고속 연산을 위한 64bit 가산기의 설계

오.재환* , 이영훈*, 김상수*, 상명희**
한남대학교 전자공학과* 공주대학교 정보통신공학과**
대전광역시 대덕구 오정동 133 한남대학교 전자공학과
ojh@ee.hannam.ac.kr

Design of high speed 64bit adder

Jeahwan Oh* , Younghun Lee*, Sangsu Kim*, Munghee Sang**
Dept. of Electronic. Eng., Hannam Univ.*, Dept. of Info. Comm. Eng., Kongju University**
133 Ojung-dong Deaduk-Gu Teajeon, Korea
ojh@ee.hannam.ac.kr

Abstract

산술연산을 수행하는 가산기는 ALU(Arithmetic Logic Unit)의 성능을 좌우하는데 매우 중요한 역할을 하며, 어떠한 캐리 생성 방식을 사용하느냐에 따라 그 성능이 결정될 수 있다. RCA(Ripple Carry Adder)는 간단하고, 쉬운 설계로 널리 사용되지만, 캐리의 전파 지연 문제로 인해 고속의 가산기 응용에는 부적합하다. 또한, CLA(Carry Lookahead Adder)방식의 가산기는 캐리의 지연시간이 가산기의 단수와 무관하므로, 연산속도를 높일 수 있는 장점이 있지만 더하고자 하는 bit의 수가 클수록 회로가 매우 복잡해지는 큰 단점을 가지고 있다. 따라서, 본 논문에서는 간단하면서도 성능이 우수한 64bit 가산기를 설계하고 시뮬레이션을 통하여 설계된 회로의 우수성을 증명하였다.

1. 서론

반도체 기술의 급격한 발전으로 인해 집적도의 향상은 최근 몇 년간 괄목할 만한 성장을 이루었다. 이러한 집적도의 향상은 system-on-chip을 가능하게 하고 있으며 이로 인해 IC를 통한 대용량 시스템의 구현이 기술적 추세가 되어 가고 있다. 그러나 system-on-chip을 구현하기 위해서는 매우 많은 양의 데이터를 빠른 시간에 처리해야 하는 어려운 문제를 해결할 수 있는 설계기술이 뒷받침되어야 한다. 특히 마이크로 프로세서나 디지털필터 등에서와 같이 연산 기능을 수행하는 경우 고속 multi-bit 덧셈이나 곱셈 등을 위한 회로 설계 기술에 대한 연구의 중요성이 계속 증가되고 있는 추세이다.

이중 덧셈 연산은 computing system이나 마이크로프로세서 등의 data path 블록에 필수적인 기능 중의 하나이며 시스템의 속도 성능에 큰 영향을 미친다. 따라서 덧셈연산의 속도를 향상시키기 위한 연구가 계속되고 있다.[1][2][3][4]

특히 bit-parallel구조의 덧셈연산은 이를 위해 여러 가지 측면의 연구가 이루어져 왔으며 그 대표적인 예로 Ri pple - Carry Adder(RCA), Carry Lookahead Add -er (CLA), Conditional -Sum Adder(CSA), Carry-s Kip A- dder(CKA) 등을 들 수 있다. 위의 방법들은 모두 덧셈 연산의 성능을 좌우하는 carry의 전파를 구현하는 방법 에 의해 구분되어지며 RCA 방식은 발생되는 carry를 다음 단에 그대로 전달시켜 순차적으로 연산을 수행하 는 방식으로 회로를 작고 간단하게 구현 할 수 있으나 캐 리 전달시 발생되는 지연으로 인해 고속연산에는 부적 합하다. 이러한 문제를 해결하고자 carry의 전파방식을 달리하는 여러 가지의 덧셈 방식이 개발되었다.

2. Adder 기능 Block의 설계

n-bit의 리플 캐리 가산기(Ripple Carry Adder)는 그 출력을 계산하는데 너무 많은 시간이 걸린다. 어떤 특정한 구현 기술에 관련된 지연시간이 받아들일 만한 것인지는 다른 프로세서 성분들의 속도와 주기억장치의 사이클 시간과의 관계에서 결정할 수 있다. 논리 게이트들의 네트워크를 통과하는데 걸리는 시간은 기본 게이트들을 제작하는데 쓰인 전자기술과 입력에서 출력까지의 게이트 수에 따라 달라질 수 있다. 특정 기술을 선택하면, 그 기술을 이용한 게이트들로 만들어진 어떤 조합 논리 네트워크일지라도, 가장 긴 경로를 따라서 논리 게이트의 지연 시간들을 모두 더하여 전체의 지연 시간을 구할 수 있다. 리플캐리 가산기의 경우에, 가장 긴 신호 전파는 최하위 비트(LSB)의 x_0, y_0, c_0 입력으로부터 최상위 비트(MSB)의 출력 c_n, S_{n-1}까지의 경로를 갖는다. 이렇게 리플캐리 가산기의 지연시간을 생각해 보면, 모든 가산기 블록은 C_i로부터 C_{n-1} 위치까지 도달하고 S_{n-1}을 만들어내는 시간에 하나의 n- bit 덧셈을 수행할 수 있다. 즉, 캐리가 최상위 비트까지 전달(Ripple)된 후에 가산이 이루어지기 때문에 속도가 느리다. 하지만, 빠른 가산기의 설계를 위한 논리 구조는 올림수 신호의 생성속도를 향상시키는 문제에 달려있다.[5] 이러한 단점을 극복하기 위해서 본 논문에서는 기본적인 가산기의 구조로 CLA방법을 이용하였다. CLA는 리플캐리 가산기에서 캐리의 전달 속도를 향상시키기 위한 기술이다. 이것은 전가산기에서 전가산기까지 리플되는 캐리를 이용하는 것이라기 보다는 직접 입력된 캐리를 생성하는 것이다. 그림 2.1은 실제 4bit CLA의 블록 다이어그램이다.

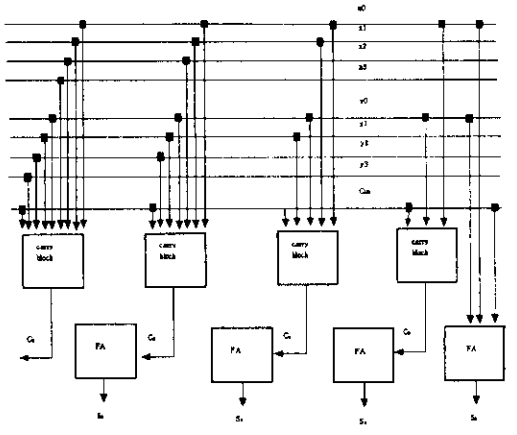

그림 2.1 4bit CLA의 블록다이어그램

그림 2.1 4bit CLA 블록다이어그램에서 캐리블럭은 전가산기에 대하여 캐리입력을 발생시킨다. 각 캐리블럭의 입력은 단지 입력의 수와 초기 캐리입력(C_0)이다. 각 캐리입력에 대한 불리안(Boolean)표현은 전가산기의 Carry-out표현을 사용함으로써 정의할 수 있다. 예를 들어,

$$C_{i+1} = x_i y_i + C_i(x_i + y_i). \qquad (2.1)$$

이므로, C_1은 아래와 같이 발생된다.

$$C_1 = x_0 y_0 + C_0(x_0 + y_0).$$

이와 유사하게, C_2는 아래와 같이 발생된다.

$$C_2 = x_1 y_1 + C_1(x_1 + y_1).$$
$$= x_1 y_1 + [x_0 Y_0 + C_0(x_0 + y_0)](x_i + y_i).$$

각 C_i에 대해 간단히 표현하기 위해 종종 g와 p 두 개의 기호를 사용하여 이 기호들은 다음과 같이 나타내면

$$g_i = x_i y_i$$
$$p_i = x_i + y_i$$

이 되므로 식(2.1)을 다음과 같이 쓸 수 있다.

$$C_{i+1} = g_i + p_i C_i \qquad (2.2)$$

g는 캐리를 생성하고 g_i가 1일 때 C_{i+1}이 1이 된다는 것이다.
p는 입력캐리에서 출력캐리까지의 지연시간을 나타낸다. 이것은 C_i와 p_i가 1을 가질 때 C_{i+1}이 1이 된다.
이들 기호를 이용하여, 아래의 식을 얻을 수 있다.

$$C_i = g_0 + C_0 p_0,$$
$$C_2 = g_1 + p_1 g_0 + p_1 p_0 C_0,$$
$$C_3 = g_2 + p_2 g_1 + p_2 p_1 g_0 + p_2 p_1 p_0 C_0,$$
$$C_4 = g_3 + p_3 g_2 + p_3 p_2 g_1 + p_3 p_2 p_1 g_0 + p_3 p_2 p_1 p_0 C_0$$

그림2.2는 C_4를 발생시키는 실제의 로직다이어그램이다. 이것들은 다섯 개의 입력을 위한 AND gate와 OR

gate이다. 역시 신호 p_3은 4개의 AND gate를 구동하기 위해 필요하다. [6]

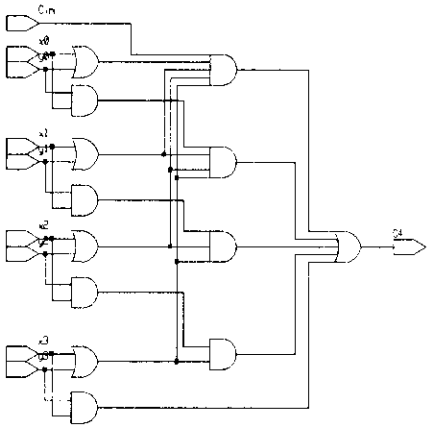

그림 2.2 C_4 발생 로직 다이어그램

일반적으로 두 개의 n-bit정수를 더하기 위해서는 n+1의 fan-in을 가지는 AND와 OR gate가 필요하다. 이것은 또 n개의 AND gate를 구동시킬 p_{n-1}개의 신호가 필요하다. 실제적으로, 이것들의 필요조건은 n>5이면 불가능하다. 이런 필요조건 이외에, 그림 2.1과 그림 2.2는 큰 n에 대해 모듈러 설계를 지원하지 않는다.

위의 문제를 해결하기 위해, 우리는 기술에 의존적인 fan-in, fan-out을 확실한 수로 제한해야 한다. 이것은 캐리예측회로에 더 많은 로직단계를 더해야 하는 결과를 낳는다. 앞의 예에서 게이트의 fan-in을 4개로 제한한다면 C_4를 계산하기 위해서 더 많은 게이트 지연이 생길 것이다. 여기에서, 캐리발생 항 G_0와 전달지연 항 P_0 두 개의 새로운 항을 정의하면, 여기에서

$$G_0 = g_3 + p_3 g_2 + p_3 p_2 g_1 + p_3 p_2 p_1 g_0$$

$$P_0 = p_3 p_2 p_1 p_0$$

이므로 다음을 얻을 수 있다.

$$C_4 = G_0 + P_0 C_0$$

3. 고속 연산을 위한 64bit 가산기의 설계

비록 CLA가 RCA에 비해 빠르다 할지라도 CLA는 RCA에 비해 많은 회로를 필요로 하기 때문에 one-chip화하기에 매우 복잡하고 더 큰 칩의 공간을 필요로 한다. 또한, 실질적인 문제는 게이트의 가능한 팬입력(fan-in)이 유한하다는 것이다. C_{i+1}에 대한 표현은 가장 큰 AND 항에 i+2개의 입력이 필요하고 가장 큰 OR항에도 i+2개의 입력이 필요하다. 일반적인 게이트에서 팬입력은 보통 8개 이하로 제한된다.[7]

게이트가 가지고 있는 기본적인 특성으로 인하여 전체회로가 대형화되는 문제점을 극복하고자 본 논문에서는 그림 2.1에서와 같은 방법으로 32bit의 CLA가산기를 설계하였고, 또한, 전체 64bit의 가산기를 설계하기 위해서 32bit의 CLA가산기를 CSA(Carry Select Adder)방식으로 연결하여 64bit의 연산결과를 얻는 Adder를 설계하였다. 그림 3.1은 64bit 전체 Adder의 블록도를 나타내고 있다.

그림 3.1에서는 3개의 32bit CLA Adder를 사용하였다. 하위 32bit를 계산하기 위해서 하나의 32bit CLA Adder를 사용하였고, 하위 32bit에서 계산되어진 결과(Sum)는 그대로 출력된다. 하위 32bit의 캐리 출력은 상위 32bit의 캐리로 입력되며 이때 MUX는 캐리가 '0' 혹은 '1'인 경우에 따라서 나머지 상위 32bit를 32bit CLA1과 32bit CLA2 중에 선택하여 출력하게 된다.

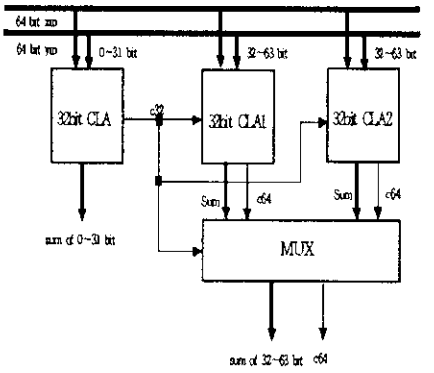

그림 3.1 64bit CLSA Adder의 블록도

4. 컴퓨터 시뮬레이션 및 성능평가

본 논문에서 설계한 고속 64bit Adder는 전자통신연구원 에서 개발한 LodeCap을 사용해서 평가하였다. 입력 신호는 worst case의 딧셈시간을 측정할 수 있도록 피가산자에 FFFFFFFFFFFFFFFF를 가산자에 0000 0000000000001을 인가시킨 후 C_{in}을 1로 toggle시켜 덧셈 의 결과과 인 0000000000000001이 출력되기까지의 지연 시간을 측정하였다. 전체 회로에 대한 LodeCap 결과 파 형을 그림 4.1에 나타냈다.

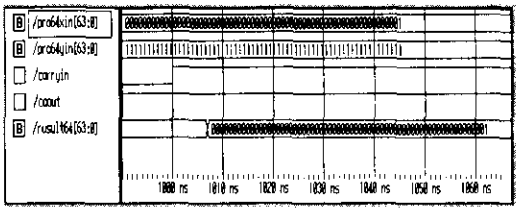

그림 4.1 고속 64bit Adder의 시뮬레이션

결과 파형에서 알 수 있듯이 설계된 64bit의 Adder는 worst case에서 약 154MHz까지 동작됨을 확인하였다.

5. 결론

본 논문에서는 유한한 fan-in을 가지는 게이트의 특성으로 인하여 더하고자 하는 bit의 수가 많아질수록 고유의 특성을 살리지 못하는 CLA 설계 방식의 단점을 극복하고, RCA의 캐리지연으로 인한 연산속도의 저하를 막고자 본 논문에서 CLSA(Carry Lookahead Select Adder)방식 제안 하였다. 제안한 CLSA방식을 이용하여 worst case에서 154MHz까지 동작하는 64bit 고속 Adder를 설계하고 시뮬레이션을 통하여 그 성능이 우수함을 증명하였다. 또한 설계된 칩을 이용하여 마이크로프로세서나 DSP 등과 같이 고속연산을 필요로 하는 경우에 전체 성능의 병목지점 중에서 하나인 Adder의 속도를 크게 향상시켰다.

참 고 문 헌

[1] Z. Guan, et al., "A simple and high speed CMOS carry chain adder auchitecture", International Journal of Electronics. vol.75, NO.4, p.p743-752, 1993

[2] H. Morinaka, et al., "A 64bit Carry Look-ahead CMOS Adder using Modified Carry Select", IEEE Proc. Custom Integrated Circuits Conference. p.p 585-588, 1995

[3] L. Dadda, V. Piuri, "Pipelined Adders", IEEE Trans. Computers, vol.45, NO.3, p.p348-356, Mar 1996

[4] K. Ueda, et al., "A 64bit Carry Lookahead Adder Using Pass Transistor BiCMOS Gates", IEEE Journal of Solid-State Circuits. vol.31, NO.6, p.p810-817, June 1996

[5] V. Carl Hamacher, Computer organization, third edition, McGraw-Hill, Newyork, 1990

[6] Mehdi R. Zargham, Computer architecture-single and parallel systems, Prentice Hall, New Jersey, 1996

[7] Sajjan. G. Shiva, Computer design and architectu- re, Little Brown, Boston, 1985