The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 18, No. 4, pp.169-175, Aug. 31, 2018. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2018.18.4.169
JIIBC 2018-4-24

임베디드 마이크로 프로세서의 전력 소비에 대한 연구

A Study on Power Dissipation of Embedded Microprocessors

이종복*

Jongbok Lee*

요 약 프로세서의 전력 소비량은 최근에 이르러 고성능 마이크로프로세서 및 멀티코어프로세서 뿐만이 아니라 임베디드 시스템 및 모바일 장치에 매우 중요하게 대두되고 있다. 이러한 전력 소비량은, 하드웨어 및 소프트웨어 설계자로 하여금 성능과 전력에 대한 올바른 타협점을 찾도록 하는 바탕이 된다. 대부분의 전력 분석 도구들은 반도체 칩 레이아웃이나 평면계획이 완료된 후에야 최소의 정확도를 갖게 되며 또한 느리다. 본 논문에서는 전력 분석기와 연동이 가능한 빠른 속도를 갖는 임베디드 마이크로프로세서 명령어 자취형 (trace-driven) 모의실험기를 개발하였다. 또한, MiBench 임베디드 벤치마크를 입력으로 모의실험을 수행하여 기존의 도구보다 훨씬 빠른 속도로 명령어 당 평균 전력 소비량을 측정하였다.

Abstract Recently, power dissipation issue is very significant not only in high-end modern processors but also in embedded systems and mobile devices. Based on the power dissipation, hardware and software designers can correctly find the power/performance tradeoffs. Most power analysis tools calculate power dissipation when chip layout or floor planning are finished. In this paper, a trace-driven simulator that can interact with power analysis tool for an embedded microprocessor has been developed. Using MiBench embedded benchmarks as input, the trace-driven simulation has been performed to estimate the average power dissipation which is faster than the conventional tools.

Key Words : power dissipation, instruction trace, simulation

Ⅰ. 서 론

전력 소비량은 고성능 마이크로프로세서나 멀티코어 프로세서에서는 물론, 임베디드 컴퓨터 또는 모바일 컴퓨터 시스템의 설계에서 주요 제약요소로 대두되고 있다. 프로세서의 기술이 발전하여 반도체 다이의 크기가 더욱 작아지고, 클럭의 주파수가 증가할수록 전력 소비량이 큰 비중으로 고려되어야 한다.

임베디드 마이크로프로세서의 성능과 전력을 측정하기 위한 모의실험에는 실행 위주(execution-driven) 방식과 명령어 자취(trace-driven) 방식이 널리 쓰이고 있다. SimpleScalar와 같은 실행 위주 방식은 정확하지만 시간이 많이 걸린다는 단점이 있다[1]. 본 논문에서는 전력 모델과 연동하여 임베디드 프로세서의 전력 소비량을 빠르고 효율적으로 측정할 수 있는 명령어 자취 모의실험기를 제안하였다. 또한, 본 모의실험기를 이용하여, MiBench

*정회원, 한성대학교 전자정보공학과
접수일자 2018년 7월 2일, 수정완료 2018년 8월 2일
게재확정일자 2018년 8월 10일

Received: 2 July, 2018 / Revised: 2 August, 2018 /
Accepted: 10 August, 2018
*Corresponding Author: jblee@hansung.ac.kr
Dept. of Electronic & Information Eng., Hansung University, Korea

임베디드 벤치마크 프로그램에 대하여 평균 전력의 소비량을 측정하고, 임베디드 프로세서의 각 하드웨어 유닛에서 소비되는 전력을 분석하였다[2].

본 논문은 다음과 같이 구성된다. 2 장에서 임베디드 프로세서의 전력 소비량을 측정할 수 있는 모델과 임베디드 프로세서 모의실험기에 대하여 살펴보고, 3 장에서 모의실험 환경에 대하여 고찰한다. 4 장에서 모의실험 결과를 보이며, 5 장에서 결론을 맺는다.

II. 임베디드 프로세서의 전력 모델 및 모의실험기

1. 임베디드 프로세서의 전력 모델

그림 1에 임베디드 프로세서의 전력분석기와 명령어 자취 모의실험기의 관계를 나타내는 전체 흐름도를 보였다. 특정한 임베디드 프로세서의 하드웨어 사양은 명령어 자취 모의실험기와 전력분석기 양쪽에 모두 공급된다. 명령어 자취 모의실험기는 위 사양과 명령어 자취를 공급받아 성능을 측정하며, 전력분석기는 각 싸이클마다 하드웨어를 접근한 회수를 입력으로 받아서 전력을 측정한다.

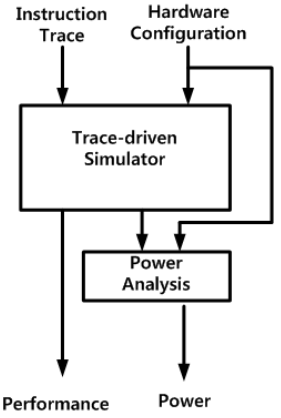

그림 1. 전력측정의 전체 흐름도
Fig. 1. Overall Flow of the Power Simulator

CMOS 임베디드 프로세서에서, 동적인 전력 소비량이 차지하는 비중이 가장 크며, 식 1과 같이 나타낼 수 있다.

$$P_d = \alpha C V_{dd}^2 f \qquad (1)$$

이 때 α는 0과 1 사이의 값으로서 하드웨어 유닛에 대한 클럭에 의한 평균 활성화율이며, C는 정전용량, Vdd는 공급되는 전원, f는 동작 주파수이다. 본 논문의 실험에서는 0.35μm CMOS 기술을 가정하였다[3].

임베디드 프로세서의 전력 소비량은 설계 및 구현 방법과 회로의 내부 정전용량에 의하여 크게 영향을 받는다. 지금까지 연구된 가장 일반적인 방법은 임베디드 프로세서를 여러 단계로 나누어 각 단계마다 RC 회로를 모델링하는 것으로, 각 단계의 지연시간을 측정할 수 있으나 임계경로를 위주로 측정하기 때문에 전력 소비량을 측정하는데 한계가 있다. 이러한 단점을 극복하기 위하여, 본 논문에서는 각 단계에서의 정전용량을 고려하고 모든 경로에서의 전력 소비량을 분석하고 합산하였다[4].

본 논문에서 임베디드 프로세서에 대한 전력 소비량의 측정은, 표 1에 나타낸 것과 같이 크게 네 가지 유형의 하드웨어 구조로 분류하였다. 첫 번째는 복합 조합회로로서, 연산유닛, 명령어 윈도우 선택 논리회로, 종속 검사 논리회로, 결과버스 등을 예로 들 수 있다. 두 번째는 배열 구조로서, 명령어 캐쉬, 데이터 캐쉬, 캐쉬 태그 배열, 레지스터 화일, 분기 예측기, 로드 스토어 큐 등이 이에 해당된다. 세 번째는 내용주소화 기억장치인 CAM (Content Addressable Memory) 구조로서, 명령어 기동 논리장치 (instruction wakeup logic)가 이에 속한다. 네 번째는 배열구조와 CAM 구조의 특징을 동시에 갖는 회로로서, 명령어 윈도우, TLB, 로드/스토어 큐 등이 여기에 속한다.

표 1. 임베디드 프로세서 하드웨어 유닛과 대응하는 전력 모델
Table 1. Embedded microprocessor hardware units and associated power model

카테고리 유형	하드웨어 유닛
복합 조합회로	명령어 선택 논리장치, , 레지스터 재명명 종속검사 장치, 정수형 연산 유닛, 실수형 연산 유닛
배열구조	명령어 캐쉬, 데이터 캐쉬, 레지스터 화일, 분기예측기
CAM 구조	명령어 기동 논리장치
배열 및 CAM 구조	명령어 윈도우, TLB, 로드/스토어큐

가. 복합 조합회로의 전력 소비량

복합 조합회로는 이전 상태를 기억하지 않고 현재의 입력에 의해서만 출력이 결정되며, 규칙적이거나 반복적인 구조를 갖지 않고 임의의 논리를 구현하는데 쓰인다.

복합 조합회로의 대표적인 사례는 ALU와 로드 스토어 장치 등의 정수형 연산유닛, 실수형 덧셈기와 실수형 곱셈기를 포함하는 실수형 연산유닛, 명령어 윈도우의 명령어 선택 논리장치와 레지스터 재명명 장치의 종속 검사 논리장치를 들 수가 있다[5,6].

나. 배열구조의 전력 소비량

배열구조는 명령어 캐쉬, 데이터 캐쉬, 레지스터 화일 및 분기 예측기와 같이 규칙적이고 반복적인 구조를 갖는 하드웨어 유닛의 전력을 측정하는데 쓰인다. 이 때, 행의 수, 열의 수, 읽기 및 쓰기 포트 수를 기준으로 하여 디코더의 크기와 개수, 워드라인 (word line) 및 비트라인 (bit line)의 개수가 결정된다. 따라서, 배열구조의 전력 소비는 디코더, 워드라인 구동기, 비트라인 방전으로 결정된다.

워드라인과 비트라인의 전력 소비량 측정은 각 라인의 총 정전용량을 계산하여 얻을 수 있다. 워드라인과 관련된 정전용량은 워드라인 구동기의 확산 정전용량 (diffusion capacitance), 셀 접근 트랜지스터의 게이트 정전용량 (gate capacitance), 워드라인 금속선의 정전용량의 세 가지로 구성된다. 한편, 비트라인의 정전용량은 선충전 트랜지스터의 확산 정전용량, 셀 접근 트랜지스터의 확산 정전용량, 비트라인 금속선의 정전용량으로부터 계산에 의하여 구할 수 있다. 이것을 종합하면, 배열구조의 지연시간은 디코드, 워드라인, 비트라인, 센스증폭기에서의 각 지연시간을 합하여 아랫 식 2로 나타낼 수 있다.

$$T_{array} = T_{decode} + T_{wordline} + T_{bitline} + T_{senseamp} \quad (2)$$

다. CAM 구조의 전력 소비량

CAM 구조는 내용으로 메모리를 어드레싱할 수 있는 점을 제외하고 배열구조와 유사하다. 이 구조에서는 배열구조의 비트라인과 워드라인이 각각 태그라인과 매치라인에 대응된다는 점만 제외하고 배열구조와 같다. CAM 구조에 속하는 하드웨어 장치는 명령어 기동 논리장치가 대표적이다. 이 때, 명령어의 이슈 및 완료 폭은 CAM 내부의 각 코어셀의 매치 및 태그 라인 수를 결정하며, 윈도우의 크기는 물리적 레지스터의 크기를 결정한다. 윈도우 기동 논리의 지연요소는 크게 3 가지로서, 태크비트를 구동하는데 소요되는 버퍼의 지연시간, 매치라인을 끌어내리기 위하여 비교기에서 소요되는 지연시간, 각 매치신호에 대한 논리합을 적용시키기 위하여 소요되는 지연시간으로 구성되며 식 3과 같다.

$$T_{wakeup} = T_{taggdrive} + T_{tagmatch} + T_{matchOR} \quad (3)$$

마지막으로, 배열 및 CAM 구조는 규칙적인 구조를 가지면서 동시에 저장된 데이터의 내용으로 장치를 접근해야하는 경우이다. 명령어 윈도우, TLB, 로드/스토어큐가 이에 해당된다. 이상, 본 논문의 전력모델은 위의 네 가지 카테고리에 대하여, Wattch 도구와 Cacti 도구를 참고로 하였다[7-9].

2. 임베디드 프로세서 모의실험기

임베디드 프로세서 모의실험 과정은 명령어 자취의 발생과 명령어자취에 대한 임베디드 프로세서의 실행으로 나누어진다. 제 1 단계에서 명령어 자취는 특정 하드웨어와 상관없이 소프트웨어의 특성에 따라 발생되었다.

제 2 단계 임베디드 프로세서 실행은 다음과 같다. 본 모의실험기는 수퍼스칼라 임베디드 프로세서로 동작할 수 있도록 프로그래밍 되었다. 따라서 한 싸이클에 2 개 이상의 명령어를 인출받을 수 있다. 제 2 단계의 과정을 자세하게 기술하면 그림 2에 나타낸 것과 같으며, 설명하면 다음과 같다.

가. 명령어 인출 및 재명명

Initialize 함수에서 초기화 작업을 거친 후에, Grouping 함수가 Create_Window 함수를 부르고 이것은 다시 Fetch_One_Instr 함수를 불러서 매 싸이클마다 새로운 명령어를 인출받는다. 한 싸이클에 1 개의 명령어를 인출받을 때와 달리, 2 개 이상의 명령어를 인출받을 때, 분기명령어를 만나면 그 이상의 인출을 하지 않고 인출을 멈춘다.

Get_Node 함수에서 인출한 명령어는 Rename 함수에서 재명명 (renaming) 작업을 거치면서 명령어 종속에 의한 타임스탬프(timestamp) 값을 설정받는다. 타임스탬프 방식은 명령어 자취를 이용하는 모의실험에서 데이터 종속성을 신속하고 효율적으로 부여할 수 있는 핵심적인 방법이다. 모든 명령어는 인출 초기에 현재 싸이클에 명령어 자체의 지연 싸이클을 더한 값을 타임스탬프로 설정 받는다.

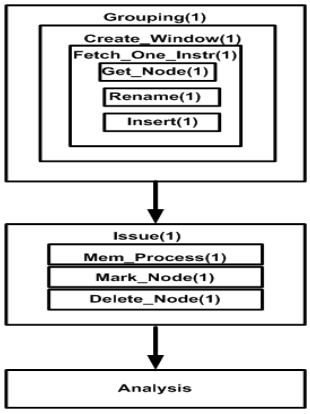

그림 2. 임베디드 프로세서 모의실험기
Fig. 2 The embedded microprocessor simulator

이후의 디코드 및 재명명 단계에서, 각 명령어의 소오스 레지스터 1과 소오스 레지스터 2에 대하여 레지스터 화일을 검색하여 같은 이름을 갖는 레지스터를 찾아서 해당하는 타임스탬프 값을 읽는다. 만일 현재 명령어의 타임스탬프 값보다 소오스 레지스터 1 또는 2의 타임스탬프 값이 더 크다면, 명령어의 타임스탬프는 그 값으로 대체된다. 또한, 명령어의 목적레지스터의 타임스탬프도 같은 값으로 대체되어, 추후에 이 목적 레지스터를 소오스 레지스터로 참조하는 명령어에 대하여 올바른 타임스탬프 값을 가질 수 있도록 한다. 레지스터 화일의 타임스탬프 값에 의하여, 임베디드 프로세서 명령어 간의 종속성이 유지되어 성능을 구하는데 반영된다. 이와 같이 재명명을 거친 명령어는 insert 함수에서 명령어 윈도우에 삽입된다.

나. 명령어 이슈

Issue 함수로 진행하면, 싸이클이 증가함에 따라서 윈도우 내의 명령어는 자체의 타임스탬프 값이 현재 싸이클 보다 작거나 같을 때 삭제될 수 있다. 그러나, 첫 단계인 Mark_Node 함수에서 즉각 삭제하지 않고 삭제 가능 표시만 하며, 다음 단계인 Delete_Node 함수에서 삭제 가능 표시를 한 명령어를 실제로 삭제한다. 이 과정에서 만일 삭제 가능 표시를 하지 않은 명령어를 만나면 그 이후의 명령어는 삭제를 즉각 종료한다. 이렇게 함으로써, 비순차실행(out-of-order execution) 방식의 수퍼스칼라 프로세서인 경우에도 순차종료(in-order completion)를 보장할 수 있다.

다. 시뮬레이션

해당 윈도우 공간에 Grouping 함수를 이용하여 명령어를 인출해서 채우고, Issue 함수로 명령어를 실행하면서 종속성에 의하여 부여된 명령어의 타임스탬프가 충족되면 삭제한다. 위의 Issue 동작은 명령어가 삭제되어 윈도우가 빈 상태가 될 때까지 반복적으로 실행된다. 윈도우가 비어있는 상태가 되면, 다시 Grouping 함수를 통하여 윈도우를 명령어로 채운다. 이 과정은 입력으로 주어진 벤치마크 프로그램의 모든 명령어가 소진될 때까지 반복된다.

위 과정이 한번 실행될 때 마다 싸이클이 증가하므로 모의실험에 입력으로 쓰인 명령어의 총 개수를 처리하기 위하여 소요된 싸이클 수로 나누어, 임베디드 프로세서 시스템의 IPC(Instruction Per Cycle)을 계산할 수 있다[10].

라. 전력 모델과의 인터페이스

전력 모델은 위에서 기술한 임베디드 프로세서 모의실험기에서 매 싸이클 당 접근되는 각 하드웨어 유닛을 추적 및 카운트하여 실행을 위하여 소비되는 전력을 계산한다. 예를 들어서, 명령어 캐쉬를 접근할 때, 명령어 윈도우에서 이슈 가능한 명령어를 명령어들을 선택할 때, 레지스터 화일에서 데이터를 읽거나 쓸 때, 산술논리연산 유닛에서 계산을 수행할 때, 데이터 캐쉬를 접근할 때 각 장치에서의 전력의 소비량을 계산한다.

Ⅲ. 모의실험 환경

본 논문의 모의실험은 운영체제 Fedora 22에서 3.1GHz로 동작하는 Intel Core i5-2400에서 시행하였다. 표 2는 모의실험에 이용된 일곱 개의 MiBench 벤치마크 프로그램이다.

표 2. MiBench 임베디드 벤치마크 프로그램
Table 2. MiBench embedded benchmark programs

benchmark	description
basicmath	simple mathematics
CRC	32 bit circular redundancy check
dijkstra	the shortest path finder of a large scale graph
FFT	fast Fourier transform
qsort	sorting of a large scale string
rijndael	block encipherment
sha	a safe hashing algorithm

이 일곱 개의 정수형 벤치마크 프로그램을 SimpleScalar를 통하여 MIPS IV 10억 개의 명령어 자취를 발생시켜서 모의실험기에 입력하였다.

표 3은 모의실험에 이용된 임베디드 프로세서 아키텍쳐의 사양을 나타낸 것이다. 임베디드 프로세서는 수퍼스칼라 방식으로 운영되므로, 매 싸이클 마다 4 개의 명령어를 인출하고 윈도우의 크기는 16이다. 임베디드 프로세서의 연산유닛은 정수형 유닛, 로드 스토어 유닛, 그리고 분기명령어로 구성된다. 임베디드 프로세서의 명령어 캐쉬와 데이터 캐쉬는 64 KB의 용량을 갖도록 설정하였다.

표 3. 임베디드 프로세서 아키텍쳐 하드웨어의 사양
Table 3. The architecture specification of each core

항목	값
1 차 명령어 캐쉬	64KB
1 차 데이터 캐쉬	64KB
1차 명령어 캐쉬 공통 사항	2 차 연관, 16 B 미스 페날티 10 싸이클
1차 데이터 캐쉬 공통 사항	직접, 32 B 미스 페날티 10 싸이클
명령어 윈도우의 크기	16
인출율, 이슈율, 퇴거율	4
연산유닛 사양	ALU(4), load & store(2), branch(1)
분기 어드레스 캐쉬	1 K 엔트리
분기 예측기	8 비트 전역 히스토리 방식 미스 페날티 6 싸이클
이슈 지연 싸이클	산술논리(1), 분기(1), 로드(1), 스토어(1),
결과 지연 싸이클	산술논리(1), 분기(1), 로드(1), 스토어(1),

1 차 명령어 캐쉬는 2 차 연관도(set associativity)를 가지나, 1 차 데이터 캐쉬는 직접 매핑을 통하여 접근된다. 2 차 캐쉬는 100 % 히트가 난다고 가정하였다. 분기 명령어는 2 단계 적응형 분기 예측 방식을 적용하였다. 본 논문의 실험에서는 600MHz에서 동작하는 0.35μm CMOS 기술을 가정하였다.

Ⅳ. 모의실험 및 결과

그림 3에 일곱 개의 MiBench 벤치마크에 대하여 모의실험을 수행하여 측정한 평균 전력 소비량의 결과를 보였다. 모의실험 결과에서 알 수 있듯이, 각 벤치마크 프로그램 별로 최저 28.2 W에서 최고 40.0 W 범위의 평균 전력을 기록하였다.

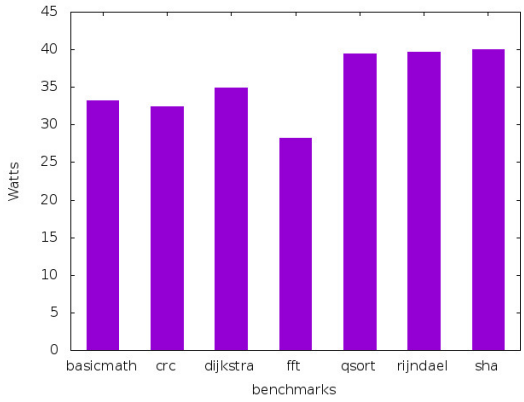

그림 3. MiBench 임베디드 벤치마크의 평균 전력 소비량
Fig. 3. The average power dissipation of MiBench embedded benchmarks

*Sha*의 경우, ALU에서의 전력 소비량이 많아서 가장 높은 전력 소비량을 기록하였으며, *fft*는 명령어 캐쉬와 데이터 캐쉬에서의 전력 소비량이 적어서 가장 낮은 전력을 소비하였다.

그림 4에 임베디드 프로세서의 싸이클 당 하드웨어 장치별 전력 소비량을 나타냈다. 이 때 싸이클 당 전력 소비는 크게 재명명 장치, 분기예측 장치, 명령어 윈도우 장치, 로드 및 스토어 장치, 레지스터 파일, 명령어 캐쉬, 데이터 캐쉬, 산술연산장치, 결과 버스에서 이루어진다.

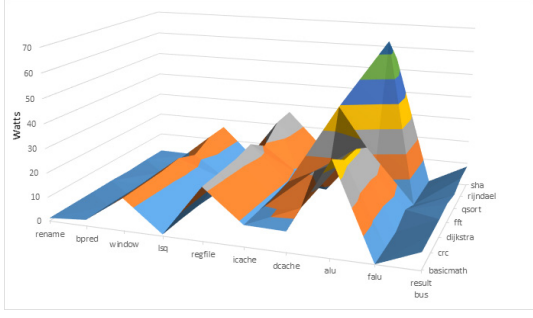

그림 4. 하드웨어 장치별 전력 소비량
Fig. 4. Power dissipation of each hardware unit

모의실험 결과에 의하면 재명명 장치, 분기예측 장치, 로드 및 스토어 장치에서의 전력 소비는 상대적으로 미

미하고, 명령어 윈도우 장치, 레지스터 파일, 명령어 캐쉬, 데이터 캐쉬, 산술연산 장치에서 대부분의 전력이 소비된다는 것을 알 수 있다.

본 모의실험을 운영체제 Fedora 22에서 3.1GHz로 동작하는 Intel Core i5-2400에서 시행한 결과, 초당 260K 개의 명령어를 처리할 수 있다. 이것은 초당 80K 개의 명령어를 처리할 수 있는 Wattch 도구보다 월등한 결과이다. 따라서, 본 논문에서 제안하는 임베디드 프로세서의 모의실험에 명령어 자취형 방식을 채택함으로써, 실행위주 방식보다 3.3 배 빠르게 전력 소비량에 대한 결과를 얻을 수 있다.

V. 결 론

본 논문에서는 0.35 μm CMOS를 기반으로 600MHz에서 동작하는 임베디드 프로세서의 전력을 명령어 자취 모의실험기를 기반으로 하여 빠르게 측정할 수 있는 전력 측정기를 개발하였다. MiBench 벤치마크에 대하여 마이크로프로세서에 대한 전력을 모의실험을 통하여 측정한 결과, 35.4 W의 평균 전력을 기록하였으며, 전력을 측정하는 속력을 Wattch 도구와 비교한 결과 3.3 배의 향상을 가져왔다.

추후로, 임베디드 멀티코어 임베디드 프로세서 및 디지털 신호처리 멀티코어 프로세서의 전력 소비량을 측정하는 연구 및 비대칭 칩 멀티코어 프로세서 구조에 대한 연구를 들 수 있다. 나아가, 최근 각광을 받는 범용 그래픽 처리장치인 GPGPU (General Purpose Graphic Processing Unit) 또는 신경망 전용 프로세서인 TPU(Tensor Processing Unit)에 대하여 전력 분석 연구를 수행할 예정이다.

References

[1] T. Austin, E. Larson, and D. Ernest, "SimpleScalar : An Infrastructure for Computer System Modeling," Computer, Vol. 35, No. 2, pp. 59-67, Feb. 2002.

[2] M. R. Guthaus, J. S. Ringenberg, D. Ernest, T. M. Austin, T. Mudge, and R. B. Brown, "MiBench: A free, commercial representative embedded benchmark suite," Workload Characterization, pp. 3-14, December 2001.

[3] S. Palacharla, N. Jouppi, and J. Smith, "Complexity-Effective Superscalar Processors", Proc. of the 24th International Symposium on Computer Architecture, 1997.

[4] S. Wilton and N. Jouppi, "An Enhanced Access and Cycle Time Model for On-chip Caches," WRL Research Report 93/5, DEC Western Research Laboratory, 1994.

[5] B. Bishop, T. Kelliher, and M. Irwin, "The Design of a Register Renaming Unit," Proc. of Grear Lakes Symposium on VLSI, 1999.

[6] R. Zimmermann and W. Fichtner, "Low-power logic styles : CMOS versus pass-transistor logic", IEEE Journal of Solid State Circuits, Vol. 32, No. 7, pp.1079-1090, 1997.

[7] M. Borah, R. Owens, and M. Irwin, "Transistor sizing for low power CMOS circuits," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems", Vol. 15, No. 6, pp. 665-671, 1996.

[8] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A Framework for Architectural-Level Power Analysis and Optimizations," Proceedings of the 27th International Symposium on Computer Architecture, pp 83-94, Jun. 2000.

[9] R. Joseph and M. Martonosi, "Run-time Power Estimation in High-Performance Microprocessors," The International Symposium on Low Power Electronics and Design, Aug. 2001.

[10] J. Lee, "A Study of Trace-driven Simulation for Multi-core Processor Architectures," Journal of The Institute of Internet, Broadcasting and Communication, vol. 12, no. 3, pp. 9-13, Jun. 2012.

저자 소개

이 종 복(정회원)

- 1964년 8월 20일생.
- 1988년 : 서울대 컴퓨터공학과 졸업.
- 1998년 : 동 대학 전기공학부 졸업(공박).
- 1998년 ~ 2000년 : LG반도체 선임연구원.
- 2000년 ~ 현재 : 한성대 전자정보공학과 교수

- Tel : 02-760-4497
- Fax : 02-760-4435
- E-Mail : jblee@hansung.ac.kr

<관심분야 : 마이크로 프로세서, 티코어 프로세서, 텐서 프로세서 유닛>

※ 본 연구는 한성대학교 교내학술연구비 지원과제임.