내장형 프로세서를 위한 동적 분기 예측기의 최적화 구성

김성은[O] 이영림 유혁
고려대학교 컴퓨터학과
{sekim[O], yrlee, hxy}@os.korea.ac.kr

Finding Optimal Configuration of Dynamic Branch Predictors for Embedded Processors

SungEun Kim[O] Young-Rim Lee Hyuck Yoo
Dept. of Computer Science & Engineering, Korea University

요 약

내장형 시스템에 보다 강력한 성능이 요구됨에 따라 내장형 마이크로 프로세서는 보다 깊은 파이프라인을 채택하고 있다. 따라서, 내장형 마이크로 프로세서는 보다 정확한 분기 예측기를 필요로 하고 있다. 이러한 상황에서 분기 예특기의 구조, 성능 및 전력 소모와 전체 시스템의 전력 소모 사이의 trade-off를 분석하는 것은 매우 중요하다. 내장형 환경에서 시스템의 전력 소모는 설계 시 매우 중요하게 고려되어야 한다. 특히 내장형 시스템의 요구사항은 동작할 응용 프로그램에 의하여 규정되고, 전력 소모도 응용 프로그램의 구조와 강하게 연관되어 있다. 본 논문의 목표는 내장형 환경에서 성능-전력 공간에서 분기 예측기를 분석하는 기법을 제시하는 것에 있다. 이를 통하여, 분기 예측기 테이블의 성능-전력을 고려한 최적화된 크기를 찾을 수 있다. 이러한 목표는 수학적 모델링을 통한 정량적 예측의 수행 및 시뮬레이션 결과와의 비교를 통한 수학적 모델링의 검증의 과정을 통하여 이루어진다. 결과는 우리의 수학적 모델이 성능-전력 공간에서 분기 예측기 테이블의 최적화된 크기 결정의 해법을 제공하고 있음을 보여주고 있다.

1. 서 론

저전력 소모는 현재 마이크로 프로세서 설계에 있어 매우 중요한 제약사항이다. 특히, 내장형 장치에 이용되는 상품들의 경우 저전력 소모는 가장 중요한 요구사항 중 하나이다. 최근 몇 년간 내장형 장치의 영역은 휴대전화에서 공장자동화기기에 이르기까지 매우 확장되었다. 이러한 내장형 장치의 발전 과정에서 다양한 기능과 성능이 내장형 장치에 추가되어왔는데, 이는 데이터 처리 속도 향상과 같은 내장형 마이크로 프로세서의 발전에 의하여 가능해졌다. 그러나, 이러한 내장형 프로세서의 성능 향상에 따라 설계자에게 전력 소모 문제에 대한 고려가 필수적으로 요청되고 있다. 그러므로, 데이터 처리 속도 향상과 같은 성능의 부분과 전력 소모의 부분의 Trade-off 관계는 설계 시 필수적으로 고려되어야 한다.

내장형 장치에 보다 강력한 성능이 요구되어짐에 따라, 내장형 프로세서는 보다 깊은 파이프라인을 채택하고 있다. 예를 들어, Intel Xscale 마이크로 프로세서는 7~8 간계의 파이프와 128개의 entry를 지니는 BTB를 가지고 있다 [1]. 이와 같은 상황에서, 정확하게 예측되지 못한 분기에 따른 해저드는 상당한 량의 프로세서 동작 시간을 낭비시키며 Instruction Level Parallelism (ILP)를 저해하는 중요 요인으로 지적되고 있다. 그러므로, 내장형 프로세서에 있어 정확한 분기 예측의 중요성은 점점 더 중요해지고 있다.

분기 예측 기술은 효율적인 분기 동작을 위하여 현재의 프로세서들에 의하여 채택되고 있다. 분기 예측은 효율적인 ILP를 구현하기 위하여 이용하는 기술 중의 하나로 자리잡고 있다. 분기 예측 중 동적 분기 예측 기술은 가장 좋은 성능을 보장하고 있다. 동적 분기 예측은 분기 예측 테이블과 같은 메모리 구조를 이용하고 있는데, 테이블은 매 클럭 주기 마다 접근되어 이전 분기 정보를 이용하여 예측을 수행한다. 그러나, 이러한 메모리 접근은 전력 비용을 수반한다. 즉, 메모리 접근은 각각의 메모리 셀의 캐패시터의 충전과 방전을 수반하며, 이에 따라 누설 전류가 발생하여 전력 소모를 증가시킨다. 현재의 마이크로 프로세서의 분기 예측기가 사용하는 전력 소모는 점점 증가하고 있으며, 전체 프로세서 사용 전력의 약 10%까지 사용하는 것으로 보고되고 있다 [2]. 이러한 이유로 인하여, 분기 예측기 구조, 프로세서 성능, 분기 예측기의 전력 소모 및 전체 시스템의 전력 소모 사이의 trade-off에 대한 분석은 매우 중요하다.

분기 예측기는 분기 예측기 테이블의 크기를 줄이거나 분기 예측기의 메모리 구조를 변경하는 것, 혹은 분기 예측기 예측 (Branch Predictor Prediction, BPP) [3] 등의 저전력 알고리즘의 적용을 통하여 저전력 소모에 적합한 구조로 구성될 수 있다. 이러한 방법들 중에서 분기 예측기 테이블의 크기를 조정하는

것이 성능 및 전력 소모에 가장 큰 영향을 준다. 분기 예측기 테이블의 크기를 증가시키면 분기 예측의 정확성을 향상시켜 분기 예측 miss-penalty를 감소시킨다. Miss-penalty의 감소는 전체 시스템의 CPI를 감소시켜 전체 프로그램을 수행하는데 필요한 전체 Cycle 수를 감소시킨다. 그러나, 증가된 분기 예측기 테이블의 크기는 분기 예측기 자체에 의한 전력 소모를 증가시킨다. 특히, 분기 예측기 테이블은 SRAM 셀로 구성되어 있기 때문에 분기 예측기에 의한 전력 소모는 전체 분기 예측기 테이블 크기에 비례하다.

내장형 환경에서 시스템에 대한 요구사항은 응용에 특화되어 있다. 그러므로, 고성능의 범용 프로세서와는 달리, 내장형 프로세서는 풍부한 자원을 기본적으로 가질 필요는 없다. 어떤 의미에서는 내장형 프로세서의 풍부한 자원은 전력 소모의 관점에서 볼 때 전력 낭비의 요인으로 작용할 수도 있다. 일반적으로 내장형 장치는 배터리에 의한 제한된 작동 시간의 특성을 지닌다. 제한된 작동 시간은 내장형 장치의 중요한 특성 중의 하나이다. 내장형 장치에 있어 전력 소모는 특정 응용 프로그램의 특성 및 구조와 밀접한 관계가 있다. 그러므로, 특정 응용에 특화된 목적에 부합하도록 마이크로 프로세서를 설계하는 것은 매우 중요하다.

본 논문의 목표는 내장형 환경에서 성능-전력 공간에서 분기 예측기를 분석하는 기법을 제시하고 최적화된 분기 예측기 테이블의 크기를 찾는 것에 있다. 본 논문은 다음과 같은 장들로 구성된다. 2장에서는 전력 소모의 관점에서 분기 예측기에 대한 관련 연구를 정리할 것이다. 3장에서는 분기 예측기의 성능-전력소모 특성을 수학적 모델링을 통하여 분석 할 것이다. 4장에서는 시뮬레이션의 환경 및 결과를 서술하고 결과와 우리의 모델링에 대한 비교 분석을 수행할 것이다. 5장에서는 우리의 작업에 대한 결론 및 요약을 수행할 것이다. 이 논문을 통하여 저전력 내장형 마이크로 프로세서의 설계에 있어 유용한 지표를 제공되기를 바란다.

2. 관련 연구

[9] 는 분기 예측기의 구성에 따른 전력-성능간의 trade-off에 대하여 기술하였다. 이 논문에서는 보다 정확한 분기 예측기에 의하여 전체 시스템의 실행 시간을 감소시킬 수 있다면, 보다 복잡한 분기 예측기에 의한 전력 소모의 증대는 보다 가치있는 것이라고 결론지었다. [10] 은 프로파일 기법을 이용하여 전역 예측 (Gselect [11])이나 지역 예측 (Bimodal [4])중 어떤 것이 적합할지 결정하였다. 이러한 방법을 통하여 이종 분기 예측기에 이용되는 meta-예측기를 제거하였다 [5].

[12] 는 프로파일링 및 동적인 예측기 자원의 조정을 통하여 전력 소모를 감소시키는 방법에 대하여 제안하였다. 이는 예측 목표 버퍼 (Branch Target Buffer, BTB)의 크기를 조절함으로써 이루어졌다.

[13] 은 힌트 명령을 컴파일 시 포함시킴으로써 분기 예측기에 대한 불필요한 접근을 차단시켜 상당량의 전력 소모를 감소시켰다. [14]는 컴파일러의 도움을 통하여 응용에 특화된 내장형 프로세서에 대한 저전력 BTB의 방법을 제안하였다. [3]은 분기 예측기 예측을 (BPP)를 제안하였다. 분기 예측기 예측은 각각의 분기 예측기 테이블을 Turnon/Turnoff를 통하여 불필요한 전력 소모를 감소시켰다. [16]은 성능 향상에 도움이 되지 않은 분기에 있어 분기 예측기에 대한 접근을 규정하고 차단하여 전력 소모를 감소시켰다. [15] 는 PABU (Power aware branch predictor update)를 이용하여 분기 예측기의 내용에 대한 불필요한 갱신을 억제하여 전력 소모를 감소시키는 방법을 제안하였다. [3]의 BPP가 분기 예측기의 접근을 제한하는 방법에 초점을 맞추었다면, PABU는 분기 예측기의 갱신을 감소시키는 방법에 초점을 맞추었다.

이전 관련 연구들은 주로 고성능의 범용 프로세서에 초점을 맞추어 진행되었고 제한된 자원을 지닌 내장형 프로세서에 대한 관련 연구는 거의 존재하지 않는다. 그리고, 이전 관련 연구는 그들의 가정에 대한 증명을 Brute-force 접근 방식을 이용하여 수행하였다. 그러나, 우리는 “내장형 프로세서”에 대한 최적화된 전력 소모를 위한 구성 방식을 수학적 모델링을 통하여 입증하고 제안한다.

3. 수학적 모델링

이 장에서는 문제를 단순화하기 위하여 간단한 구조를 지닌 Bimode 분기 예측기 [4]에 초점을 맞추어 모델링을 진행할 것이다. 물론 gshare와 gselsect [5]와 같은 전역 이력 구조를 지닌 분기 예측기가 보다 정확한 예측이 가능하지만 전역 이력과 관련된 변수를 취급해야한다는 측면에서 분석을 복잡하게 만든다. 더구나 일반적인 Intel Xscale과 같은 내장형 마이크로 프로세서는 간단한 bimode 분기 예측기를 사용하고 있다. 그러므로, 먼저 우리는 Bimode 분기 예측기에 기반하여 분기 예측기 테이블 크기에 따른 miss prediction rate의 함수를 모델링할 것이다. 그리고, 이를 확장하여 전체 시스템의 전력 소모에 관련된 모델링을 수행할 것이다. 그 이후 저전력 소모를 위한 최적화된 분기 예측기 테이블 크기를 구할 것이다.

3.1 Miss Prediction Rate

Aliasing 효과는 miss prediction rate의 가장 중요한 요인으로 알려져 있다 [6][7]. Aliasing은 2개 이상의 분기 주소가 하나의 테이블 엔트리에 맵핑되었을 때 발생한다. 만약 프로세서에 충분한 개수의 엔트기가

존재한다면, Aliasing 효과는 발생하지 않는다. 그러나, 전력과 Chip 크기의 제한으로 내장형 프로세서는 제한된 개수의 엔트리만 가질 수 있다. 작은 엔트리의 개수는 Aliasing 효과의 발생 가능성을 높인다.

Aliasing은 다음의 두가지 기준에 의하여 나뉠 수 있다. 첫번째 기준은 해당 Aliasing이 Constructive나 Destructive 중 어떤 효과를 지니는가이다 [6][7]. Constructive Aliasing이 발생하면 분기 예측기는 분기의 방향을 운좋게 정확하게 예측한다. 그러나 Destructive Aliasing은 분기의 방향을 잘못 예측하는 효과를 지닌다. 두번째 기준은 해당 Aliasing이 어떠한 조치에 의하여 개선 가능한가이다. Compulsory/Conflict Aliasing은 해당 분기 명령이 처음 나타났을 때와 같이 결코 피할 수 없는 Aliasing이다. 반대로 Capacity aliasing은 프로그램의 분기 명령의 수가 많아서 분기 예측기의 테이블의 엔트리 수가 모자를 경우에 발생하며, 이 경우 간단하게 분기 예측기의 테이블 크기를 증가시킴으로써 개선 가능하다 [8]. 우리는 이러한 Aliasing의 종류에 따라서 수학적 모델링을 수행할 것이다.

수학적 모델링을 수행하기 위하여 몇 가지 가정을 수행하였다. 분기 예측기 테이블은 N의 크기를 지니고, 주요 분기 명령이 반복적으로 나타나는 사이는 M개의 거리를 지니고 있다. 분기 예측기 테이블의 크기가 무한대일 경우 Compulsory/Conflict Aliasing에 의하여 발생하는 Miss prediction은 B의 확률을 지니고 있다.

먼저 miss prediction rate을 두 개의 부분으로 분리하였다. $r_{mp,a}$ 는 capacity aliasing이 발생하였을 때이고, $r_{mp,na}$ 는 capacity aliasing이 발생하지 않았을 경우이다.

$$r_{mp} = r_{mp,a} + r_{mp,na} \quad (1)$$

$$r_{mp,a} = 2b(1-b)p_a \,,\; r_{mp,na} = B(1-p_a) \quad (2)$$

b는 분기가 취해질 확률이므로, $2b(1-b)$ 는 destructive aliasing이 발생할 확률이 된다. 그리고, p_a 는 capacity aliasing이 발생할 확률이다. 따라서 우리는 p_a 를 다음과 같이 정의할 수 있다.

$$p_a = 1-(1-\frac{1}{N})^M \quad (3)$$

$2b(1-b)$ 를 A라 한다면, miss prediction rate r_{mp} 은 다음과 같이 표현된다.

$$r_{mp} = (A-B)(1-\frac{1}{N})^M + B \quad (4)$$

3.2 전력 소모 모델링

전체 전력 소모는 분기 예측기의 전력 소모량 및 나머지 시스템의 전력 소모량으로 나누어 표현될 수 있다.

$$E_{sys} = E_{rest} + E_{bp} \quad (5)$$

(5)의 식을 확장하면 다음과 같다.

$$\begin{aligned} E_{sys} &= P_{rest}\cdot CPI\cdot N_{Inst}\cdot T_{cycle} + \frac{1}{2}CV_{dd}{}^2 N_{branch}N_{cell}K \\ &= P_{rest}\cdot N_{Inst}\cdot T_{cycle} + P_{rest}\cdot T_{cycle}\cdot N_{branch} \\ &\cdot (misspenalty)\cdot (missrate) + N_{branch}N_{cell}K_{bp} \\ &= E_{ideal} + N_{branch}\left(P_{rest}\cdot T_{cycle}\cdot (misspenalty)\cdot (missrate) + N_{cell}K_{bp}\right) \end{aligned} \quad (6)$$

K_{bp} 는 분기 예측기의 구조 및 반도체 공정 기술과 관련된 상수이다. 식 (6)에서 첫번째 항 E_{ideal} 은 어떠한 miss prediction도 발생되지 않았을 때의 전력 소모량이며 두번째 항은 분기 예측기와 전체 시스템 사이의 전력 소모량의 Trade-off를 나타낸다. 그러므로, 최적화된 분기 예측기 테이블 크기를 결정하기 위하여 두번째 항만 고려하여 수식을 전개하면된다. 두번째 항을 전개하면 다음과 같다.

$$E_{tradeoff}(N_{cell}) = P_{rest}\cdot T_{cycle}\cdot (misspenalty)\cdot r_{mp}(N_{cell}) + N_{cell}K_{bp} \quad (7)$$

최적화된 테이블 크기에 따른 최적화된 값을 찾기 위하여 (7)의 식을 N_{cell} 로 미분하여 전개하면 다음과 같다.

$$E_{tradeoff}{}'(N_{cell}) = P_{rest}\cdot T_{cycle}\cdot (misspenalty)\cdot r_{mp}{}'(N_{cell}) + K_{bp} \quad (8)$$

$$r_{mp}{}'(N_{cell}) = -(A-B)M(1-\frac{1}{N_{cell}})^{M-1}\frac{1}{N_{cell}{}^2} \quad (9)$$

$E_{tradeoff}{}'(N_{cell})$ 이 0이 되는 N_{cell} 의 값을 구하면 $E_{tradeoff}(N_{cell})$ 값이 최소값을 지닐 수 있게된다. 그러므로, 해당 N_{cell} 은 최적화된 지점이라고 할 수 있다.

$$\frac{(N_{cell}-1)^{M-1}}{N_{cell}^{M+1}} = \frac{K_{bp}}{P_{rest}\cdot T_{cycle}\cdot(misspenalty)(A-B)M} \quad (10)$$

보통 N_{cell} 의 값은 큰 값을 지니므로 $\frac{(N_{cell}-1)^{M-1}}{N_{cell}^{M+1}}$ 는 $\frac{1}{N_{cell}^{2}}$ 으로 근사화 가능하다. 이를 통하여 (10)의 식을 정리하면 다음과 같다.

$$N_{cell,opt} \approx \sqrt{\frac{P_{rest}\cdot T_{cycle}\cdot(misspenalty)(A-B)M}{K_{bp}}} \quad (11)$$

3.3 모델 분석

그림 1은 식 (4)를 이용하여 응용 특성에 따른 miss prediction rate의 변화를 보여주고 있다. 점선의 왼편에 있는 부분은 capacity aliasing이 지배적으로 작용하고 있음을 보여주고 있다. 점선의 위치는 M 값에 의하여 변화한다. 오른쪽으로 점선이 이동하는 것은 capacity aliasing을 방지하기 위하여 보다 많은 분기 예측기 테이블의 엔트리가 필요함을 의미한다. 이러한 결론은 식 (11)과 잘 부합하고 있다. 식 (11)은 큰 M 값일 경우 보다 큰 $N_{cell,opt}$ 이 필요함을 보여준다. 그림 1 (b)는 $(A-B)$ 의 변화에 따른 $N_{cell,opt}$ 과 M 사이의 관계를 보여준다. $(A-B)$ 의 값은 0에서 0.5 까지 변화하도록 하였을 때 모든 경우 $N_{cell,opt}$ 의 값이 128보다 큰 값을 지닌다. Intel Xscale과 ARM9과 같은 일반적인 내장형 프로세서의 경우 128 엔트리의 분기 예측 테이블 크기를 가진다 [1][17]. 전력 소모의 관점에서 이러한 크기는 작게 설계되었다고 판단된다. 분기 예측기의 설계에 있어 칩 크기, 온도, 속도 등의 시스템적 요인이 고려되어야하나, 우리는 전력 특성이 가장 중요한 요인이라고 생각한다. 그러므로, 해당 내장형 프로세서의 분기 예측 테이블의 크기는 보다 더 증가되어야한다.

4. 시뮬레이션 환경 및 결과

4.1 실험 환경

전력-성능 시뮬레이션을 위하여 Sim-Panalyzer [18]를 이용하였다. Sim-panalyzer는 Cycle Accurate 시뮬레이터인 SimpleScalar [19]에 전력 소모를 계산하기 위하여 내부 스위칭 전력, IO 스위칭 전력, 누설 전류 등에 대한 시뮬레이션 기능을 추가하였다. 실험 구성은 내장형 환경을 반영하기 위하여 Xscale PXA255 프로세서를 유사하게 구성하였으며 표 1에 나타냈다.

벤치마크 프로그램은 MiBench [20]를 이용하였다. MiBench는 내장형 환경을 목표로 구성되었기 때문에 범용 벤치마크인 SPEC을 대신하여 사용하였다. 모든 MiBench 프로그램은 GCC 2.95.2에 기반한 ARM 크로스 컴파일러를 이용하였다. 성능-전력 간의 trade-off를 측정하는 것이 목적이기 때문에, 벤치마크 프로그램의 분기 명령의 개수는 중요하다. 각각의 벤치마크 별 특성은 표 2에 나타내었다.

bimode 분기 예측기 테이블의 크기를 1에서 2^{20}까지 증가 시키며 시뮬레이션을 진행하였다. 각각의 엔트리는 2bit 카운터로 구성되었으며, 분기 명령의 하위 비트들을 인덱스로 이용하는 구조로 구성되었다.

4.2 시뮬레이션 결과 및 분석

그림 2는 miss prediction rate에 대한 Sim-panalyzer의 시뮬레이션 결과와 식 (4)를 이용한 계산 결과값을 보여주고 있다. 시뮬레이션 결과와 우리의 모델링에 의한 결과가 잘 부합하고 있음을 확인할 수 있다. Miss prediction rate은 분기 예측기 테이블

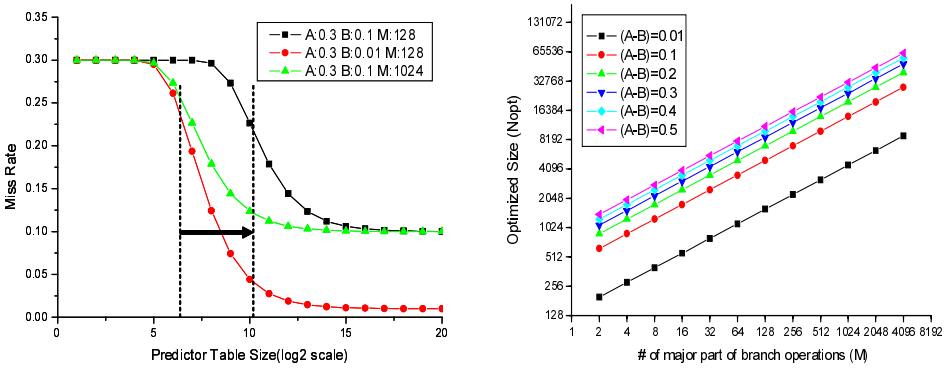

그림. 1 계산 결과 (a) miss-prediction rate (b) $N_{cell,opt}$

표 1. Base Configuration

Fetch queue (instruction)	4
Branch predictor	Bimodal / Gshare 4 cycles of miss-penalty
Fetch & Decode width	1
Issue width	1
Functional units	1 int ALU, 1FP mult, 1FP ALU, 1 FP mult
Instruction L1 cache	32k byte fully associativity cache, FIFO
Data L1 cache	32k byte fully associativity cache, FIFO
L2cache	None
Memory (bus width, first block latency)	4byte, 12 cycle

표 2. Characterization of MiBench

benchmarks	Dynamic conditional branch	Static conditional branches
madplay	445080	7425
djpeg	192863	8056
ispell	8467918	6212
basicmath	5914616	5295
fft	1636079	5043
susan.edges	783850	62415
tiffdither	7837003	7807

크기의 함수의 형태로 점선으로 표시되고 있다. Tiffdither, susan, djpeg의 경우 작은 M 값으로 인하여 테이블 크기를 늘림에 따라 큰 개선이 이루어지고 있지 않다. 그러나, 반대로 basic_math의 경우 테이블의 크기가 2^{10}을 넘더라도 capacity aliasing이 발생됨을 확인할 수 있다. 엔트리의 수가 2^{10} 을 넘는 경우 compulsory aliasing이 miss prediction의 주요 원인으로 작용하고 있으며, 이는 gshare와 같은다른 전역 이력 분기 예측기를 이용함으러써 개선될 수 있다.

시뮬레이션 결과를 통하여 A , B 및 M 값을 캘리브레이션을 통하여 추출할 수 있었다. 이러한 응용 특성 값들을 식 (11)에 대입하여 최적화된 분기 예측 테이블의 크기 $N_{cell,opt}$ 값을 구하였다. 그림 3 및 표 3은 해당 결과값을 나타내고 있다. 결과값을 보면 시뮬레이션 결과와 우리의 모델 사이에 $N_{cell,opt}$ 과 관련된 에러값이 존재한다. 이는 전력 소모 그래프의 평평한 부분이 특정 구간에 존재하는 것에 기인한다고 판단된다. 그림 3에서 보이는 바와 같이, 최저의 전력 소모를 이루고 있는 부분은 매우 넓게 펼쳐져 있다. Djpeg의 경우 $N_{cell,opt}$ 는 2^6 to 2^{13}의 영역 어디에라도 존재 가능하다. 이러한 영역이 발생하는 것은 거의 모든 벤치마크 프로그램에서 동일하게 나타나고 있다. 그러므로, 설계 단계의 테이블 크기의 결정에 있어 설계 유동성을 지닐 수 있다.

5. 결론

본 논문에서 우리는 내장형 프로세서에 대한 성능-전력 공간에서 분기 예측기에 대하여 분석 기술을 제안하였다. 그리고, 분기 예측기 테이블 최적화된 크기를 구할 수 있는 방법을 제시하였다. 이는 수학적 모델링, 시뮬레이션 결과와의 비교 및 수학적 모델에 대한 검증을 통하여 수행되었다. 결과는 우리의 모델이 성능-전력 공간에서 최적화된 분기 예측기 크기를 구하는 방법을 제시하고 있음을 보여주고 있다. 우리의 수학적 모델은 최저의 전력 소모를 위한 A , B 및 M 과 같은 응용 프로그램에 특화된 특성들을 제시하고 있다.

6. 참고 문헌

1 Intel® XScale™ Microarchitecture Technical Summary. Available: http://www.intel.com

2 D. Parkikh, K. Skadron, Y. Zhang, M. Barcella, and M. R. Stan, “Power Issues Related to Branch Prediction,” Proc. of HPCA, 2002.

3 A. Baniasadi, A. Moshovos, “Branch Predictor Prediction: A Power-Aware Branch Predictor for High-Performance Processors,” Proc. of ICCD, 2002.

4 J.E. Smith. “A study of branch prediction strategies,” Proc. of ICPACT, pages 199-206, Oct. 2000

5 S. McFarling, “Combining branch predictors: Technical Report,” DEC, 1993

6 S. Sechrest, C.C. Lee, and T. Mudge, “Correlation and Aliasing in Dynamic Branch Predictors,” Proc. of AISCA, May 1996

7 C. Young, N. Gloy, and M.D. Smith, “A Comparative Analysis of schemes for correlated branch prediction,” Proc. of AISCA, June 1995

8 P. Michaud, A. Seznec, and R. Uhlig, “Trading Conflict and Capacity Aliasing in Conditional Branch Predictors,” Proc. of ISCA, 1997

9 D.Parikh, K. Skadron, Y. Zhang, and M. Stan, “Power-aware branch-prediction: Characterization and design,” IEEE Trans. Comput. 53, 2, 168–186. 2003

11. S.-T. Pan, K. So, J. T. Rahmeh, “Improving the Accuracy of Dynamic Branch Prediction Using Branch Correlation,” Proc.of ASPLOS V, 1992.

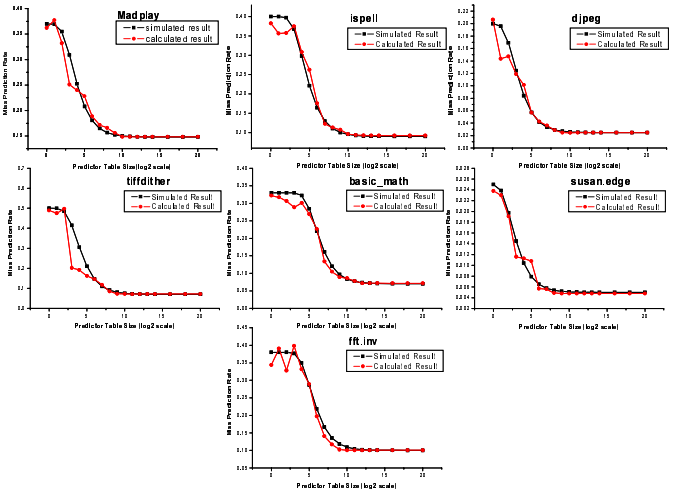

그림. 2 Miss-prediction rate

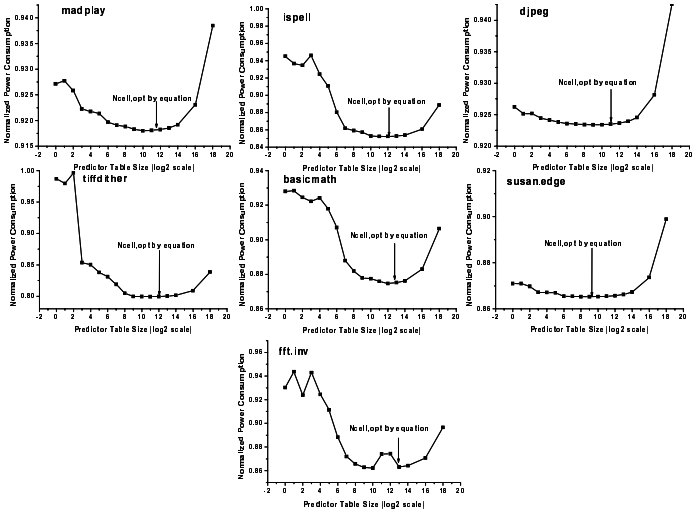

그림. 3 테이블 크기에 따른 전력 소모

표 3. Application specific characterization of benchmarks

benchmarks	M	A	B	$N_{cell,opt}$ simulation	$N_{cell,opt}$
madplay	20	0.37	0.148	2^{10}	$2^{11.53}$
djpeg	13	0.2	0.0247	2^{9}	$2^{11.05}$
ispell	35	0.4	0.09	2^{12}	$2^{12.17}$
basicmath	110	0.33	0.07	2^{12}	$2^{12.87}$
fft.inverse	70	0.38	0.1	2^{10}	$2^{12.59}$
susan.edge	10	0.025	0.005	2^{9}	$2^{9.29}$
tiffdither	25	0.5	0.07	2^{11}	$2^{12.17}$

12. D. Chaver, L. Pinuel, M. Prieto, F. Tirado, and M.C. Huang, “Branch Prediction on Demand: an Energy-Efficient Solution,” Proc. of ISLPED, 2003.
13. M. Monchiero, G. Palermo, M. Sami, C. Silvano, V. Zaccaria, R. Zafalon, “Power-Aware Branch Prediction Techniques: A Compiler-Hints Based Approach for VLIW Processors,” Proc.of GLSVLSI, 2004.
14. Peter Petrov and Alex Orailoglu. "Low-Power Branch Target Buffer for Application-Specific Embedded Processors" Euromicro Symposium on Digital System Design (DSD), pp.158-165, September 2003.
15. Baniasadi, A.: “Power-aware branch predictor update for high-performance processors”. Proc. Int. Workshop on Power and Timing Modeling, Optimization and Simulation, September 2003, pp. 420–429
16. A. Baniasadi, A. Moshovos, “SEPAS: A Highly Accurate Energy-Efficient Branch Predictor,” Proc. of ISPLED, 2004.
17 ARM11 Technical Reference Guide. Available: http://www.arm.com
18 Sim-Panalyzer Project, http://www.eecs.umich.edu/~panalyzer/
19 SimpleScalar Project, http://www.simplescalar.com/
20 M. R. Guthaus, J. S. Ringenberg, D. Ernst, T. M. Austin, T. Mudge, and R. B. Brown, “Mibench: A free, commercially representative embedded benchmark suite,” In IEEE 4th Annual Workshop on Workload Characterization, Austin, TX, Dec. 2001