Proceedings of KIIS Spring Conference 2008 Vol. 18, No. 1.

OFDM 해상통신방식용 저전력 32-point FFT 알고리즘에 관한 연구

A Study on Low Power 32-point FFT Algorithm for OFDM Maritime Communication

조승일[1] · 이광희[1] · 조하나[1] · 김근오[1] · 이충훈[1] · 박계각[2] · 조주필[3] · 차재상[4] · 김성권©
Seung-il Cho, Kwang-Hee Lee, Hana Jo, Keun-O Kim, Chunghoon Lee, Gye-Kack Park, Juphil Cho, Jaesang Cha and Seung-Kweon Kim

[1]목포해양대학교 대학원 해양전자통신공학과
E-mail : whtmddlf@mmu.ac.kr
[2]목포해양대학교 해상운송시스템학부
E-mail : gkpark@mmu.ac.kr
[3]군산대학교 공과대학 전자정보공학부
E-mail : stefano@kunsan.ac.kr
[4]서울산업대학교 매체공학과
E-mail : chajs@snut.ac.kr
©목포해양대학교 대학원 해양전자통신공학과
E-mail : skkim12632@mmu.ac.kr

요 약

유비쿼터스 네트워크의 실현을 위한 4세대 통신방식의 유력한 후보로 부상하는 OFDM (Orthogonal Frequency Division Multiplexing) 통신방식이 육상에서 주목받고 있으며, 고속 데이터 전송을 위한 무선랜의 표준기술로 확정되어 있다. 해상 통신의 경우에서도 OFDM 통신방식은 단파대역을 이용한 데이터 전송방식으로 제안되고 있으며, ITU (International Telecommunication Union)는 해상통신에서 32-point FFT를 사용하도록 권고하고 있다. 해상 통신에서는 해양사고 및 조난 시에도 통신이 이루어져야 하는 한계상황을 고려하면, OFDM 통신방식의 중요 디바이스인 FFT는 저전력으로 동작되어야 한다. 따라서 본 논문에서는 OFDM 방식의 중요 디바이스인 32-point FFT를 저전력으로 동작시키기 위해 radix-2와 radix-4를 이용하여 저전력 32-point FFT 알고리즘을 제안한다. 최적화된 설계로 32-point FFT를 저전력 동작이 가능하도록 설계하였으며, 제안한 알고리즘은 VHDL로 구현하고 FPGA Spartan3 board에 장착하여 Matlab의 이론값과 비교, 검증하였다, 제안된 32-point FFT는 해상통신에서의 OFDM 적용을 위한 선도기술로 유용할 것이다.

Key Words : OFDM (Orthogonal Frequency Division Multiplexing), 단파대, 저전력, 32-point FFT

1. 서 론

유비쿼터스 시대를 맞이한 현대 사회는 음성, 사진, 동영상 등의 멀티미디어 통신 및 고속 데이터 통신에 대한 수요가 증가함에 따라 다양한 무선통신 방식의 출현과 시스템 광대역화 현상이 두드러지게 나타나고 있다. 최근 고속 데이터 통신을 지향하는 차세대 무선통신기술 개발에 대한 연구가 활발한 가운데 OFDM (Orthogonal Frequency Division Multiplexing) 기술이 주목받고 있다 [1].

OFDM은 IEEE802.11a 및 IEEE802.11g의 작업그룹에 의하여 5GHz대역 및 2GHz대역에서 고속 데이터 전송을 위한 WLAN(Wireless Local Area Network)의 표준기술로 확정되어있으며, 육상 통신뿐만 아니라 해상 통신에서도 OFDM 통신방식에 대해 주목하고 있다[2]. 해상통신에서 데이터 통신을 위해 위성통신을 사용하지만, 통신료가 비싸므로 중,소형 선박에서는 사용할 수 없기 때문에 음성 통신이 주로 사용되었던 HF (High Frequency) 및 VHF(Very High Frequency) 대역에서 데이터 통신을 하기 위한 연구가 활발히 진행 되고 있고, HF 및 VHF 대역의 주파수 효율적인 사용과 대용량 고속 데이터 통신을 위하여 OFDM 통신방식이 제안되고 있다[3]. ITU(International Telecommunication Union)에서는 해상통신에서 OFDM의 적용에 대한 제안들에 대해 2006년 9월에 회의를 하고 해상통신에서 OFDM을 적용할 경우 32-point FFT(Fast Fourier Transform)를 사용하도록 권고하였다 [4].

무선통신의 가장 큰 쟁점은 통신의 지속성과 이동성이

감사의 글 : 본 연구내용의 일부는 해양수산부 특정연구개발사업의 연구비 지원 (과제번호 F10702407A220000110) 에 의해 수행되었습니다.

며, 해상통신에서도 예외일 수는 없다. 특히 긴급한 대처가 요구되며, 자체 전력을 손실하는 해양사고 및 조난 시에도 통신이 이루어져야 한다. 이러한 상황에서도 통신의 지속성과 이동성을 보장 받기 위해서는 통신시스템의 전력소비 문제를 해결해야 한다. 결국 통신시스템은 저전력으로 동작되어야 한다.

OFDM의 부반송파는 직교성을 유지할 수 있도록 IFFT (Inverse Fast Fourier Transform)와 FFT 프로세서를 이용하여 신호를 변조하기 때문에 고성능의 FFT 프로세서를 구현하는 것이 OFDM 방식의 고속 무선 데이터 통신을 구현하기 위한 핵심 사항이라 할 수 있다. 또한 FFT/IFFT 프로세서는 OFDM 방식의 물리계층에서 가장 큰 면적과 전력을 소모한다 [6-9]. 따라서 저전력 OFDM 해상통신시스템을 위해서는 저전력 32-point FFT 구현이 필요하다.

본 논문에서는 32-point FFT를 저전력으로 동작시키기 위해 radix-2와 radix-4를 이용하여 곱셈기의 수를 줄인 저전력 32-point FFT 알고리즘을 제안한다.

2. FFT 알고리즘

2.1 FFT의 개념

푸리에 변환은 시간영역의 신호를 주파수 성분으로 변환하는 방법으로 신호가 가지고 있는 주파수 성분을 통해 스펙트럼 분석에 사용된다. 또한 신호처리 연산의 계산량를 줄이는데 푸리에 변환이 이용된다 [5].

디지털 시스템의 이산 샘플신호의 스펙트럼 분석에는 DFT (Discrete Fourier Transform)가 사용된다. N개의 유한한 샘플 시퀀스를 가지는 DFT 정의는 식 (1)과 같고 행렬로 나타내면 식 (2)와 같다. 여기서 N은 이산 샘플 값의 개수이다.

$$X[k]=\sum_{n=0}^{N-1} x[n]\, W_N^{kn}, \quad k=0,1,\cdots,N-1 \tag{1}$$

$$\begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(N-1) \end{bmatrix} = \begin{bmatrix} W_N^0 & W_N^0 & \cdots & W_N^0 \\ W_N^0 & W_N^1 & \cdots & W_N^{N-1} \\ \vdots & \vdots & \ddots & \vdots \\ W_N^0 & W_N^{(N-1)} & \cdots & W_N^{(N-1)^2} \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[N-1] \end{bmatrix} \tag{2}$$

N-point 시퀀스의 DFT를 직접 구현하는 계산량은 $N(N-1)$의 복소 덧셈과 N^2의 복소 곱셈연산이 필요하다. DFT는 N^2에 비례하는 방대한 계산량과 이에 따른 계산 시간 때문에 N의 크기가 큰 경우 산술 연산을 직접 계산하기 어려워지며, 연산기 수의 증가로 H/W 크기도 커지게 된다. 그러나 회전인자 W_N의 대칭적이고 주기적인 성질은 DFT의 계산 효율을 향상시킴으로써 이러한 문제점들을 개선할 수 있다. 1965년에 Cooley와 Tukey는 DFT 연산 중에 수행 할 계산의 양을 실질적으로 줄이는 알고리즘을 개발하였다. 이러한 효율적인 알고리즘을 FFT라고 한다 [6].

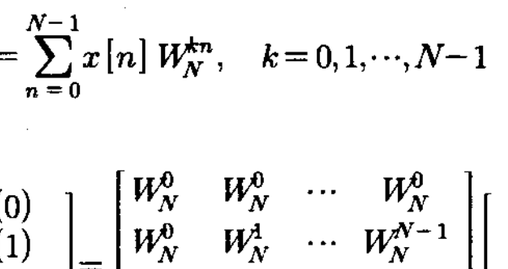


그림 1. radix-2 버터플라이 구조
Fig 1. radix-2 butterfly structure

2.2 Radix-2 알고리즘

N의 크기가 2의 승수일 경우, N은 2×2×2×...×2로 인수분해 되며, N-point Cooley-Tukey 알고리즘은 radix-2 알고리즘으로 전개할 수 있다. radix-2 알고리즘 구현방식을 이해하기 위해 처음 N 을 $2\times N/2$으로 분해하면 $N_1=2$, $N_2=N/2$이 되고 n과 k를 식 (3)으로 나타낼 수 있으며 식 (1)은 식 (4)로 나타낼 수 있다 [6].

$$\begin{aligned} n &= \frac{N}{2}n_1+n_2, \left(n_1=0,1, n_2=0,1,\cdots,\frac{N}{2}-1\right) \\ k &= k_1+2k_2 \quad \left(k_1=0,1, k_2=0,1,\cdots,\frac{N}{2}-1\right) \end{aligned} \tag{3}$$

$$\begin{aligned} X[k_1+2k_2] &= \sum_{n_2=0}^{N/2-1}\left[\left(\sum_{n_1=0}^{1} x\left[\frac{N}{2}n_1+n_2\right] W_2^{k_1n_1}\right) W_N^{k_1n_2}\right] W_{N/2}^{k_2n_2} \\ &= \sum_{n_2=0}^{N/2-1}\left[\left(x[n_2]+W_2^{k_1}x\left[n_2+\frac{N}{2}\right]\right) W_N^{k_1n_2}\right] W_{N/2}^{k_2n_2} \\ &= \sum_{n_2=0}^{N/2-1} \widetilde{B_{N/2}^{k_1}}[n_2]\, W_{N/2}^{k_2n_2} \end{aligned} \tag{4}$$

여기서 $\widetilde{B_{N/2}^{k_1}}[n_2]$는 그림 1과 같이 radix-2 버터플라이 연산을 의미한다. 첫 번째 버터플라이 연산은 $N/2$의 거리를 갖는 두 입력 시퀀스 간에 radix-2 버터플라이 연산을 한다. 첫 번째 버터플라이 연산 후 회전인자와 곱한다. 결국 각 열의 $N/2$-point 변환을 수행하여 출력 시퀀스를 얻게 된다. $N/2$-point 변환 역시 radix-2 알고리즘의 반복적인 처리를 통해 최종적으로 2-point 변환만 남을 때까지 수행하게 된다.

2.3 Radix-4 알고리즘

N의 크기가 4의 승수일 경우, N은 4×4×4×...×4로 인수분해 되며, N-point Cooley-Tukey 알고리즘은 radix-4 알고리즘으로 전개할 수 있다. 처음 N을 4×N/4으로 분해하면 $N_1=4$, $N_2=N/4$이 되고, n과 k를 식 (5)로 나타낼 수 있으며 식 (1)은 식 (6)으로 나타낼 수 있다 [5].

$$\begin{aligned} n &= \frac{N}{4}+n_2, \left(n_1=0,1,\cdots,3, n_2=0,1,\cdots\frac{N}{4}-1\right) \\ k &= k_1+4k_2, \left(k_1=0,1,\cdots,3, k_2=0,1,\cdots\frac{N}{4}-1\right) \end{aligned} \tag{5}$$

$$\begin{aligned} X[k_1+4k_2] &= \sum_{n_2=0}^{n/4-1}\left[\left(\sum_{n_1=0}^{3} x\left[\frac{N}{4}n_1+n_2\right] W_4^{k_1n_1}\right) W_N^{k_1n_2}\right] W_{N/4}^{k_2n_2} \\ &= \sum_{n_2=0}^{n/4-1}\left[\left(x[n_2]+W_4^{k_1}x\left[n_2+\frac{N}{4}\right]+W_2^{k_1}x\left[n_2+\frac{N}{2}\right]+W_4^{3K_1}\left[n_2+\frac{3N}{4}\right]\right)\right. \\ W_N^{k_1n_2}\Big] W_{N/4}^{k_2n_2} &= \sum_{n_2=0}^{n/4-1} \widetilde{B_{N/4}^{k_1}}[n_2]\, W_{N/4}^{k_2n_2} \end{aligned} \tag{6}$$

여기에서 $\widetilde{B_{N/4}^{k_1}}[n_2]$는 그림 2와 같이 radix-4 버터플라이 연산을 의미한다. 첫 번째 버터플라이 연산은 행의 $N/4$의 거리를 갖는 인접한 4개의 원소 간에 radix-4 버터플라이 연산을 한다. 첫 번째 버터플라이 연산 후 회

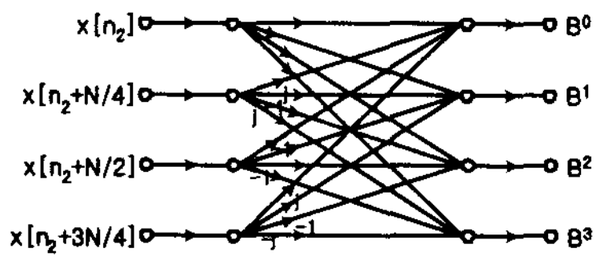


그림 2. radix-4 버터플라이 구조

Fig 2. radix-4 butterfly structure

전인자와 곱한다. 결국 각 열의 *N/4*-point 변환을 수행하여 출력 시퀀스를 얻게 된다. *N/4*-point 변환 역시 radix-4 알고리즘의 반복적인 처리를 통해 최종적으로 4-point 변환만 남을 때까지 수행하게 된다.

3. 제안된 FFT 알고리즘

FFT 알고리즘 중에서 radix-2는 덧셈기 2개, 곱셈기 1개, radix-4는 덧셈기 8개, 곱셈기 3개를 필요로 한다. 본 논문에서는 저전력 32-point FFT 알고리즘을 위해 radix-2와 radix-4의 조합을 통하여 곱셈기를 줄이는 알고리즘을 제안한다.

$$32 = 2^x 4^y \tag{7}$$

32-point에서 radix-2와 radix-4을 혼합할 경우 사용되는 stage 수는 식 (7)과 같다 *x=5, y=0*인 경우 radix-2만 5 stage가 필요하며, *x=3, y=1*인 경우 radix-2는 3 stage, radix-4는 1 stage가 필요하다. 그리고 *x=1, y=2*인 경우 radix-2는 1 stage, radix-4는 2 stage가 필요하다. 여기서 32는 4의 승수가 아니기 때문에 radix-4는 첫 단에 배치할 수 없다.

3.1 알고리즘에 대한 곱셈기 수 산출 및 비교

R2-R2-R2-R2-R2 알고리즘은 32-point FFT를 radix-2를 이용하여 5개의 stage로 나타낸 것이다. 이 알고리즘은 DFT의 기본식 (1)을 식 (8)과 같이 나타낼 수 있다.

$$\begin{aligned} &X[k_1+2k_2+4k_3+8k_4+16k_5] \\ &= \sum_{n_5=0}^{1}\left[\left(\sum_{n_4=0}^{1}\left[\left(\sum_{n_3=0}^{1}\left[\left(\sum_{n_2=0}^{1}\left[\left(\sum_{n_1=0}^{1} x[16n_1+8n_2+4n_3+2n_4+n_5]\right.\right.\right.\right.\right. \\ &\left.\left.\left.\left.\left. W_{32}^{16k_1n_1}\right) W_{32}^{k_1(8n_2+4n_3+2n_4+n_5)}\right] W_{32}^{16k_2n_2}\right) W_{32}^{2k_2(4n_3+2n_4+n_5)}\right] W_{32}^{16k_3n_3}\right) \\ &W_{32}^{4k_3(2n_4+n_5)}\Big] W_{32}^{16k_4n_4}\Big) W_{32}^{8k_4n_5}\Big] W_{32}^{16k_5n_5} \end{aligned} \tag{8}$$

총 버터플라이 stage 수는 5이며, 160개의 복소 덧셈기와 64개의 복소 곱셈기가 필요하다.

R2-R4-R2-R2 알고리즘은 32-point FFT를 radix-2와 radix-4를 이용하여 4개의 stage로 나타낸 것이다. 이 알고리즘은 DFT의 기본식 (1)을 식 (9)와 같이 나타낼 수 있다.

$$\begin{aligned} &X[k_1+2k_2+8k_3+16k_4] \\ &= \sum_{n_1=0}^{1}\left[\left(\sum_{n_3=0}^{1}\left[\left(\sum_{n_2=0}^{3}\left[\left(\sum_{n_1=0}^{1} x[16n_1+4n_4+2n_4+n_4] W_{32}^{16k_1n_1}\right)\right.\right.\right.\right. \\ &\left.\left.\left.\left. W_{32}^{k_1(4n_2-2n_3-n_4)}\right] W_{32}^{8k_2n_2}\right) W_{32}^{2k_2(2n_3-n_4)}\right] W_{32}^{16k_3n_3}\Big) W_{32}^{8k_3n_4}\Big] W_{32}^{16k_4n_4} \end{aligned} \tag{9}$$

총 버터플라이 stage 수는 4이며, 160개의 복소 덧셈기와 56개의 복소 곱셈기가 필요하다.

R2-R2-R2-R4 알고리즘은 32-point FFT를 radix-2와 radix-4를 이용하여 4개의 stage로 나타낸 것이다. 이 알고리즘은 DFT의 기본식 (1)을 식 (10)과 같이 나타낼 수 있다.

$$\begin{aligned} &X[k_1+2k_2+4k_3+8k_4] \\ &= \sum_{n_4=0}^{3}\left[\left(\sum_{n_3=0}^{1}\left[\left(\sum_{n_2=0}^{1}\left[\left(\sum_{n_1=0}^{1} x[16n_1+8n_2+4n_3+n_4] W_{32}^{16k_1n_1}\right)\right.\right.\right.\right. \\ &\left.\left.\left.\left. W_{32}^{k_1(8n_2+4n_3+n_4)}\right] W_{32}^{16k_2n_2}\right) W_{32}^{2k_2(4n_3+n_4)}\right] W_{32}^{16k_3n_3}\Big) W_{32}^{4k_3n_4}\Big] W_{32}^{8k_4n_4} \end{aligned} \tag{10}$$

총 버터플라이 stage 수는 4이며, 160개의 복소 덧셈기와 48개의 복소 곱셈기가 필요하다.

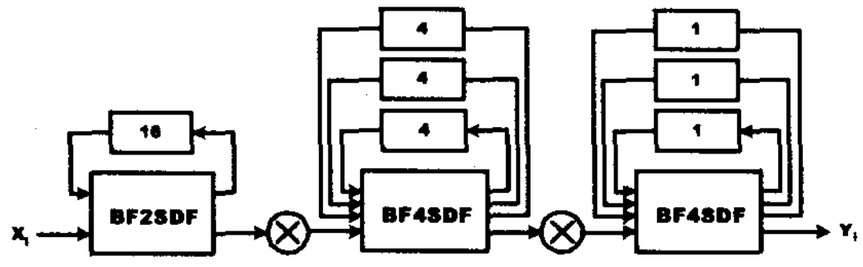
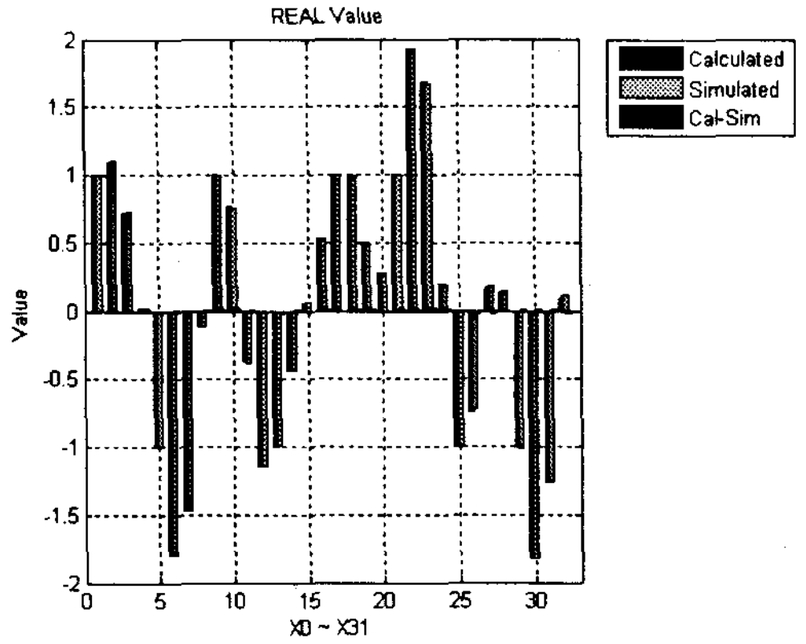
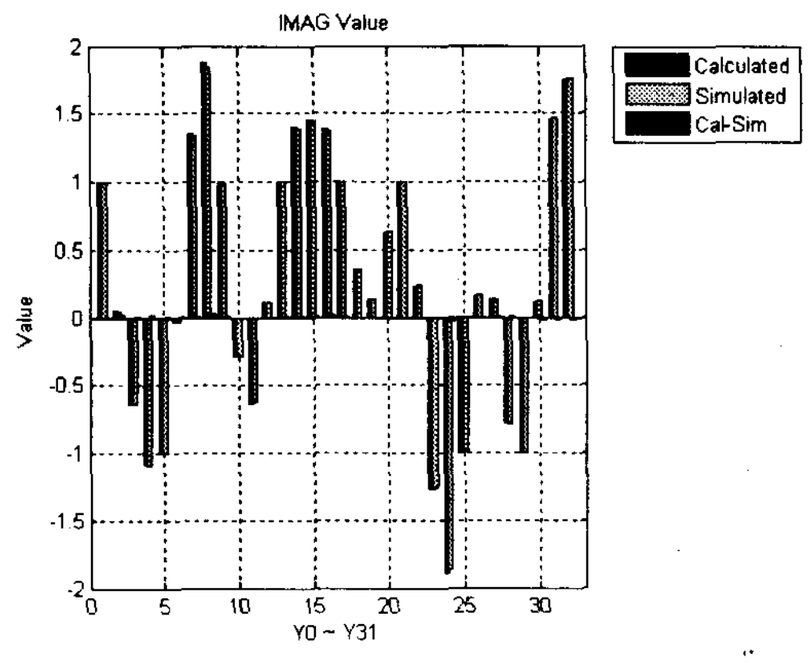


그림 3. 32-point FFT R2-R4-R4 SDF의 block diagram

Fig 3. block diagram of 32-point FFT R2-R4-R4 SDF structuire



(a) 실수 결과 값 비교



(b) 허수 결과 값 비교

그림 4. Matlab을 이용한 결과 값 비교

Fig 4. comparison between results using Matlab

R2-R4-R4 알고리즘은 32-point FFT를 radix-2와 radix-4를 이용하여 3개의 stage로 나타낸 것이다. 이 알고리즘은 DFT의 기본식 (1)을 식 (11)과 같이 나타낼 수 있다.

$$X[k_1+2k_2+8k_3] = \sum_{n_3=0}^{3}\left[\left(\sum_{n_2=0}^{3}\left[\left(\sum_{n_1=0}^{1} x[16n_1+4n_2+n_3]\,W_{32}^{16k_1n_1}\right)W_{32}^{k_1(4n_2+n_3)}\right]W_{32}^{8k_2n_2}\right) W_{32}^{2k_2n_3}\right]W_{32}^{8k_3n_3} \quad (11)$$

총 버터플라이 stage 수는 4이며, 160개의 복소 덧셈기와 40개의 복소 곱셈기가 필요하다.

각 알고리즘에 대해 산출한 덧셈기수와 곱셈기수를 비교하면, 32-point FFT는 알고리즘에 관계없이 덧셈기의 수는 160개로 일정하지만 곱셈기의 수는 알고리즘 마다 다르다. radix-4를 마지막 stage에 사용하는 것이 곱셈기를 줄이는 가장 효율적인 방법이며, R2-R4-R4 알고리즘이 가장 적은 곱셈기를 사용하는 것을 알 수 있다. 따라서 저전력 32-point FFT 프로세서의 구현을 위해서는 곱셈기를 가장 적게 사용하는 R2-R4-R4 알고리즘이 적합하다.

3.2 제안된 FFT 프로세서의 구현 및 결과

파이프라인 방식 중 제어가 용이하고 메모리를 적게 사용함으로써 저전력으로 동작이 가능한 SDF (Single-path Delay Feedback)를 이용하여 본 논문에서 제안한 32-point FFT R2-R4-R4 알고리즘을 구현하였다 [7, 8]. 32-point FFT R2-R4-R4 알고리즘을 SDF 구조로 구현할 경우 전체적인 block diagram은 그림 3과 같다, 알고리즘은 ModelSim XE Ⅲ를 이용하여 VHDL (VHSIC hardware description language)로 설계하였고, Spartan3 FPGA (field-programmable gate array) board의 출력값과 Matlab을 이용해 계산된 이론값을 비교 검증하였다.

4. 결 론

대용량 고속 데이터 통신을 지향하는 차세대 무선통신으로 OFDM 통신방식이 주목받고 있는 가운데 육상통신뿐만 아니라 해상통신에서도 적용을 추진 중이다. OFDM 통신방식이 단파대 해상 통신에 적용될 경우 ITU에서는 32-point FFT 프로세서의 사용을 권고하였다. 또한 해상 통신에서는 해양사고 및 조난 시에도 반드시 통신이 이루어져야 하므로 OFDM 통신시스템의 중요 디바이스인 FFT는 저전력으로 동작하여야 한다.

본 논문에서는 radix-2와 radix-4를 조합한 저전력 32-point FFT R2-R4-R4 알고리즘을 제안하였고, 이를 기반으로 파이프라인 구조 32-point FFT R2-R4-R4 SDF 방식을 VHDL로 설계하여 알고리즘을 검증하였다. 제안된 알고리즘 R2-R2-R2-R2-R2, R2-R4-R2-R2, R2-R2-R2-R4 그리고 R2-R4-R4 중에서 덧셈기의 수는 모두 같지만 R2-R4-R4의 곱셈기 수가 가장 적어 제안된 알고리즘 중에서 가장 최적화되어, 하드웨어의 크기를 줄이고, 전력 소비도 줄여 저전력화 하였다. VHDL로 구현된 파이프라인 구조 32-point FFT R2-R4-R4 SDF 방식의 출력값은 Matlab의 이론값과 일치하였다.

FFT는 OFDM 통신시스템의 중요 디바이스이므로 제안된 32-point FFT 알고리즘은 단파대 해상통신에서 OFDM을 적용한 저전력 데이터 통신 위한 선도 기술로 유용할 것이다.

참 고 문 헌

[1] 정연호,"고속 다중 사용자 데이터 전송 환경에서 고유의 펄스 형성화 기술을 적용한 적응 OFDM 시스템의 개발에 관한 연구", 한국과학재단, 2004.5.
[2] 오정렬, "저 면적 및 저 전력 복소 곱셈기를 갖는 파이프라인 방식의 FFT 프로세서 설계에 관한 연구", 박사학위논문, 전북대학교 대학원, 2005.8.
[3] 김정년, "디지털 漁業通信을 위한 SSB 모뎀 開發에 關한 硏究", 박사학위논문, 목포해양대학교 대학원, 2007.6
[4] ITU "Document 8/161-E", 2006.9.
[5] 장영범 "DSP 이론과 실무", 생능출판사 2004. 3.
[6] J. W. Cooley, J. W. Tukey, "An Algorithm for the Machine Calculation of Complex Fouier Series", Math. Comp, Vol.19, pp.297-301, 1965.4.
[7] E. H. Wold, A. M. Despain, "Pipeline and Parallel Pipeline FFT Processors for VLSI Implementation", IEEE Trans. Comput, C-33(5), pp.414-426, May 1984.
[8] S. He and M. Torkelson, "Designing pipeline FFT processor for OFDM (de)Modulation",in Proc. IEEE URSI Int. Symp. Signals, Syst., Electron., pp.257-262, 1998.