고속 검사합 모듈의 덧셈구조에 관한 비교 연구

°김대현, 한상원, 공진흥
광운대학교 컴퓨터공학과
Tel : 02-940-5126, Fax : 02-940-5121, E-mail : abracsus@explore.kwangwoon.ac.kr

A comparative study on the addition architecture of high-speed checksum module

°Dae-Hyeon Kim, Sang-Won Han, Jin-Hyeung Kong
Department of Computer Engineering, Kwangwoon University
Tel : 02-940-5126, Fax : 02-940-5121, E-mail : abracsus@explore.kwangwoon.ac.kr

Abstract

In this paper, a comparative study is presented to evaluate the addition architecture of the high-speed checksum module in TCP/IP processing. In order to speed up TCP/IP processing, H/W implementation offers concurrent and parallel processing to yield high speed computation, with respect to S/W implementation. This research aims at comparing two addition architectures of checksum module, which is the major bottleneck in TCP/IP processing. The 16-bit and 8-bit byte-by-byte addition architecture are implemented by the full custom design, and compared, in analytical and experimental manner, from standpoint of space and performance. For LG 0.6㎛ TLM process, the 8-bit addition implementation requires the area, 1.3 times larger than the 16-bit one, and it operates at 80MHz while the 16-bit one runs by 66MHz.

1. 서론

최근 컴퓨터 통신의 환경은 매우 급격한 변화를 겪고 있다. 광통신 기술의 발전 및 상용화에 따라서 데이터 전송 대역폭이 증가하였으며, 컴퓨터 통신의 가입자는 실시간 통신 및 대용량의 광대역 멀티미디어 응용 서비스를 요구하고 있다.

그러나 컴퓨터 통신에서 TCP/IP 프로토콜 처리는 통신에서 물리층의 처리 속도를 상위 프로토콜 계층으로 전달하지 못하는 병목현상을 가져오게 된다. 이러한 TCP/IP의 병목현상을 해결하기 위해서는 TCP/IP 프로토콜 처리의 HW 구현에 대한 연구가 요구된다. TCP/IP HW구현에 관한 연구는 지금까지 활발히 진행되었는데, 대표적인 예를 보면 프로그래머블 프로토콜 VLSI Engine 구현[1], TCP/IP 성능향상을 위한 HW/SW Architecture 연구[2], 그리고 프로토콜 처리의 병렬성을 이용한 멀티프로세서의 구현[3] 및 전용 VLSI 구현[4]등이 있다. 이와 같은 연구들에서 타이머 관리, 버퍼 관리 등과 같이 TCP/IP 처리성능을 제한하는 병목현상이 제기되었는데 그 중 가장 주된 것은 전송도중 발생하는 에러를 검출하는 검사합이다. 검사합은 실시간으로 많은 양의 데이터를 가산 처리해야 하는데, 함수를 호출하여 순차적으로 가산하는 TCP/IP 프로토콜 SW에서는 이러한 넓은 대역폭을 제공하지 못하고 있다. 따라서 검사합 처리의 병목현상을 해결하기 위해 CLA(Carry Look-ahead Adder)와 같은 고속의 가산기를 사용한 HW 구현이 필요하며 알고리즘의 최적화 및 캐리전달지연을 최소화 할 수 있는 방법이 개발되어야 한다. 실제로 RFC1071[5]은 16-비트 덧셈구조와 8-비트 byte-by-byte 덧셈구조를 모두 제시하고 있는데 HW구현에 앞서 이에 대한 성능의 비교분석이 요구된다.

본 논문에서는 검사합 알고리즘 최적화와 CLA를 이용한 덧셈구조의 설계 및 비교 평가를 통하여 고속의 검사합 모듈에 적합한 HW 구조를 제시하고자 한다. 2장에서는 검사합 알고리즘과 문제점을 소개하고 최적화 된 덧셈구조를 제시하였으며, 3장에서는 이를 완전주문형 방법으로 설계하였다. 4장에서 설계된 덧셈구조를 면적 대비 성능의 관점에서 평가하였다.

2. 검사합 알고리즘

2.1 검사합 알고리즘 개요

TCP/IP 검사합의 알고리즘은 그림 1과 같다. 송신의 경우 두 옥텟(8-비트)을 한 쌍으로 16-비트 integer를 구성하고, 이러한 16-비트 integer들의 1's complement 가산이 수행된다. TCP/IP 패킷에서 가산

결과의 1의 보수가 검사합 필드 값으로 결정되어 전송된다. 수신의 경우 검사합을 수행하기 위해서는 검사합 필드를 포함하여 수신된 패킷의 1's complement 가산이 수행되고 그 결과가 모두 1 이면 에러 없이 검사가 성공적으로 수행된 것이고, 0 이면 에러가 발생했다는 것을 의미한다. 이는 전송된 패킷에서 검사합 필드와 이를 제외한 나머지를 덧셈한 값은 서로 1의 보수관계에 있기 때문에 0이 나오는 것이다. 다시 그 값의 보수를 취하면 모든 비트가 1이 되기 때문에 에러가 없음을 나타낸다.[5]

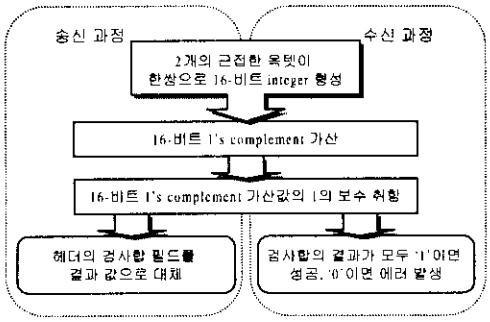


그림 1. 검사합 알고리즘

2.2 검사합 알고리즘의 문제점 및 최적화

검사합에서 수행되는 1's complement 덧셈은 최상위 비트에서 발생하는 캐리를 다시 캐리 입력으로 하는 end around carry[6] 구조를 갖고 있다. 따라서 덧셈 후에 최상위 캐리를 한번 더 더해주어야 하기 때문에 2배의 클럭 주기(clock cycle)가 소요된다. 이러한 end around carry의 문제점은 어떤 덧셈구조에서나 공통적으로 발생된다. 이를 해결하기 위해 누적연산에서 발생하는 캐리를 다음 덧셈의 캐리입력으로 하여 그림 2와 같이 파이프라인 처리함으로써 해결할 수 있다.

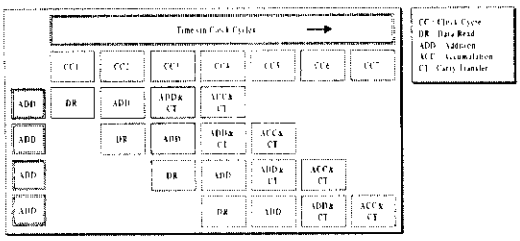


그림 2. 1's complement 덧셈의 파이프라인 동작

또한 16-비트 덧셈의 경우 캐리전달로 인한 지연이 크기 때문에 속도저하의 주원인이 된다. 16-비트 덧셈의 캐리전달지연은 8-비트 byte-by-byte[5] 구조로 해결할 수 있는데 그림 3은 이러한 8-비트 byte-by-byte 구조를 16-비트 덧셈과 비교하고 있다. 즉 16-비트 데이터를 8-비트로 나누어 각각을 더하고 최종 캐리를 서로 엇갈려 더해줌으로써 16-비트 덧셈과 동일한 결과가 나타난다. 이를 통해 16-비트 덧셈으로 인해 발생되는 캐리전달지연을 최소화 할 수 있으며 클럭의 속도(clock speed)를 증가시킬 수 있다.

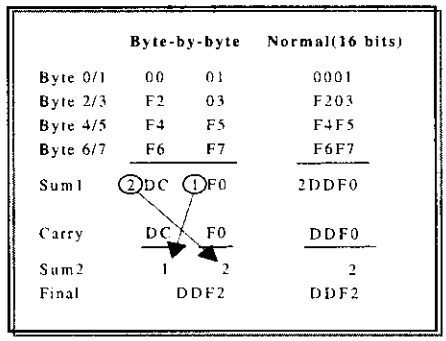


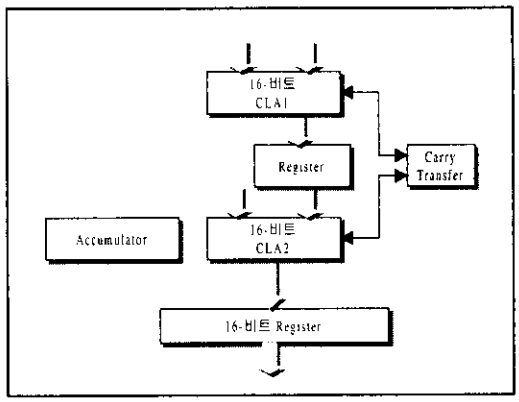
그림 3. 16-비트 덧셈 vs. 8-비트 Byte-by-Byte 덧셈 비교

3. 검사합 덧셈구조의 HW 설계

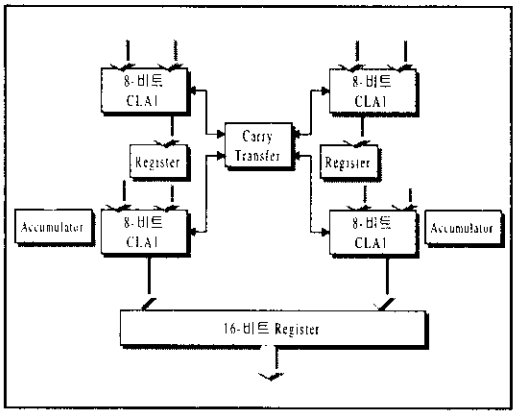
3.1 HW 덧셈구조의 비교

최적화 된 검사합 알고리즘을 바탕으로 덧셈구조의 성능을 비교하기 위해 16-비트 CLA를 이용한 덧셈구조와 8-비트 CLA를 이용한 byte-by-byte 덧셈구조의 블록도를 설계하였다.

먼저 16-비트 CLA를 이용한 경우 그림 4(a)와 같이 누적연산을 위해 16-비트 CLA 2개와 1개의 누산기(accumulator)를 사용하였으며, end around carry로 인한 오버헤드를 줄이기 위해 캐리전달 블록을 두어 다음 덧셈과 병렬처리되도록 하였다. 8-비트 byte-by-byte 구조의 경우 8-비트 CLA 4개와 누산기 2개를 사용하였으며, 위와 동일하게 캐리전달 블록을 두어 다음 덧셈에서 캐리가 엇갈리게 전달되도록 하였다. 즉 그림 4(b)와 같이 CLA1에서 발생한 캐리는 CLA4에 전달되고, CLA2에서 발생한 캐리는 CLA3에 전달된다.

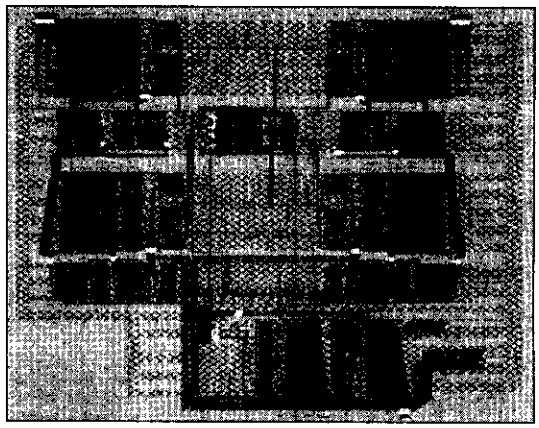


(a) 16-비트 덧셈구조



(b) byte-by-byte 덧셈구조

그림 4. 덧셈구조의 비교 블록도

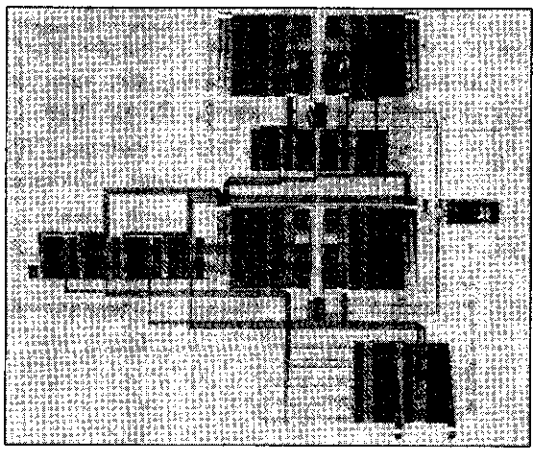


(b) byte-by-byte 덧셈구조 레이아웃

그림 5. 덧셈구조의 설계

3.2 덧셈구조의 HW설계

검사합 덧셈구조의 설계 및 구현은 LG 0.6㎛ TLM(Triple Layer Metal) 공정을 이용하여 완전 주문형(full custom)으로 제작되었다. 기초 셀(primitive cell)과 매크로 셀(macro cell)의 레이아웃 (layout)설계는 공개된 CAD tool인 Magic을 사용하였고, 최종 단계인 전체 P&R(Placement&Routing) 및 DRC(Design Rule Check)와 LVS(Layout Versus Schematic)는 Cadence에서 진행하였다. 그림 5(a)는 16-비트 CLA를 사용한 덧셈구조의 레이아웃을, 그림 5(b)는 8-비트 CLA를 사용한 byte-by-byte 덧셈구조의 레이아웃을 각각 보여주고 있다.



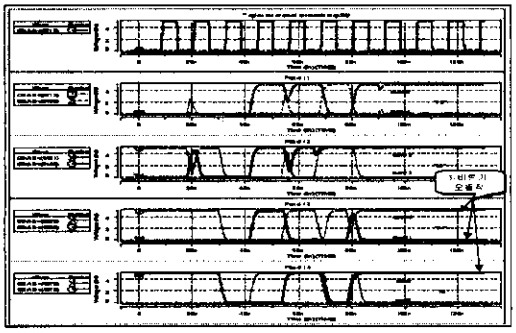
(a) 16-비트 덧셈구조 레이아웃

4. 실험 및 성능 분석

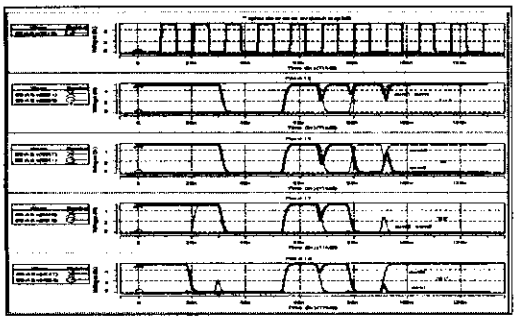
앞서 설계된 두 덧셈구조의 성능 분석은 Avant사의 Star-Hspice[7]를 통해 수행하였다. 캐리전달지연의 측면에서 worst case인 경우 성능을 비교하기 위하여 5개의 32-비트 16진수 데이터 "AAFFFF01", "FFFFFFFF", "FFAACCFF", "FF013323", "FFFF0000"를 테스트 벡터(test vector)로 하여 결과 값 "A8D1"이 출력되는 것을 확인하였다.

면적 대비 성능에 대한 비교 분석에서 먼저 게이트 수를 보면 16-비트 덧셈구조는 2832개이고, 8-비트 덧셈구조는 2884개로 52개의 게이트 수 차이를 보였다. 또한 인터커넥션(Interconnection) 측면에서 각 모듈간의 연결에 관련된 길이를 고려하여 가로, 세로의 면적을 살펴보았다. 8-비트 덧셈구조는 15068㎛ × 11861㎛인 반면 16-비트 덧셈구조는 10624㎛ × 12405㎛를 차지하여 8-비트 덧셈구조가 1.3배 더 큰 면적을 차지한다는 것을 알 수 있었다. 성능 분석은 클럭주기를 15nsec, 12nsec, 10nsec로 구분하여 동작여부를 검사하는 방법으로 진행되었다. 검증 결과를 보면 16-비트 덧셈구조는 15nsec에서 동작하여 66M의 동작속도를 가지며, 그 이상의 클럭주기에서는 오동작을 하였다. 8-비트 덧셈구조는 15nsec 및 12nsec에서 동작하여 66M이상 80M의 동작속도를 보였으며 10nsec에서는 오동작을 하였다. 그림 6은 16-비트 덧셈구조가 12nsec에서 3-비트 오동작 한 검증결과를 보이고 있으며, 그림 7은 8-비트 덧셈구조가 12nsec에서 올바르게 동작한 결과를 보여주고 있다.

위의 실험결과 면적의 측면에서는 8-비트 byte-by-byte 덧셈구조가 1.3배 많은 면적을 차지하고는 있으나, 동작속도가 80M로 16-비트 덧셈구조 보다 1.2배 더 빠른 동작속도를 가지고 있다. 따라서 면적은 다소 크나 고속 가산을 위해서는 더 빠른 동작속도가 요구되기 때문에 8-비트 byte-by-byte 덧셈구조가 검사합의 HW구현에 적합하다.

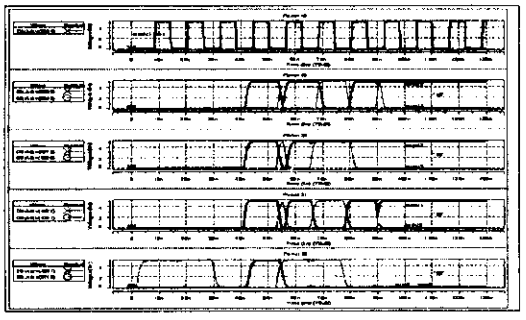


상위비트 sum15~sum8 ("A6")

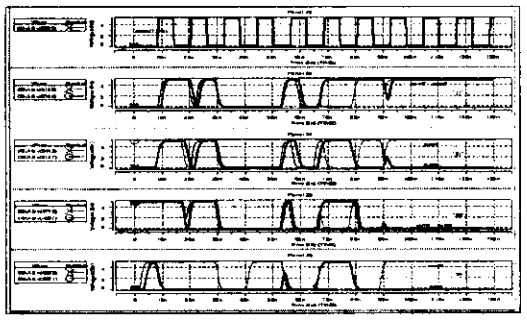


하위비트 sum7~sum0 ("D1")

그림 6. 16-비트 덧셈구조의 simulation 파형 (80M)



상위비트 sum15~sum8 ("A8")



하위비트 sum7~sum0 ("D1")

그림 7. 8-비트 덧셈구조의 simulation 파형 (80M)

5. 결론 및 추후과제

본 논문에서는 TCP/IP 프로토콜 처리를 위한 HW 구현 연구의 일환으로 병목현상의 주된 요인으로 작용하는 검사합을 고속으로 처리하기 위한 HW 구현에 관한 연구를 수행하였다.

본 연구는 8-비트 byte-by-byte 덧셈구조가 16-비트 덧셈구조에 비해 캐리전달지연을 감소시켜 성능을 향상시킬 수 있다는 점을 고려하여 수행되었다. 실제로 두 모델을 HW로 구현하여 성능을 비교 평가해 본 결과 8-비트 덧셈구조가 면적은 1.3배 더 소요되지만 80M에서 동작되어 66M에서 동작하는 16-비트 덧셈구조에 비해 1.2배 더 빠르다는 것을 밝혔다. 따라서 8-비트 덧셈구조가 고속의 처리속도를 필요로하는 검사합에 더 적합하다는 결론을 얻었다.

추후과제로는 HW로 구현된 검사합 모듈을 통해 병목현상을 해결함으로써 TCP/IP 프로토콜 처리의 고속화를 실현하는 것이 요구된다.

참고문헌

[1] A. S. Krishnakumar, W. C. Fischer, and Krishan Sabnani, "The Programmable Protocol VLSI Engine(PROVE)", IEEE Transactions on Communications. Vol. 42, No. 8, pp. 2630-2641, August 1994.

[2] Roy A. Sutton, Sameer M. Jalnapurkar, "Hardware/Software Architecture for TCP/IP Acceleration on UNIX Workstation", http://infopad.eecs.berkeley.edu/~rsutton/class_project/cs252/final_report

[3] T.F. La Porta and M. Schwartz, "A high-speed protocol parallel implementation", IFTP High Performance Networking, December 1992.

[4] H. Abu-Amara, T. Balraj, T. Barzilai, and Y. Yemini, "A silicon compiler for very fast protocol processing", IFIP, pp 181-195, 1989.

[5] R. Braden, D. Borman, and C. Partridge, "RFC 1071 : Computing the Internet Checksum", September 1988.

[6] Charles H. Roth, "Fundamentals of Logic Design", WEST Publishing Company, 1992.

[7] "HSPICE User's Manual : Analysis and Methods", Meta-Software, Inc., March, 1995.

※ 본 연구는 반도체설계교육센터(IDEC)로부터 부분적인 지원을 받아 이루어 졌음.