韓國 OA學會 論文誌
第7卷 第4號, 2002. 12.

2002-7-4-2-1

32비트 ALU 설계에 대한 연구

황 복 식* 이 영 훈**

A study on the design of a 32-bit ALU

Bok-sik Hwang* Young-hun Lee**

요 약

본 논문에서는 32비트 DSP에 사용가능한 ALU를 설계하였다. 이 ALU는 32비트 연산을 기본 단위로 하고 있으며 5단 파이프라인 중에서 execution 단계에 해당된다. ALU에서 지원하는 기능은 덧셈, 뺄셈, 나눗셈과 같은 산술연산, AND, XOR과 같은 논리연산, 그리고 쉬프트 등이다. 기능별로 여러 기능 블록을 사용하지 않는 대신 몇 개의 기능 블록만을 만들고, 회로 동작이 이 기능 블록들을 공유하도록 설계하였으며, ALU를 설계하기 위해 각 기능 블록을 HDL로 기술하여 시뮬레이션을 수행하였다. 이 ALU는 32 비트 DSP에 사용 가능하도록 설계되었다.

Abstract

This paper describes an ALU core which is suitable for 32-bit DSP. This ALU operates in 32-bit data and occupies the third stage, execution, among 5 stage pipeline strcture. The supplied functions of the ALU are arithmetic operations, logical operations, shifting, and so on. For the implementation of this ALU core, each fuctional block is described by HDL. And the functional verification of the ALU core is performed through HDL simulation. This ALU is designed to use the 32-bit DSP.

* 한남대학교 전자공학과 박사과정
** 한남대학교 전자공학과 교수

논문접수 : 2002. 9. 15
심사완료 : 2002. 11. 30

Ⅰ. 서론

전자·정보 기술이 발전함에 따라서 라디오, 텔레비젼, 컴퓨터 등의 문명의 이기들이 개발되어 사회생활에 큰 변화를 가져왔다. 특히 컴퓨터 성능의 급격한 발전은 음성, 영상신호의 처리가 동시에 이루어지는 멀티미디어 데이터 처리와 고품질의 통신 서비스를 위하여 기존의 아날로그 신호처리 방식이 디지털 신호처리 방식으로 전환되고 있으며 디지털 신호처리를 위한 다양한 알고리즘이 개발되고 있다.[1~2]

일반적인 디지털 신호처리 알고리즘은 데이터의 압축 및 복원, 동기화, 시각 및 청각화 등을 처리하기 위하여 많은 양의 승산과 가감산을 요구하므로 기존의 마이크로프로세서로 실시간 시스템을 구현하기에는 많은 어려움이 있었다. 이러한 문제점을 극복하기위해 디지털 신호처리의 수학적 계산을 고속으로 처리하는 DSP가 펌웨어 형태로 출현하게 되었다. DSP는 수학적 계산이 많은 복잡한 디지털 신호처리 알고리즘을 하드웨어로 쉽고 빠르게 구현할 수 있도록 내부에 곱셈기, barrel shifter, 함수 테이블 등의 특수한 기능블럭을 포함하고 있다. 또한 성능이 우수하고 작은 면적과 효율면에서 우수한 특성을 가지고 있어야 한다. 따라서 여러 가지 기능과 하드웨어를 부가시켜야 하므로 회로의 집적이 증가하고 복잡하게 되어 테스팅이 더욱 어려워지고 있다.[1~3]

따라서, 본 논문에서는 이러한 멀티미디어와 디지털 신호처리에 응용하기 위해 32비트 DSP 설계에 사용 가능하며 지연시간이 최소가 되고, 성능이 우수한 ALU(arithmetic Logic Unit)를 설계하고 시뮬레이션을 통하여 그 특성이 우수함을 입증하는데 목적이 있다.

Ⅱ. ALU의 설계

1. DSP 기능 블록

DSP 프로세서는 디지털 신호 처리에서 많은 양의 데이터를 처리하거나 FFT와 같은 특수한 알고리즘을 빠르게 처리하기 위한 특징을 가지고 있다.

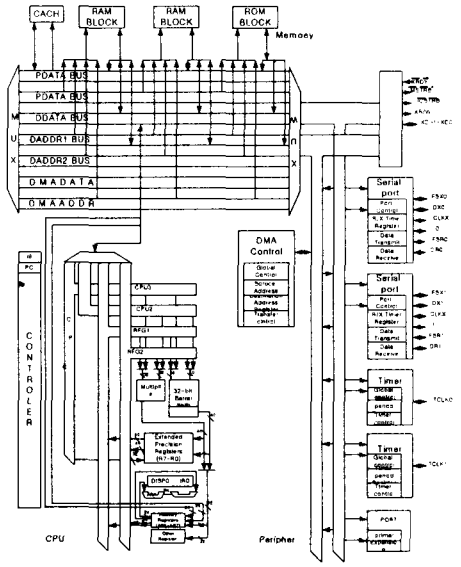

그림1. DSP 블럭도

DSP 블럭도를 살펴보면 5단계의 파이프라인으로 즉, fetch, decode, execution, memory, write back으로 구성되어 있으며 매 사이클마다 하나의 명령어를 수행한다. [4~6]

2. 논리연산 구조 설계

그림2는 ALU의 블록도를 나타낸 것이다. 본 논문에서는 ALU는 연산을 위하여 ALU에 입력되는 두 개의 데이터의 자리수를 맞추는 기능과 산술연산의 데이터 출력에 대하여 normalization 기능을 수행하는 barrel shifter 블럭들과 데이터 선택을 위한 MUX로 구성하였다. 또한, 본 논문에서는 논리연산 및 산술 연산을 위한 가산기는 면적의 오버헤드가 있지만 속도가 빠른 CLA(Carry-Lock-Ahead) 구조를 사용하였다. 논리연산을 위한 논리회로는 별도로 구현하지 않고 가산기 회로에서 캐리를 발생하는 부분과 캐리를 전파하는 부분을 이용하여 AND, OR 그리고, XOR 연산을 수행하도록 그림3과 같이 설계하였다.

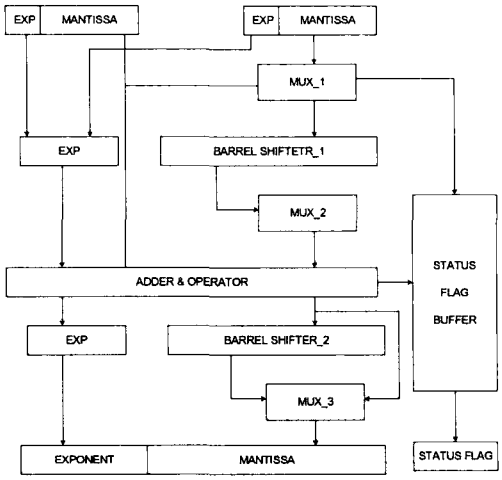

그림2. ALU의 블록도

그림 3의 회로에서 덧셈기는 CLA 구조로 설계하였으며, 이것은 캐리와 두 개의 연산자가 입력으로 필요로 하게 된다. 뺄셈 기능을 수행할때는 피연산자를 1의 보수로 만들고 캐리 입력을 1로 하여 덧셈을 수행하도록 하였다.

3. 배럴 쉬프터 설계

본 논문에서는 그림4의 블록과 같은 barrel shift를 설계하였다. 설계된 배럴쉬프터는 회전/평이동 어레이(Rotate/Shift Array)와 제어 논리부로 이루어져 있다.

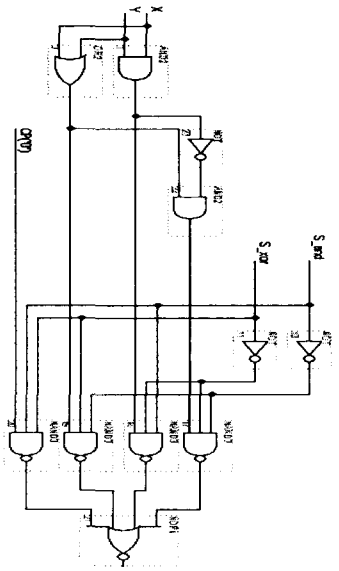

그림3. 논리 연산 회로

회전/평이동 어레이는 1비트 회전 어레이, 바이트 회전/평이동 어레이(Byte Rotate/Shift Array)와 비트 회전/평이동 어레이(Bit Rotate/Shift Array)로 이루어져 있고, 제어 논리부는 2×4 디코더, 3×8 디코더, 보간 회로등으로 구성되어 있다. 회전/평이동 어레이와 바이트 평이동 어레이는 기본적으로 32개의 4×1 MUX가 병렬로 구성되어 있으며, 바이트 평이동 어레이에서는 1칸을 회전/평이동하면 실제로는 1바이트 단위로 회전/평이동하는 것이다. 비트 평이동 어레이(0-7 bit rotate/shift array)는 32개의 8×1 mux가 병렬로 구성되어 있으며 비트 평이동 어레이는 1비트 단위로 회전/평이동 하도록 설계하였다.

회로의 면적과 게이트 수를 줄이기 위하여 본 논문에서 설계한 배럴 쉬프터의 회전/평이동 어레이는 왼쪽방향으로만 회전/평이동하도록 설계하였다.

왼쪽방향으로의 어레이에서 오른쪽 방향으로의 회전/평이동 명령을 정확히 수행하기 위하여 회전/평이동 어레이에 2×1 MUX를 기본 셀로 하는 1비트 회전 어레이를 추가하였다.

왼쪽 방향으로의 어레이에서 오른쪽으로 회전/평이동

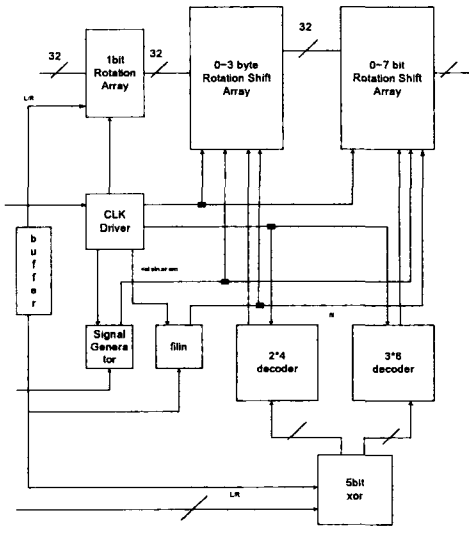

그림4. 배럴 쉬프터 블럭도

하는 방법은 어떤 데이터가 k 만큼 오른쪽 방향으로 회전을 수행한 결과는 32-k 만큼 왼쪽으로 회전을 한 것과 같은 결과를 얻을 수 있다. 여기서 32-k는 k의 2의 보수이며, 그값은 "k의 1의 보수 +1"의 값과 동일하다. 1의

보수를 취하는 방법은 간단하게 평이동 구간의 각각의 비트를 인버팅하므로써 얻을 수 있고, 더하기 1을 해준다는 것은 왼쪽으로 1비트 더 회전되는 것이므로 1비트 회전 어레이를 이용하였다.

배럴 쉬프터의 동작원리는 5비트 중에서 상위 2비트를 이용하여 최소 0에서 최대 3바이트까지 회전 평이동을 할 수 있으며, 하위 3비트를 가지고, 최소 0에서 최대 7비트까지 회전/평이동을 할 수 있고, 바이트 평이동 어레이와 비트 평이동 어레이 2단을 구성하였다. 2단을 이용하면 최소 0에서 최대 31비트까지 모든 구간을 회전/평이동 할 수 있으며, 64비트로 확장하는 경우에도 어레이 단을 추가하지 않고 바이트 어레이 셀을 8×1 MUX로의 교체만으로 확장이 가능하도록 설계하였다.

배럴 쉬프터의 명령어는 다음과 같은 내용으로 구성하였다. 여기서, L/R은 좌/우 방향을 나타내는 비트이고,

L/R	R/S	L/A	L_4	L_3	L_2	L_1	L_0

R/S는 회전/평이동 명령 형태를 나타내며, L/A는 평이동 명령을 수행할 때 논리/ 연산인지를 나타내주는 비트이다. 마지막으로 L4~0는 얼마만큼 회전/평이동할 것인지를 나타낸다. 예를 들어, 왼쪽으로 8비트 연산 평이동 동작을 수행하려면 명령어는 01101000과 같이 된다.

4. Leading Zero 검사 회로 설계

Leading zero 검사는 오퍼랜드 값에서 몇 번째 비트에 '0'이 아닌 값이 나오는지 그 특정 비트를 알려주며, 위와 같은 연산을 수행하기 위해서는 2 사이클이 필요로 하게 된다. 1사이클에서는 오퍼랜드의 각 4비트 단위로 8개의 그룹으로 NOR 연산을 수행하여 8개의 출력을 만들고 leading zero 발생기 부분에서는 8개의 값을 이용하여 어느 그룹까지 '0'으로 되어 있는지를 찾아 T[4:2]의 값에 할당한다. 여기서 오퍼랜드의 값이 모두 0이면 T[5:]의 값을 100000으로 세트 한다. 그 다음 사이클 때 앞서 설정된 값 T[4:2]을 이용하여 쉬프터에서 1단계 쉬프트를 수행하고, 쉬프트한 결과를 이용하여 leading zero 발생기에서 T[1:0]의 값을 세팅하기 위해서 결과의 최상위 3개의 비트를 검사하여 T[1:0]를 만들고 leading zero 검사기의 최종 출력 T[5:0]은 ALU의 블록인 논리 연산 결과와 멀티플렉싱되어 출력된다. 예를 들어 XR[31:0]=000...000일 때 T[5]=1이 되고

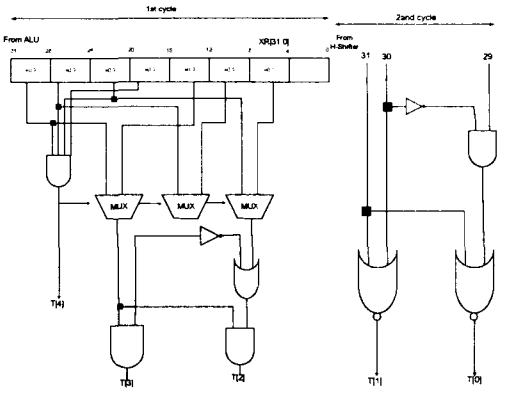

그림5. Leading zero 수의 발생

T[4:0]=00000이된다. T[1:0]의 경우는 leading zero 수를 발생기에서 구해진 T[4:2]를 다음 사이클에 쉬프터의 1단계에서 쉬프트 양으로 사용하게 된다. 이렇게 쉬프트된 결과에서 T[1:0]를 구하게 된다. 그리고, 이러한 방법으로 구해진 T[5:0]의 값을 출력하게 된다.

Ⅲ. 컴퓨터 시뮬레이션

본 논문에서는 논리연산을 위한 논리회로는 따로 구현하지 않고 가산기 회로에서 캐리를 발생하는 부분과 캐리를 전파하는 부분을 이용하여 AND, OR 그리고, XOR 연산을 수행하도록 설계하였으며 altera사의 maxplus을 이용하여 그림6과 같이 시뮬레이션 수행하였다. 덧셈기는 그림6에서 보여주는 바와 같이 2개의 32비트 입력과 1개의 캐리를 입력받아 덧셈한 결과와 캐리를 만들었고, 뺄셈의 경우에는 피연산자를 2의 보수를 만들어 덧셈을 수행하였다. 또한 그림 7에서와 같이 배럴 쉬프터는 쉬프터 내부에서 부호 확장과 '0'으로 채워 넣는 기능을 포함하지 않고 쉬프터 입력 전에 부호 확장을 수행하였으며, 이것은 배럴 쉬프터가 2개의 32비트 입력을 받기 때문이다. 그림8은 leading zero 에 대한 시뮬레이션 결과이다.

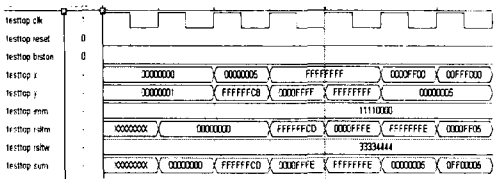

그림6.HDL을 이용한 덧셈 연산 시뮬레이션 결과

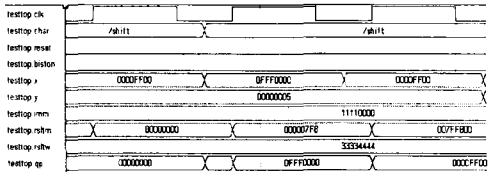

그림7. HDL을 이용한 배럴쉬프트 시뮬레이션 결과

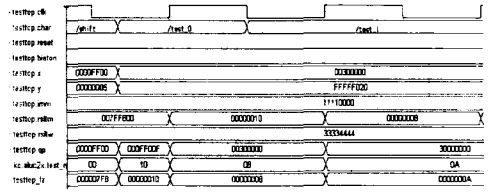

그림8. HDL을 이용한 L-z 수의 시뮬레이션 결과

Ⅳ. 결론

본 논문에서는 32비트 DSP 코어에서 사용가능한 ALU를 설계하였으며, 설계된 기능 블럭들은 논리연산을 위한 논리회로기와 배럴 쉬프터 그리고 오퍼랜드 값에서 몇 번째 비트에 '0'이 아닌 값이 나오는지 그 특정 비트를 알려주는 leading zero 검사 회로를 각각 설계하였다. 설계된 각각의 기능 블록들은 Alter사의 Mexplus를 이용하여 VHDL로 기술하였고 기술된 VHDL을 시뮬레이션하였으며, VHDL로 기술된 코드를 자동합성기에 의해서 표준 셀 라이브러리의 네트리스트로 변환하고, VHDL 검증시 추출된 데스트 벡터를 사용하여 타이밍 시뮬레이션을 수행한 결과 동작전압 3.0v, 온도65C에서 15.46ns으로 나타나 64MHz의 주파수 특성을 얻을 수 있었다. 따라서, 본 논문에서 설계한 ALU는 32비트 DSP 등에 사용할 수 있다.

참고문헌

[1] Michael Dolle, Manfred Schlett, "A Cost-Effective RISC/DSP Microprocessot for Embebded Systems", IEEE Micro, pp. 32-40, October 1995.

[2] Phil Lapsley, Jeff Bier, Amit Shomnam, DSP Processor Fundamentals, IEEE press, 1997.

[3] "TMS320C54x DSP CPU and Peripherals", Texas Instruments.

[4] "TMS320C54x DSP Mnemonic Instrucion Set", Texas Instruments.

[5] Douglas J. Smith, "HDL Chip Design", Doone Publications, 1996

[6] Boaz Port, : A Course in Digital Signal Processing"

[7] 서지근, "32비트 부동소수점 호환 DSP의 설계 및 구현", 부산대학교 전자공학과.

[8] "IKOS VirtuaLogic Guide"

저 자 소 개

황 복 식

2000.2 한남대학교 전자공학과 졸업(석사)

2002. 현재 한남대학교 전자공학과 재학(박사)

관심분야 :

아날로그 회로 설계, 혼성모드 회로 설계

이 영 훈

한남대학교 전자공학과 교수