FPGA 를 사용한 radix-2 16-points FFT 알고리즘 가속기 구현

이규섭 [1], 조성민 [2], 서승현 [3]
[1] 한양대학교 전자공학과 석사과정
[2] 한양대학교 전자공학과 박사과정
[3] 한양대학교에리카 전자공학부 교수

r22jiw0n@hanyang.ac.kr, smcho3315@ hanyang.ac.kr, seosh77@ hanyang.ac.kr

Radix-2 16-points FFT accelerator implementation using FPGA

Gyu Sup Lee[1], Seong-Min Cho[2], Seung-Hyun Seo[3]
[1, 2]Dept. of Electrical Engineering, Hanyang University
[3]School of Electrical Engineering, Hanyang University ERICA

요 약

본 논문에서는 FPGA 를 활용하여 radix-2 Fast Fourier Transform(FFT) 알고리즘을 빠르고 효율적으로 구현하는 연구에 대해 기술한다. 본 논문에서 zybo z7-20 FPGA 를 사용하여 Processing System(PS)에서만 동작하는 구현과 Programmable Logic(PL)에서 동작하며 파이프라인과 병렬처리를 사용한 FFT 구현 결과를 비교한다. 또한 유사한 논문과의 결과 비교를 통해 본 구현 방법의 연산 시간 및 리소스 사용의 효율성을 분석한다.

1. 서론

시간의 연속적인 신호를 주파수 영역으로 변환하는 Fast Fourier Transform(FFT)은 과학, 공학, 의학 등 다양한 분야에서 널리 사용되는 중요한 알고리즘이다. 특히, 신호처리, 주파수 특성 분석, 암호학 등의 분야에서 FFT 연산은 필수적인 역할을 수행한다. 최근에는 FFT 의 변형인 Number Theoretic Transform(NTT) 연산이 Post-Quantum Cryptography(PQC)의 알고리즘 중 격자기반 알고리즘에 주로 사용되면서 FFT 와 관련된 연구가 활발하게 이루어지고 있다.

이러한 FFT 알고리즘을 빠르고 효율적으로 구현하는 방법 중 하나로 Field Programmable Gate Array(FPGA)를 사용하여 알고리즘을 가속화하는 기술이 각광받아왔다. FPGA 는 고성능 연산 처리, 높은 병렬처리 능력 등의 다양한 장점을 갖고 있어, 임베디드 및 IoT 환경에 적합한 하드웨어이다. 또한 FPGA 를 활용한 가속화 기법은 FFT 뿐만 아니라 다양한 알고리즘에 적용이 가능하여 연구의 중요성이 높다.

본 논문에서는 Zybo Z7-20 FPGA 를 사용하여 16-point radix-2 FFT 를 구현하고, Processing System(PS)에서만 작동하는 구현과 Programmable Logic(PL)에서 동작하면서 파이프라인 및 병렬처리를 사용한 구현결과를 비교한다. 또한 유사한 논문과의 비교를 통해 본 구현 방법의 연산 시간 및 리소스 사용의 효율성을 분석한다. 이를 통해 FPGA 의 장점을 활용한 FFT 의 구현이 얼마나 빠르고 리소스를 효율적으로 사용할 수 있는지에 대해 기술한다.

2. 배경 지식

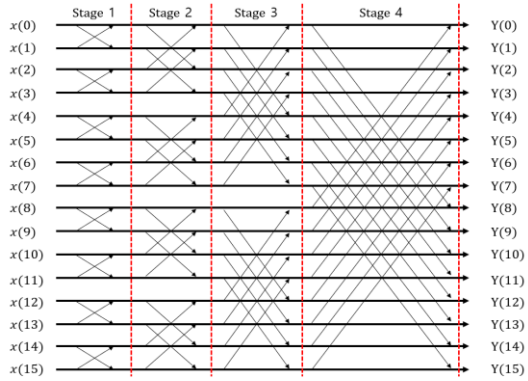
2.1 Radix-2 Fast Fourier Transform

FFT 는 시간 영역의 연속적인 신호를 주파수 영역으로 변환하여, 신호의 주파수 성분을 분석할 수 있는 알고리즘이다. 이 변환은 특히 주기적인 패턴이나 주파수 영역에서 표현되는 특성을 갖는 신호를 분석하는데 유용하다.

FFT 는 Cooley-Tukey 알고리즘을 기반으로 하며, 이는 기존의 Discrete Fourier Transform(DFT) 알고리즘을 개선한 것으로 데이터를 절반씩 나눠서 처리하며 계산 복잡도를 $O(N^2)$에서 $O(N \log N)$으로 줄였다. Radix-2 FFT 는 잘 알려진 FFT 알고리즘 중 하나로 2 의 거

듭제곱 크기의 input 데이터를 다루는 것이 특징이며, 본 논문에서는 16(2^4)개의 데이터 포인트를 갖는 radix-2 FFT 를 구현했다.

FFT 에서 핵심적으로 사용되는 butterfly 연산은 (그림 1)과 같이 두 입력 데이터의 덧셈과 회전자(twiddle factor)를 곱한 차를 출력한다.



(그림 1) 16-point FFT 의 butterfly 연산 과정

2.2 Field Programmable Gate Array

FPGA 는 사용자가 프로그래밍할 수 있는 집적회로로, 변경이 가능한 논리 게이트와 다양한 역할을 하는 블록들을 연결하는 interconnect 로 구성이 되어있어서 사용자가 특정한 기능이나 응용프로그램을 구현할 수 있게 한다.

FPGA 는 하드웨어의 구성을 통해 고성능 연산 처리를 가능하게 하여, 병렬 처리를 적용할 수 있는 다양한 알고리즘에 활용할 수 있다.

FPGA 는 ASIC(Application Specific Integrated Circuit)과 비교했을 때, 높은 유연성과 낮은 개발 비용 등의 장점을 갖고 있다. 이로 인해 FPGA 는 회로의 프로토타입, 학술 연구, 고성능 컴퓨팅 등 다양한 분야에서 널리 사용되고 있으며 낮은 전력 소모를 바탕으로 임베디드 시스템이나 IoT 환경에서 적합한 하드웨어로 사용되고 있다.

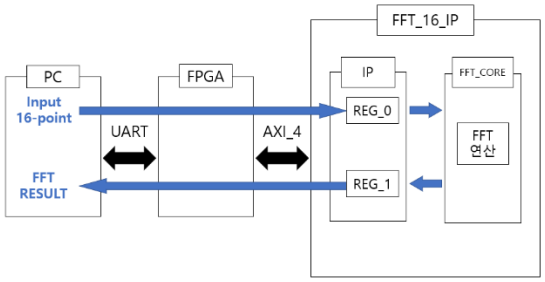
3. FPGA 구현

3.1 구현 방법

실험에서는 Digilent 社의 zynq-7000 시리즈 중 zybo z7-20(XC7Z020-1CLG400C) FPGA 를 사용했다. 개발환경은 Xilinx 社의 Vivado 2020.2 이며 PS 및 PL 에 필요한 어플리케이션 코딩은 Vitis2020.2 에서 진행했다.

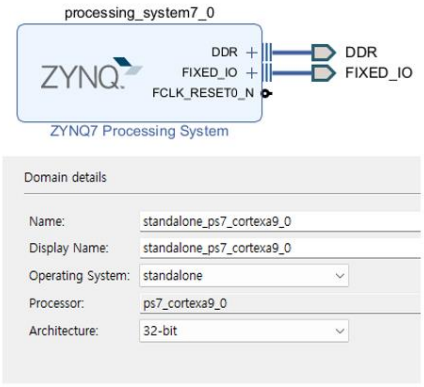
전체적인 구현은 (그림 2)와 같다. PC 에서 input 으로 사용되는 16 개의 point 를 입력하면, UART 통신을 통해 FPGA 보드로 전달되며, 보드 내에서 AXI4_LITE 인터페이스를 통해 FFT 연산이 이뤄진다. 이때 연산의 결과값은 다시 레지스터에 저장되고, PC 로 결과값이 전달된다.

FFT_CORE 의 설계는 16 개의 input 에 대하여 4 개의 stage 에 따라 butterfly 연산이 8 번씩 진행되며, butterfly 연산은 모듈화를 통해 연산이 진행될 때마다 호출된다. 또한 연산과정에서 반복적으로 사용되는 butterfly 연산에 대해 반복문이 아닌 unrolled 구조로 구현했으며, 각 stage 별 연산 결과를 저장하여 pipeline 구조를 활용했다. 또한 각 연산에 필요한 twiddle factor 를 미리 계산하여 9 비트 파라미터로 저장함으로써 복잡한 계산을 최소화했다.



(그림 2) FFT 연산의 FPGA 전체 구현 흐름도

Logic 을 사용하지 않고 PS 에서만 FFT 연산을 수행하는 C 코드의 구현은 (그림 3)과 같이 Zynq 에서 제공하는 프로세서를 사용했다. PL 에서 구현한 FFT 와 동일한 기능을 구현하였으며, FPGA 에 내장된 667MHz dual-core Cortex-A9 을 사용하여 연산을 수행한다.



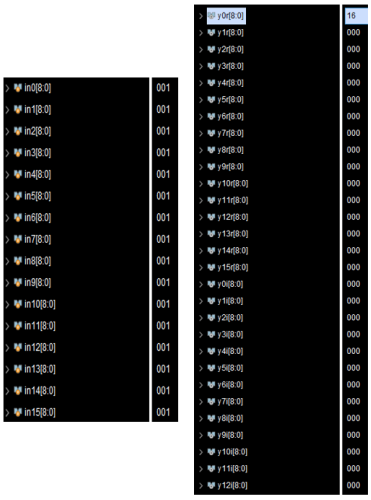
(그림 3) PS 연산의 프로세서

3.2 결과

(그림 4)는 FFT 연산 모듈의 구현 정확성을 확인하는 테스트 벤치의 결과이다. input 으로 [1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1]인 값을 보냈을 때, output 으로 [16,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,]이 정상적으로 출력됨을 확인할 수 있다. <표 1>은 구현한 FFT_CORE 모듈을 ZYNQ 프로세서 모듈과 결합하여 합성 및 구현을 진행할 때 사용된 resource 를 보여준다.

<표 3> 유사 연구 비교 표

Method	FPGA	Radix	N	No. LUT	No. FF	No. banded IOB	No. slice
Proposed	Zybo z7-20	r-2	16	497	802	130	227
[1]	Spartan6	r-4	8	296	112	352	-
[2]	Zynq 7z020clg484-1	r-4	16	1039	836	-	-
[3]	Spartan6	r-2	16	1004	557	295	-
[4]	XC5VLX330-2FF1760	r-4	16	1894	3469	-	-
[5]	(VHDL language)	r-2	16	(fully used LUT-FF pairs) 295		51	2401



(그림 4) FFT_CORE 모듈의 테스트 벤치 결과

<표 1> 사용된 resource 및 utilization

Resource	Utilization	Available	Utilization %
LUT	497	53200	0.93
LUTRAM	60	17400	0.34
FF	802	106400	0.75
BUFG	1	32	3.13

연산과정에서 소요된 시간은 <표 2>와 같다. PS 에서의 연산시간이 PL 에서의 연산시간보다 평균적으로 약 3 us 길었다. 또한 소요 clock cycle 은 약 1000 cycle 정도의 차이가 발생했다.

<표 2> PS 와 PL 의 연산 시간

평균	PS	PL
연산시간 (us)	21	18
Clock cycle	14,236	13,396

4. 비교 분석 및 결론

4.1 비교 분석

<표 3>은 본 논문에서 연구한 구현 방식과 유사한 연구들을 비교한 표이다. 비교 과정에서 각 연구마다 사용한 FPGA 보드, FFT 기법, 성능 등이 달라서 정확한 비교는 어렵지만, 전반적으로 본 연구에서 제안한 구현 방식이 리소스 사용에 있어서 효율적임을 보인다.

4.2 결론

본 논문에서는 FPGA 를 활용하여 radix-2 FFT 알고리즘을 빠르고 효율적으로 구현하였다. <표 2>에서 볼 수 있듯이, 본 연구에서 제안한 구현이 효율적으로 작동함을 확인할 수 있다. 이는 FPGA 의 고성능 연산 처리, 높은 병렬처리 등의 장점을 활용하여 성능 향상을 이룬 것으로 해석할 수 있다. 또한 유사한 논문과의 결과 비교를 통해 본 연구의 구현 방식이 다른 연구들과 비교하여 리소스 사용에 있어서 효율성을 보이는 것을 확인할 수 있다. 이러한 결과는 본 논문에서 제시한 FPGA 기반의 FFT 구현 방식이 다양한 분야에서 빠른 연산 처리와 효율적인 리소스 사용이 요구되는 알고리즘에 적용될 수 있음을 시사한다.

본 연구를 바탕으로 향후 FPGA 를 활용한 다양한 알고리즘의 구현 및 최적화 방법에 대해 더 깊이 연구할 수 있으며, 그 결과로 다양한 분야에서 더 나은 성능과 효율성을 달성할 수 있을 것으로 기대된다.

Acknowledgement

본 연구는 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2021R1A2C1095591)

참고문헌

[1] V. Patil and T. M. Manu "Fpga implementation radix-2 dit fft using fixed point arithmetic and reduced arithmetic complexity" 2021 International Conference on Intelligent Technologies (CONIT), Karnataka, India, 2021.

[2] G. Akkad, et al. "Fft radix-2 and radix-4 fpga acceleration techniques using hls and hdl for digital communication systems" 2018 IEEE international multidisciplinary conference on engineering technology (IMCET). Beirut, Lebanon, 2018.

[3] L. Santhosh and A. Thomas "Implementation of radix 2 and radix 2^2 FFT algorithms on Spartan6 FPGA" 2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT). Tiruchengode, India, 2013.

[4] A. Mankar, AD. Das, and N. Prasad. "FPGA implementation of 16-point radix-4 complex FFT core using NEDA" 2013 Students Conference on Engineering and Systems (SCES). Allahabad, India, 2013.

[5] S. J. Saenz, et al. "FPGA design and implementation of radix-2 Fast Fourier Transform algorithm with 16 and 32 points" 2015 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC). Ixtapa, Mexico, 2015.