파이프라인 FFT 프로세서 설계을 위한 하드웨어 구조 분석

정성완, 정용진
광운대학교 전자통신공학과
e-mail : overjsw@hotmail.com , yjjeong@kw.ac.kr

Comprehensive Analysis of Hardware Architectures of Pipeline FFT Processor

Sungwan Jung and Yong-Jin Jeong
Department of Electronics and Communications Engineering
Kwangwoon University

Abstract

FFT(Fast Fourier Transform)는 멀티미디어 통신 및 디지털 신호처리 분야, 특히 무선통신이나 디지털 방송 등에서 쓰이는 OFDM(Orthogonal Frequency Division Multiplexing)에서 필수적인 역할을 하고 있다. 본 논문에서는 파이프라인 FFT 프로세서 설계의 다양한 알고리즘 및 하드웨어 구조에 대해 살펴보고 이를 한 눈에 파악할 수 있는 설계 가이드라인을 제시한다. 또한 분석 중 Radix-2 Single-path Delay Feedback의 복소곱셈기의 비효율적인 면을 찾고 새로운 R2SDF 구조를 제안한다.

I. 서론

FFT는 DFT(Discrete Fourier Transform)를 보다 작은 DFT로 연속해서 분해하여 주기적으로 같은 값을 갖는 회전인자(Twiddle Factor)의 성질을 이용하여 N^2번의 곱셈을 NlogN번의 곱셈으로 줄여주는 효율적 알고리즘이다. 현재까지 Radix-2, Radix-4, Radix-8, Radix-16, Radix-2^2, Radix-2^3, Radix-2^4, Radix-2/4, Radix-2/8, Radix-2/4/8등 다양한 FFT 알고리즘이 제안되어 왔다. 이 알고리즘들은 각각이 갖는 특징이 다르며 하드웨어 설계 구조에 따라 또 다른 특성들을 갖는다. FFT Processor 설계 방식에는 크게 Single Butterfly (이하 BF) 방식과 Pipeline 방식, 병렬처리 방식 3가지가 있지만, 본 논문에서는 하드웨어 설계 시 가장 효율적인 Pipeline FFT의 다양한 구조에 대해 분석한다.

II. FFT Algorithms

식(1) DFT의 연산을 줄이는 핵심요소는 바로 신호의 백터 성분 중 곱셈을 유발시키는 회전인자(Twiddle Factor)의 특성에 있다. 이 회전인자는 식 (2)같이 코싸인 성분과 싸인 성분, 싸인 성분에는 허수($j=\sqrt{-1}$)가 있다. FFT는 그림1과 같이 이 Twiddle Factor의 삼각 함수와 허수의 특성을 이용하여 그 연산량을 줄인다.

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi j}{N}nk} \qquad k = 0, 1, ..., N-1 \qquad (1)$$

$$W_N^{nk} = e^{-\frac{2\pi j}{N}nk} = \cos(\frac{2\pi nk}{N}) - j\sin(\frac{2\pi nk}{N}) \qquad (2)$$

본 논문은 서울시 나노 SoC 클러스터 사업단의 지원으로 이루어졌습니다.

1. Radix-2

Radix-2는 Twiddle Factor가 실수축에 걸릴 때 즉, $W_N^{kN/2} = e^{-\frac{j2\pi k}{2}} = \cos(\frac{2\pi k}{2}) - j\sin(\frac{2\pi k}{2})$ 일 때 계수가 1 또는 -1이 되는 성질을 이용한다.

2. Radix-4

Radix-4는 Radix-2의 이점을 취하면서 Twiddle Factor가 허수축에 걸리는 $W_N^{kN/4}$일 때 j 또는 -j가 되는 점을 이용하여 Radix-2보다 더 적은 연산량으로 DFT를 처리한다. 계수가 j나 -j 값을 갖을 때 식(3)과 같이 실수부를 허수부로 허수부를 실수부로 바꾸어 주고 -1만 곱해주면 된다.

$$x(n) * j = (a+jb) * j = ja - b \quad \text{,since } j*j = -1 \qquad (3)$$

3. Radix-8

Radix-8 역시 Radix-2, 4의 이점을 동시에 취하면서 Twiddle Factor가 실수와 허수 중간축에 걸리는 $W_N^{kN/8}$일 때 $\pm(\frac{\sqrt{2}}{2} - j\frac{\sqrt{2}}{2})$와 $\pm(\frac{\sqrt{2}}{2} + j\frac{\sqrt{2}}{2})$이 되고, 이는 복소수 곱셈 대신 덧셈기 4개만으로 처리가 가능하다. [1]

III. Architectures for the pipeline FFT processor design

Pipeline FFT Processor는 Data-path 방식에 따라 Single-path와 Multi-path로 나뉘며, Data Buffering 방식에 따라 Delay Feedback과 Delay Commutator로 나뉜다.

1. MDC(Multi-path Delay Commutator)

Radix-r개의 데이터 패스를 두고 각 Stage사이에 데이터의 지연을 담당하는 Commutator를 두는 방식. 상대적으로 연산이 빠른 반면, 낮은 효율의 많은 하드웨어 자원을 필요로 한다.

2. SDF(Single-path Delay Feedback)

BF 연산을 위해 데이터간 떨어진 거리만큼 지연소자를 두어 기다린 후 해당 데이터가 입력될 때 BF연산을 하고 복소수곱셈이 필요한 데이터는 다시 지연소자로 Feedback하여 메모리와 곱셈기의 효율을 높인 구조이다. 상대적으로 느리지만

적은 하드웨어 비용만으로도 구현이 가능하다.

3. SDC(Single-path Delay Commutator)

SDC 구조는 BF를 한번에 하나의 데이터만 출력시키도록 간소화하여 BF의 효율성을 100%까지 끌어올린 구조이다. BF가 간단해지고 System Frequency가 올라간다는 장점이 있지만 지연소자를 많이 필요로 한다. 다소 복잡한 구조와 늘어나는 메모리 때문에 Radix-4에만 적용되어진다.

IV. Comparison and Analysis

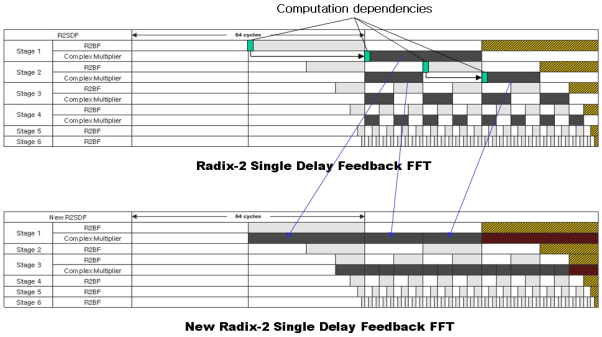
1. New R2SDF

앞서 언급한 알고리즘 및 구조로 지금까지 다양한 아키텍처들이 제안되어 왔으며, 이러한 구조들은 FFT에서 가장 연산복잡도가 높은 회전인자의 곱, 즉 복소곱셈량과 그 곱셈기의 효율을 얼마나 높이는가에 중점을 두고 있다. 하나의 복소곱셈기는 실수 곱셈기 4개, 실수 덧셈기 2개, 약 16,921 게이트로 이루어져 있어 FFT Processor의 대부분을 차지하므로 이 복소곱셈기의 개수를 줄이는 것은 곧 FFT Processor의 하드웨어 비용을 줄이는 것을 의미하기 때문이다.

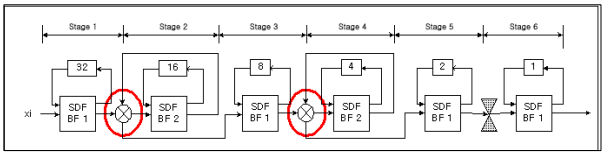
	Hardware Requirement			Performance	Hardware Utilization		
	Complex Multipliers	Complex Adders	Memory	Cycles	Complex Multiplier	BF	Memory
R2SDF	log2N-1	4log4N	N-1	N	50%	50%	100%
R2MDC	log2N-1	4log4N	3N/2-2	N/2	50%	50%	50%
R2^2SDF	log4N-1	4log4N	N-1	N	75%	50%	100%
R2^2MDC	2(log4N-1)	4log4N	3N/2-2	N/2	37.5%	50%	50%
R2^3SDF	log8N-1	(6+2T)log8N-1	N-1	N	87.5%	50%	100%
R2^3MDC	2(log8N-1)	(6+2T)log8N-1	3N/2-2	N/2	43.75%	50%	50%
R4SDF	log4N-1	8log4N	N-1	N	75%	25%	100%
R4MDC	3(log4N-1)	8log4N	5N/2-4	N/4	25%	25%	25%
R4SDC	log4N-1	3log4N	2N-2	N	75%	100%	100%
R8SDF	log8N-1	(24+2T)log8N	N-1	N	87.5%	12.5%	100%
R8MDC	7(log8N-1)	(24+2T)log8N	9N/2-8	N/8	12.5%	12.5%	12.5%

표 1. Hardware requirement and utilization comparison for N-point FFT architectures

표 1에서와 같이 High-radix알고리즘을 이용한 구조는 필요한 복소곱셈기의 개수를 줄일 수 있지만 데이터 콘트롤과 BF복잡도가 올라가면서 구현이 어려운 반면, Low-radix알고리즘을 이용한 구조는 구현이 단순하지만 상대적으로 많은 복소곱셈기를 필요로 하게 된다. 본 논문에서는 각 아키텍처를 분석하면서 간단한 구조를 갖고 있는 R2SDF의 50% 효율을 갖고 있는 복소곱셈기의 연산이 연산 의존도에 의해 미리 연산되어질 수 있고 이로 인해 다음 스테이지에서 일어날 복소곱셈 연산의 타이밍과 일치하게 되어 그림 1과 같이 복소곱셈기를 반으로 줄이고 그 효율성을 100%로 올릴 수 있다는 것을 알게 되었다.



(a) Timing table of process elements of New R2SDF



(b) 64-point New R2SDF Architecture

그림 1. New R2SDF

2. Design guideline for Pipeline FFT Processor

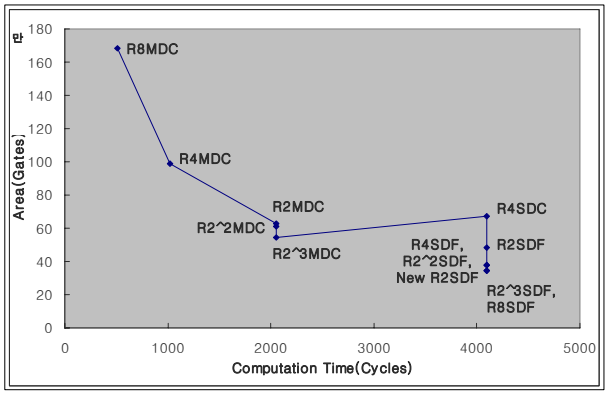


그림 2. Hardware cost in proportion to computation time (측정된 Area는 각 구조의 16-bit 복소곱셈기와 지연소자만 고려하여 나타낸 것임)

그림 2는 4096 point의 다양한 Pipeline FFT Processor 구조들을 한눈에 보여줌으로서 설계의 가이드라인을 제시한다. 상대적으로 연산이 빠른 구조인 R8MDC, R4MDC등은 많은 하드웨어 자원을 요구하지만 MIMO OFDM (Multiple-Input Multiple-Output Orthogonal Frequency Division Multiplexing) 시스템에서 그 효용가치가 높다고 알려져 있다. [2] 한편, 속도는 느리지만 가장 적은 하드웨어 자원을 요구하는 구조는 R8SDF이다. 이 경우 설계의 복잡성은 동일한 알고리즘이지만 구조를 단순화 한 R2^3SDF를 설계함으로서 해결할 수 있다. [1] 위 그래프는 몇 point를 대상으로도 비슷한 양상을 보이며 각 점들간의 거리는 포인트가 증가할수록 비례적으로 멀어진다. (단, R4SDC의 경우 1024 points이하에서는 R2SDF보다 더 적은 하드웨어 자원을 필요로 한다.)

V. 결론 및 향후 과제

FFT 알고리즘과 아키텍처들을 결합된 상태로 비교하여 Pipeline FFT Processor 설계 시 그 가이드라인이 될 수 있도록 하였다. 하지만 FFT Processor는 단일 알고리즘 또는 단일 구조가 아닌 복합 구조로도 설계가 가능하기 때문에 다각적 분석이 필요하다. 향후 요즘 가장 많이 쓰이는 4k, 8k point를 대상으로 여러 가지 구현사례를 분석해보고 상황에 따른 최적화된 구조를 찾을 수 있는 가이드라인을 제시하려 한다.

참고 문헌

[1] Lihong Jia, Yonghong Gao and Hannu Tenhunen, A New VLSI-Oriented FFT Algorithm and Implementation, Electronic System Design Laboratory, Royal Institute of Technology, Stockholm, Sweden
[2] Sangmin Lee, Yunho Jung, Jaeseok Kim, Efficient pipelined FFT processor for the MIMO-OFDM systems