 격자계수(twiddle factor) 비트수 등을 지정하면 자동으로 파라메터 값을 설정하여, 총 640가지의 FFT/IFFT 코어 중 하나의 Verilog-HDL 코드를 생성한다. 표 1은 FCore_Gen의 파라메터 선택 범위를 나타낸 것이며, 그림 1은 MFC로 구현된 FCore_Gen의 실행 화면을 나타낸다.

표 1. FCore_Gen의 파라메터 선택범위
Table. 1 Parameter selection rages of the FCore_Gen

Parameter	Selection Range
FFT length	64~8,192-point
Input bit width	6~14-bit with 2-bit step
Internal bit width	10~18-bit with 2-bit step
Twiddle bit width	8~18-bit with 2-bit step

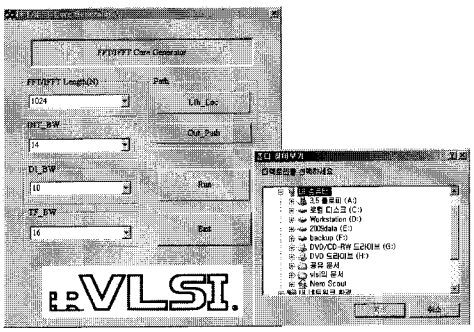


그림 1. FCore_Gen의 실행 화면
Fig. 1 The Gui of the FCore_Gen

2.1 FFT/IFFT 코어의 구조

FCore_Gen에서 생성되는 FFT/IFFT 코어는 그림 2와 같이 단일 메모리 구조를 기반으로 하며, 하나의 나비 연산기와 하나의 복소수 곱셈기, 격자계수 메모리, 스케일 블록, 그리고 이들 블록의 동작을 제어하기 위한 제어블록 등으로 구성된다.

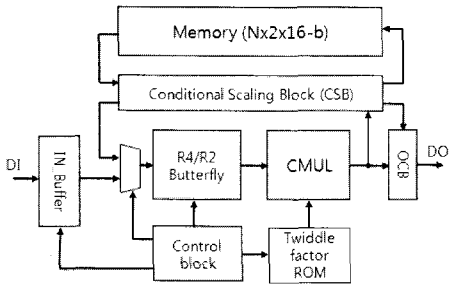


그림 2. FCore_Gen에서 생성되는 FFT 코어의 구조
Fig.2 The architecture of the FFT core generated by the FCore_Gen

나비 연산기는 radix-4 연산을 기본으로 하며, 선택적으로 radix-2 연산이 수행되어 64점에서 8192점 범위의 FFT/IFFT를 연산한다. FFT의 길이 N에 대해 $N=4^k$ $(3 \le k \le 6)$인 경우에는 k번의 radix-4 연산이 반복 수행되며, $N=2^{2k+1}$인 경우에는 k번의 radix-4 연산과 한 번의 radix-2 연산이 수행된다. 각 연산 stage의 중간 결과 값은 내부 데이터에 의해 선택되며, 조건적 스케일링 블록 CSB(Conditional Scaling Block)은 각 연산 stage 결과 값의 크기에 따라 조건적으로 스케일링하여 유한한 내부 비트 수에 따른 연산 오차가 최소화되도록 하였다. 메모리 주소는 각 연산 stage의 중간 결과 값의 읽기주소와 쓰기주소가 동일하게 사용되는 in-place 방식을 사용하여 데이터 간 충돌을 방지 하였다. 복소수 곱셈기 CMUL(Complex MULtiplier)는 RB (Redundant Binary) 수치계를 이용하여 설계되었다. RB 수치계를 이용하는 방법은 2진 수치계를 이용하는 방법에 비해 부분 곱의 수가 1/3로 감소되어 회로의 단순화와 함께 곱셈 속도의 향상, 전력소모 감소 등의 장점을 갖는다[9]. 입력단의 IN_Buffer는 순차적으로 입력되는 데이터를 받아 버퍼링하여 연속적인 실시간 동작이 가능하도록 하며, 출력 보상 블록 OCB(Output Compensation Block)은 역스케일링을 통해 최종 출력 데이터의 크기를 보정한다.

2.2 나비 연산기

나비 연산기는 radix-4 연산과 radix-2 연산이 선택적으로 수행될 수 있도록 그림 3과 같은 구조로 설계되었다. radix-4 연산을 기본으로 하며, 그림 3에서와 같이 select_r2 신호에 의해 radix-2 연산이 선택적으로 수행된다. 생성된 FFT/IFFT 코어는 저면적 구현을 위해 하나의 복소수 곱셈기만을 사용하므로, 나비 연산기는 한 사이클 당 하나의 결과를 출력한다.

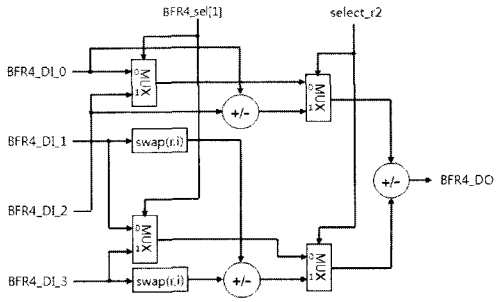


그림 3. radix-4/radix-2 나비 연산기
Fig. 3 The radix-4/radix-2 butterfly unit

$s = \lfloor \log_4 N \rfloor$ // Number of stage
$b = \log_2 N$ // Number of address bit
$a_0[b-1:0] = cnt[b-1:0]$
for k from 1 to s begin
 if (k=1)
  $a_k = concat(a_0[1:0], a_0[b-1:2])$
 else
  if (k=s)
   $a_k = a_0$
  else
   if (k=2)
    $a_k = concat\{swap(a_{k-1}[b-2k+3:b-2k+2], a_{k-1}[b-2k+1:b-2k]), a_{k-1}[b-2k-1:0]\}$
   else
    $a_k = concat\{a_{k-1}[b-1:b-2k+4], swap(a_{k-1}[b-2k+3:b-2k+2], a_{k-1}[b-2k+1:b-2k]), a_{k-1}[b-2k-1:0]\}$

그림 4. 메모리 주소 생성 알고리듬
Fig. 4 An algorithm for the memory address generation

따라서 radix-4 연산에 4 사이클이 소요되며, radix-2 연산에 2 사이클이 소요된다.

2.3 메모리 주소 생성

설계된 FFT/IFFT 코어는 in-place 방식을 사용한 단일 메모리 구조이므로, 효율적인 메모리 주소의 생성이 중요하다. 본 논문에서는 그림 4와 같은 메모리 주소 생성 알고리듬을 이용하여 메모리 주소 생성회로를 설계하였다. 그림 4에서 concat(x, y)는 x와 y를 순서대로 결합하는 것을 의미하며, swap(x, y)는 x와 y의 위치 교환을 의미한다. 그림 4의 알고리듬을 이용한 메모리 주소 생성회로는 그림 5와 같이 0에서 N-1까지 증가하는 증가 계수기와 $b(\log_2 N)$ 개의 MUX로 구성된다. 그림 6은 메모리 주소 생성 회로를 이용하여 생성되는 주소이며, in-place 방식을 사용하기 때문에 읽기와 쓰기 주소는 동일하다.

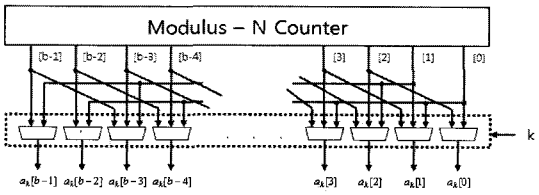


그림 5. 메모리 주소 생성기 블록도
Fig. 5 The block diagram of the memory address generator

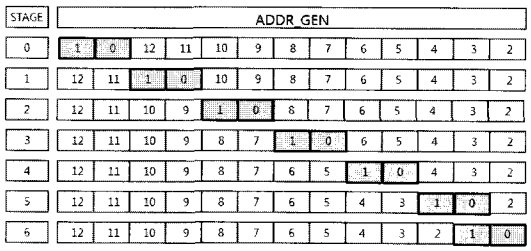


그림 6. 생성된 메모리 주소 (N=8,192)
Fig. 6 The generated memory address (N=8,192)

2.4 조건적 스케일링

FFT 연산은 나비 연산과 복소수 곱셈으로 구성되는 연산단계를 거치면서 내부 데이터의 비트수가 지속적으로 증가하게 된다. 따라서 고정된 메모리 비트를 갖는 단일 메모리 구조에서는 증가된 데이터의 스케일링이 필수적이다.

본 논문에서는 각 연산 stage 마다 중간 결과 값의 크기에 따른 조건적 스케일링을 통해 오차를 최소화하는 방법을 고안하여 적용하였다. 조건적 스케일링 블록 CSB은 각 연산 stage의 복소수 곱셈결과로부터 SSI (Stage Scaling Index)를 찾고, 이를 이용하여 메모리에 저장된 중간 결과 값을 스케일링하여 다음 stage의 연산에 사용되도록 한다.

FCore_Gen에서 생성되는 FFT/IFFT 코어는 각 연산 stage의 처리가 연속적으로 이루어지며, 따라서 그림 4와 같이 k번째 stage의 마지막 복소수 곱셈과 (k+1)번째 stage의 첫번째 나비연산이 동시에 이루어지게 된다. k번째 stage의 SSI는 해당 stage의 마지막 복소수 곱셈이 완료되는 T3시점에서 확정되며, 이는 (k+1)번째 stage가 시작된 이후가 된다. 따라서 (k+1)번째 stage의 첫 번째 나비 연산 입력 데이터는 k번째 stage의 SSI를 적용하여 스케일링하는 것이 불가능하다. 이와 같은 문제를 해결하기 위해 각 연산 stage의 첫 번째 나비 연산에 사용되는 데이터는 그림 7의 T1시점에서 생성되는 임시 스케일링 지수 TSI(Temporal Scaling Index)를 적용하여 임시 스케일링하고, 복소수 곱셈이 완료되는 T3시점에서 정확한 SSI가 확정되면 TSI와 SSI의 차이만큼 보상해준다. 한편, 매 연산 stage의 두 번째 나비 연산의 입력부터는 확정된 SSI를 이용한 스케일링 결과가 사용된다.

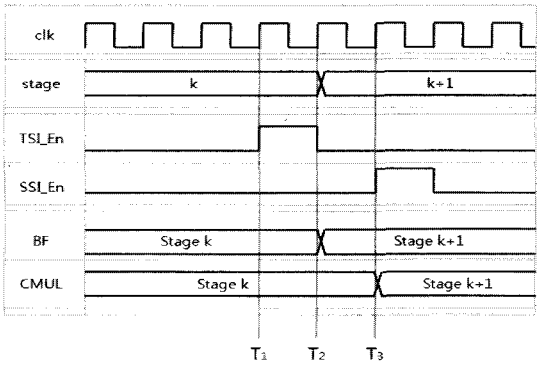


그림 7. TSI/SSI 타이밍도
Fig. 7 The TSI/SSI timing diagram

연산 stage를 거치면서 중간 결과 값들은 SSI 만큼 스케일링이 이루어지므로, 최종 출력 데이터는 역 스케일링을 통한 보상이 필요하다. 출력보상 블록 OCB은 각

연산 stage를 통해 누적된 스케일링 지수 ACSI (ACcumulation Scaling Index) 만큼 역스케일링을 통해 최종 출력 데이터를 보상을 한다. 또한, OCB에는 FFT/IFFT를 선택적으로 수행하기 위해 실수부와 허수부를 교환하는 스위치 블록이 포함된다.

2.5 격자계수 생성기

복소수 곱셈에 사용되는 격자계수(W_N^{nk})는 sine 파형과 cosine 파형의 한 주기를 샘플링 하여 ROM에 저장하는데, FFT 길이 N이 긴 경우에는 격자계수를 저장하기 위한 ROM의 크기가 매우 커지게 되어 면적과 전력소모가 증가하게 된다. 본 설계에서는 sine 함수와 cosine 함수의 대칭성을 이용하여 1/8 주기만을 ROM에 저장한 후, ROM의 읽기 주소를 적절히 제어하여 각 연산 단계에서 필요한 격자계수가 생성되도록 하였다. 그림 8은 격자계수 생성기를 도식화 한 것이며, 격자계수 생성기는 격자계수 주소 생성기, ROM, 그리고 디코더로 구성된다. 격자계수 주소 생성기는 그림 9와 같은 알고리듬을 이용하여 쉬프트 연산으로 구현하였다.

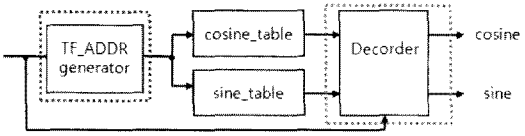


그림 8. 격자계수 생성기의 블록도
Fig. 8 The block diagram of the twiddle factor generator

```
TW_CNT= CNT/4;
TW_RST= CNT%4;

leftshift (x, c) return (x << c);

    case(STAGE){
        0 :     TW_CON_ADDR = leftshift (TW_CNT, 0);
        1 :     TW_CON_ADDR = leftshift (TW_CNT, 2);
        2 :     TW_CON_ADDR = leftshift (TW_CNT, 4);
        3 :     TW_CON_ADDR = leftshift (TW_CNT, 6);
        4 :     TW_CON_ADDR = leftshift (TW_CNT, 8);
        5 :     TW_CON_ADDR = leftshift (TW_CNT, 10);
        6 :     TW_CON_ADDR = leftshift (TW_CNT, 12);
      default : TW_CON_ADDR = 0;
    }

TW_ADDR  = TW_CON_ADDR*TW_RST;
```

그림 9. 격자계수 주소 생성 알고리듬
Fig. 9 An algorithm for the twiddle factor address generation

III. 설계 검증 및 성능 분석

생성된 FFT 코어는 ModelSim과 Matlab을 이용하여 성능 평가를 수행하였다. 그림 10은 성능평가 과정을 보인 것이며, 2진 랜덤 신호를 생성하여 변조(QPSK, 16-QAM, 64-QAM)한 후, 부동점(floating-point)연산을 갖는 이상적인 IFFT와 이득 조정을 거쳐 양자화 된 데이터를 입력 데이터로 사용하였다.

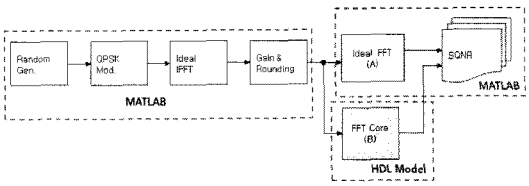


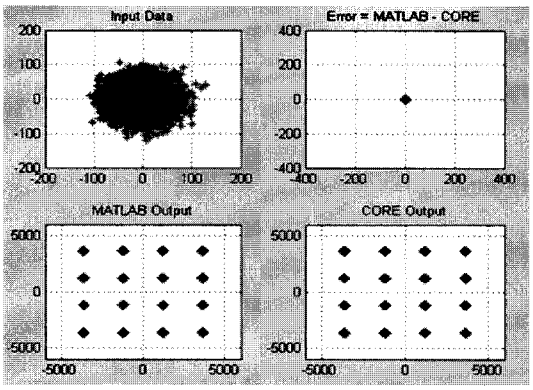
그림 10. FFT 코어의 성능평가 과정
Fig. 10 The performance evaluation procedure of the FFT core

SQNR은 식 (1)에 의해 계산 되었으며, 식 (1)에서 A는 Matlab에서 연산된 결과 값을 나타내며 B는 생성된 FFT 코어에서 연산된 결과 값을 나타낸다.

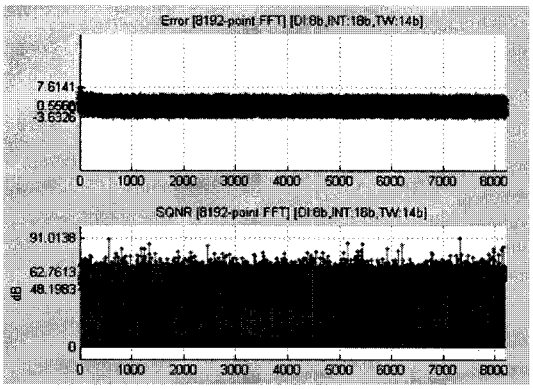
$$SQNR = \frac{\sum[Re(A)]^2 + \sum[Im(A)]^2}{\sum[Re(A) - Re(B)]^2 + \sum[Im(A) - Im(B)]^2} \quad (1)$$

그림 11-(a)는 8,192점 FFT 모드의 성능을 평가한 것인데, 사용된 입력 데이터는 좌측 상단의 성상도와 같으며, 좌측 하단의 성상도는 Matlab에서 얻어진 이상적 FFT 출력이고, 우측 하단의 성상도는 생성된 FFT 코어에서 얻어진 FFT 출력이다. 연산오차는 우측 상단의 성상도와 같이 매우 작은 범위를 나타낸다. 연산 정확도는 SQNR 분석을 통해서도 알 수 있으며, 그림 11-(b)는 연산오차와 SQNR 특성을 주파수에 따라 나타낸 것이다.

그리고 ModelSim을 이용한 결과 파형은 그림 12와 같이 나타나 올바른 데이터가 출력되는 것을 확인하였다.



(a)



(b)

그림 11. 생성된 FFT 코어의 성능 평가 (N=8,192)
(a) 성상도 (b) 오차와 연산정밀도
Fig. 11 The performance evaluation of the FFT core (N=8,192) (a) Constellations (b) Error and SQNR

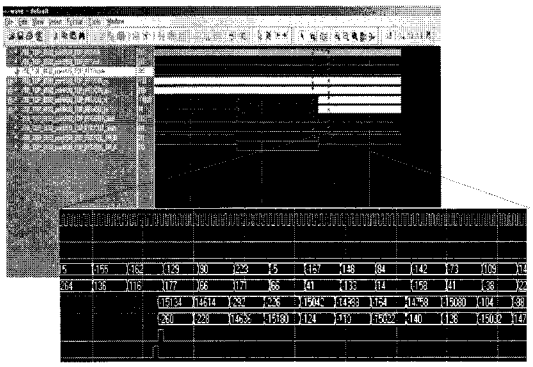
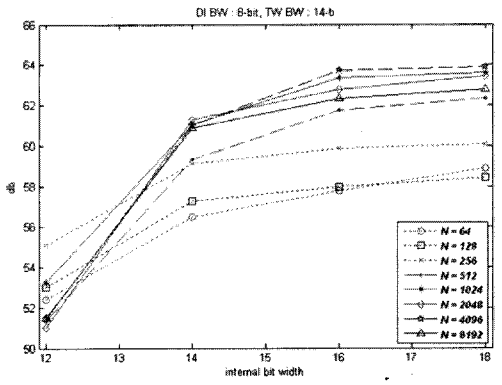
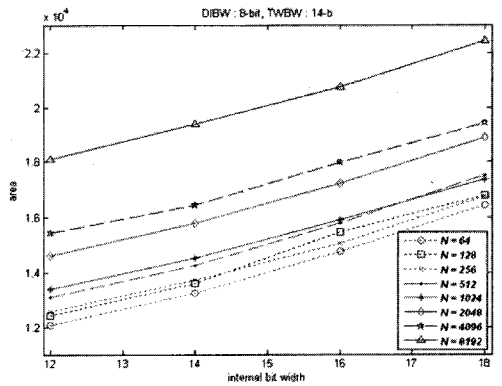


그림 12. 모델심을 이용한 기능 검증 (N=8,192)
Fig. 12 The functional simulation in ModelSim (N=8,192)

그림 13-(a)는 입력 8-bit, 격자계수 14-bit일 때 8 가지의 FFT 길이에 대한 SQNR을 분석한 결과이며, y 축은 db을 나타내고 x 축은 내부비트수를 의미한다. 8,192점 FFT 연산의 경우 내부 중간 결과 값의 비트수가 12-bit, 14-bit, 16-bit, 18-bit일 때의 SQNR이 각각 51.4975, 60.8845, 62.3297, 62.7613으로 측정되어 내부 중간 결과 값이 14-bit 이상인 경우, 성능의 차이가 미미한 것으로 나타났다. 또한 0.35-μm CMOS 표준 셀로 합성한 결과, 그림 13-(b)와 같은 면적을 확인 할 수 있었다. 그림 13-(b)의 y 축은 면적을 나타내고 x 축은 내부비트수를 나타내며, 내부 중간 결과 값의 비트수가 커짐에 따라 면적이 증가하는 것을 확인 할 수 있다.



(a)



(b)

그림 13. 내부 비트수에 따른 성능 분석
(a) SQNR 분석 (b) 면적 분석
Fig. 13 The performance analysis with internal bit widths (a) the SQNR analysis (b) the Area analysis

표 2는 입력 10-bit, 격자계수 14-bit, 내부 중간 결과 값의 비트수를 14-bit로 설정한 경우, FFT 길이 N에 따른 SQNR, 면적, 연산시간, 그리고 통신 시스템에 따른 유효 심볼 길이를 나타낸 것이다. FFT/IFFT 코어의 연산 정밀도는 최소 58-dB(N=8,192)에서부터 최대 63-dB (N=64)의 SQNR을 가지며, 생성된 코어의 연산시간이 각 통신 시스템에서 요구하는 조건을 모두 만족하는 것으로 나타났다.

표 2. FFT길이에 따른 성능 분석
Table. 2 The performance analysis with FFT lengths

N	SQNR (avg)	Area		연산 시간 [μs]	유효심볼 길이[μs] (통신 시스템)
		gate	Mem [byte]		
64	63.86	14,548	2,048	2.55	3.2 (WLAN)
128	61.22	15,031	4,096	6.81	160 (ACIS)
256	62.90	14,994	8,192	13.62	125 (DAB Ⅲ)
512	60.43	15,556	16,384	34.05	250 (DAB Ⅱ) 234 (ADSL)
1024	60.22	16,113	32,768	68.10	500 (DAB Ⅳ) 102.4 (Wi-Bro)
2048	59.19	16,940	65,536	163.43	1000 (DAB Ⅰ) 224 (DVB-T:2k)
4096	58.50	17,874	131,072	326.86	448(DVB-H:4k)
8192	58.43	21,297	262,144	762.68	896 (DVB-T:8k)

표 3. 1,024점 FFT 코어의 성능 비교
Table. 3 The performance comparison of 1,024-point FFT cores

	Spiffee [4]	CS2410 [10]	CFMR [11]	This paper
Technology[μm]	0.7	0.18	0.18	0.35
Gate count	115,000	39,000	37,000	16,113
Frequency[MHz]	173	100	100	75
Memory[KByte]	4.5	4.0	8.0	4.0
Radix	radix-2	mixed-radix	mixed-radix	radix-2/4
In-place	Yes	Yes	Yes	Yes
Word length[bit]	18	16	16	14
Computation cycles	5,100	5,175	1,280	5,120

표 3은 1024점 FFT 코어를 기준으로 성능을 비교한 것이다. Spiffee의 cached memory 방식[4]의 115,000게이트, mixed-radix 방식의 CFMR [11]의 37,000게이트에 비하여 각각 86%와 56% 감소된 16,113게이트로 구현되었으며, 따라서 본 논문의 FFT 코어는 저면적/저전력 특성이 우수하다.

Ⅳ. 결 론

본 연구에서는 다중 표준 OFDM 통신 시스템에 활용하기 위하여 FFT 길이 $N=64\times 2^k$ $(0\le k\le 7)$의 8가지에 대해 FFT/IFFT 연산이 가능하도록 하였으며, 파라메터의 선택에 따라 총 640가지의 FFT/IFFT 코어를 Verilog-HDL 코드로 생성하는 생성기인 FCore_Gen을 구현하였다.

생성되는 FFT/IFFT 코어의 구조는 크기 N의 단일 메모리 구조를 가지며, in-place 방식을 사용하여 데이터 간의 충돌을 피하고 메모리 주소 생성 알고리듬을 통하여 효과적으로 구현하였다. 나비 연산기와 복소수 곱셈기를 각각 하나씩 사용하였으며, 메모리 크기 감소와 정밀도 향상을 위해 조건적 스케일링을 적용하였다.

생성되는 FFT/IFFT 코어는 75-MHz@3.3-V로 동작 가능하여 8,192점 FFT의 연산시간은 762.7-μs로서 DVB-T 표준에서 제시하는 유효 심볼 구간인 896-μs을 충분히 만족하며, WLAN (802.11a), DAB, DVB-T, DVB-H, WiBro, 그리고 VDSL과 같은 OFDM 기반의 다양한 통신 시스템에 폭 넓게 사용될 수 있을 것이다.

감사의 글

반도체설계교육센터(IDEC)의 CAD Tool 지원에 감사드립니다.

참고문헌

[1] 조용수, 무선 멀티미디어 통신을 위한 OFDM 기초, 대영사, 2001.

[2] J. C. Kuo, C. H. Wen, A. Y. Wu, "Imple- mentation of a

programmable 64-2048- point FFT/IFFT processor for OFDM- based communication systems", *Proceedings of the IEEE international symposium on circuits and systems,* vol. 2, pp. 121-124, May 2003.

[3] 이진우, 신경욱, 김종환, 백영석, 어익수, "OFDM 모뎀용 FFT/IFFT IP 자동 생성기", 한국통신학회, 제31권, 제3A호, pp. 368-376, 2006.

[4] Bevan M. Baas, "A low-power, high- performance, 1024-point FFT processor", *IEEE J. Solid-State Circuits*, vol. 24, no. 3, pp. 380-387, Mar. 1999.

[5] Shousheng He and M. Torkelson, "Design and implementation of a 1024-point pipeline FFT processor", Proceedings of the IEEE Custom Integrated Circuits Conference, pp. 131-134, 1998.

[6] E. Bidet, D. Castelain, C. Joanblanq, P. Senn, "A fast single-chip implementation of 8192 complex point FFT", IEEE Journal of Solid-State Circuits, vol. 30 Issue:3, pp. 300-305, March 1995.

[7] Bevan M. Baas, "A low-power, high- performance, 1024-point FFT processor", IEEE Journal of Solid-State Circuits, vol.34, issue:3, pp. 280-387, March 1999.

[8] Y.J. Hongil and J. Kim, "New efficient FFT algorithm and pipeline implementation result for OFDM/DMT applications", IEEE Trans. on Consumer Electronics, vol. 49, no. 1, pp. 14-20, Feb. 2003.

[9] Kyung-Wook Shin, Bang-Sup Song, Kantilal Bacrania, "A 200-MHz complex number multiplier using redundant binary arithmetic", IEEE Journal of Solid-State Circuits, vol. 33, no. 6, pp. 904-909, June, 1998.

[10] Amphion, 8-1024 Point FFT/IFFT, CS2410, Jul. 2001.

[11] B. G. Jo and M. H. Sunwoo, "New continuous-flow mixed-radix (CFMR) FFT processor using novel in-place strategy", *IEEE Trans. on Circuits and Systems I: Regular Papers,* vol. 52, no. 5, pp. 911-919, May 2005.

저자소개

임창완(Chang-Wan Yeem)

2008.2 금오공과대학교 전자공학과 졸업
2008.2~~현재 금오공과대학교 대학원 전자공학과 석사과정
※관심분야 : 통신/신호처리, 집적회로 설계, SoC 설계

전흥우(Heung-Woo Jeon)

1980.2 한국항공대학교 전자공학과 (공학사)
1982.2 고려대학교 대학원 전자공학과 (공학석사)
1988.8 고려대학교 대학원 전자공학과(공학박사)
1989.3~현재 금오공과대학교 전자공학부 교수
※관심분야 : 집적회로설계, 신경망회로

신경욱(Kyung-Wook Shin)

1984.2 한국항공대학교 전자공학과 (공학사)
1986.2 연세대학교 대학원 전자공학과(공학석사)
1990.8 연세대학교대학원 전자공학과(공학박사)
1990.9~1991.6 한국전자통신연구소
1991.7~현재 금오공과대학교 전자공학부(교수)
1995.8~1996.7 University of Illinois at Urbana-Champaign (방문교수)
2003.1~2004.1 University of California at San Diego (방문교수)
※관심분야 : 통신 및 신호처리용 SoC 설계, 정보보호 SoC 설계, 반도체 IP 설계