32 비트 RISC/DSP 프로세서를 위한 17 비트 x 17 비트 곱셈기의 설계

박 종 환, 문 상 국, 홍 종 욱, 문 병 인, 이 용 석
연세대학교 전자공학과
서울시 서대문구 신촌동 134 번지
Tel 02)361-2872, Fax 02)312-4584
stanza@dubiki.yonsei.ac.kr

17 x 17-b Multiplier for 32-bit RISC/DSP Processors

JongHwan Park, Sangook Moon, JongWook Hong, Byung In Moon, Yongsurk Lee
Dept. of Elec. Eng., Yonsei University
134, Shinchon-dong, Seodaemun-gu, Seoul
Tel 02)361-2872, Fax 02)312-4584
stanza@dubiki.yonsei.ac.kr

Abstract

The paper describes a 17 x 17-b multiplier using the Radix-4 Booth's algorithm, which is suitable for 32-bit RISC/DSP microprocessors. To minimize design area and achieve improved speed, a 2-stage pipeline structure is adopted to achieve high clock frequency. Each part of circuit is modeled and optimized at the transistor level, verification of functionality and timing is performed using HSPICE simulations. After modeling and validating the circuit at transistor level, we lay it out in a 0.35 ㎛ 1-poly 4-metal CMOS technology and perform LVS test to compare the layout with the schematic.

The simulation results show that maximum frequency is 330MHz under worst operating conditions at 85℃, 3V. The post simulation after layout results shows 187MHz under worst case conditions. It contains 9,115 transistors and the area of layout is 0.72mm by 0.97mm.

1. 서 론

지속적으로 정보화 사회로 변해가는 과정과 함께 각종 정보 통신 기기들의 소형화가 병행하게 된다. 내장형으로 사용이 되는 CPU 는 대부분이 RISC(Reduced Instruction Set Computer)구조를 채택하고 있고 고성능화가 될수록 DSP 를 처리할 수 있는 하드웨어가 필수 불가결한데 이러한 DSP 처리 유닛의 핵심은 곱셈기이다. 이러한 배경에서 17 비트 오퍼랜드의 쌍을 Radix-4 Booth 알고리즘을 이용하여 곱셈 연산을 수행하는 곱셈기를 설계하였다. 속도를 빠르게 하기 위하여 2 단 파이프라인 구조로 설계를 하였고 월레스 트리의 레이아웃을 규칙적으로 하기 위하여 4:2 CSA 를 사용하였다.

본 논문에서는 멀티미디어 기능을 강화한 32 비트 RISC/DSP 마이크로프로세서를 위한 17 x 17 곱셈기를 설계하였다. 기존의 회로보다 적은 면적을 차지하면서도 고속을 구현하는데 중점을 두었다. 본 논문의 제 2 장에서는 곱셈기의 구조를 설명하였으며, 제 3 장에서는 의 각 부분에 대한 곱셈기의 설계를 기술하였다. 제 4 장에서는 회로의 검증을 위한 시뮬레이션 과정과 레이아웃을 다루었고, 제 5 장에서 최종 결과를 분석하였다.

2. 곱셈기의 구조

본 논문의 곱셈기는 DSP 유닛에서 사용이 되며 2 개의 16 x 16 곱셈기를 사용해서 32 x 32 곱셈의 수행을 한다. 16 x 16 곱셈 연산이 아닌 17 x 17 signed 곱셈인 이유는 32 비트의 오퍼랜드를 2 개의 16 비트 오퍼랜드로 나눌 때 16 비트 오퍼랜드는 signed 일수도 unsigned 일수도 있다. 그러므로 같은 값을 가지는 17 signed 비트 오퍼랜드로 변환하여 사용하였다. 곱셈은 실제로 17 x 17 signed 곱셈이며 따라서 본 논문은 17 x 17 signed 곱셈이다.

곱셈기의 전체 동작은 2 단 파이프라인의 구조로 되어 있다. 클락과 함께 승수와 피승수의 17 비트의 입력이 플립플롭으로 저장이 되어 Booth 인코더를 지나 부분 곱을 선택하는 신호를 만들어 부분 곱 생성 블록으로 들어가서 이 블록에서 9 개의 부분 곱(partial

※ 본 연구의 일부는 과학 재단의 특정 기초 연구 과제 연구비 지원으로 이루어 졌음. (97-0100-0701-2)

product)이 나와 월레스 트리(Wallace tree)에서 연산이 되어져서 33 비트의 합 벡터와 33 비트의 캐리 벡터가 플립플롭에 저장이 되는 첫 번째 단의 파이프라인과 그 다음의 플립플롭에서 클락과 함께 생성이 된 33 비트의 최종 합 벡터와 캐리 벡터가 33 비트 자리 올림 선택 가산기(Carry select adder)로 들어가 최종의 결과로 다시 플립플롭으로 저장이 되는 두 번째의 파이프라인으로 구성이 된다. 그림 1 은 곱셈기의 전체 블록 다이어그램이다.

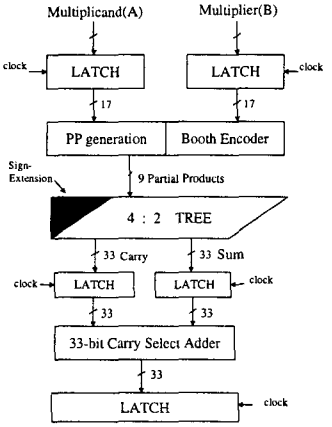

그림 1. 곱셈기의 전체 블록 다이어그램

17 비트 입력 A 에 대한 부분 곱은 Radix-4 Booth's algorithm 에 의해 입력 B 의 시퀀스에 따라 결정되어 각각 9 개의 18 비트 부분 곱이 생성된다.[1] 생성된 부분 곱은 4:2 덧셈기로 구성된 월레스 트리로 보내어져 33 비트 합 벡터와 33 비트 캐리 벡터로 표현된 다음 두 번째 플립플롭 단으로 넘겨진다. 여기서 sign-extension 때문에 생기는 과도한 계산량을 줄이기 위해서 sign-generation 이라는 방법을 사용하여 불필요한 하드웨어를 사용하지 않도록 하였다.[2]

최종적으로 생성된 캐리 벡터와 합 벡터는 33 비트 CSA(Carry-Select Adder)에서 처리되는데 기본 덧셈기 블록으로 캐리 체인 가산기(Carry chain adder)를 사용하였다.[3]

3. 곱셈기의 설계

3.1 Booth 인코더 (Booth encoder)

Booth's algorithm 에 따르면 입력 B 의 시퀀스에 따라서 입력 A 의 형태에 대한 +2A, +1A, 0A, -1A, -2A 중 하나를 택해야 하는데 이러한 선택이 이루어지는 블록이다. 3 비트씩 그룹을 하여 5 개의 선택 신호를 출력한다. 입력이 17 비트의 쌍이기 때문에 9 개의 부분 곱을 구해야 하는데 첫 번째부터 8 번째 부분 곱을 구하는 회로까지는 B 입력의 이전 비트와 현재 두 비트를 참고하기 때문에 8 개의 인코더가 동일하지만 9 번째 부분 곱에 대한 인코더는 표 1 에서 보듯이 b15 와 b16 만 참고하면 가상적으로 sign-extension 을 시켜 b17 을 알 수 있기 때문에 b15 와 b16 만으로 인코딩을 함으로써 b17 를 참고한 것과 똑 같은 결과임을 나타낸 것이므로 회로를 보다 간단하게 할 수 있다.

표 1. 가상적으로 생성되는 b17 비트의 값

b17	b16	b15	Select
0	0	0	0A
1	1	0	-1A
0	0	1	+1A
1	1	1	0A

3.2 부분 곱 생성 블록 (Partial product generation block)

입력 A 가 첫번째 플립플롭 단을 거친 다음 입력 B 의 시퀀스에 따라서 부분 곱을 생성하는 블록이다. 1 에서 8 번째 부분 곱을 생성하는 블록에서는 5:1 mux 를 사용하고 9 번째 부분 곱을 생성하는 블록에서는 3:1 mux 를 사용하였다.[3] 그림 2 는 weak PMOS 를 사용한 mux 의 구조이다.

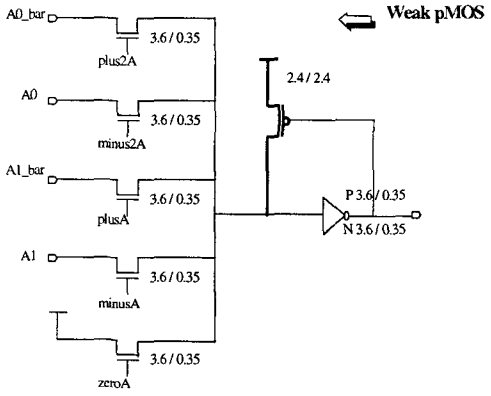

그림 2. weak PMOS 를 사용한 mux 의 구조

입력 B 의 시퀀스에 따라서 +2A, +1A, 0A, -1A, -2A 중 어느것을 선택할 것인지가 정해지면 select 비트 중 하나만이 일정 시간에 활성화 되고 그에 따른 입력 A 의 값들이 각각에 해당하는 형태로 쉬프트 되거나 보수화가 되어서 출력되어 월레스 트리로 들어가게 된다.

3.3 월레스 트리 (Wallace tree)

이전 단에서 생성된 9 개의 부분 곱들을 고속으로 처리하기 위하여 CSA(Carry Save Adder) 구조를 사용하는데 3:2 덧셈기 2 개를 엇갈리게 사용하여 캐리의 전

파시간과 더하는 연산이 오버랩이 되므로 인한 지연시간의 감소를 얻을 수 있고, 트리의 형태를 역 삼각형 구조가 아닌 레이아웃 상에서 규칙적인 구조를 이루기 때문에 본 논문에서는 4:2 덧셈기를 사용하였다. 그림 3 은 4:2 CSA 의 회로도이다.

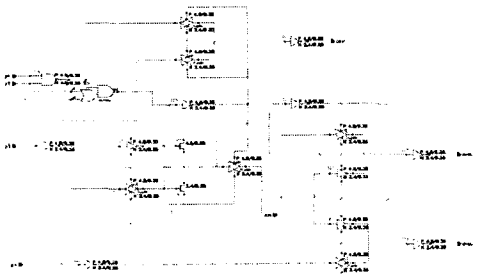

그림 3. 4:2 CSA 의 회로도

월레스 트리 안에서는 부호비트 확장 때문에 생기는 불필요한 계산을 하지 않기 위해서 부호 확장 알고리즘(sign-generation method)을 사용하여 면적을 최소화하고자 하였다. 또한 multiplicand 가 가지는 음의 부호 즉 -2A, -A 의 경우 2's compliment 를 지원하는 neg1 ~ neg9 비트가 월레스 트리에서 계산이 된다. 그림 4 는 월레스 트리에서의 곱셈 과정이다.

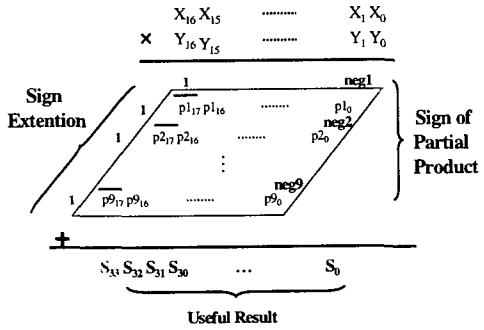

그림 4. 월레스 트리에서의 곱셈과정

3.4 자리 올림 선택 가산기 (Carry-select adder)

월레스 트리에서 계산되어 나온 캐리 벡터와 합 벡터는 일단 플립플롭에 담겨진다. 플립플롭은 간단한 형태의 마스터-슬레이브 형태로 구성하였다. 다음 사이클에서 플립플롭의 출력들은 33 비트 CSA 로 들어가게 되는데 덧셈기의 기본 셀로는 캐리 체인 덧셈기를 사용하였다. 자리 올림 선택 가산기에서 8 비트의 캐리 벡터(Cvec)와 합 벡터(Svec)를 받아서 두 번째 단에서 8 비트 덧셈을 수행하는데, 면적을 최소화하고 속도도 늦추지 않도록 하기 위하여 캐리 체인 가산기(Carry chain adder)를 사용을 하였다.[3]

4. 검증 및 구현

레이아웃의 수행 전에 회로에서 netlist 를 추출하여 HSPICE 시뮬레이션한 예상되는 지연 경로와 지연 시간 그림을 나타내었다.[4] 그림 5 는 임계 경로의 각 블록에서의 지연 시간이다.

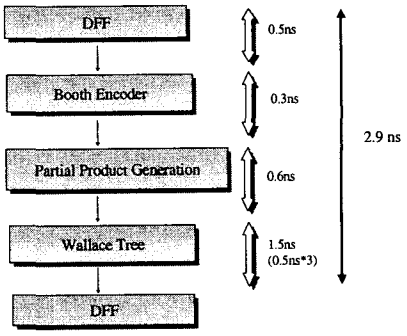

그림 5. 임계 경로의 각 블록에서의 지연 시간

플립플롭(0.5ns) + Booth 인코더(0.3ns) + 부분 곱 생성 블록(0.6ns) + 월레스 트리(0.5ns*3) = 2.9ns

각각의 블록의 회로도는 모두 HSPICE 를 이용한 시뮬레이션으로 동작과 타이밍을 검증하였다. SPICE 시뮬레이션은 다음과 같은 과정을 거쳤다. Schematic 툴을 사용하여 회로도의 넷리스트(netlist)를 추출한 다음, 여기에 테스트 입력을 집어넣어서 결과 파형을 확인하였다. 그림 6 은 SPICE 시뮬레이션 파형을 나타낸 것이다. 월레스 트리까지 통과하는데 걸리는 시간, 즉 임계 경로에서의 지연 시간은 최악 조건(worst condition)에서 3.3ns 가 걸리고, 최상 조건(best condition)에서 2.9ns 가 걸렸다. 최악 조건은 85℃ 3.0V 전원전압 공급의 경우로 설정하였고, 최상 조건은 25℃ 3.3V 전원전압 공급, 그리고 정상 조건(normal condition)은 25℃ 3.3V 전원전압 공급으로 하였다.[4] 그림 6 은 임계경로에서의 시뮬레이션 파형이다.

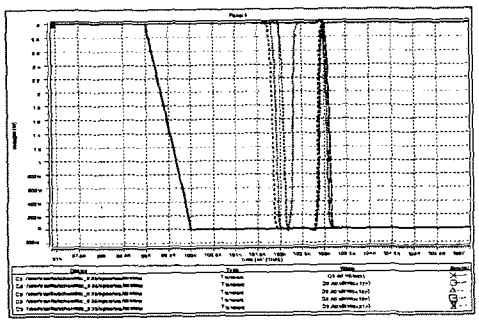

그림 6. 임계경로에서의 시뮬레이션 파형

SPICE 시뮬레이션을 통하여 검증된 회로는 0.35um CMOS 공정을 이용하여 완전 주문형(full-custom)방식으로 설계되었다. 모든 회로는 트랜지스터 수준에서 설계되었으며, 레이아웃을 마친 뒤에 DRC(Design rule check)와 LVS(Layout versus Schematic) 검사를 하여 레이아웃이 제대로 설계되었음을 검증하였다 다음으로, LVS를 통과한 레이아웃에서 기생 커패시턴스를 고려하였을 경우, 간단한 테스트 패턴을 넣고 시뮬레이션 하여 보았다. 그림 8은 post 시뮬레이션의 파형이다. 웰 레스 트리를 통과하는데 걸리는 시간은 최악 조건에서 5.33ns가 걸렸다. 이것은 최악 조건에서 187MHz에서 동작함을 보여준다. 그림 7은 Post 시뮬레이션 파형이다.

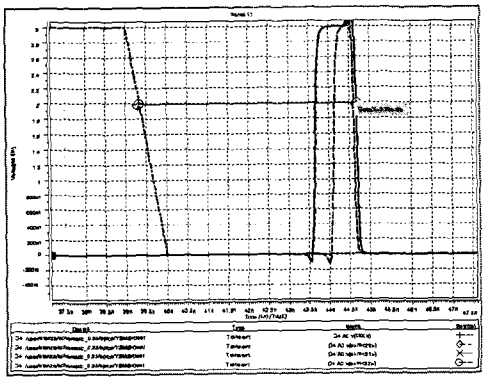

그림 7. Post 시뮬레이션 파형

5. 결 론

본 논문에서는 32비트 RISC/DSP CPU에 적합한 곱셈기를 설계하였다. SPICE 시뮬레이션의 결과는 회로가 정상적으로 동작함을 보여준다. 이렇게 검증된 회로는 0.35um 1-poly 4-metal CMOS 공정을 이용한 완전 주문형 (full-custom) 방식으로 설계되었다. 이렇게 설계된 레이아웃은 다시 DRC와 LVS 검사를 통하여 최종 검증되었다. 그림 8은 곱셈기의 레이아웃이다. 최대 동작 가능 주파수는 정상 조건에서 350MHz이며, 최악 조건에서도 300MHz이었다. 기생 커패시턴스를 고려한 post 시뮬레이션에서는 최악 조건에서 187MHz에서 동작함도 확인하였다. 코어(core)의 면적은 0.72mm x 0.97mm이고 총 트랜지스터의 수는 9,115개이다.

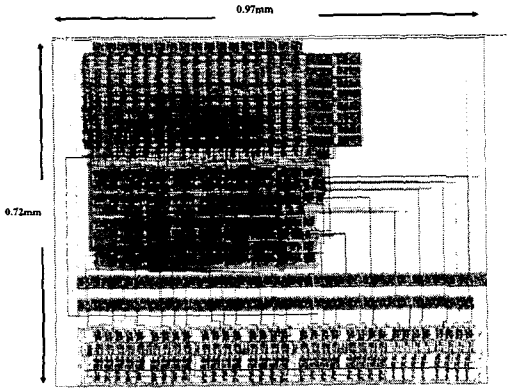

그림 8. 곱셈기 전체 레이아웃

6. 참 고 문 헌

[1] ISRAEL KOREN, *"Computer Arithmetic Algorithms"*, University of Massachusetts, Amherst

[2] M. Annaratone, W.Z Shen, "The Design of a Booth Multiplier : nMOS vs. CMOS Technology"

[3] Yong S. Lee, "A 4 Clock Cycle 64 x 64 Multiplier with 60MHz Clock Frequency", *Kite Journal of Electronics Engineering*, Vol. 2, No. 2 Dec. 1991.

[4] *HSPICE User's Manual : Elements d Device models*, Meta-Software, vol. 1-3, 1996