Wide Bit-Width 프로세서를 위한 최적의 Fault-Tolerant Adder Scheme 분석

*공병용, 박인철
KAIST 전기 및 전자공학과
e-mail : *bykong.ics@gmail.com, icpark@kaist.edu*

Analysis of Optimal Fault-Tolerant Adder Schemes for Wide Bit-Width Processors

*Byeong Yong Kong, In-Cheol Park
Department of Electrical Engineering
KAIST

Abstract

Effectiveness of various fault-tolerant adder schemes on wide bit-width processors is examined. By expanding the concept of quadruple time redundancy (TR) adder while satisfying constraints for achieving high hardware efficiency, the concept of 2^n-tuple TR adder schemes is proposed. Among the various schemes including the proposed ones, an optimal scheme is determined for each bit-width of a processor based on implementation results.

I. 서론

프로세서의 bit-width는 여러 가지 애플리케이션에 의해 요구되는 고성능에 대한 수요와 더불어 증가해왔다. 16-bit 프로세서는 이미 그 용처가 현저히 줄어들었으며, 현재는 32-bit 또는 64-bit의 프로세서가 널리 사용되고 있다. 학계 및 산업계에서는 128-bit 프로세서에 관한 연구 결과가 발표되는 추세이며, 미래에는 256-bit 이상의 프로세서가 사용될 것으로 예측되고 있다.

그럼에도 불구하고 기존의 fault-tolerance 기술에 대한 연구는 현재 주로 사용되고 있는 32-bit 또는 64-bit의 프로세서에 집중되어왔다. 따라서 기존 기술이 미래형 wide bit-width 프로세서에 그대로 적용될 경우, 그 효용성을 확신할 수 없다는 문제가 있다.

이 문제를 해결하기 위하여 본 논문에서는 여러 가지 fault-tolerant adder 기술이 wide bit-width 프로세서에 적용되었을 때의 효용성에 대해 논한다. 기존 기술 중 하나인 quadruple time redundancy adder [1]의 개념을 확장하여 더 높은 order의 2^n-tuple time redundancy adder를 제안하며, 구현 결과에 입각하여 각 bit-width에 대한 최적의 fault-tolerant adder scheme은 무엇인지 기술한다.

II. Fault-Tolerant Adders

2.1 Triple Modular Redundancy (TMR)

TMR [2]은 매우 널리 사용되는 fault-tolerance 기술로서, 같은 모듈 3개를 동시에 사용하여 연산하고, voting을 통하여 3개의 결과 중 다수의 결과를 최종 결과로 취하는 방식이다. Critical path에 오직 voter만 추가되므로 딜레이 오버헤드가 거의 없다는 장점이 있으나, 하드웨어 오버헤드가 3배 이상이라는 큰 단점이 있다. 그림 1은 TMR에 기반하여 16-bit adder를 구현했을 때의 block diagram이다.

2.2 Time Shared TMR (TSTMR)

TMR과 같이 모듈의 redundancy를 사용하는 방식은 하드웨어 오버헤드가 매우 크므로, 이를 경감하기 위해 time redundancy가 사용된다. Time redundancy란 복수의 하드웨어 대신 복수의 반복 연산을 통해 redundancy를 얻는 기술이다.

TSTMR [3]은 TMR의 개념을 그대로 적용하여 3개의 같은 모듈을 두되, 하드웨어 오버헤드를 줄이기 위해 n-bit를 3분할하고, 3분할된 operand를 한 번에 하나씩 총 3번의 연산을 반복하는 scheme이다. 예를 들어, 16-bit adder는 그림 2에 나타나 있듯 *ceiling*[16/3]-bit adder, 즉 3개의 6-bit adder로 분할되며, 이 3개의 adder를 가지고 3번의 6-bit 덧셈을 하면 최종적으로 16-bit 덧셈을 할 수 있게 된다. 3분할된 operand 중 하나를 선택하기 위하여 3:1 multiplexer (MUX)가 사용되며 입력 carry와 앞선 계산에 의한 carry 중 하나를 선택하기 위한 2:1 MUX가 하나 필요하다. 또한 앞선 계산에 의한 carry를 저장하기 위한 1-bit 레지스터가 필요하다.

이 scheme은 TMR에 비해 하드웨어 오버헤드가 현저히 줄어든다는 장점이 있으나, 3번의 연속 연산으로 인하여 딜레이가 늘어난다는 단점이 있다. 한 번에 수행하는 덧셈에서의 bit수는 약 1/3로 줄었기 때문에 한 cycle의 시간은 줄어들지만, 3번의 반복 연산을 하게 되어 딜레이의 총합은 늘어난다.

2.3 Quadruple Time Redundancy (QTR)

TSTMR은 앞서 언급한 단점 외에 다음과 같은 약점이 있다.

1) 실제로 사용되는 프로세서의 bit-width는 16, 32, 64와 같은 숫자들이기 때문에 3으로 나눌 경우 나누어떨어지지 않아 bit의 efficiency가 떨어진다. 그림 2에 나타나있듯 6-bit 3:1 MUX의 입력이 16이 아닌 18로서, 2-bit의 잉여 bit가 생긴다.
2) Operand가 3분할 되어있으므로 MUX는 3개의 operand를 차례로 선택해야한다. 그런데 3은 2^n이 아니므로 MUX의 control signal이 optimal하지 않다. 예를 들어 00, 01, 10이 3:1 selection을 위해 사용된다면 11은 잉여 조합으로 남게 된다.

위 두 가지 약점을 보완하기 위하여 QTR이 제안되었다. QTR은 n-bit를 3분할이 아닌 4분할을 하고, 4번의 연속 연산을 수행하는 방법이다. 이를 통해 모든 bit가 실질적 연산을 담당하게 되어 bit efficiency가 높아지고, control signal이 optimal하게 된다. 그림 3에는 QTR에 기반해 구현한 16-bit adder의 block diagram이 그려져 있다.

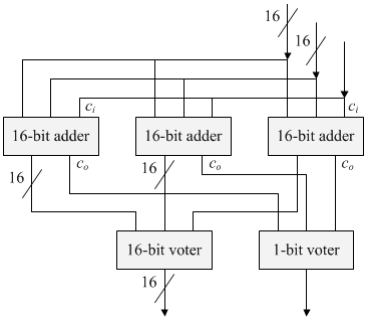

그림 1. TMR 16-bit adder

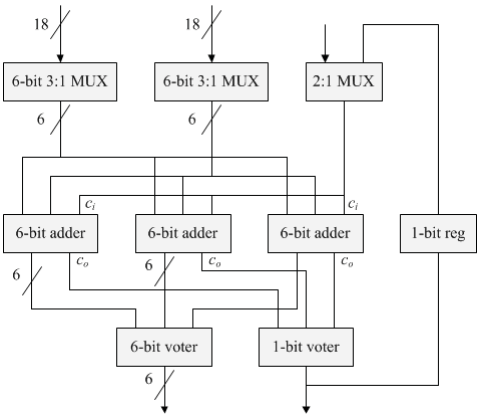

그림 2. TSTMR 16-bit adder

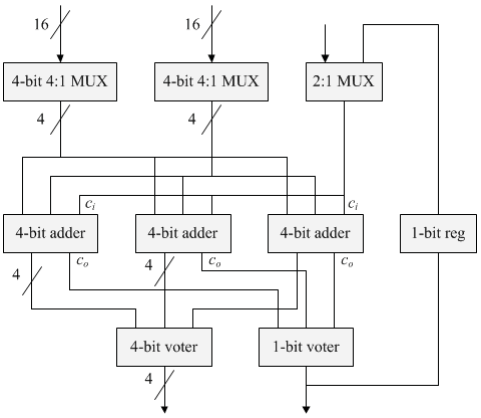

그림 3. QTR 16-bit adder

2.4 Proposed 2^n-tuple Time Redundancy (TR)

QTR은 TSTMR의 두 가지 약점을 효과적으로 보완하였다. 하지만 그 두 단점을 보완할 수 있는 divisor는 4로 한정되어있지 않다. 즉, 모든 2^n의 divisor가 높은 hardware efficiency를 위한 두 개의 조건을 만족시킬 수 있다. 따라서 QTR을 확장하여 2^n-tuple, 예를 들어 8-tuple, 16-tuple TR adder를 만들 수 있다.

Ⅲ. 구현

Section Ⅱ에서 설명한 scheme들로 여러 bit-width의 adder를 구현하고 그 결과를 제시하였다. 그림 4는 equivalent gate count로 하드웨어의 복잡도를 나타낸다. 그림 5는 logic depth로 critical path 딜레이를 나타낸다. 그림 6은 figure of merit [4]로, (equivalent gate count) × (logic depth)^{2}로 계산되며, 아무런 fault-tolerance 기술을 적용하지 않은 nonredundant scheme에 대한 각 scheme의 비율을 표시하였다.

그림 4를 통해, 하드웨어의 복잡도는 bit-width가 커짐에 따라 모든 scheme의 경우에 대해 exponential하게 증가함을 알 수 있다. 그림 5를 보면, logic depth는 bit-width가 커짐에 따라 모든 scheme에 대해 점점 nonredundant case로 수렴한다. 그림 6에서, 각 bit-width에 대해 최소의 figure of merit를 보이는 scheme을 찾음으로써 optimal fault-tolerance scheme을 정할 수 있으며, 이는 표 1에 정리되어있다.

표 1. 각 Bit-Width에 대한 Optimal Scheme

Bit-Width	8	16	32	64
Optimal Scheme	TMR		QTR	
Bit-Width	128	256	512	1024
Optimal Scheme	8-TR		16-TR	

또한 그림 6에서 32-TR의 기울기를 보면, 1024 보다 더 큰 bit-width의 경우에는 32-TR이 가장 optimal해질 것이라고 유추할 수 있다. 따라서 bit-width가 커질수록 더 높은 order의 TR adder가 optimal해진다는 결론을 내릴 수 있다.

Ⅳ. 결론 및 향후 연구 방향

본 논문에서는 프로세서의 bit-width 변화에 따라 여러 가지 fault-tolerant adder scheme의 효용성이 어떻게 변하는지 논하였다. Bit-width가 증가할수록 높은 order의 TR adder가 효용성이 높았으며, 이는 미래형 프로세서에는 현재 사용되는 기술을 그대로 적용하기 보다는 bit-width에 따라 기존 기술을 확장시켜 적용하는 것이 바람직함을 의미한다.

Acknowledgement

이 논문은 2012년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 20120000982)

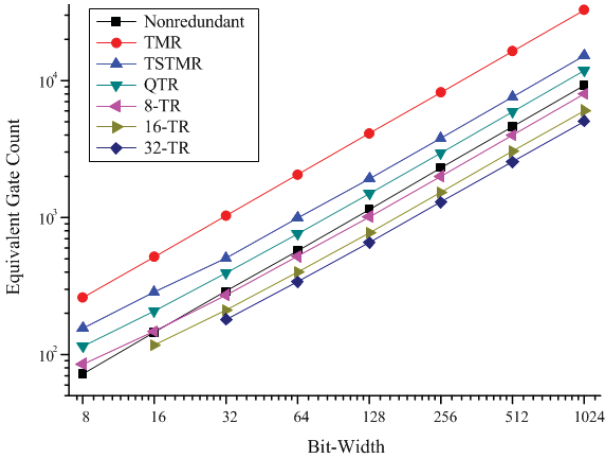

그림 4. Equivalent Gate Count

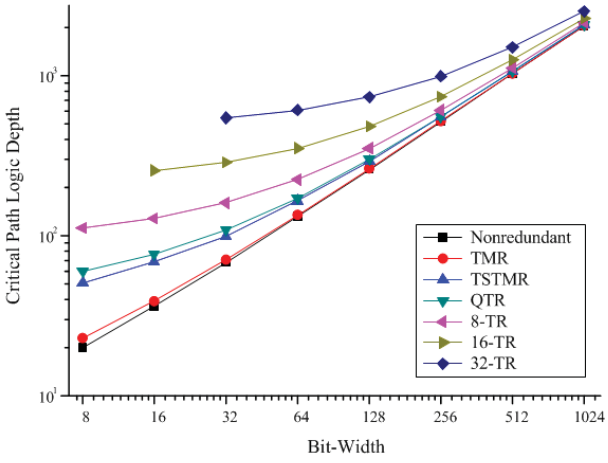

그림 5. Critical Path Logic Depth

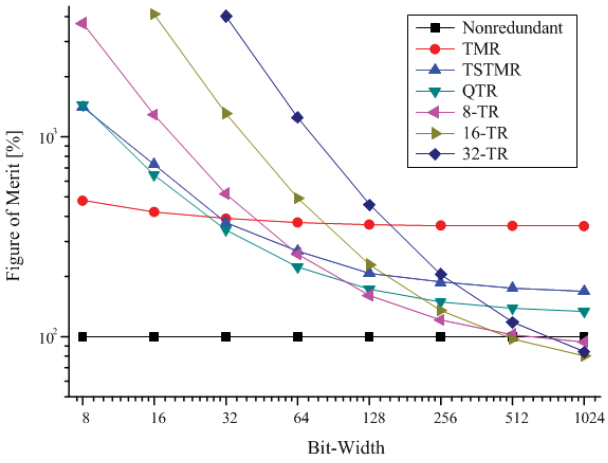

그림 6. Figure of Merit

참고문헌

[1] W. J. Townsend, J. A. Abraham, and E. E. Swartzlander, Jr., "Quadruple Time Redundancy Adders," in *Proc. IEEE International VLSI Test Symposium on Defect and Fault Tolerance in VLSI Systems*, pp. 250-256, Cambridge, MA, USA, Nov. 2003.

[2] R. E. Lyons and W. Vanderkulk, "The Use of Triple-Modular Redundancy to Improve Computer Reliability," *IBM Journal of Research and Development*, vol. 6, p. 200-209, April 1962.

[3] Y.-M. Hsu, "Concurrent Error Correcting Arithmetic Processors," *Ph.D. Dissertation*, The University of Texas at Austin, August 1995.

[4] R. J. Lipton and R. Segewick, "Lower bounds for VLSI," in *ACM Symposium on Theory of Computing*, pp. 300-307, Milwaukee, WI, USA, May 11-13, 1981.