논문 2009-46SD-1-3

임베디드 프로세서의 L2 캐쉬를 위한 오류 정정 회로에 관한 연구

(A Study on an Error Correction Code Circuit for a Level-2 Cache of an Embedded Processor)

김 판 기*, 전 호 윤*, 이 용 석**

(Pan-Ki Kim, Ho-Yoon Jun, and Yong Surk Lee)

요 약

정확한 연산이 필요한 마이크로프로세서에서 소프트 에러에 대한 면밀한 연구들이 진행되었다. 마이크로프로세서 구성원 중에서도 메모리 셀은 소프트 에러에 가장 취약하고, 소프트 에러가 발생했을 때 중요한 정보들과 명령어들을 가지고 있기 때문에 전체 프로세스와 동작에 큰 영향을 미치게 된다. 아키텍처 레벨에서 이러한 소프트 에러를 발견하고 정정하기 위한 방법으로 오류 검출 및 정정 코드가 많이 사용되고 있으며, Itanium, IBM PowerPC G5등의 마이크로프로세서는 Hamming 코드와 Hasio 코드를 L2 캐쉬에 사용하고 있다. 하지만 이러한 연구들은 대형 서버에 국한되었으며 전력 소모에 대한 고려는 되지 않았다. 고집적 저전력 임베디드 마이크로프로세서의 출현과 함께 동작과 문턱 전압이 낮아짐에 따라 임베디드 마이크로프로세서에서도 오류 검출 및 정정 회로의 필요하게 되었다. 본 논문에서는 SimpleScalar-ARM을 이용하여 L2캐쉬의 입출력 데이터를 분석하고, 임베디드 마이크로프로세서에 적합한 32 비트 오류 검출 및 정정 회로의 H-matrix를 제안한다. 그래서 H-spice를 사용하여 modified Hamming 코드와 비교한다. 본 실험을 위해 MiBench 벤치마크 프로그램과 TSMC 0.18um 공정이 사용되었다.

Abstract

Microprocessors, which need correct arithmetic operations, have been the subject of in-depth research in relation to soft errors. Of the existing microprocessor devices, the memory cell is the most vulnerable to soft errors. Moreover, when soft errors emerge in a memory cell, the processes and operations are greatly affected because the memory cell contains important information and instructions about the entire process or operation. Users do not realize that if soft errors go undetected, arithmetic operations and processes will have unexpected outcomes. In the field of architectural design, the tool that is commonly used to detect and correct soft errors is the error check and correction code. The Itanium, IBM PowerPC G5 microprocessors contain Hamming and Hasio codes in their level-2 cache. This research, however, focuses on huge server devices and does not consider power consumption. As the operating and threshold voltage is currently shrinking with the emergence of high-density and low-power embedded microprocessors, there is an urgent need to develop ECC (error check correction) circuits. In this study, the in-output data of the level-2 cache were analyzed using SimpleScalar-ARM, and a 32-bit H-matrix for the level-2 cache of an embedded microprocessor is proposed. From the point of view of power consumption, the proposed H-matrix can be implemented using a schematic editor of Cadence. Therefore, it is comparable to the modified Hamming code, which uses H-spice. The MiBench program and TSMC 0.18 um were used in this study for verification purposes.

Keywords: ECC, low-power ECC, embedded ECC

I. 서 론

α-파티클(particle)과 대기중의 중성자에 의한 소프트 에러가 May와 Woods의 보고에 되었으며, 이러한

* 학생회원, ** 정회원, 연세대학교 전기전자공학부
(Department of Electrical Electronic Eng., Yonsei University)
접수일자: 2007년12월10일, 수정완료일: 2009년1월8일

소프트 에러가 전자 회로의 신호와 데이터를 변형시키는 것을 발견하였다[1]. 현재 반도체 공정과 전압의 변화에 따라 소프트 에러의 발생률이 증가하고 있으며, 마이크로프로세서의 신뢰성을 위협하고 있다. 그 예로 한 실험에서 Alpha 칩을 사용하여 실험을 하였을 때 5000개 이상의 소프트 에러가 검출되는 것으로 보고되었다[2]. 이러한 소프트 에러는 마이크로프로세서의 일시적으로 데이터를 변형시키고, 원하지 않는 오동작을 일으키게 된다.

그중에서도 캐쉬와 메모리는 이러한 소프트 에러에 직접적인 영향을 받게 되고, 공정이 발달할수록 점점 더 그 영향이 커지고 있다[3]. 그림 1에서 SIA (Semiconductor Industry Association)에서 예상하고 있는 공정의 발달에 따른 소프트 에러율의 증가 예상치를 보여주고 있다. 그래서 이런 소프트 에러를 방지할 수 있는 방법들이 대형 서버를 중심으로 연구되었다. 대표적인 예로 Itanium, IBM PowerPC G5등의 상용 마이크로프로세서에서 캐쉬의 데이터를 보호하기 위한 방법을 사용하고 있다. L1 캐쉬에서는 디코딩의 속도가 중요하므로 패리티 비트 체크가 사용되고, L2 캐쉬에서는 Hamming 코드와 Hasio 코드같이 오류 검출뿐만 아니라 오류가 발생한 곳을 정정할 수 있는 코드들이 사용되었다. 그리고 추가적인 멀티쓰레딩을 이용하여 오류를 검출해 내는 방법을 사용하였다[4~5].

임베디드 마이크로프로세서의 반도체 공정 또한 작아지고, 트랜지스터의 집적도는 높아졌다. 그리고 이런 추세와 함께 한정된 에너지를 효율적으로 사용하기 위해서 전력 소모와 회로의 열발산을 줄이기 위한 관심이 높아지면서, 공급전압과 문턱전압(threshold voltage)이 낮아지고 있다. 하지만 이런 변화와 함께 대기중의 방사성 입자에 의한 소프트 에러는 기하급수적으로 증가하고, 임베디드 마이크로프로세서의 설계에 제약 사항으로 작용했다[6]. 그림 2에서 SRAM에서 공급전압이 낮아짐에 따라 소프트 에러가 기하급수적으로 증가하는 것을 보여주고 있다. 그리고 이러한 소프트 에러에 의한 오류를 방지할 수 있는 오류 검출 및 정정 회로가 사용되었다. 하지만 기존의 오류 검출 및 정정 회로들은 서버용에 초점을 맞추어왔으며, 전력소모에 대한 고려는 이루어지지 않았다. 임베디드 마이크로프로세서에서도 오류 검출 및 정정회로를 사용하면서 이 회로의 전력소모를 줄이기 위한 연구들이 진행되었다. 몇몇 연구에서 면적과 지연에 영향을 주지 않으면서 오류 검출

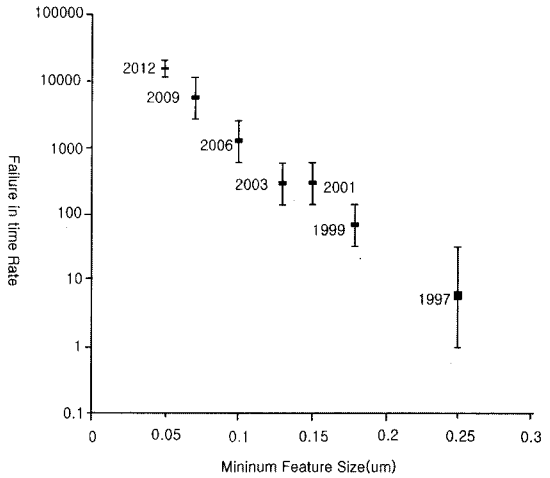


그림 1. SIA의 소프트 에러율 증가 전망
Fig. 1. SIA Road map of SER.

및 정정 회로의 전력 소모를 감소시킬 수 있는 방법들이 제시되었다.

본 연구에서는 H-matrix를 임베디드 마이크로프로세서의 L2 캐쉬에 적합하게 최적화하고 전력 소모를 비교한다. Ⅱ장에서는 임베디드 프로세서의 오류 검출 및 정정 회로의 필요성과 현재 사용되는 오류 검출 및 정정 회로를 소개한다. 그리고 오류 검출 및 정정 회로의 전력 소모 감소를 위한 관련 연구들을 설명한다. 그리고 Ⅲ장에서 SimpleScalar를 이용해서 L2캐쉬의 입출력 데이터 분석과 함께 32비트 H-matrix를 제안한다. Ⅳ장에서 H-spice를 사용하여 실험을 하고 그 결과를 분석한다. 그리고 마지막 Ⅴ장에서 결론을 맺게 된다.

Ⅱ. 임베디드 프로세서의 ECC 회로의 필요성과 전력 소모 감소

1. 임베디드 환경에서의 오류 검출 및 정정 회로의 중요성

휴대 기기의 발달과 함께 한정된 전지의 양을 효율적으로 사용하기 위한 저전력 기술이 개발되고, 공정 기술의 발달로 회로의 디바이스의 면적은 줄어들고 ,고집적화 되었다. 이러한 기술의 발전과 함께 공급 전압과 문턱 전압이 낮아지면서, 외부의 입자에 의한 소프트 에러는 기하급수적으로 증가하고 있다.

일정한 전압에서는 에너지양이 많기 때문에 외부적인 요인에 의하여 전압이 크게 바뀌지 않지만 전압이 낮아졌을 때 소프트 에러는 급격하게 증가된다. 이와 함께 외부적인 입자의 양이 증가할수록 소프트 에러율

은 증가하게 된다. 그래서 새로운 기술이 나올수록 소프트에러에 대한 고려한 설계가 이슈가 될 것으로 전망하고 있다. 특히 마이크로프로세서는 신뢰성 있는 동작이 필요하기 때문에 이를 고려한 설계는 더욱 중요하다.

이러한 경향과 함께 임베디드 프로세서의 급격한 수요에 따라 몇몇 안정성을 요구하는 중요한 프로그램을 다루는 임베디드 프로세서와 실시간으로 작업을 처리해야하는 프로세서에서도 이러한 신뢰성에 대한 필요가 제기되었다. 특히 지상에서 보다 상공에서는 소프트 에러를 일으킬 수 있는 입자의 양은 1000배가 증가하며 이러한 입자량에 따라 소프트 에러는 선형적으로 증가하는 것이 아니라 기하급수적으로 증가한다. 항공장비나 군사장비에 들어가는 임베디드 프로세서의 경우 잘못된 연산이나 실시간으로 연산을 처리하지 못하는 경우 큰 사고로 이어질 수 있다. 군사적 목적이나 항공 장비 외에도 교통 시스템이나 차량에 들어가는 프로세서 등은 오류가 날 경우 사고를 불러일으킬 수 있다.

임베디드 프로세서에서 오류 검출 및 수정이 필요한 다른 이유 중 하나는 임베디드 환경에서 중요시 되는 실시간(real time) 개념 때문이다. 실시간이라는 것은 정해진 시간 안에 주어진 작업을 해야 하지만 소프트에러 등에 의해서 주어진 시간 안에 일을 처리 하지 못하는 일뿐만 아니라 심지어 재부팅이 필요한 상황에 이르게 된다. 이러한 실시간으로 임베디드 프로세서가 작업을 끝내지 못하는 상황은 공정이 발달할수록 더욱 더 빈번하게 발생하게 될 것으로 전망되어지고 있다.

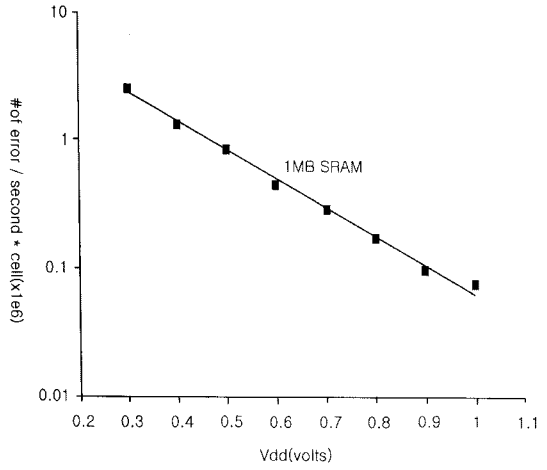


그림 2. 1MB SRAM에서의 공급전압과 소프트 에러율
Fig. 2. SER and Supply voltage in 1MB SRAM.

심지어 유비쿼터스 세상과 함께 모든 전자 기기에 프로세서가 내장되면서 안정성을 중시하는 특수한 임베디드 프로세서 외에도 일반적으로 사용되는 임베디드 프로세서에서의 소프트 에러에 대한 관심이 점차 높아지고 있다. 통신, 사무 등의 장비들이 개인에게 중요한 업무 도중에 재부팅이나 데이터가 잘 못되어 버리는 일들은 전자 제품의 신뢰성을 떨어트리고 사용자의 금전적 시간적 소모를 가져오게 된다. 이러한 소프트 에러에 의한 위험성과 손실을 방지하기 위해서는 오류를 발견하고 이 정보를 바르게 정정 할 수 있는 추가적인 방법들이 사용되어야한다.

2. 프로세서의 ECC 회로의 필요성

소프트 에러는 지구 외부에서부터 이온화된 입자 등에 의해서 커패시터의 값이 의도하지 않은 상태로 변환되어지는 것을 말한다. 이온화는 자유전자를 생성하게 되고 이것은 저장 노드의 축전 전압을 변화하게 만들고 잘못된 정보를 가지게 된다.

지금 현재 대부분의 고속의 저장 매체는 데이터 정보를 저장하고 있으며 SRAM도 예외는 아니다. 이러한 소프트 에러가 발생했을 때 오류가 발견되지 않는다면 연산이나 프로세스의 과정이 의도하지 않은 결과를 발생시키게 되며 이러한 결과는 치명적인 결함이 될 수 있다. 이 소프트 에러를 발견하는지는 전적으로 프로세서가 이러한 오류 검출 및 정정의 기능을 지원할 것인가에 의해 좌우된다.

3. 현재 상용 프로세서의 ECC회로

신뢰성을 가진 프로세서를 만들기 위해서 현재 많은 프로세서가 캐쉬에 ECC회로를 장착하고 있다. 대표적인 예로 Power4[5]와 Itanium[4] 프로세서들이 있다. Itanium 프로세서는 Intel IA-64 구조를 처음으로 구현한 마이크로프로세서로 L1 캐쉬는 패리티 비트 체크를 통해서 소프트 에러를 체크하게 된다. 이 구조는 바이트(byte)당 한 개의 패리티 비트를 가지고 있으며, 명령어 수행이 끝나기 전에 한 싸이클 만에 에러가 가지고 있는지를 알아낸다. 그리고 만약 오류가 발견되었을 때, 정지를 시키고 기다리게 된다. L2 통합 캐쉬는 데이터 당 8비트 오류 정정 회로에 의해서 매 64비트의 데이터가 수행되어진다. 이것은 SECDED(Single Error Correction Double Error Detection)을 보장해준다. 그러나 프로세스 중에는 8, 16, 32비트 단위의 캐쉬 읽기-

수정-쓰기의 과정이 필요한 경우가 있다. 이를 방지하기 위해 32비트의 오류 검출 및 정정 회로를 지원해주는데 이러한 32비트 오류 검출 및 정정을 위해서는 총 7비트의 오류 체크 비트가 필요하게 된다.

Power4 프로세서는 1.3GHz로 동작하며 125 Gbytes/s의 속도로 데이터가 L2캐쉬에서 프로세서 코어로 넘겨준다. Power4 칩은 혁신적으로 L2의 Miss를 다루게 된다. 실험에 의해서 이 프로세서에서 5,000개 이상의 소프트 에러가 발생하게 되었다. 이것을 완화하기 위해서 Power4는 오류 정정 회로를 지원하게 된다. 이곳에서 사용되는 오류 정정 기법은 메모리에서 표준적으로 사용되고 있는 Hamming ECC 기법을 사용한다. 이 프로세서 또한 SECDED(Single Error Correction Double Error Detection)을 보장해준다.

ARM은 최초로 임베디드 프로세서에 소프트 에러를 고려한 코어를 선보였다[6]. 대표적으로 ARM1156T2(F)-S 코어는 캐쉬를 소프트 에러에서 보호할 수 있는 오류 검출 및 정정 기술과 회로를 추가하였다. 이 마이크로프로세서의 캐쉬는 태그에서는 오버헤드가 가장 작은 패리티 체크를 통한 오류 발견을 제공해준다. ARM Cortex-R4[7]는 차세대 휴대폰, 하드디스크 드라이브, 프린터 및 자동차 설계를 위해 개발된 새로운 프로세서이다. 그리고 이러한 오류 검출 및 정정 회로를 지원한다. 이 마이크로프로세서는 L1 캐쉬와 태그는 패리티 체크를 통해서 오류 검출을 수행하고, L2 캐쉬는 Hamming 코드, 웨이트(weight) 기반 등의 코드를 통해 오류 정정을 선택적으로 지원해 주고 있다.

4. 오류 검출 및 정정 회로의 전력소모 감소

임베디드 메모리와 캐쉬에서 오류 검출 및 정정 회로의 사용이 늘어나면서 이 회로의 전력소모에 대한 연구가 활발히 진행되었다. Favalli는 오류 검출 및 정정회로의 전력 소모를 줄이기 위해서 처음으로 체크 트리의 확률적인 방법을 사용하였다[8]. 입력이 두개인 게이트의 체크 트리에서 입력의 순서를 변형시킴으로써 지연이나 회로의 면적이 늘어나지 않으면서 전력을 소모시킬 수 있는 방법을 제안하였다. Mohanram은 오류 검출 및 정정 회로의 전력 소모를 줄이기 위해서 공간지역성을 이용하여 체크 트리의 입력 데이터를 최적화하는 알고리즘을 제안하였다[9]. 그리고 Ghosh는 비트 변화를 최소화하는 데 그 목적을 두고, 각각의 프로그램에 오류 검출 및 정정 회로를 최적화하는 두 가지 알고리즘을

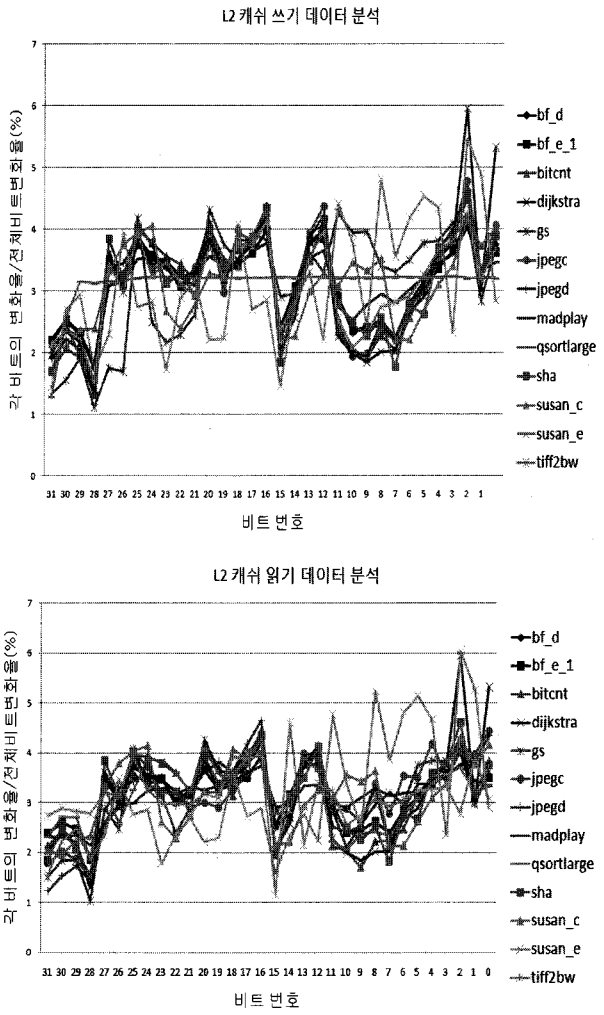


그림 3. 임베디드 프로그램의 캐쉬 읽기 쓰기 분석
Fig. 3. Analysis of bit-changing rate during cache-read and cache-write.

제안하고, 메모리에서 전력소모를 중심으로 오류 검출 및 정정 회로를 프로그램에 적합하게 H-matrix를 구현하였다[10]. SPEC95와 2000 벤치마크 프로그램을 사용하여 실험한 결과 Hamming 코드 보다는 최소 5%에서 최대 41%를, Hasio 코드 보다는 최대 27% 전력을 줄일 수 있었다. 이런 H-matrix의 최적화가 가능했던 것은 H-matrix를 구성하는 지에 따라 오류 검출 및 정정 회로의 특성이 달라지기 때문이다.

예를 들어 (72,64) Odd-weight-column코드에서 체크 비트 8개가 필요며 H-matrix를 보았을 때 열에서 1이 하나만 가지고 있게 된다. 그리고 1의 개수가 3개인 열은 C_3^8인 56개가 필요하고 1의 개수가 5개인 열이 8개가 필요하다. 여기에서 중요한 점은 1의 개수가 5개인 열은 C_5^8인 56개 중에서 8개를 고르게 되어 있다. 이중

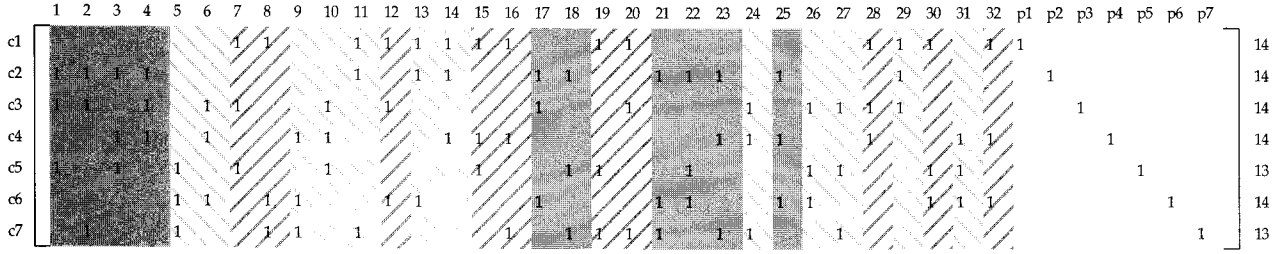


그림 4. 제안하는 32비트 H-matrix
Fig. 4. Proposed 32bit H-matrix.

어떤 것을 선택하느냐에 따라 전력소모와 오류 검출 및 정정 회로의 특성이 달라진다.

위 연구에서는 오류 검출 및 수정 회로의 전력 소모를 감소하기 위한 많은 연구들이 이루어졌지만 대부분의 연구는 메모리에 초점이 맞추어져있었다. 그리고 프로그램 마다 H-matrix의 최적화를 시도하였다. 하지만 범용으로 이러한 프로그램으로 최적화된 회로를 사용할 수 없다.

본 연구에서는 임베디드 프로세서에 주로 사용되는 프로그램의 캐쉬 접근의 관계성을 일반화하고 공통적으로 사용될 수 있는 적합한 H-matrix를 제안한다. 임베디드 프로세서에서 주로 사용되는 멀티미디어와 수학 연산 프로그램의 경우 상위 비트의 변화보다 하위 비트의 변화가 더 많다. 캐쉬의 경우에는 메모리보다 이러한 데이터의 지역적과 시간적 관계성이 두드러지게 나타나게 된다. 특히 명령어의 경우 특정 루프 등을 통해서 이러한 특징은 더 잘 나타난다. 이러한 특성을 잘 활용해서 H-matrix를 작성하고 오류 검출 및 수정 회로를 설계할 수 있다.

Ⅲ. 제안하는 ECC회로의 최적화

1. 임베디드 데이터의 특징 분석

본 연구에서는 Simplescalar-ARM을 통해서 MiBench가 L2 캐쉬에 읽기와 쓰기를 하는 동안의 데이터의 비트의 변화를 분석한다. 32비트 오류 검출 및 정정 회로를 만들 것이기 때문에 32비트 씩 데이터를 분석한다. 벤치마크 프로그램이 각각 수행되는 시간과 데이터를 필요로 하는 양이 각각 틀리기 때문에 단순히 L2 캐쉬에 접근하는 수만을 더하게 되면, 데이터 접근이 많은 프로그램에 의해서 데이터의 비트 변화를 차지하게 된다. 따라서 다음과 같은 수식에 의해서 그래프로 표현한다. 이 수식에서 α는 각 프로그램의 n번째 비트의 변화율이고, β는 n번째 비트의 변화수이다.

$$\alpha_n = \frac{\beta_n}{\sum_{1}^{32}\beta_n} \times 100 \quad (1)$$

데이터 분석을 그림 3에서 보여주고 있다. 캐쉬의 읽기 시에 MSB(Most Significant Bit)를 31번째 비트라고 했을 때 31~28 번째 비트와 15 번째 비트, 14번 째 비트가 비트의 변화의 수가 작다. 반면 2, 19, 24, 11비트의 순으로 비트의 변화가 많다. 그리고 캐쉬의 쓰기의 경우 역시 31~28 번째 비트의 변화가 적으며, 다음으로 15번째와 7번째의 비트 변화가 적다. 반면 0, 27, 2, 21, 3, 22번째의 비트 변화가 많다. 이러한 비트변화의 특징은 임베디드 프로그램의 데이터가 연산을 위해 사용되는 데이터가 많으며, 연산을 위한 데이터는 상위비트가 하위비트보다 비트 변화가 작기 때문이다. 15비트의 비트 변화가 작은 것은 그림 파일들의 경우 16비트 단위로 연산되어지는 프로그램들이 존재하기 때문이다.

오류 검출 및 정정 회로에서 면적을 고려하여 체크 비트 생성기는 읽기와 쓰기 시 동시에 사용된다. 따라서 읽기와 쓰기 시의 비트 변화율이 틀리다면, 공통으로 체크 비트 생성기를 사용했을 때 최적화를 시킬 수 없다. 하지만 그림 3에서와 같이 L2 캐쉬의 읽기와 쓰기 동안 비트의 변화 수가 작은 비트와 많은 비트가 비슷하다는 것을 알 수 있다.

2. 제안한 오류 검출 및 정정 회로의 H-matrix

임베디드 프로세서의 주요 작업은 지도의 좌표계산이나 물체의 인식, 그림의 변형 등의 수학적인 연산이다. 이 수학적인 연산에 쓰이는 대부분의 데이터는 상위비트의 변화보다 하위비트의 변화가 더 많다. 위 분석에서 임베디드 프로세서에서 주로 변화가 일어나는 비트를 알아보기 위해 SimpleScalar-ARM을 통해서 캐

쉬에서 읽혀지거나 쓰이는 데이터의 변화를 앞 절에서 설명하였다. 그 결과 그림 3에서처럼 읽혀지거나 쓰일 때 특정 비트의 변위가 작게 나타나는 것을 알 수가 있다. 대표적으로 MSB가 31번째 비트이라고 가정했을 때 31~28번째 비트의 상위비트는 변화가 작다.

본 연구에서 제안하는 H-matrix는 트리에서의 비트 변화를 최소화하는 데 그 목적이 있다. 비트의 변화를 계산하기 위해서 다음과 같은 수식을 이용한다. 모든 신호는 {0, 1} 집합에 포함된다고 했을 때 x노드의 신호가 1일 때의 확률을 $P_s(x)$로 표현하고, 비트 변화의 확률을 $P_t(x)$라고 가정한다. 그 때 이들의 관계는 다음 수식(2)와 같다. 그리고 본 연구에서는 전체 트리의 비트의 변화율을 계산하기 위해서 그림 5와 같은 트리 구조에서 각 노드의 비트의 변화율을 수식(3)과 같이 합한다.

$$P_t(x_i) = 2 \times P_s(x_i) \times P_s(\overline{x_i}) = 2 \times P_s(x_i) \times (1 - P_s(x_i)) \quad (2)$$

$$P_t(total) = P_t(y_1) + P_t(y_2) + P_t(y_3) \quad (3)$$

두개의 입력을 가지는 게이트에서 {A, B}와 {B, A}와 같이 입력 순서가 바뀌어도 같은 트리이다. 따라서 이런 트리는 배제하여 입력 함수의 할당을 고려한다. 예를 들어 그림 5에서와 입력 함수의 조합은 4!개가 존재하지만 토폴로지가 같지 않은 입력 함수의 할당 방법은 3개가 존재할 수 있다. 확률적인 파워 예측 모델에서 이러한 각 노드의 비트 변화 확률이 가장 작은 입력 신호를 선택한다. 각 비트의 변화율은 앞 절에서의 분석을 통해서 알 수가 있고, 이를 토대로 전력 소모를 줄일 수 있는 H-matrix를 설계한다.

제안되는 오류 검출 및 정정 회로에서는 체크를 구성할 때 변화가 많은 그룹과 변화가, 작은 그룹을 먼저 나누어 체크를 구성한다. 이를 위해서 32비트 오류 정정

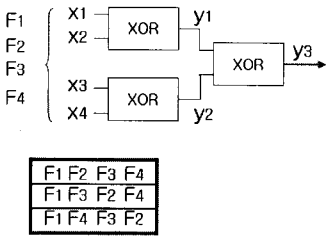
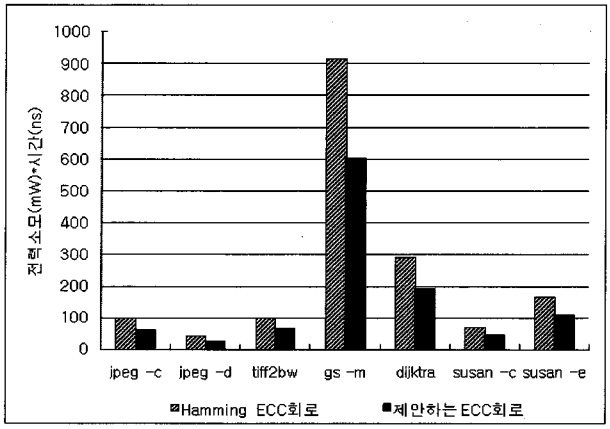


그림 5. XOR 트리의 입력함수 구성
Fig. 5. Combination of Input fuction in XOR tree.

회로에서 선택할 수 있는 H-matrix의 열은 35개이다. 이들 중 32개를 선택하는 방법에 따라 오류정정회로의 특성이 달라진다. 첫 번째 체크의 구성원에서 우선적으로 변화가 많은 그룹을 먼저 선택하기 위해서 체크 1이 1로 되어 있는 열을 우선적으로 변화가 많은 비트위치에 배정을 한다. 그리고 체크 2가 1로 되어 있는 열을 우선적으로 변화가 작은 비트의 위치에 배정한다. 체크 1과 체크 2가 동시에 1로 되어있는 열은 5개이다. 이들은 변화가 중간 정도가 되는 비트들에 배열을 한다. 위 방법으로 배치한 H-matrix가 그림 4와 같다.

Ⅳ. 실 험

본 연구에서는 임베디드 프로세서에서 전력소모가 적은 적합한 2-레벨 캐쉬의 오류 정정 회로구현하고 이 회로를 Modified Hamming 코드 회로와 비교를 통해 검증한다. 본 연구에서는 현재 임베디드 프로세서로 가



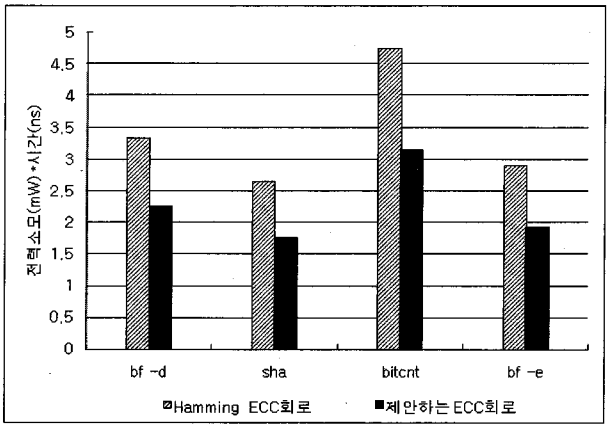


그림 6. Modified Hamming 오류 검출 및 정정 회로와 전력 소모 비교
Fig. 6. Comparison of power consumption with Modified Hamming ECC.

장 널리 사용되고 있는 ARM 마이크로프로세서를 모델로 하여 실험을 한다.

ARM 모델에서 시뮬레이션 하기 위해서 SimpleScalar-ARM을 이용한다. SimpleScalar에서는 ARM 프로세서를 모델로 하여 실제적인 캐쉬에 접근 데이터를 얻어 낼 수 있다. SimpleScalar 코드에서 READ_MEM* 매크로는 주소 값을 이용하여 캐쉬의 물리적인 접근 값을 얻어낼 수 있다. 실험에서 SimpleScalar-ARM에서 설정된 선택사항을 그대로 사용한다. 표 7-1에서 이 선택사항을 정리하고 보여준다. 그리고 CISC 구조를 위해 inorder 옵션을 선택한다.

일반적으로 마이크로프로세서의 L2 캐쉬에서는 Modified Hamming 코드 회로가 일반적으로 사용되어진다. 이 Modified Hamming 코드로 만들어진 오류 검출 및 정정 회로와 제안한 오류 검출 및 정정 회로의 전력 비교를 위해서 Synopsis사의 Star-Hspice를 사용하여 평균 전력 소모를 비교한다. 공정으로는 TSMC 0.18um 라이브러리를 사용하였으며 1.8V 전압을 사용하였다.

기존 연구에서 오류 검출 및 정정 회로의 최적화에서 면적과 지연에 영향을 주지 않고 최적화를 하였다. 본 연구에서는 이러한 방법을 임베디드 마이크로프로세서에 적합하게 하였다. 그래서 면적과 지연의 비교는 수행하지 않고 전력 비교만을 수행한다. Star H-spice에서 전력 소모를 측정했을 때의 결과는 그림 6과 같다. jpeg -c 와 tiff2bw, gs-m가 Modified Hamming 오류 검출및 정정 회로와 비교했을 때 전력 소모가 각각 36.4 %, 34%, 34.1 %로 가장 많이 줄어들었다. 반면 susan -c 벤치마크 프로그램이 가장 작게 줄어들어서 32.8%가 줄어들었다.

표 1. SimpleScalar-ARM의 기본 설정
Table 1. Setting of SimpleScalar-ARM.

Parameter	Option
Fetch/decode/commit width	4
IntALU/IntMUL/FpALU/FpMUL/Mem	4/1/2/1/2
Local register file size(per FU)	32
Issue queue and load/store queue	8-entry issue queue per FU, 16-entry load/store queue
register update unit (RUU) size	16
load/store queue (LSQ) size	8
I-cache L1	16KB direct-mapped, 32-byte lines, 1-cycle latency, LRU
D-cache L1	4-way set associative 16KB direct-mapped, 32-byte lines, 1-cycle latency, LRU
I/D-cache L2	4-way set associative 256KB, Unified L2-cache, 64-byte lines, 1-cycle latency

기존 연구에서 각각의 프로그램마다 메모리에 최적화했을 때 최소 5%에서 최대 41% 전력 소모에 낮춘 것에 비해서는 전력 소모가 높지만, 임베디드프로그램을 수행하고 있는 임베디드 마이크로프로세서의 L2캐쉬에서 공통적으로 사용할 수 있는 최적화 오류 검출 및 정정 회로를 만들었을 때 평균적으로 33.6%의 전력을 감소시킬 수 있었다.

V. 결 론

반도체 공정 기술의 발달로 칩의 면적이 줄어들고, 저전력을 위해서 공급 전압의 감소하면서 전자 회로는 소프트 에러에 의한 오류에 대한 위협에 놓이게 되었다. 이와 함께 신뢰성 있는 마이크로프로세서를 위해서 소프트 에러에 의한 오류를 효과적으로 줄일 수 있는 방법들에 대한 필요성이 제기되었다. 대형 서버를 중심으로 이러한 마이크로프로세서의 소프트 에러에 의한 일시적인 오류를 방지하기 위해서 방법들이 사용되고 연구되었다. 대표인 방법으로 추가적인 프로그램의 연산을 통한 오류 검출을 하거나, 특정 회로를 만들어서 소프트 에러에 의한 오류를 방지했다. 하지만 이런 방법들은 자원의 소모도 많이 요구되고, 전력 소모를 줄이는 것에는 초점을 두지 않았다.

유비쿼터스 시대와 함께 임베디드 마이크로프로세서가 널리 쓰이면서 몇몇 오류가 발생 시 심각한 사고를 일으킬 수 있거나, 상공에서 사용되는 항공장비에서 쓰이는 임베디드 프로세서를 중심으로 소프트 에러를 고려한 설계가 필요하게 되었다. 그래서 ARM사에서는 최초로 임베디드 프로세서에서 소프트 에러를 방지할 수 있는 패리티 체크와 오류 검출 및 정정 코드들을 지원해주게 되었다. 이런 추세와 함께 몇몇 연구는 오류 검출 및 정정 회로에서 메모리에 H-matrix를 최적화함으로써 전력 소모를 줄일 수 있는 가능성을 보였다. 하지만 이들 연구에서는 각각의 프로그램마다 최적화할 수 있는 방법에 대해서만 서술을 하고 공통적으로 사용될 수 있는 회로에 대해서는 연구하지 않았다. 그리고 메모리를 대상으로 했지만 캐쉬의 오류 검출 및 정정에서 전력 소모를 줄일 수 있는 방법에 대해서는 연구를 하지 않았다.

본 연구에서는 임베디드 마이크로프로세서의 L2 캐쉬의 입출력 데이터를 SimpleScalar-ARM에서 MiBench를 구동하여 분석하였다. 분석 결과 대체로 상위 비트의 변화가 작았으며, 이와 함께 특정 비트의 변화가 두드러지게 작거나 크게 나타났다. 이러한 특성을 이용하여 임베디드 마이크로프로세서의 L2 캐쉬에서 전력 소모를 줄일 수 있는 오류 검출 및 정정회로를 제안하였다. 비트의 변화를 최소화 할 수 있게 32비트 H-matrix를 설계하고, Star H-spice를 이용하여 전력 소모를 분석하였다. 그래서 전력 소모 분석 결과 Modified Hamming 코드 회로와 비교했을 때 평균적으로 33%정도의 전력 소모를 감소시킬 수 있었다. 이 연구를 통해서 임베디드 프로그램의 데이터의 특성에 맞게 오류 검출 및 정정 회로를 설계할 수 있으며, 사용전력 소모를 감소시킬 수 있음을 보였다. 본 연구에서는 비트의 변화율에만 관심을 두었다. 앞으로 캐쉬의 시간 지역성과 공간 지역성을 활용하는 연구를 통해 오류 검출 및 정정 회로의 전력 소모를 더욱 감소시킬 수 있을 것으로 기대한다.

참 고 문 헌

[1] T. C. May, M. H. Woods, "Alpha-particle-induced soft errors in dynamic memories," Electron Devices, IEEE Transactions on , vol.26, no.1, pp. 2-9, Jan 1979.

[2] N. Cohen, T.S. Sriram , N. Leland , D. Moyer, S. Butler,R. Flatley, "Soft error considerations for deep-submicron CMOS circuit applications," Electron Devices Meeting, 1999. IEDM Technical Digest. International, Washington, DC, pp. 315-318, USA, 1999.

[3] N. Seifert, D. Moyer, N. Leland, R. Hokinson, "Historical trend in alpha-particle induced soft error rates of the AlphaTM microprocessor," Reliability Physics Symposium 2001, Proceedings. 39th Annual. 2001 IEEE International , pp. 259-265, 2001.

[4] Nhon Quach, "High availability and reliability in the itanium processor," IEEE Micro , vol.20, no.5, pp. 61-69, Sept.-Oct. 2000.

[5] D. C. Bossen, J. M. Tendler, and K. Reick. "POWER4 system design for high reliability." IEEE Micro, Vol.22, no.2, pp.16-24, March-April 2002.

[6] Phelan, R., "Addressing Soft Errors in ARM Core-based SoC. Dec," ARM Ltd, 2003.

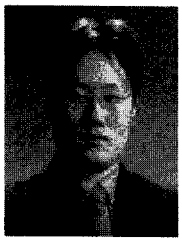
[7] John Penton , Shareef Jalloq, "Cortex-R4 : A mid-range processor for deeply-embedded applications,",ARM Ltd, May 2006.

[8] K. Favalli, C. Metra,"Design of Low-Power Cmos Two-rail Checkers," Journal of Microelectronics Systems Integration, vol. 5, no. 2, pp.101-110, 1997.

[9] K. Mohanram , N.A. Touba , "Input ordering in concurrent checkers to reduce power consumption," Defect and Fault Tolerance in VLSI Systems, DFT 2002. Proceedings. 17th IEEE International Symposium on , pp. 87-95, 2002.

[10] S. Ghosh, S. Basu, N.A. Touba, "Reducing power consumption in memory ECC checkers," Test Conference, 2004. Proceedings. ITC 2004. International , pp. 1322-1331, 26-28 Oct. 2004.

저 자 소 개



김 판 기(학생회원)
2006년 2월 숭실대학교 전자공학과 학사
2008년 2월 연세대학교 전기전자공학과 석사
2008 년 2월 삼성전자 연구원

<주관심분야 : 저전력 프로세서, 프로세서 테스트>

이 용 석(정회원)
1973년 2월 연세대학교 전기공학과 학사
1977년 2월 University of Michigan, Ann Arbor 석사
1981년 2월 University of Michigan, Ann Arbor 박사
1993년~현재 연세대학교 전기전자공학과 교수
<주관심분야 : 마이크로프로세서, 네트워크 프로세서, 암호화 프로세서, SoC>

전 호 윤(학생회원)
2002년 2월 홍익대학교 컴퓨터정보통신학부 학사 취득
2004년 2월 홍익대학교 컴퓨터공학과 석사 취득
2006년 3월 연세대학교 전기전자공학과 박사과정
<주관심분야 : 마이크로프로세서, 저전력 캐쉬, SoC>