



저작자표시-동일조건변경허락 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.
- 이차적 저작물을 작성할 수 있습니다.
- 이 저작물을 영리 목적으로 이용할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



동일조건변경허락. 귀하가 이 저작물을 개작, 변형 또는 가공했을 경우에는, 이 저작물과 동일한 이용허락조건하에서만 배포할 수 있습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

2010년 8월
박사학위 논문

듀얼 FPU 와 열전도 지연 공간을
활용한 저온도 마이크로프로세서
구조 설계

조선대학교 대학원

컴퓨터학과

이병석

듀얼 FPU 와 열전도 지연 공간을 활용한 저온도 마이크로프로세서 구조 설계

Thermal-Aware Microprocessor Design utilizing
Dual FPU and Heat Conduction Delay Space

2010 년 8 월 25 일

조선대학교 대학원

컴퓨터학과

이병석

듀얼 FPU 와 열전도 지연 공간을
활용한 저온도 마이크로프로세서
구조 설계

지도교수 이 정 아

이 논문을 이학 박사학위신청 논문으로 제출함

2010 년 4 월

조선대학교 대학원

컴 퓨 터 학 과

이 병 석

이병석의 박사학위논문을 인준함

위원장	조선대학교	교수	<u>李 曉 浩</u> (인)
위원	조선대학교	교수	<u>丁 日 鎭</u> (인)
위원	조선대학교	부교수	<u>牟 相 晚</u> (인)
위원	전남대학교	조교수	<u>金 慈 弘</u> (인)
위원	조선대학교	교수	<u>李 禎 娥</u> (인)

2010 년 6 월

조선대학교 대학원

목 차

I. 서론	1
A. 연구 배경 및 목적	1
B. 연구 내용.....	5
C. 논문의 구성	7
II. 마이크로프로세서 발열 관리 연구 동향	8
A. 마이크로프로세서 온도 모델링	8
1. HotSpot 온도 모델	8
2. 성능계수기를 이용한 온도 모델.....	9
B. 마이크로프로세서 온도 관리 기법	11
1. 정적인 온도 관리 기법	12
2. 동적인 온도 관리 기법	15
III. 부동소수점 연산을 위한 마이크로프로세서 발열 관리 기법.....	18
A. 발열 관리 조건	18
B. 동적 이관 구조	21
C. 동적 이관을 위한 유닛 선택 기법	25
1. 일정 비율 기준	27
2. 주 유닛의 온도 기준.....	28
3. 동작중인 유닛의 온도 기준	29
4. 유닛의 온도와 냉각 임계 온도	30

5. 동작 유닛과 유틸 유닛의 온도 차	31
D. 열전도 지연 공간	33
IV. 실험 및 결과	38
A. 실험 방법	38
1. 모의 실험 도구의 구조	38
2. 실험 환경	40
3. 마이크로프로세서 코어의 플로어플랜	45
B. 실험 결과 분석	50
1. 동적 이관을 위한 유닛 선택 기법 분석	50
2. 열섬 현상 분석	58
3. 발열 안정성 평가	65
4. 마이크로프로세서 성능 평가	67
V. 결론 및 향후 연구	72
참고문헌	73

표 목 차

(표 1) 온도적 RC모델과 열 역학 비교.....	9
(표 2) 정수형과 부동소수점형 벤치마크에서 유닛의 최고 온도.....	20
(표 3) Alpha 21264 마이크로프로세서 특성.....	41
(표 4) SPEC CPU2000 부동소수점형 벤치마크 종류.....	44
(표 5) 유닛의 면적.....	46
(표 6) 플로어플랜 구성에 따른 칩의 면적과 증가 비율.....	46
(표 7) 유닛 선택 기법에 대한 온도 실험 결과.....	57
(표 8) 동적 온도 관리 기법 적용시 유닛 선택 기법의 성능 비교.....	57
(표 9) ev6 플로어플랜을 기준으로 한 온도 차이 결과.....	65
(표 10) ev6 플로어플랜을 기준으로 한 성능 개선 비교.....	70

그림 목 차

(그림 1) 무어의 법칙에 따른 마이크로프로세서 발전.....	1
(그림 2) 마이크로프로세서의 발열 추세	2
(그림 3) 마이크로프로세서 온도 관리 기법 분류	11
(그림 4) 활성 이관 기법의 구성 예	13
(그림 5) Alpha 21264 마이크로프로세서의 유닛에 대한 최고 온도	19
(그림 6) 부동소수점 가산기 동적 이관 구조	23
(그림 7) 동적 이관 기법을 위한 구조.....	26
(그림 8) 일정 비율이 기준인 정적 이관 기법의 과정.....	27
(그림 9) 주 유닛의 온도가 기준인 동적 이관 기법의 과정	28
(그림 10) 동작중인 유닛 온도가 기준인 동적 이관 기법의 과정	29
(그림 11) 동작 유닛의 온도와 냉각 임계 온도가 기준인 동적 이관 기법의 과정	30
(그림 12) 동작 유닛과 유휴 유닛의 온도 차를 기준으로 유닛을 활성화 하는 과정	32
(그림 13) 열전도 지연 공간 배치 구조.....	34
(그림 14) 인접한 유닛에 대한 HotSpot RC 모델	35
(그림 15) 마이크로프로세서 수준의 모의 실험 도구 구조	40
(그림 16) 플로어플랜 구성 예제와 할당되는 코어 영역의 위치.....	43
(그림 17) ev6 플로어플랜	48
(그림 18) ev6 _{+FPAdd2} 플로어플랜	48
(그림 19) ev6 _{+FPAdd2+HCDS} 플로어플랜	49

(그림 20) SAM-1:1 기법에서 유닛의 선택과 온도 변화	53
(그림 21) DAM-PU 기법에서 유닛의 선택과 온도 변화	54
(그림 22) DAM-AU 기법에서 유닛의 선택과 온도 변화	54
(그림 23) DAM-CT 기법에서 유닛의 선택과 온도 변화	55
(그림 24) DAM-DT 기법에서 유닛의 선택과 온도 변화	55
(그림 25) 유닛 선택 기법에 따른 부동소수점 가산기 유닛의 최고 온도	57
(그림 26) mgrid 벤치마크 실행 결과에 따른 유닛의 최고 온도 분포	60
(그림 27) mgrid 벤치마크 실행에 따른 온도 변화와 DTM, DAM-DT 수행 결과	63
(그림 28) 플로어플랜 종류별 mgrid 벤치마크 결과	64
(그림 29) 동적 온도 관리 기법의 설정과 플로어플랜에 따른 마이크로프로세서의 최고 온도	66
(그림 30) 동적 온도 관리 기법의 설정과 플로어플랜에 따른 부동소수점 벤치마크 실행 시간 (긴급 단계 온도는 85℃)	68
(그림 31) 동적 온도 관리 기법의 설정과 플로어플랜에 따른 부동소수점 벤치마크 실행 시간 (긴급 단계 온도는 90℃)	69

ABSTRACT

Thermal–Aware Microprocessor Design utilizing Dual FPU and Heat Conduction Delay Space

By Lee, Byeong–Seok

Advisor: Prof. Lee, Jeong–A, Ph.D.

Department of Computer Science,

Graduate School of Chosun University

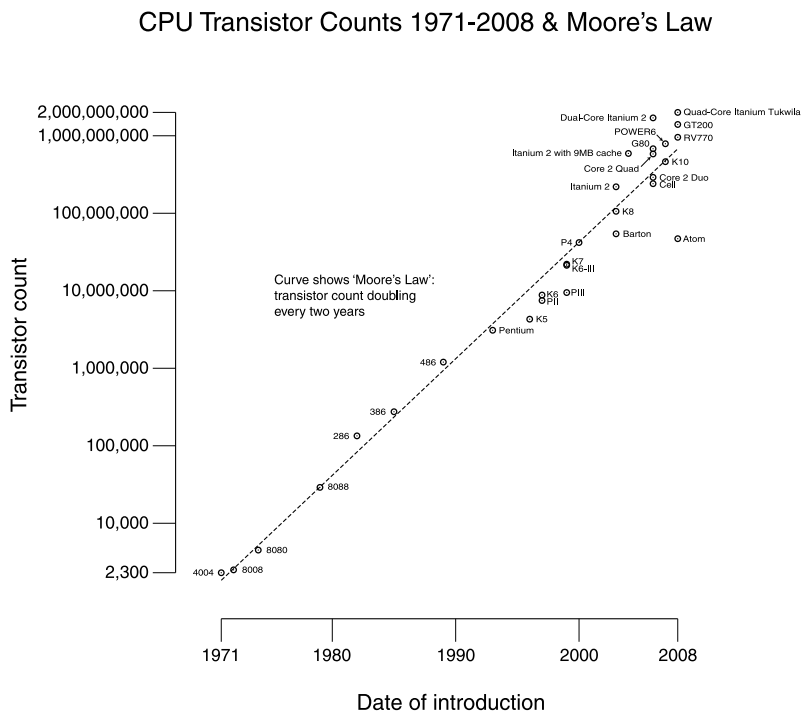
As the process technology scales down, the microprocessor performance improves but we face thermal problems due to the increased power density caused by the increase in the number of transistors. To solve thermal problems effectively, Dynamic Thermal Management (DTM) has been proposed and has been used to balance thermal reliability and cooling cost. However, the DTM introduces the problem of slowing down the microprocessor because dynamic voltage and frequency scaling, dynamic voltage scaling and instruction fetch throttling are applied. In this thesis, we propose the utilization of dual floating–point adder units to deal with thermal problems while minimizing the performance

loss for the floating-point applications. Only one of two floating-point adder units is used selectively in the proposed architecture, resulting in the reduction of the peak temperature in the microprocessor. Temperature has become a key constraint on the microprocessor performance. Activity migration, which enables moving computation between multiple replicated units, can be a solution for reducing peak junction temperature. Activity migration techniques can be divided into two categories: Static activity migration and dynamic activity migration. In static activity migration, the toggling time for each computation unit is fixed. Static activity migration requires simple hardware support but it cannot react to thermal emergencies effectively due to the fixed operation time per unit. On the contrary, in dynamic activity migration, the toggling time is decided dynamically based on the information from the digital thermal sensor such that thermal problems can be handled as they are detected by the sensor and comparator. We also propose a new floorplanning technique, which includes a Heat Conduction Delay Space (HCDS) in the microprocessor for overcoming the thermal problem of Heat Conduction between adjacent hot units. To estimate the temperature with these proposed schemes, we extended the DTM version by modifying the Watch, HotSpot, and hotfloorplan tools. From the experiments, we show that the peak temperature drops by 5.3°C on the average (maximum 10.8°C) for the microprocessor where the DTM with dual FPU and Heat Conduction delay space is utilized. In addition, we show that the performance of the microprocessor is improved by 45% on the average by reducing the stall time due to the DTM.

I. 서론

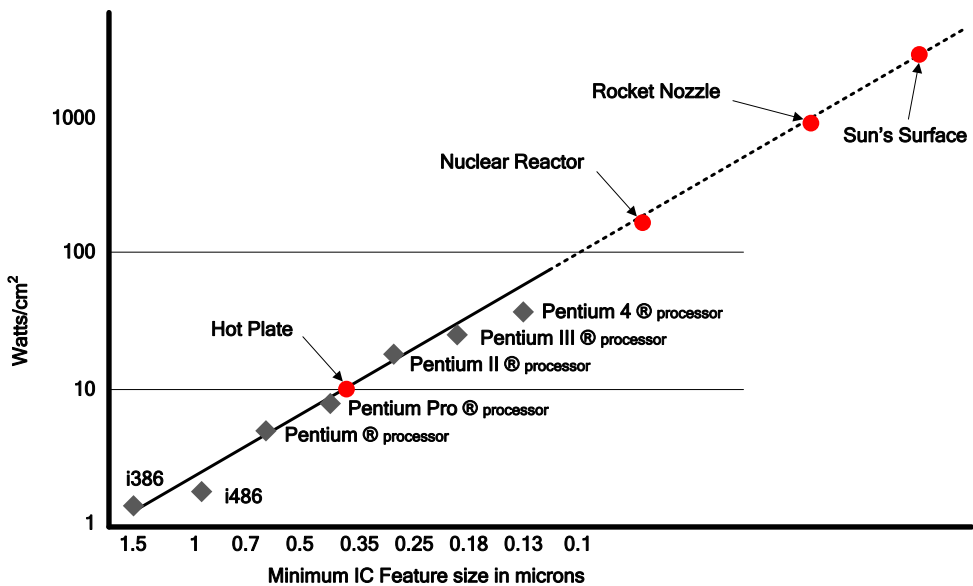
A. 연구 배경 및 목적

무어의 법칙(Moore's Law)[1]으로 잘 알려져 있는 마이크로프로세서의 반도체 집적도는 (그림 1)과 같이 매우 빠르게 발전하고 있다. 1971년 세계 최초의 마이크로프로세서인 4004는 $10\mu\text{m}$ 반도체 공정에 약 2천 개의 트랜지스터 집적도로 개발되었으나, 2002년에 개발된 Intel Itanium®2 마이크로프로세서는 $0.18\mu\text{m}$ 반도체 공정에 약 6억 개의 트랜지스터가 칩에 집적되었다[2].



(그림 1) 무어의 법칙에 따른 마이크로프로세서 발전

반도체 공정 기술 발전으로 칩의 트랜지스터 집적도가 높아지고 성능이 향상되면서 소비 전력 증가와 함께 마이크로프로세서의 발열 현상이 심각한 문제로 등장하였다. 마이크로프로세서는 칩의 단위 면적당 소비하는 전력량이 높을수록 뜨거워진다. 즉, 마이크로프로세서가 소비하는 전력량이 많을수록, 칩의 면적이 작을수록 마이크로프로세서의 온도는 높아진다. 물론, 반도체 공정 기술 중 하나인 전압 스케일링(Voltage Scaling)로 인해 칩의 소비 전력량 감소가 가능하다. 하지만, 칩의 트랜지스터 크기는 더욱 작아지면서 단위 면적당 소비하는 전력량은 점점 늘어나고 있다. (그림 2)는 마이크로프로세서의 발열 문제가 매우 심각한 수준인 것을 단적으로 보여주고 있다. (그림 2)를 보면, 마이크로프로세서의 단위 면적당 소비 전력량은 이미 전열기 수준을 넘었으며, 지금과 같은 추세로 발전하면 머지 않아 원자로, 로켓 노즐에서 태양 표면의 수준까지 다다를 수 있다고 발표하였다[3].



(그림 2) 마이크로프로세서의 발열 추세

마이크로프로세서의 온도는 바로 칩의 신뢰성과 매우 밀접한 관계가 있다[4]. 칩 일부분의 온도가 급격하게 높아지는 열섬(Hotspot) 현상은 마이크로프로세서가 오동작하거나 심할 경우에는 칩의 회로가 손상되어서 신뢰성을 저하시키는 원인이 되기도 한다. 온도가 높은 환경에서는 트랜지스터의 스위칭 속도가 느려지면서 회로가 동작 주파수 이내에 실행할 수 없었어 오동작하는 경우가 발생할 수 있다. 또한, 마이크로프로세서가 높은 온도에서 동작하면 노화(Aging) 현상이 가속화되면서 수명이 줄어들거나, 심한 경우에는 칩이 동작하지 않는 문제가 발생할 수 있다.

칩의 냉각을 위해 일반적으로 사용되는 히트 스프레더(Heat Spreader), 방열판(Heat Sink), 냉각 팬(Cooling Fan) 등을 이용한 물리적인 냉각 방법들은 물리적인 한계와 함께 냉각 비용이 상승하는 문제점을 가지고 있다[5]. 최근에는 심각한 발열 현상을 보이는 고성능 마이크로프로세서 뿐만 아니라 비용과 성능에 민감한 임베디드 마이크로프로세서 또한 디자인 특성으로 인해 공간적인 제약 조건을 만족하면서 최대한 열섬을 줄여 열적 분포(Thermal Profile)를 균등하게 하는 온도 지향적 설계가 요구되고 있다[6]. 따라서 최신 마이크로프로세서를 설계할 때에는 성능 향상과 신뢰성 확보를 위해 발열 문제를 반드시 고려해야 한다.

마이크로프로세서의 발열 문제를 해결하기 위해 주로 사용되는 동적 온도 관리(DTM: Dynamic Thermal Management)[7] 기법은 냉각 비용을 감소시킴과 동시에 칩의 신뢰성을 높이는 장점이 있지만, 마이크로프로세서의 성능이 저하된다는 단점이 있다. 그러므로 마이크로프로세서의 발열 문제를 해결하면서 동시에 성능 저하를 최소화할 수 있는 해결책이 요구된다. 지금까지 수행된 성능 저하를 최소화 하면서 발열 문제를 해결하기 위한 기법들은 대부분 마이크로프로세서 내에서 가장 뜨거운 정수 레지스터 파일(Integer Register File)에 집중되었다. 하지만 정수 레지스터 파일의 온도를 효과적으로 제어하더라도, 부동소수점 연산(Floating-

Point Operation)이 많은 응용 프로그램을 수행하면서 부동소수점 연산과 관련이 있는 유닛의 발열로 인해 칩의 안정성과 성능에 문제가 발생할 수 있다[8]. 또한, 멀티미디어 재생 기술에서 서비스 품질(QoS: Quality of Service)을 유지하기 위해서는 동적 온도 관리 기법의 사용을 최소화하는 기법을 요구하고 있다[9]. 대부분의 부동소수점 응용 프로그램은 고속 연산을 요구하기 때문에 동적 온도 제어로 인한 성능 감소 또한 최소화 되어야 한다.

DSP(Digital Signal Processor)나 멀티미디어와 같은 부동소수점 응용 연산을 주로 수행하는 마이크로프로세서에 대한 발열 연구는 거의 이루어지지 않다가, 최근 들어서 GPU(Graphics Processing Unit)에서 안개 효과(Fog Effect), 텍스처 매핑(Texture Mapping)에 대한 발열 분석[10], MPEG-2와 MPEG-4 비디오 디코딩에서 GOP(Group on Picture)의 작업 부하를 추정하여 동적 온도 관리에 적용한 기법[11, 12], 멀티미디어 응용에서 서비스 품질(QoS) 만족을 위해 MPEG-4, H.264/AVC 등 비디오 디코딩 분야에서 동적 온도 관리에 따라 동영상 프레임율 제어하는 기법[9] 등이 연구되었다.

지금까지 마이크로프로세서의 발열 문제에 대한 연구 중에서 부동소수점 연산과 관련된 연구는 부동소수점 응용 프로그램 실행에 따른 발열 현상 분석, 멀티미디어 재생에 따른 발열 분석 및 온도 제어 등 일부 제한된 분야에서 연구가 진행되었다. 일반적인 정수 연산과 비교하면 부동소수점 연산에 관한 연구는 매우 부족하다. 특히, 하드웨어 동적 온도 관리 기법 연구는 매우 부족하다. 따라서 본 논문에서는 부동소수점 응용 프로그램을 수행하는 경우 마이크로프로세서 내부 온도를 효과적으로 관리 하기 위한 구조를 설계하고자 한다.

B. 연구 내용

마이크로프로세서 발열 관리는 칩의 안전성과 성능을 결정하는 중요한 설계 요소이다. 지금까지 마이크로프로세서 구조적인 발열 관리 연구는 유닛의 온도가 가장 뜨거운 정수 레지스터 파일에 집중되었다. 하지만, 정수 응용 프로그램과 부동소수점 응용 프로그램을 수행하는 유닛은 다르다. 따라서 새로운 발열 연구가 필요하다. 최근 스마트 폰(Smartphone), PMP(Portable Media Player), 휴대용 게임기 등은 멀티미디어, 게임, 이동 통신을 위해 고성능의 부동소수점 연산 처리를 요구하고 있다. 모바일 시스템에서 발열 문제를 해결에 있어 가장 큰 제한 사항이 있다. 휴대성과 디자인을 강조하는 모바일 임베디드 시스템은 제품의 크기로 인해 많은 부피를 차지하는 물리적인 냉각 기법을 적용할 수 없다. 따라서 동적 온도 관리 기법을 사용하여 마이크로프로세서의 발열을 관리해야 한다. 여기서 또 하나의 새로운 문제가 발생한다. 동적 온도 관리 기법의 사용에 따른 성능 저하 현상이다. 동적 온도 관리 기법은 기본적으로 마이크로프로세서의 성능을 저하시키는 방법으로 칩의 온도를 제어한다. 이는 고성능을 요구하는 부동소수점 응용 프로그램을 수행에 있어 우선적으로 해결해야 할 문제이다. 따라서 동적 온도 관리 기법을 사용함에 있어 성능 저하를 최소화시키는 연구가 필요하다. 이를 해결하는 방법은 매우 간단하다. 바로 칩의 온도를 낮추면 된다. 칩의 온도가 낮을수록 동적 온도 관리 기법이 사용하는 시간이 줄어들면서 자연스럽게 낮은 성능으로 마이크로프로세서가 동작하는 시간을 줄일 수 있다.

본 논문에서는 부동소수점 응용 프로그램을 수행할 때의 발열 문제와 이로 인한 성능 저하 문제를 해결하기 위하여 듀얼 부동소수점 가산기 구조의 중복 배치와 플로어플랜 상에서 전략적인 열전도 지연 공간 추가를 제안한다. 부동소수점 응용 프로그램은 정수 응용 프로그램과는 다르게 부동소수점 연산기의 발열이 심한

편이기 때문에, 동일한 기능을 수행하는 또 하나의 부동소수점 연산 유닛을 중복 배치하여 상호 접근을 분산시킴으로써 온도 상승을 지연시킬 수 있다. 특히, 분산된 부동소수점 가산기 유닛을 주변 유닛의 발열을 고려하면서 동작할 유닛을 선택하는 동적 이관(DAM: Dynamic Activity Migration) 기법을 제안한다. 이와 더불어, 온도가 높은 유닛으로부터 열전도(Heat Conduction)로 인해 인접한 유닛의 온도가 함께 상승하는 문제를 해결하기 위하여 열전도 지연 공간(HCDS: Heat Conduction Delay Space)을 추가 배치하는 방법도 제안한다.

본 논문이 연구하고 제안한 구조와 기법에 대한 결과를 분석하기 위해서는 모의 실험 프로그램과 벤치마크 프로그램이 필요하다. 모의 실험 프로그램은 마이크로프로세서 구조 레벨에서 수정이 가능해야 한다. 그리고 벤치마크 프로그램의 수행에 따라 마이크로프로세서의 유닛이 소비하는 전력을 계산할 수 있어야 한다. 이와 함께 마이크로프로세서 온도 모델을 통해 유닛의 온도를 분석할 수 있어야 한다. 또한 동적 온도 관리 기법의 사용에 따라 마이크로프로세서의 성능을 제한시킬 수 있어야 한다. 본 논문은 모의 실험 프로그램에 듀얼 부동소수점 구조, 동적 이관 기법, 열전도 지연 공간을 실험할 수 있도록 기능을 추가한 후, 벤치마크 프로그램을 이용하여 모의 실험을 진행한다. 그리고 동적 온도 관리 기법을 사용했을 때와 비교하여 본 논문이 제안한 기법에 대한 발열 안전성과 성능을 비교 분석한다.

C. 논문의 구성

본 논문의 구성은 다음과 같다. I장 서론에 이어서 II장에서는 지금까지의 마이크로프로세서의 발열 연구 동향을 설명한다. 현재 많이 사용하는 마이크로프로세서의 온도 모델링 종류를 설명하고, 정적인 발열 관리 기법과 동적인 발열 관리 기법을 함께 설명한다. III장은 부동소수점 응용 환경에서 본 논문이 제안한 듀얼 부동소수점 가산기 구조와 이를 위한 동적 이관 기법을 통한 동적인 발열 관리 기법을 기술한다. 또한, 열전도 지연 공간을 추가로 배치한 정적인 발열 관리 기법을 함께 기술한다. IV장에서는 모의 실험 구조와 환경을 설명하고, 벤치마크 부동소수점 응용 프로그램을 이용하여 본 논문이 제안하는 기법에 대한 결과를 기술한다. 마지막으로 IV장은 본 논문의 연구 결론과 향후 연구 방향을 기술한다.

II. 마이크로프로세서 발열 관리 연구 동향

A. 마이크로프로세서 온도 모델링

마이크로프로세서 설계 시 온도에 대한 모델링과 모의 실험은 매우 중요하다. 이는 마이크로프로세서의 온도 안정성을 미리 시험해볼 수 있고, 제품의 타임 투마켓(Time to Market)과 비용을 줄이는 데에도 중요한 역할을 한다. 본 논문에서는 가장 많이 사용하고 있는 HotSpot 모델[4]과 마이크로프로세서의 런타임에 온도를 측정하기 위해 HotSpot에서 약간 변형된 성능계수기[13]를 이용한 모델을 간단하게 설명한다.

1. HotSpot 온도 모델

HotSpot 온도 모델은 현재 컴퓨터 구조 커뮤니티에서 가장 널리 쓰이고 있는 모델이다. HotSpot 온도 모델은 온도적 RC 모델에 기초하고 있으며, 마이크로프로세서 상의 유닛 또는 부분들을 전기저항과 전기용량으로 모델링 하였다. 실리콘이 실제 칩 부분이고, 칩 안에는 여러 컴포넌트 또는 유닛으로 나누어져 있다. 그리고 히트 스프레더와 방열판도 같이 모델링 되어 있다. (표 1)은 온도적 RC 모델과 실제 열역학에서 쓰이는 요소들을 비교한 표이다. RC 모델은 4차 Runge-Kutta 방법을 통해 해를 구할 수 있으며, (표 1)에서 보이는 바와 같이 전압을 온도 차이로 바꿀 수 있다. HotSpot은 열전도에 중용한 역할을 하는 인접성을 고려하기 위하여 플로어플랜(Floor Plan) 정보를 입력 받는다. 여기서 인접성은 열이 전도되는 데에 매우 중요한 역할을 한다. 플로어플랜은 각각의 유닛들이 어느 위치에 배치되어 있

는지를 나타내는 정보이다.

(표 1) 온도적 RC모델과 열 역학 비교

열량	단위	전기량	단위
열(Q)	J	전하(q)	C
온도(T)	K	전위차(ϕ)	V
온도 차	K	전압(V)	V
열 흐름, 전력(P)	W	전류(I)	A
열전도율(k)	$W/(m \cdot K)$	전기 전도체(σ)	$1/(m \cdot \Omega)$
열적 흐름, 전력밀도(q) = $-k\nabla T$ (푸리에 법칙)	W/m^2	전류밀도(j) = $-\sigma\nabla\phi$ (옴의 법칙)	A/m^2
열저항(R_{th})	K/W	전기저항(R)	$\Omega = V/A$
열용량(C_{th})	J/K	전기용량(C)	$F = C/V$
온도적 RC상수 $\tau_{th} = R_{th} \cdot C_{th}$	s	전기적 RC 상수 $\tau = R \cdot C$	s

2. 성능계수기를 이용한 온도 모델

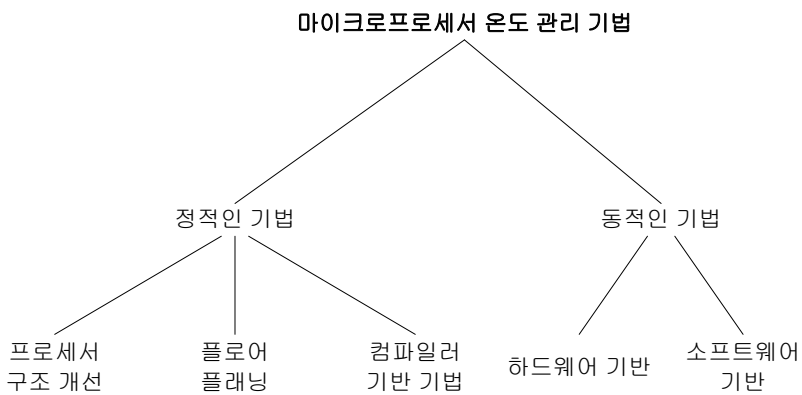
성능계수기를 이용한 온도 모델[13]은 마이크로프로세서를 실시간으로 온도를 측정하는 모델로, 성능계수기를 통해 각 유닛의 접근 횟수를 이용하여 각 유닛의 전력 소모량을 측정한다. 전력 소모량은 (식 1)과 같이 측정될 수 있다[14].

$$Power_{Total} = Access\ Rate \times Architectural\ Scaling \times Power_{Max} + Power_{non\ Gated\ Clock\ Power} \quad (\text{식 } 1)$$

(식 1)에서 *Access Rate* 는 성능계수기에서 추출한 접근 횟수이고, *Architectural Scaling* 은 아키텍처 구조에 따라서 적절한 전력 소모량 값을 만들기 위해 달라지는 계수, $Power_{Max}$ 는 접근 당 최대 파워 소모량, 그리고 $Power_{non\ Gated\ Clock\ Power}$ 는 클럭 파워 소모량이다. 각 유닛 별 파워 소모량을 추출한 후에 HotSpot을 이용하여 온도를 계산한다. Intel Pentium 4 마이크로프로세서에서 성능 계수기를 이용하여 실시간으로 파워를 측정하는 연구가 있다[14]. 최근에는 Intel Core2 Duo 마이크로프로세서를 모델로 온도 인지 동적 전압 및 주파수 스케일 (Temperature-Aware Dynamic Voltage and Frequency Scaling)에 관한 연구도 있다[15].

B. 마이크로프로세서 온도 관리 기법

지금까지 마이크로프로세서의 발열 문제를 해결하기 위한 온도 관리 기법에 대한 연구를 구분하면 (그림 3)과 같이 크게 정적인 기법과 동적인 기법으로 분류할 수 있다[16]. 정적인 기법으로는 발열에 강한 마이크로프로세서 구조를 설계하는 마이크로프로세서 구조 개선, 발열을 고려하여 각 유닛(Function Unit)을 코어(Core)에 배치하는 플로어플래닝(Floorplanning), 발열이 낮도록 프로그램의 명령어를 최적화하는 컴파일러 기반 기법 등이 있다. 동적인 기법으로는 마이크로프로세서가 자체적으로 온도를 제어하는 하드웨어 기반 기법과 운영체제(OS: Operating System) 및 응용 프로그램(Application Program)에서 온도를 제어하는 소프트웨어 기반 기법 등이 있다.



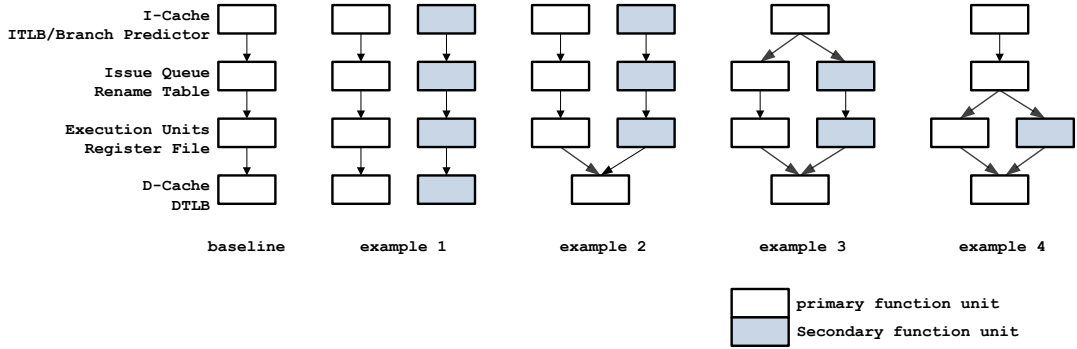
(그림 3) 마이크로프로세서 온도 관리 기법 분류

1. 정적인 온도 관리 기법

a. 마이크로프로세서 구조 개선

정적인 기법 중 하나인 마이크로프로세서 구조 개선은 칩에서 발생한 온도를 고려하여 마이크로프로세서를 설계하는 기법이다. 초기에는 슈퍼스칼라(Superscalar) 마이크로프로세서에서 발열에 대비해 두 개의 파이프라인을 각각 운영하는 이중 파이프라인 구조를 연구하였다[17]. 정상시에는 비순서 파이프라인(Out-of-Order Pipeline)를 사용하다가, 마이크로프로세서의 발열이 심해지면 비순서 파이프라인보다 전력 소모가 적은 순서 파이프라인(In-Order Pipeline)을 사용한다. 기존의 비순서 파이프라인은 다시 사용할 때까지 클럭 게이팅(Clock Gating)을 통해 클럭 공급을 중지한다. 클럭이 정지하면 활성 전력(Active Power)이 차단되면서 자연스럽게 비순서 파이프라인은 동작하지 않게 된다.

다른 연구로 활성 이관(Activity Migrating)[18] 기법이 있다. 이는 동일한 역할을 수행하는 유닛을 여러 개 두어서, 한 개의 유닛에 대한 사용이 많아지면 다른 유닛으로 실행을 전환한다. (그림 4)에서는 활동 이관을 지원하기 위한 구성을 보여준다. (그림 4)의 example 1에서는 데이터패스(Data path)상의 모든 유닛이 동일한 역할을 수행하는 보조 유닛(Secondary Function Unit)과 함께 구성되었으며, 주 유닛(Primary Function Unit)의 발열이 심할 경우에는 보조 유닛으로 전환하여 실행할 수 있다. (그림 4)의 example 2부터 example 4 역시 데이터패스 상에 있는 유닛을 필요에 따라 중복되게 구성하여 마이크로프로세서를 설계할 수 있다. 마이크로프로세서 설계자는 온도 시뮬레이션을 이용하여 어느 유닛을 중복하여 설계할 것인지 결정한다.



(그림 4) 활성 이관 기법의 구성 예

활성 이관 기법과 비슷한 연산 이관(Migrating Computation)[4] 기법은 마이크로프로세서에서 정수 레지스터 파일이 가장 뜨겁다는 점을 이용한다. 이 기법은 보조 정수 레지스터 파일을 추가하여 정수 레지스터 파일이 뜨거워지면, 정수 레지스터의 값을 보조 정수 레지스터 파일에 복사한 후, 보조 정수 레지스터 파일을 사용하여 정수 레지스터의 온도를 낮추는 대표적인 기법이다. 이러한 과정 중에서 연산 이관을 실행할 때에는 파이프라인을 멈춘 후 데이터를 다른 레지스터 파일에 복사해야 하므로 마이크로프로세서의 성능이 낮아지는 문제점이 있다.

정수 레지스터 파일의 온도를 낮추는 연구로 하나의 정수 레지스터 파일을 두 영역으로 분할한 후, 분할된 영역을 상호 번갈아 가면서 이용하는 연산 분할(Computation Division)[19] 기법이 있다. 이 기법 또한 분할된 레지스터의 위치가 상호 인접하기 때문에 서로 열이 전달되어 정수 레지스터 파일 영역에서 열섬이 발생하는 경우에는 마이크로프로세서의 실행을 중지해야 하는 문제점이 발생한다.

정수 레지스터 파일을 활성 이관 기법으로 구성하고, 두 개의 정수 레지스터 파일의 접근 비율을 조절하여, 동적 온도 관리 기법의 사용에 따른 성능 저하와 발열 문제를 해결하기 위한 듀얼 레지스터 파일 구조도 있다[8].

다른 정수 레지스터 파일의 구조 개선 연구로는 뱅크 레지스터 파일(Banked Register File)을 이용한 구조가 있다[20]. 이 기법은 정수 레지스터 파일이 일반적으로 마이크로프로세서에서 가장 뜨겁다는 사실에 착안하여 뱅크 레지스터 파일 구조를 적용, 정수 레지스터 파일의 온도를 낮추고 동적 온도 관리 기법이 사용하는 횟수를 줄여서 성능저하 문제를 해결하는 구조도 있다.

b. 플로어플래닝

플로어플래닝(Floorplanning)은 마이크로프로세서를 설계하는 EDA (Electronic Design Automation)나 CAD (Computer Aided Design) 도구에서 유닛을 배치하는 것을 의미한다. 코어에서 유닛의 위치와 상호 인접한 유닛의 종류에 따라 마이크로프로세서의 발열이 달라진다. 특히, 마이크로프로세서의 발열을 고려하여 플로어플래닝을 할 경우에는 발열이 심한 유닛을 인접하여 배치해서는 안 된다. 그 이유는 발열이 심한 유닛이 상호 인접하게 되면, 열전도 현상으로 인하여 두 유닛 모두 더 뜨거워지는 현상이 발생하게 된다. 따라서 발열이 심한 유닛이 인접하지 않도록 멀리 배치하는 것이 발열을 고려한 플로어플래닝 기법이라 할 수 있다.

연구용으로 널리 알려진 온도 시뮬레이션 도구인 HotSpot[21]에 포함되어 있는 HotFloorplan 도구는 플로어플래닝 알고리즘인 Simulated Annealing 알고리즘을 적용한 도구이다[22]. 이 도구는 칩 영역의 넓이, 온도, 그리고 선 길이(Wire Length)를 고려하여 유닛의 배치를 결정한다. Simulated Annealing 알고리즘을 기반으로 유닛 사이의 열전도량을 추가적으로 고려한 플로어플래닝 기법도 있다[23].

위에서 소개된 기법과는 달리 명령어당 클럭 사이클 수인 CPI (Clock cycle Per Instruction)를 고려한 기법들도 있다. 위의 두 기법 역시 유닛 사이의 선 길이

를 고려하지만, 이것만으로 정확한 성능을 반영할 수는 없다. 따라서, 성능을 비교할 수 있는 척도인 CPI를 이용하여 HotFloorplan와 비교하면 온도와 성능이 우수한 결과를 보여주었다[24, 25].

c. 컴파일러 기반 기법

컴파일러 기반 기법은 응용 프로그램의 명령어를 온도에 최적화 하도록 재구성하는 기법이다. 이 기법은 마이크로프로세서의 온도를 관리하기 위해 필요한 추가적인 하드웨어가 필요 없다. 따라서 하드웨어 추가에 따른 칩의 영역 면적 증가에 따른 비용 상승과 함께 소비 전력이 증가되는 문제가 발생하지 않는다.

컴파일러 기반 기법의 온도 최적화는 VLIW (Very Long Instruction Word) 구조를 중심으로 연구되어 왔다. VLIW 구조는 명령어 스케줄링을 컴파일러에 의존하므로 컴파일러 기반 기법을 통해 온도 최적화가 매우 중요하다. 이러한 기법들로 유닛을 균형되게 사용하여 특정 부분이 과열되지 않도록 하는 로드 밸런싱(Load Balancing) 기법[26]과 마이크로프로세서의 과열이 예상되는 지점에 컴파일러가 NOP (No-Operation) 명령어를 중간에 삽입하는 기법[27]이 있다.

2. 동적인 온도 관리 기법

a. 하드웨어 기반 기법

동적 온도 관리 기법인 DTM (Dynamic Thermal Management)은 마이크로프로세서에 내장된 디지털 온도 센서(Digital Temperature Sensor)를 이용하여 동적

으로 온도를 관리한다. 마이크로프로세서의 온도가 사전에 설정한 임계 온도 이상으로 상승하게 되면, 동작 전압과 클럭 주파수를 낮추는 DVFS (Dynamic Voltage & Frequency Scaling), 명령어의 인출 지연(Fetch Throttling), 저전력 모드(Low Power Mode) 전환 등으로 칩의 온도를 관리한다. 저전력 모드 전환은 클럭 게이팅(Clock Gating) 또는 전력 게이팅(Power Gating)을 사용하여 소비하는 전력을 줄임으로써 마이크로프로세서의 온도를 낮추는 방식이다.

동적 온도 관리 기법은 온도 관리를 위해 마이크로프로세서를 제어하는 동안은 성능이 저하되는 큰 문제점이 있다. 이러한 문제점과 관련하여 동적 온도 관리 기법이 성능에 미치는 영향에 관한 연구가 진행되었다[7]. 열은 전도되는데 일정한 시간이 필요하기에 동적 온도 관리 기법을 사용한다고 온도가 바로 내려가지는 않는다. 그래서 이러한 시간들을 고려하면 최적의 성능을 보이는 동적 온도 관리 기법을 고안할 수 있다.

동적 온도 관리 기법을 사용할 때, 성능 저하를 최소화 하기 위하여 계층적 동적 온도 관리(Hierarchical DTM)[28]와 하이브리드 동적 온도 관리(Hybrid DTM)[29] 기법 등이 제안되었다. 계층적 동적 온도 관리 기법은 마이크로프로세서의 온도에 따라서 계층적인 동적 온도 관리 기법을 적용하는 것으로, 마이크로프로세서의 온도가 올라감에 따라서 처음에는 필터 캐시(Filter Cache)같은 성능 저하를 발생시키지 않는 기법부터 적용하고, 마이크로프로세서의 온도가 더 높아졌을 때에는 클럭 게이팅 같이 온도를 크게 낮출 수 있지만 그 만큼 성능 저하를 유발시키는 기법을 적용한다. 하이브리드 동적 온도 관리 기법은 계층적 동적 온도 관리 기법과 비슷하지만, 마이크로프로세서의 온도에 따라 유연한 동적 온도 관리 기법을 적용하는 점이 다르다. 마이크로프로세서 상의 발열 스트레스(Thermal Stress)의 정도를 감지하여 스트레스가 심하지 않을 때는 명령어 인출 지연 기법

을 적용하고, 스트레스가 심할 경우 DVFS와 같은 강력한 동적 온도 관리 기법을 사용한다.

b. 소프트웨어 기반 기법

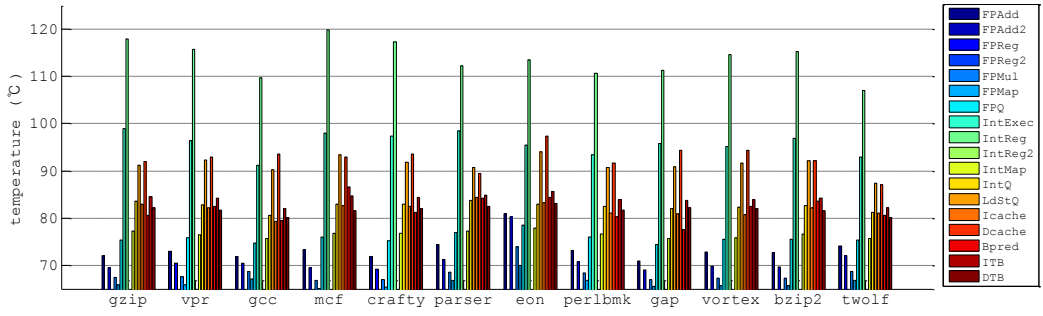
소프트웨어 기반 기법은 운영체제 또는 시스템 소프트웨어 계층에서 마이크로프로세서 온도 관리를 지칭한다. 운영체제 계층에서의 마이크로프로세서 온도 관리는 프로세스(Process) 또는 태스크(Task) 스케줄링 기법이 대부분이다. 이제까지의 연구들을 살펴보면, 발열에 강한 태스크 스케줄링 기법들은 성능 오버헤드를 최소한으로 줄이면서 뜨거운 태스크와 차가운 태스크를 혼합하여 마이크로프로세서의 과열을 막는 것이다. 최근 소프트웨어 기반 기법에 대한 연구들은 멀티 코어 혹은 SMT(Simultaneous Multi-Threading)를 대상으로 한 연구들이 많았다. 이는 태스크 스케줄링에 있어서 시간적 슬랙(Temporal Slack) 뿐 아니라 공간적 슬랙(Spatial Slack)도 같이 활용하여 성능 오버헤드를 최소화하면서 발열 관리를 할 수 있기 때문이다. 예를 들어 Heat-and-Run 기법은 스레드(Thread)들을 모아서 한 코어를 뜨겁게 한 뒤, 정지(Idle)한 코어로 옮기는 방식이다[30]. 다른 기법으로 뜨거운 태스크들은 우선 순위를 낮게 하고, 차가운 태스크들은 우선 순위를 높게 해서 스케줄링을 하는 기법이 제안되었다[31, 32]. 이 기법은 긴급 온도(Thermal Emergency) 상황이 오면 하드웨어에서 지원하는 클럭 게이팅 등의 기법을 이용하여 소프트웨어와 하드웨어의 협업을 통한 발열 관리를 제공한다. 열 균형(Heat Balancing), 뜨거운 작업의 지연 실행(Deferred Execution of Hot Jobs) 역시 멀티 코어 또는 멀티 스레딩 마이크로프로세서에서 존재하는 공간적 슬랙을 이용한 기법[33]이라고 볼 수 있다.

III. 부동소수점 연산을 위한 마이크로프로세서 발열 관리 기법

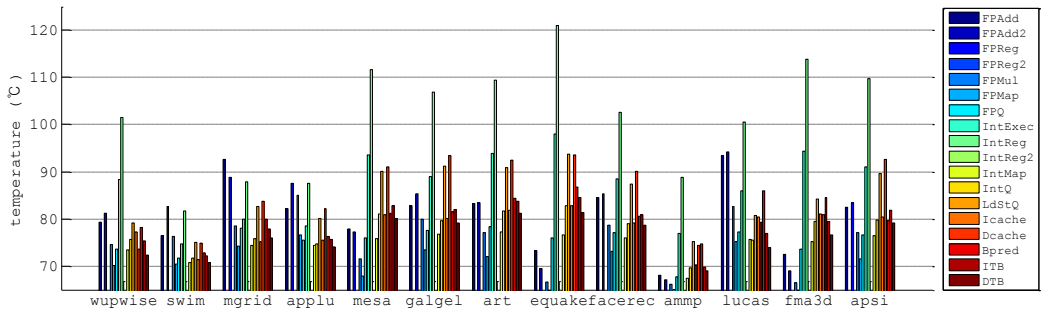
A. 발열 관리 조건

국제 반도체 기술 로드맵(ITRS: International Technology Roadmap for Semiconductors)에서는 반도체 공정이 $0.13\mu\text{m}$ 이하에서는 칩의 최고 적합 온도(Maximum Junction Temperature)가 90°C 보다 낮아야 한다고 발표하였다[34]. 이 조건을 만족하기 위하여 표본 집단 중에서 최악의 경우에 해당하는 응용 프로그램(Worst Typical Application)이 수행되는 동안의 온도를 추적하여 동적 온도 관리를 사용하는 기법이 제안되었다[35].

(그림 5)의 데이터는 Alpha 21264 마이크로프로세서에서 SPEC CPU2000 벤치마크[36]를 실행할 때, 각 유닛의 최고 온도를 추적한 결과이다(모의 실험 구성은 IV장에서 자세히 설명한다). (그림 5)의 (a)는 정수형 벤치마크 프로그램인 CINT2000 (Integer Component of SPEC CPU2000)의 실행 결과로써, 정수 연산과 관련이 있는 정수 레지스터 파일(IntReg) 유닛과 정수 연산기(IntExec) 유닛의 온도가 가장 높다는 것을 볼 수 있다. 부동소수점형 벤치마크인 CFP2000 (Floating Point Component of SPEC CPU2000)의 실행 결과인 (그림 5)의 (b)를 보면, 정수형 벤치마크와 비슷하게 정수 레지스터 파일(IntReg) 유닛과 정수 연산기(IntExec) 유닛의 온도가 가장 높은 것을 볼 수 있다. 하지만, swim, mgrid, applu, lucas 벤치마크 프로그램의 경우에는 부동소수점 연산과 관련이 깊은 부동소수점 레지스터 파일(FPReg), 부동소수점 가산기(FPAdd) 유닛의 온도가 높다는 것을 볼 수 있다.



(a) 정수형 벤치마크(CINT2000)



(b) 부동소수점형 벤치마크(CFP2000)

(그림 5) Alpha 21264 마이크로프로세서의 유닛에 대한 최고 온도

(그림 5)의 유닛 온도 결과를 이용하여 정수형 벤치마크 프로그램과 부동소수점형 벤치마크 프로그램 별로 최고 온도 순서로 유닛을 분류하면 (표 2)와 같다. (표 2)를 보면 정수 레지스터 파일(IntReg)과 정수 연산기(IntExec) 유닛의 온도가 모든 벤치마크에서 온도가 높은 것을 볼 수 있다. 그런데 이 다음부터는 정수형과 부동소수점형 벤치마크 프로그램에 대한 최고 온도 결과가 다른 것을 볼 수 있다. 정수형 벤치마크는 데이터 캐시(Dcache), 적재/저장 큐(LdStQ) 순서로 유닛의 온도가 높지만, 부동소수점형 벤치마크는 부동소수점 레지스터(FPReg), 적재

/저장 큐(LdStQ) 순서로 온도가 높은 것을 볼 수 있다. 또한, 최고 온도가 90℃보다 높은 유닛을 보면 정수형 벤치마크는 IntReg, IntExec, DCache, LdStQ 등 총 4개가 있지만, 부동소수점형 벤치마크는 부동소수점 레지스터 파일(FPReg)와 부동소수점 가산기(FPAdd) 유닛이 추가되어서, 정수형 벤치마크보다 2개 더 많은 총 6개의 유닛이 존재하고 있음을 볼 수 있다.

(표 2) 정수형과 부동소수점형 벤치마크에서 유닛의 최고 온도

최고 온도 순서	정수형 벤치마크(CINT2000)		부동소수점형 벤치마크(CFP2000)	
	유닛 이름	최고 온도(℃)	유닛 이름	최고 온도(℃)
1	IntReg	119.6	IntReg	121.0
2	IntExec	98.9	IntExec	98.1
3	Dcache	97.4	FPReg	94.2
4	LdStQ	94.0	LdStQ	94.7
5	Bpred	86.6	Dcache	93.6
6	ITB	85.7	FPAdd	93.5

B. 동적 이관 구조

부동소수점 응용 프로그램을 주로 수행하는 DSP나 멀티미디어 시스템은 시스템 품질(QoS)이 중요한 요소이다. 이를 만족하기 위해 부동소수점 응용 프로그램은 고속의 연산 장치를 통한 빠른 연산 시간을 요구한다. 따라서 동적 온도 관리의 사용 시간을 최소화하여서 마이크로프로세서의 성능 저하를 최소화 할 필요가 있다. 부동소수점 응용 프로그램이 실행되는 도중에는 부동소수점 연산에 필요한 유닛의 발열뿐만 아니라, 동적 온도 관리로 인한 성능 저하 문제까지 함께 고려해야 한다. 본 논문은 부동소수점 벤치마크 프로그램의 실행에 따른 유닛의 온도를 보여준 (그림 5)의 (b)를 참고하여 열섬 가능 인자(Hotspot-Possible Factor)를 선택한다. 먼저, 부동소수점 레지스터 파일(FPReg) 유닛을 열섬 가능 인자로 가정한다. 부동소수점 레지스터 파일을 연산 이관(Migrating Computation)[35] 기법을 적용한다면, 보조 레지스터 파일 유닛에 데이터를 복사하는 과정에서 마이크로프로세서의 성능이 저하되는 문제가 발생할 수 있다. 다른 방법인 듀얼 레지스터 파일 구조[8]를 적용하면 부동소수점 레지스터 파일 유닛의 발열 문제는 해결할 수 있지만, 부동소수점 가산기 유닛에서 발생하는 발열 문제까지는 해결할 수는 없다. 그리고 부동소수점 레지스터 파일(FPReg) 유닛은 빠른 연산을 위해 부동소수점 가산기(FPAdd)와 곱셈기(FPMu1) 유닛이 인접해야 한다. 듀얼 레지스터 파일 구조에서 두 개의 부동소수점 레지스터 유닛 중 하나는 부동소수점 가산기나 곱셈기 유닛과의 거리가 멀어지면서 데이터 전송 지연이 발생하여 성능이 저하되는 문제가 발생할 수 있다.

본 논문은 부동소수점 응용 프로그램을 실행할 때, 성능 저하를 최소화하면서 효과적으로 발열을 관리하기 위하여 부동소수점 가산기(FPAdd)를 열섬 가능 인자로 선택하고, 유닛의 발열에 대한 안정성 향상과 동적 온도 관리로 인한 성능 저하

를 최소화 하는 기법을 제안하고자 한다.

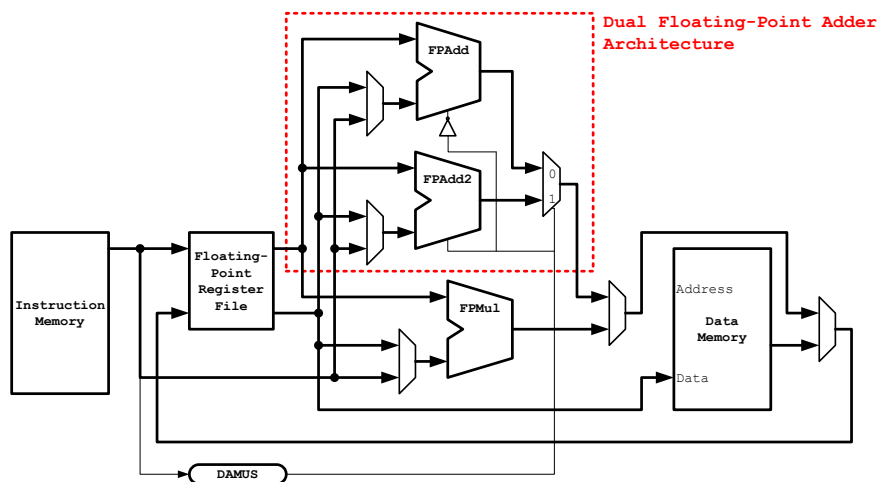
유닛의 온도는 해당 유닛이 동작 중에 소비하는 전력량(Power)과 비례한다. 따라서 소비 전력을 줄이면 유닛의 온도 또한 낮아지게 된다. (식 2)를 보면, 소비 전력인 P 는 유닛의 동작 빈도 수의 평균인 α , 부하 용량(Load Capacitance)인 C , 마이크로프로세서의 동작 주파수(Clock Frequency)인 f 에 비례하고, 공급 전압인 Vdd 에는 제곱으로 비례한다. 반도체 공정에 따라 결정되는 C 를 제외한 α, Vdd, f 를 낮추게 되면 소비 전력은 자연스럽게 줄일 수가 있다.

$$P = \alpha CVdd^2 f \quad (\text{식 2})$$

공급 전압인 Vdd 와 동작 주파수인 f 를 낮추어서 동적으로 온도를 관리하는 기법이 바로 DVFS이다. DVFS 기법은 매우 효과적인 온도 관리 기법이지만, 마이크로프로세서의 성능이 저하되는 문제점이 발생한다. 따라서 마이크로프로세서의 성능 저하 없이 유닛의 온도를 낮추기 위해서는 유닛의 동작 빈도인 α 를 줄이는 방법이 가장 효과적이다. 활성 이관(Activity Migration) 기법[18]이 바로 유닛의 동작 빈도 수를 낮추어서 마이크로프로세서의 온도를 관리하는 기법이다.

본 논문에서는 동적 온도 관리로 인해 마이크로프로세서의 추가적인 성능 저하 없이 부동소수점 가산기 유닛에서의 온도 상승을 최소화하기 위하여 부동소수점 가산기를 두 개로 분산하여 유닛의 동작 빈도를 분산시키는 듀얼 부동소수점 가산기 구조(Dual Floating-Point Adder Architecture)를 제안한다. 제안하는 부동소수점 가산기 구조는 보조 부동소수점 가산기를 추가로 배치함으로써 부동소수점 가산기에 대한 동작 빈도수를 분산하는 기술이다. 하나의 부동소수점 가산기 유닛이 동작하면 유닛이 전력을 소비하면서 유닛이 발열하지만, 동작하지 않는 다른 부

동소수점 가산기 유닛은 전력을 소비하지 않으면서 유닛이 냉각되는 원리를 이용하여 부동소수점 가산기 유닛의 온도를 관리한다. (그림 6)은 본 논문이 제안한 듀얼 부동소수점 가산기 구조이다.



(그림 6) 부동소수점 가산기 동적 이관 구조

(그림 6)에서 보여주고 있는 부동소수점 가산기 구조는 다음과 같이 동작한다. 먼저, 부동소수점 가산 명령어가 명령어 메모리(Instruction Memory)에서 인출되면, 명령어를 해석한 후 동적 유닛 선택기(DAMUS: Dynamic Activity Migration Unit Selector)가 두 부동소수점 가산기 유닛의 온도에 따라 동작할 유닛을 선택한다. 부동소수점 연산에 필요한 인자 값은 부동소수점 레지스터 파일로부터 주 부동소수점 가산기(FPAdd)와 보조 부동소수점 가산기(FPAdd2) 유닛 모두에 전송한다. 두 개의 부동소수점 가산기는 연산에 필요한 인자 값을 받지만, 동적 유닛 선택기(DAMUS)가 선택한 부동소수점 가산기 유닛만이 동작하면서 전력을 소비한다. 이와 같은 구조를 활용하여 부동소수점 연산 동작 빈도를 두 개의 부동소수점 가산기로 분산하면, 각 부동소수점 가산기의 소비 전력이 낮아지면서 발열

역시 낮아지게 될 것으로 기대된다. 마지막으로 부동소수점 연산을 수행한 부동소수점 가산기 유닛의 출력 값을 멀티플렉서(MUX)를 통하여 부동소수점 레지스터 파일에 저장한다. 제안하는 구조는 동적 온도 관리의 사용으로 인해 동작할 유닛으로 이관되는 과정에서 발생하는 지연 문제가 없다는 장점이 있다.

제안하는 듀얼 부동소수점 가산기 구조는 다음과 같은 문제점을 가지고 있다. 첫 번째는 하나의 부동소수점 가산기 유닛이 추가되는 만큼 칩 면적이 증가된다. 두 번째는 하나의 부동소수점 레지스터 파일 유닛에서 두 개의 부동소수점 가산기 유닛으로 데이터 전송 경로를 연결해야 하기 때문에 부동소수점 가산기 유닛과 부동소수점 레지스터 파일 유닛 사이의 데이터 패스가 복잡해진다는 점이다. 만약 부동소수점 레지스터 파일 유닛과 부동소수점 가산기 유닛 사이가 멀어지면 데이터 전송 경로는 더욱 복잡해지는 문제가 발생한다. 따라서 제안하는 구조에서 부동소수점 레지스터 파일 유닛과 부동소수점 가산기 유닛은 상호 인접해야 한다.

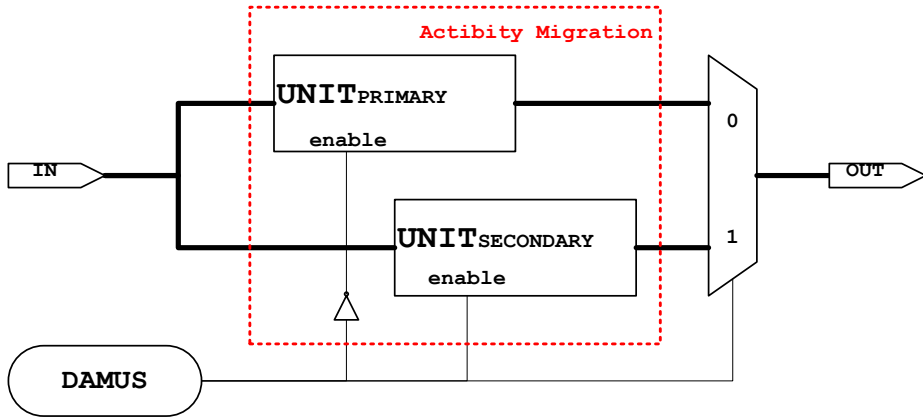
C. 동적 이관을 위한 유닛 선택 기법

활성 이관(Activity Migration) 기법을 수행하기 위해서는 동작할 유닛을 선택하는 기준이 필요하다. 지금까지는 일정한 비율로 동작할 유닛을 선택하는 기법등이 연구되었다[4, 8]. 그 결과, 동일한 값이 저장된 두 개의 정수 레지스터 파일을 1:2 비율로 접근하는 기법이 동적 온도 관리 기법을 사용하는 환경에서 발열 안전성과 마이크로프로세서의 성능이 우수하다는 연구 결과가 있었다[8].

본 논문은 위와 같이 일정한 비율로 동작할 유닛을 활성화 하는 기법을 정적 이관(SAM: Static Activity Migration) 기법이라고 정의하고자 한다. 정적 이관 기법은 부가적인 하드웨어가 적고, 유닛을 선택하는 방법이 단순하다는 장점이 있지만, 예상치 못한 원인으로 인하여 유닛이 과열되더라도 동적으로 대처 할 수 없다는 단점이 존재한다. 이러한 문제를 해결하기 위해 유닛의 발열에 따라 동적으로 대처할 수 있는 동적 이관(DAM: Dynamic Activity Migration) 기법이 필요하다.

동적 이관 기법은 유닛의 발열 문제를 즉각적으로 대처할 수 있다는 장점이 있지만, 온도 센서와 비교기(Comparator)의 추가로 인한 하드웨어 복잡도 증가와 유닛을 선택하는 기법이 잘못될 경우에는 하나의 유닛만이 과열 될 수 있다는 문제점이 있다. 따라서 동적 이관 기법은 유닛을 선택하는 기법에 따라 발열 안전성과 성능이 결정된다고 할 수 있다.

본 논문에서는 효율적인 동적 이관 기법 구현을 위해 다양한 유닛 선택 기법을 제안하고, 동작할 유닛을 선택하는 다양한 기준들을 비교, 분석하고자 한다. 마이크로프로세서의 성능 저하 없이 동적 이관을 수행하기 위하여 (그림 7)과 같은 구조로 유닛을 구성할 수 있다



(그림 7) 동적 이관 기법을 위한 구조

(그림 7)의 동적 이관 구조는 주 유닛($UNIT_{PRIMARY}$)과 보조 유닛($UNIT_{SECONDARY}$), 동작할 유닛 선택기(DAMUS), 멀티플렉서 등으로 구성된다. 각 유닛은 enable의 신호에 따라 유닛이 동작하며, 유닛 선택기(DAMUS)가 유닛의 동작을 제어한다. 유닛 선택기(DAMUS)는 동작할 유닛의 기준에 따라 '0' 또는 '1' 신호를 출력한다. 그리고 주 유닛($UNIT_{PRIMARY}$)의 enable 신호 핀 전단에는 인버터(Inverter)를 추가한다. 이러한 동적 이관 구조는 유닛 선택기(DAMUS)의 신호 값에 따라 주 유닛($UNIT_{PRIMARY}$) 또는 보조 유닛($UNIT_{SECONDARY}$)만이 동작하도록 구성할 수 있다. 데이터는 주 유닛($UNIT_{PRIMARY}$)과 보조 유닛($UNIT_{SECONDARY}$)에 같이 입력되지만, 유닛 선택기(DAMUS)가 선택한 유닛만이 동작하면서 전력 소비와 함께 열을 발생시킨다. 동작하지 않는 유닛은 전력 소비가 거의 없기 때문에 열을 발산하지 않고 생각된다. 마지막으로 멀티플렉서를 이용하여 동작중인 유닛만이 데이터가 출력되도록 한다.

다음은 정적 이관 기법과 동적 이관 기법에서 유닛 선택을 위한 기준과 이관 기법이 동작하는 과정을 설명한다.

1. 일정 비율 기준

지금까지 연구된 정적 이관 기법(SAM)으로, 일정한 비율에 따라 유닛을 활성화 하는 기법이다. 이 기법은 클럭에 의해 증가된 클럭 카운터를 기준으로 활성화 할 유닛을 선택한다. 클럭 카운터 값을 저장하는 레지스터를 R_{COUNT} , 하나의 유닛을 활성화 하기 위해 비교 값을 저장한 $R_{COMPARE}$, 주 유닛을 $UNIT_{PRIMARY}$, 보조 유닛을 $UNIT_{SECONDARY}$ 라고 할 때, 유닛을 활성화 하는 과정은 (그림 8)에서 보이는 바와 같다. 1단계에서 클럭 주기에 의해 클럭 카운터 값이 업데이트 되면, 2단계에서 미리 설정한 유닛 비율과 클럭 카운터 값을 비교한다. 3 ~ 4단계에서는 비교한 결과를 이용하여 3단계의 주 유닛이나 4단계의 보조 유닛을 선택한다.

```
1:  $R_{COUNT} = R_{COUNT} + 1$ 
2: IF ( $R_{COUNT} \& C_{COMPARE}$ ) THEN
3:   SELECT  $UNIT_{SECONDARY}$ 
   ELSE
4:   SELECT  $UNIT_{PRIMARY}$ 
```

(그림 8) 일정 비율이 기준인 정적 이관 기법의 과정

1:1 비율로 동작할 유닛을 선택하는 정적 이관(SAM) 기법은 클럭 카운터 값을 이용하여 간단하게 동작할 유닛을 선택 할 수 있다는 장점이 있다. 하지만, 유닛을 동작시킬 시간 간격에 따라 발열에 대한 성능 차이가 발생하는 문제가 있다. 유닛을 선택하는 시간 간격이 길어지면 하나의 유닛이 동작하는 시간이 길어지면서 발열하는 시간이 길어져서 유닛의 온도가 높아지는 현상이 발생할 수 있다. 이와 반대로 유닛 상호간 동작하는 시간 간격이 짧으면 유닛이 냉각할 수 있는 시간이 없으면서 두 개의 유닛 모두 온도가 상승하는 문제가 발생할 수 있다. 또한, 정

적인 이관 기법(SAM)은 외부의 영향에 따른 온도 변화에 동적으로 대처할 수 없다는 단점이 있다.

2. 주 유닛의 온도 기준

마이크로프로세서에 내장된 온도 센서를 이용하면 유닛의 발열에 따라 동적으로 보조 유닛을 동작 시킬 수가 있다. 이러한 기능을 이용하면 주 유닛의 온도가 미리 설정한 임계 온도(Threshold Temperature)보다 높을 때 보조 유닛이 동작하도록 선택하는 기법을 생각할 수 있다. 주 유닛의 온도를 $T_{PRIMARY}$, 유닛 활성을 결정하는 임계 온도를 $T_{THRESHOLD}$ 라고 할 때, 유닛을 선택하는 과정은 (그림 9)와 같다. 이 기법은 1단계에서 주 유닛의 온도가 미리 설정한 임계 온도보다 높을 경우에는 2단계처럼 보조 유닛이 동작하도록 선택한다.

```
1: IF ( $T_{PRIMARY} \geq T_{THRESHOLD}$ ) THEN  
2:   SELECT  $UNIT_{SECONDARY}$   
   ELSE  
3:   SELECT  $UNIT_{PRIMARY}$ 
```

(그림 9) 주 유닛의 온도가 기준인 동적 이관 기법의 과정

주 유닛의 온도를 기준으로 하는 동적 이관(DAM-PU: Dynamic Activity Migration - Primary Unit) 기법은 동작할 유닛을 선택 하는 기법이 단순하다는 장점이 있지만, 주 유닛의 온도가 임계 온도보다 높을 경우에는 보조 유닛만 지속적으로 동작하면서 오히려 보조 유닛이 과열 되는 문제점이 발생할 수 있다.

3. 동작중인 유닛의 온도 기준

주 유닛의 온도를 기준으로 동작할 유닛을 선택하는 동적 이관 기법의 문제를 해결하기 위해, 현재 동작중인 유닛의 온도 센서를 이용하여서, 동작중인 유닛의 온도가 임계 온도보다 높을 때, 동작하지 않는 다른 유닛을 선택하여 동작시키는 기법이다. 보조 유닛의 온도 값을 $T_{\text{SECONDARY}}$ 라고 할 때, 동작할 유닛을 선택하는 과정은 (그림 10)와 같다. 동작 과정은 크게 주 유닛이 동작할 때의 1 ~ 3단계 과정과 보조 유닛이 동작할 때의 4 ~ 6단계 과정으로 구분할 수 있다. 1단계에 주 유닛 동작하면서 2단계와 같이 발열 온도가 임계 온도보다 높을 경우에는 3단계처럼 동작하지 않는 다른 유닛을 선택한다.

```
1: IF (UNITPRIMARY) THEN  
2:   IF ( $T_{\text{PRIMARY}} \geq T_{\text{THRESHOLD}}$ ) THEN  
3:     SELECT UNITSECONDARY  
4: ELSE IF (UNITSECONDARY) THEN  
5:   IF ( $T_{\text{SECONDARY}} \geq T_{\text{THRESHOLD}}$ ) THEN  
6:     SELECT UNITPRIMARY
```

(그림 10) 동작중인 유닛 온도가 기준인 동적 이관 기법의 과정

동작중인 유닛의 온도를 기준으로 하는 동적 이관(DAM-AU: Dynamic Activity Migration - Active Unit) 기법은 주 유닛의 온도가 임계 온도보다 높을 경우 보조 유닛이 지속적으로 동작하는 문제점을 해결할 수 있지만, 두 유닛의 온도가 모두 임계 온도보다 높을 경우에는 상호 유닛이 번갈아 동작하면서 이로 인해 두 유닛 모두 온도가 동반하여 상승하는 문제가 발생할 수 있다.

4. 유닛의 온도와 냉각 임계 온도

동적 이관 기법(DAM)에서는 유닛의 발열과 냉각이 적절하게 조합되어야만 제대로 된 효과를 볼 수 있다. 따라서 냉각을 위한 새로운 기준이 필요하다. 이에 따라 동작중인 유닛의 온도가 임계 온도보다 높을지라도, 동작해야 할 유닛의 온도가 냉각 임계 온도(Cooling Threshold Temperature)보다 낮을 때까지 기다리는 기법을 생각할 수 있다. 냉각의 기준이 되는 냉각 임계 온도의 값을 T_{COOLING} 이라고 할 때, (그림 10)와 같이 1단계에서 주 유닛이 동작하고, 2단계와 같이 유닛의 온도가 임계 온도보다 높다 하더라도, 3단계의 조건에서 보조 유닛의 온도가 미리 설정한 냉각 임계 온도보다 낮을 경우에만 4단계처럼 보조 유닛을 선택하는 과정으로 동적 이관이 된다. 즉, 비활성화된 유닛이 냉각 임계 온도 이하로 냉각되어야만 동작을 이관시킬 수 있다.

```
1: IF (UNITPRIMARY) THEN  
2:   IF (TPRIMARY >= TTHRESHOLD  
3:     and TSECONDARY <= TCOLLING) THEN  
4:       SELECT UNITSECONDARY  
5: ELSE IF (UNITSECONDARY) THEN  
6:   IF (TSECONDARY >= TTHRESHOLD  
7:     and TPRIMARY <= TCOLLING) THEN  
8:       SELECT UNITPRIMARY
```

(그림 11) 동작 유닛의 온도와 냉각 임계 온도가 기준인 동적 이관 기법의 과정

동작중인 유닛의 온도와 냉각 임계 온도를 기준으로 하는 동적 이관 (DAM-CT; Dynamic Activity Migration - Cooling Temperature) 기법은 두 유닛의 온도

가 모두 임계 온도보다 높은 경우에는 동작하지 않고 있는 유닛이 냉각할 수 있는 시간을 주어서 두 유닛의 온도가 같이 올라가는 문제를 해결할 수 있다. 그러나, 동작하지 않는 유닛이 주변 유닛의 열전도로 인해 냉각 임계 온도만큼 온도가 떨어지지 않는 시점에는 동적 이관을 진행할 수 없어, 동작중인 유닛만이 계속 발열되는 문제가 있다.

5. 동작 유닛과 유휴 유닛의 온도 차

임계 온도를 기준으로 동적 이관 기법(DAM)을 위한 유닛 선택 기법을 보면, 동작하지 않는 유닛이 충분히 냉각할 수 있는 기회를 주는 것이 매우 중요하다는 것을 알 수 있다. 여기서 비활성화된 유닛의 냉각 기준이 고정되지 않고 유닛의 온도에 따라 유동적으로 변하도록 하여 유닛이 냉각할 수 있는 기회를 주는 기법을 생각할 수 있다. 즉, 동작중인 유닛의 온도가 임계 온도보다 높을지라도, 동작하지 않는 유닛의 온도가 일정한 차이만큼 충분히 냉각될 때까지 기다린 다음에 동작할 유닛을 선택하는 기법이다. 주 유닛과 보조 유닛의 온도 차이 값을 $S_{DIFFERENCE}$, 냉각 기준이 되는 온도 차이 값을 $T_{DIFFERENCE}$ 라고 할 때, (그림 12)과 같이 동작을 이관할 수 있다. 먼저, 1단계에서는 주 유닛과 보조 유닛의 온도 차이 값은 아날로그 비교기(Analog Comparator)를 이용하여 구할 수 있다. 2단계와 같이 주 유닛이 동작하면서, 3단계처럼 주 유닛의 온도가 임계 온도보다 높다 하더라도, 4단계와 같이 보조 유닛과의 온도 차이가 미리 설정한 온도 차이만큼 발생하는 경우에만 5단계의 보조 유닛을 선택하는 과정으로 유닛을 선택한다.

```

1: SDIFFERENCE = |TPRIMARY - TSECONDARY|
2: IF (UNITPRIMARY) THEN
3:   IF (TPRIMARY >= TTHRESHOLD
4:     and SDIFFERENCE >= TDIFFERENCE) THEN
5:     SELECT UNITSECONDARY
6: ELSE IF (UNITSECONDARY) THEN
7:   IF (TSECONDARY >= TTHRESHOLD
8:     and SDIFFERENCE >= TDIFFERENCE) THEN
9:     SELECT UNITPRIMARY

```

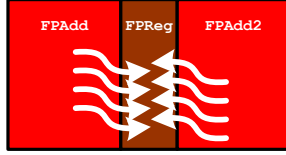
(그림 12) 동작 유닛과 유희 유닛의 온도 차를 기준으로 유닛을 활성화 하는 과정

동작중인 유닛의 온도와 유희 유닛간의 온도차를 기준으로 하는 동적 이관 (DAM-DT: Dynamic Activity Migration - Difference Temperature) 기법은 활성화 할 유닛이 냉각할 수 있는 시간을 주는 것과 동시에 냉각 임계 온도가 동적으로 변하므로 주변 유닛의 상황이 변하더라도 동작 이관을 수행할 수 있다는 장점이 있다.

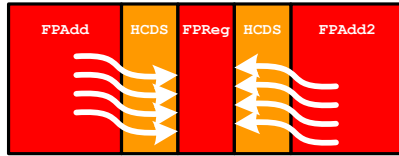
D. 열전도 지연 공간

열은 온도가 높은 곳에서 낮은 곳으로 흐른다. 즉, 두 개의 유닛이 서로 인접해 있으면 온도가 높은 유닛에서 온도가 낮은 유닛으로 열이 전도(Heat Conduction)된다. 그러므로 온도가 낮은 유닛이라도 온도가 높은 유닛에 인접해 있으면, 동작 빈도와는 관계 없이 온도가 상승하면서 발열 안정성과 마이크로프로세서의 성능에 큰 영향을 줄 수 있다. 본 논문에서는 열전도로 인한 유닛의 온도 상승 문제를 해결하기 위하여, 인접한 두 유닛 사이에 열전도 지연 공간(HCDS: Heat Conduction Delay Space)을 마이크로프로세서 코어(Core) 내부에 추가로 배치하는 정적 온도 관리 방법 또한 제안하고자 한다.

앞서 기술한 바와 같이 본 논문이 제안한 듀얼 부동소수점 가산기 구조는 데이터 패스로 인한 성능 저하를 최소화하기 위해서 (그림 13)의 (a)와 같이 배치가 이루어진다. 이러한 경우 두 개의 부동소수점 가산기(FPAdd와 FPAdd2) 유닛의 발열로 인하여 인접한 부동소수점 레지스터 파일(FPReg)의 온도가 더불어 상승하는 현상이 발생할 수 있다. 마이크로프로세서의 성능 저하를 최소화 하기 위해서는 부동소수점 레지스터 파일의 온도를 낮추는 방법 또한 필요하다. (그림 13)의 (b)처럼 두 개의 부동소수점 가산기 유닛(FPAdd와 FPAdd2)과 부동소수점 레지스터 파일 유닛(FPReg) 사이에 열전도 지연 공간(HCDS)를 추가로 배치하는 정적인 온도 관리 방법을 제안한다.



(a) 열전도 지연 공간이 없는 유닛 배치



(b) 열전도 지연 공간을 이용한 유닛 배치

(그림 13) 열전도 지연 공간 배치 구조

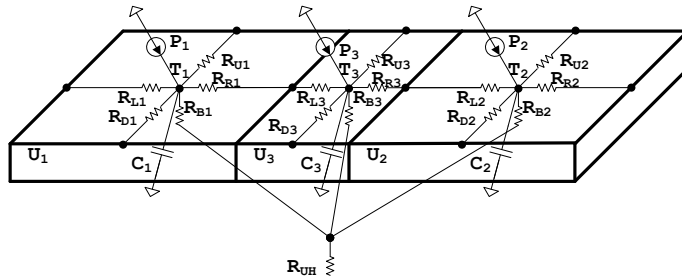
열전도 지연 공간에 대한 발열 성능은 HotSpot 온도 모델을 이용하여 실제 열 역학을 RC 모델로 표현할 수 있으며, 열 저항 네트워크를 이용하여 온도를 계산할 수 있다. 먼저, (식 3)을 보면, 열저항(Thermal Resistance)인 R_{th} 은 재료의 두께인 t 에 비례하고, 재료의 열 전도성(Thermal Conductivity)인 k 와 유닛의 면적인 A 에 반비례한다. 여기서 열 전도성의 경우 칩의 재료가 실리콘이면 $100W/m \cdot K$ 이고, 구리의 온도가 $85^{\circ}C$ 일 때에는 $400W/m \cdot K$ 이다. 열용량(Thermal Capacitance)인 C_{th} 는 단위 면적당 열 용량인 c 와 함께 두께와 면적에 비례한다. 여기서 열 용량의 경우 재료가 실리콘이면 $1.75 \times 10^6 J/m^3 \cdot K$ 이고, 구리이면 $3.55 \times 10^6 J/m^3 \cdot K$ 이다[4].

$$R_{th} = \frac{t}{k \cdot A}, \quad C_{th} = c \cdot t \cdot A \quad (\text{식 3})$$

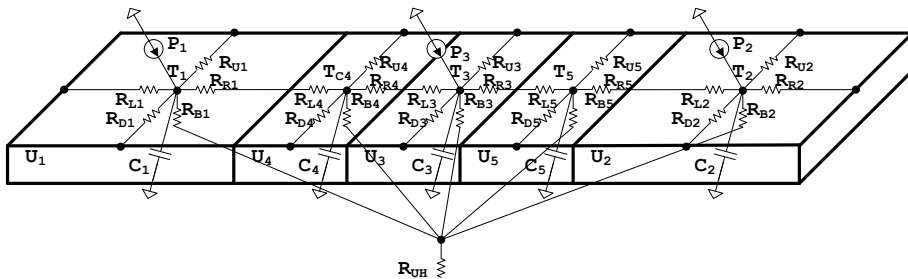
그리고 칩의 온도인 ΔT 는 (식 4)와 같이 구할 수 있다. 여기서 Q 는 열 흐름(Heat Flow)을 의미한다.

$$\Delta T = Q \cdot R \quad (\text{식 4})$$

인접한 유닛과 열전도 지연 공간을 (표 1)의 온도적 RC 모델을 이용하여 (그림 14)와 같이 HotSpot 모델로 표현할 수 있다. (그림 14)에서 U 는 유닛을 의미하고, P 는 전력(W), R 은 열 저항, C 는 열 용량, T 는 온도를 의미한다. 이 그림에서는 히트 스프레더와 방열판 모델은 생략하였다.



(a) 열전도 지연 공간을 사용하지 않는 온도 모델



(b) 열전도 지연 공간(U_4 와 U_5)을 사용한 온도 모델

(그림 14) 인접한 유닛에 대한 HotSpot RC 모델

(그림 14)의 (a)에서 U_3 는 R_{R1} 과 R_{L3} 열저항으로 U_1 과 연결되어 있으며, U_2 와 R_{L3} 과 R_{R2} 열저항으로 연결되어 있다. 이러한 열저항 네트워크를 통해 열은 온도가 높은 유닛에서 낮은 유닛으로 흐르게 된다. 그리고 코어에 있는 모든 유닛은 히트 스프레더의 열 저항 R_{UH} 에 연결되어 있으며, 히트 스프레더와 방열판을 통해 공기 중으로 열은 흐르게 된다. (그림 14)의 (b)는 U_3 주변에 열전도 지연 공간인 U_4 와 U_5 를 추가한 온도 모델이다. (그림 14)의 (b)를 보면 U_3 와 U_1 , U_3 와 U_5 사이에 열저항과 열용량이 추가된 것을 볼 수 있다. (그림 14)의 (a)와 (b)의 열 흐름과 관련하여 U_1 의 온도가 U_3 보다 높다고 가정할 때, U_3 유닛의 온도는 (식 5)와 (식 6)과 같다.

$$\Delta T_{3:1 \rightarrow 3} = \frac{T_1 - P_1(R_{R1} + R_{L3}) + T_3}{2} \quad (\text{식 5})$$

$$\Delta T_{3:1 \rightarrow 4 \rightarrow 3} = \frac{T_1 - P_1(R_{R1} + R_{L4} + R_{R4} + R_{L3}) + T_3}{2} \quad (\text{식 6})$$

(식 5)에서 $\Delta T_{3:1 \rightarrow 3}$ 는 U_3 의 온도이며, 열은 U_1 에서 U_3 로 흐른다. (식 6)의 $\Delta T_{3:1 \rightarrow 4 \rightarrow 3}$ 역시 U_3 의 온도지만, U_1 과 U_3 사이에 열전도 지연 공간인 U_4 가 추가되었고, 열은 U_1 에서 U_4 를 통과하여 U_3 으로 흐르게 된다. 따라서 $\Delta T_{3:1 \rightarrow 4 \rightarrow 3}$ 의 온도는 $\Delta T_{3:1 \rightarrow 3}$ 온도보다 $P_1(R_{L4} + R_{R4})/2$ 만큼 온도가 더 내려간다. 이외에도 열전도 지연 공간은 몇 가지 장점이 더 있다. 먼저, 열 용량의 증가로 인해 순간적인 발열에 대한 대책 효과가 더 좋아진다. (그림 14)의 (a)에서 U_1 , U_2 , U_3 유닛에 대한 열 용량은 $C_1 + C_2 + C_3$ 이다. 여기에 열전도 지연 공간을 추가한 (그림 14)의 (b)를 보면, 열 용량이 $C_1 + C_2 + C_3 + C_4 + C_5$ 로 증가한 것을 볼 수 있다. 다음으로, 열전도 지

연 공간은 칩의 냉각에도 탁월한 효과가 있다. U_1, U_2, U_3 유닛의 열이 공기 중으로 전달하는 경로인 히트 스프레더와의 연결을 보면, 열전도 지연 공간을 사용하지 않았을 경우에는 $\{R_{B1}, R_{B2}, R_{B3}\}$ 을 통해 열이 히트 스프레더로 흘러 가지만, 열전도 지연 공간을 추가하면 $\{R_{B1}, R_{B2}, R_{B3}, R_{B4}, R_{B5}\}$ 를 통해 공기 중으로 열이 흐르게 된다. 즉, 열을 배출하는 경로가 증가하면서 칩의 냉각 효과가 좋아진다.

열전도 지연 공간을 추가하면 인접한 유닛으로 발생한 열의 전달로 인해 유닛의 온도가 더 올라가는 문제를 완화시킬 수 있다. 물론, 열전도 지연 공간은 데이터를 전송하는 유닛 사이에 존재하면서 데이터 패스가 길어지는 문제가 발생한다. 이는 유닛간 데이터 전송에 필요한 주기가 증가하면서 성능이 저하되는 문제가 발생할 수 있다. 그렇지만, 데이터 전송 시간이 상대적으로 늘어나더라도 동적 온도 관리 기법의 사용으로 인한 성능 저하를 개선시키면 마이크로프로세서를 사용하는 듀티 사이클(Duty Cycle)이 증가하면서 성능이 향상될 수 있다[37].

IV. 실험 및 결과

본 장에서는 모의 실험을 위한 실험 도구의 구조, 실험 환경, 코어의 플로어플랜 구성을 설명한다. 실험 도구는 마이크로프로세서의 성능과 함께 소비 전력과 발열을 계산하고, 동적 온도 관리로 마이크로프로세서를 제어할 수 있는 통합된 실험 도구를 이용하였다. 그리고 마이크로프로세서의 코어에 부동소수점 가산기와 열전도 지연 공간을 추가하여 플로어플랜을 구성하였다. 마지막으로 벤치마크 응용 프로그램과 입력 데이터를 이용하여 모의 실험을 진행하였다.

모의 실험을 통하여 본 논문이 제안한 동적인 동작 이관 기법을 위한 활성화 유닛 선택 종류에 대한 특성과 발열 안정성을 분석한다. 또한, 동적인 동작 이관 기법을 적용한 부동소수점 가산기 구조와 열전도 지연 공간(HCDS)을 마이크로프로세서의 코어에 추가하였을 경우에 대해, 칩의 온도 변화를 살펴봄으로써 발열에 대한 안정성과 온도 상승에 따른 동적 온도 관리(DTM)의 동작으로 인한 성능 저하 정도를 평가한다.

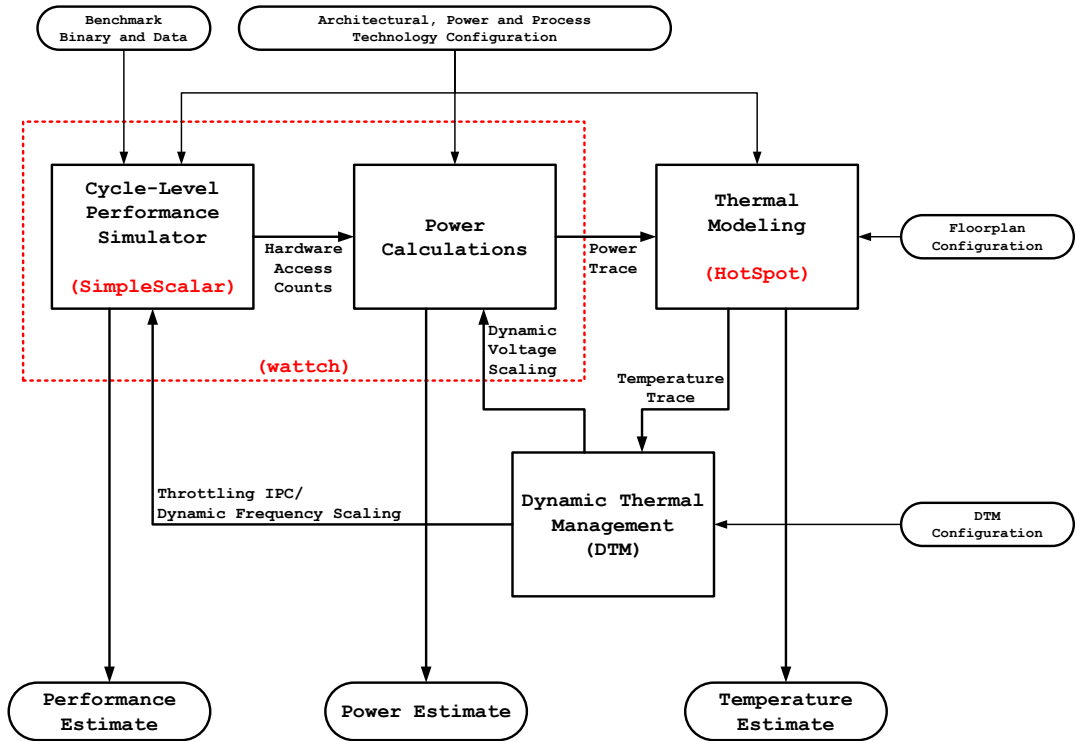
A. 실험 방법

1. 모의 실험 도구의 구조

제안된 부동소수점 응용 프로그램을 위한 저온도 마이크로프로세서 설계 기법의 효율성을 분석하기 위하여 본 논문에서는 마이크로프로세서 구조에 따른 성능, 소비 전력 및 온도 변화를 사이클 단위로 측정할 수 있는 통합된 모의 실험 도구를 사용하였다. 통합된 모의 실험 도구는 (그림 15)과 같이 마이크로프로세서의

성능 실험 도구(Cycle-Level Performance Simulator), 소비 전력 계산(Power Calculations), 온도 모델링(Thermal Modeling), 동적 온도 관리(DTM: Dynamic Thermal Management)로 구성한다. 마이크로프로세서 구조 수준의 모의 실험 도구는 wattch[38], HotSpot[21] 프로그램에 동적 온도 관리 (DTM) 기능이 하나로 통합하여 수행한다. wattch 실험 도구는 SimpleScalar[39]를 성능 실험 엔진 (Performance Simulation Engine)으로 이용하여서 한 사이클 단위로 마이크로프로세서의 성능을 모의 실험하고 소비 전력을 계산한다. HotSpot 실험 도구는 각 유닛의 소비 전력 데이터를 이용하여 마이크로프로세서의 온도 분포를 예측한다. 동적 온도 관리(DTM)는 HotSpot 실험 도구에서 사이클 단위로 계산된 각 유닛의 온도를 감시하고, 미리 설정된 환경에 따라 동적 온도 관리(DTM) 기법의 사용을 결정한다. 모의 실험의 결과물로는 벤치마크 프로그램 실행에 따른 마이크로프로세서 성능과 함께 사이클 단위로 각 유닛의 소비 전력과 온도 값을 얻을 수 있다.

본 논문이 제안한 듀얼 부동소수점 가산기 구조의 실험을 위해 통합된 모의 실험 도구에서 소비 전력 계산(Power Calculations) 부분을 추가하였으며, 동적인 동작 이관 기법을 수행하기 위해 동적 온도 관리 부분을 추가하였다. 그리고 이를 위한 실험 환경과 결과를 분석하기 위한 데이터 출력 기능을 추가하였다.



(그림 15) 마이크로프로세서 수준의 모의 실험 도구 구조

2. 실험 환경

a. 마이크로프로세서 환경 구성

모의 실험을 위한 목표 마이크로프로세서로 Alpha 21264 마이크로프로세서 [40]를 사용하였으며, 0.18 μm 반도체 공정, 1.3V 공급 전압, 3GHz 동작 주파수를 갖는 슈퍼스칼라 구조로 가정하였다. 목표 마이크로프로세서의 특성 값[35]은 (표 3)와 같이 설정하였다.

(표 3) Alpha 21264 마이크로프로세서 특성

마이크로프로세서 코어(Microprocessor Core)	
Instruction Window Physical Register Memory Order Queue	RUU: 80, LSQ: 64 Instruction Fetch Queue: 8 IntQ: 20, FPQ: 15
Fetch width Decode width Issue(Instruction) width Commit width	4 4 4(INT: 4, FP, 2, LSQ: 2) 4
Function Unit	Int ALU: 4, Int MULT/DIV: 1 FP ALU: 2, FP MULT/DIV: 1 Memory Port: 2
분기 예측(Branch Prediction)	
Local History Table Local Predict Global History Register Choice Predict	Combined, Bimodal 4K Table 2-Level (GAg) 1 Table, 12bit History 4K Chooser
Branch Target Buffer Return Address Stack	2K Entry, 2-Way 32 Entry
메모리 구조(Memory Hierarchy)	
L1 D-Cache size L1 D-Cache Associativity	64K, 2-Way (LRU) 64B Blocks, 2 Cycle Latency
L1 I-cache size L1 I-cache Associativity	64K, 2-Way (LRU) 64B Blocks, 2 Cycle Latency
TLB size	128 Entry Fully Associativity, 4K Pages 30 Cycle Miss Latency Memory Access Latency (255 / 2)

약어 설명

RUU: Register Update Unit	LSQ: Load/Store Queue
IntQ: Integer Queue	FPQ: Floating Point Queue
Int: Integer	FP: Floating Point
MULT: Multiplier	DIV: Divider
ALU: Arithmetic Logic Unit	LRU: Least Recently Used
TLB: Translation Lookaside Buffer	

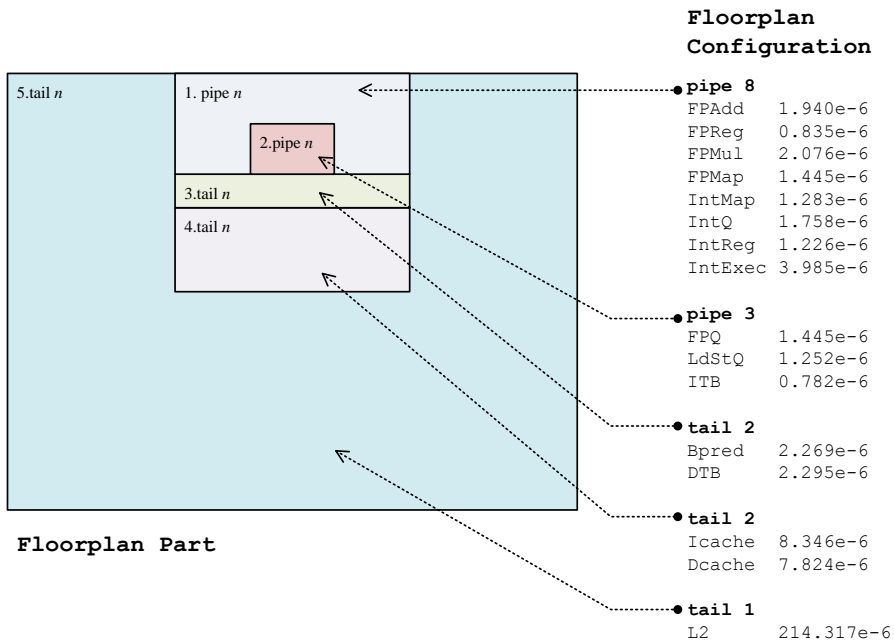
b. 동적 온도 관리 환경 구성

모의 실험에서는 마이크로프로세서의 온도가 일정 수준에 도달하면 동적 온도 관리(DTM)가 수행되면서 온도를 관리한다. 본 논문에서는 하이브리드 동적 온도 관리(Hybrid DTM)로 구성하였으며, 칩의 온도 제어는 2단계로 수행된다. 첫 번째 단계인 구동 단계(Trigger Threshold)에서는 명령어 인출 지연 방법으로 온도를 조절한다. 두 번째 단계인 긴급 단계(Emergency Threshold)에서는 가변 전압 및 주파수 조정 방법(DVFS)으로 온도를 조절한다. 긴급 단계에서는 온도를 급격하게 낮춤으로써 칩의 온도가 최대 한계를 넘지 않도록 조절한다. 이러한 2단계 정책은 고성능 마이크로프로세서에서 성능 손실을 적게 하면서 온도를 조절하는데 도움이 된다. 본 논문은 연구에서는 동적 온도 제어의 구동 단계 온도를 긴급 단계 온도보다 2.0°C 낮게 설정하고 실험을 수행하였다.

온도 모델링 모의 실험에서 HotSpot의 설정은 HotSpot 모델링의 명세[4]에 따른다. 정밀한 모의 실험 결과를 얻기 위하여 모든 유닛이 60.0°C에서 시작하도록 온도를 설정하고, 3억 개의 명령어를 마이크로프로세서의 예열(Warm-Up) 기간으로 설정한다[8]. 예열 기간이 끝나면, HotSpot이 출력하는 안정 상태 온도(Steady State Temperature)로 마이크로프로세서의 초기 온도를 설정한다. 캐시와 분기 예측의 준비 기간은 예열 기간보다 작은 2억 개의 명령어로 설정한다. 모의 실험에는 많은 시간이 필요하기에 명령어의 실행 개수를 5억 개로 제한한다.

c. 플로어플랜 환경 구성

플로어플랜(Floorplan)은 각 유닛의 발열과 열전도에 매우 밀접한 관계가 있다. 모의 실험에서는 Alpha 21264 EV6 코어를 참조하여서 플로어플랜을 구성하였으며, 설계는 hotfloorplan[22]을 이용하였다. 플로어플랜 설계는 코어의 위치 영역과 유닛의 면적을 입력하면 hotfloorplan이 최적화된 위치와 영역을 결정한다. (그림 16)에서 우측이 Alpha 21264 EV6 마이크로프로세 플로어플랜을 설계하는 구성 값 예제이며, 좌측이 hotfloorplan에 의해 배치된 코어의 영역을 보여준다. 플로어플랜 구성은 총 5개의 영역으로 구분되며, 각각의 영역에 대한 유닛의 배치 방법과 유닛의 수, 유닛의 이름과 면적을 입력한다. 유닛의 배치 방법 중에서 pipe는 반 시계방향으로, tail은 왼쪽에서 오른쪽 방향으로 유닛을 할당한다.



(그림 16) 플로어플랜 구성 예제와 할당되는 코어 영역의 위치

d. 벤치마크 프로그램

모의 실험 도구의 실행 프로그램으로 SPEC(System Performance Evaluation Cooperative) CPU2000[36]중에서 부동소수점형 벤치마크(CFP2000)를 수행하였다. 총 14개의 부동소수점 벤치마크 중에서 13개를 수행하였으며, 그 종류는 (표 4)와 같다. 나머지 부동소수점 벤치마크인 sixtrack은 수행 시간이 다른 벤치마크보다 평균 6배 이상 소요되어서 실험에는 제외하였다. 모든 벤치마크 프로그램은 Alpha 명령어 집합에 기반한 실행파일이며, 벤치마크 샘플 데이터 중에서 ref 데이터를 입력 자료로 사용하였다.

(표 4) SPEC CPU2000 부동소수점형 벤치마크 종류

이름	설명
wupwise	양자 색 역학
swim	얕은 물 모델
mgrid	3차원 포넨설계에서의 다중 그리드 풀이법
applu	포물선/타원형 편미분 방정식
mesa	3차원 그래픽 라이브러리
galgel	계산 유체 역학
art	신경회로망을 이용한 화상 인식
equake	지진파 전달 시뮬레이션
facerec	얼굴 화상 인식
ampp	계산 화학
lucas	소수 판정법
fma3d	유한요소법을 사용한 충돌 시뮬레이션
apsi	기상학; 오염물질 확산

3. 마이크로프로세서 코어의 플로어플랜

본 논문은 총 세 가지 형태의 플로어플랜을 구성하여 모의 실험을 진행하였다. Alpha 21264 EV6 코어를 기본으로 본 논문이 제안한 부동소수점 가산기(FPAdd) 유닛과 열전도 지연 공간(HCDS)을 코어에 추가하는 방식으로 플로어플랜을 구성하였다.

모의 실험에서 Alpha 21264 EV6 코어의 각 유닛에 대한 면적 명세는 (표 5)과 같다. 여기서 열전도 지연 공간인 HCDS1과 HCDS2는 정수 레지스터 파일(IntReg) 유닛의 면적으로, HCDS3과 HCDS4는 부동소수점 레지스터 파일(FPReg) 면적과 동일하게 설정하였다. (표 6)은 세 가지의 플로어플랜 이름, 부동소수점 가산기(FPAdd)와 정수 레지스터 파일(IntReg) 유닛의 개수, 코어의 회로 면적과 증가 비율을 보여주고 있다. (표 6)에서 ev6 플로어플랜은 Alpha 21264 EV6 코어를 의미한다. +FPAdd2는 본 논문이 제안한 듀얼 FPU 동적 이관 기법을 위해 보조 부동소수점 가산기(FPAdd2)를 추가한 것이며, +HCDS는 열전도 지연 공간(HCDS) 추가를 의미한다.

칩의 전체 면적을 보면 부동소수점 가산기(FPAdd)를 추가하면 0.77%, 열전도 지연 공간(HCDS)을 더 추가하면 1.62% 증가한다. 이는 칩의 면적 증가로 인해 칩의 비용이 증가할 수 있다는 단점이 될 수 있지만, 기존의 코어 대비 1.62% 면적 증가는 그리 크지 않는 수치라 판단된다. 또한, 최근 반도체 공정의 발전으로 칩의 면적이 전체 시스템에 차지하는 비중이 점차 줄어들고 있고, 칩의 안전성과 성능이 향상될 수 있으면 면적에 대한 단점을 상쇄할 수 있을 것으로 판단된다.

(표 5) 유닛의 면적

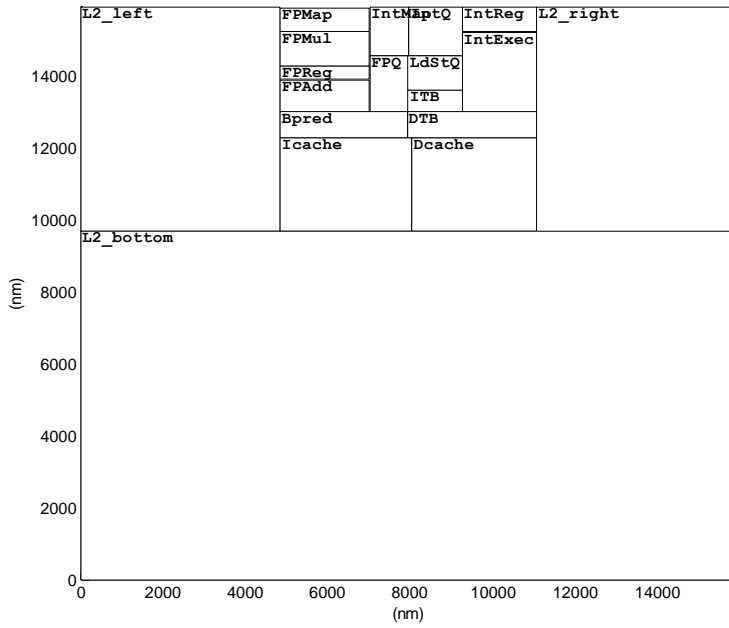
이름	면적(μm^2)	이름	면적(μm^2)	이름	면적(μm^2)
FPAdd	1.940	FPReg	0.835	FPMul	2.076
FPMap	1.445	IntMap	1.283	IntQ	1.758
IntReg	1.226	IntExec	3.985	FPQ	1.445
LdStQ	1.252	ITB	0.782	Bpred	2.269
DTB	2.295	Icache	8.346	Dcache	7.824
L2Cache	214.317	HCDS1/2	0.835	HCDS3/4	1.226

(표 6) 플로어플랜 구성에 따른 칩의 면적과 증가 비율

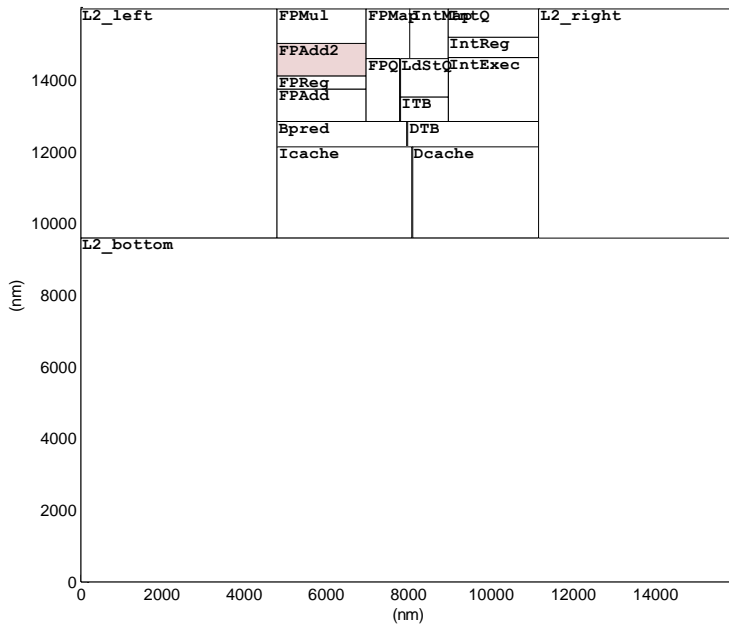
플로어플랜 이름	유닛 및 열전달지연 공간의 개수		전체 면적 (μm^2)	증가한 면적* (μm^2)	증가 비율* (%)
	FPAdd	HCDS			
ev6	1	0	253.078	-	-
ev6 _{+FPAdd2}	2	0	255.018	1.940	0.77%
ev6 _{+FPAdd2+HCDS}	2	4	259.140	6.062	1.62%

(* ev6 플로어플랜을 기준으로 함.)

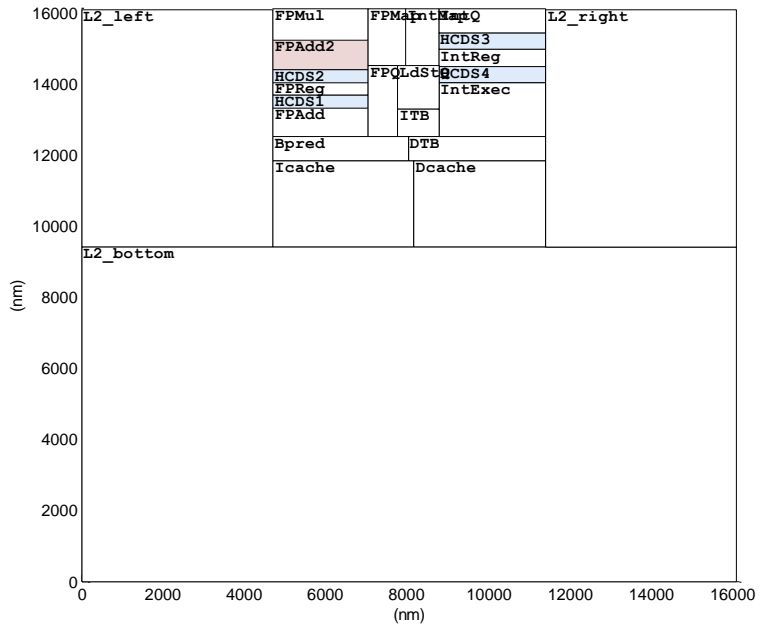
(그림 17)부터 (그림 19)는 (표 5)와 (표 6)의 값으로 hotfloorplan[22]을 이용하여 설계한 최적화된 코어 플로어플랜이다. (그림 17)은 기본적인 Alpha 21264의 ev6 코어의 플로어플랜을 보여준다. (그림 18)은 본 논문이 부동소수점 연산을 위해 제안된 듀얼 부동소수점 연산기 구조를 위해 보조 부동소수점 가산기(FPAdd2) 유닛을 추가한 ev6+FPAdd2 코어의 플로어플랜이다. 플로어플랜 상에서 적색 영역이 추가된 보조 부동소수점 가산기(FPAdd2) 유닛이며, 주 부동소수점 가산기(FPAdd) 유닛과 보조 부동소수점 가산기(FPAdd2) 유닛은 유닛간 데이터 전송 지연 문제를 최대한 방지하기 위해 부동소수점 레지스터 파일(FPReg)와 인접한 위치에 배치를 한다. (그림 19)의 ev6+FPAdd2+HCDS 코어에서 녹색 영역은 ev6+FPAdd2 코어에 열전도 지연 공간(HCDS)을 추가한 플로어플랜이다. 여기서 열전도 지연 공간은 열전도 현상으로 인해 뜨거워지는 부동소수점 레지스터 파일(FPReg)과 일반적으로 가장 뜨거운 정수 레지스터 파일(IntReg) 주변에 배치한다.



(그림 17) ev6 플로어플랜



(그림 18) ev6+FPAdd2 플로어플랜



(그림 19) ev6+FPAdd2+HCDS 플로어플랜

B. 실험 결과 분석

1. 동적 이관을 위한 유닛 선택 기법 분석

본 논문이 제안한 동적 이관(DAM) 기법을 위한 유닛 선택 기법을 모의 실험을 통해 발열의 안전성과 성능을 비교 분석한다. 기존 연구와 비교를 위해 정적 이관(SAM) 기법 중에서 1:1 비율로 동작할 유닛을 선택하는 SAM-1:1 기법과 함께 동적 이관 기법(DAM) 중에서 주 유닛의 온도를 기준으로 하는 DAM-PU(Dynamic Activity Migration - Primary Unit), 동작중인 유닛의 온도를 기준으로 하는 DAM-AU(Dynamic Activity Migration - Active Unit), 동작중인 유닛의 온도와 냉각 임계 온도를 기준으로 하는 DAM-CT(Dynamic Activity Migration - Cooling Temperature), 동작중인 유닛의 온도와 유휴 유닛간의 온도 차를 기준으로 하는 DAM-DT(Dynamic Activity Migration - Difference Temperature) 기법을 분석한다.

a. 모의 실험 환경 구성

모의 실험을 위해 (그림 18)의 플로어플랜과 같이 듀얼 부동소수점 가산기 구조로 모의 실험을 진행한다. 주 유닛인 $UNIT_{PRIMARY}$ 에는 부동소수점 가산기(FPAdd)를, 보조 유닛인 $UNIT_{SECONDARY}$ 에는 부동소수점 가산기와 동일한 기능을 수행하는 보조 부동소수점 가산기(FPAdd2)로 동적 이관 구조를 구성하였다. 동적 이관 기법에 대한 유닛의 최고 온도 변화를 모의 실험 하기 위해 동적 온도 관리 기법(DTM)은 사용하지 않는다. 벤치마크 프로그램은 SPEC CPU2000 중에서 부동소수점 가산기(FPAdd) 유닛의 온도가 가장 높은 lucas 부동소수점 벤치마크를

사용하였다. SAM-1:1 기법에서 선택 주기는 128MHz 클럭 주기로 설정한다. 동적 이관 기법(DAM)에서 필요한 임계 온도(Threshold Temperature)는 70℃, 냉각 임계 온도(Cooling Threshold Temperature)는 68℃, 두 유닛의 온도 차이(Different Temperature)는 2℃로 설정한다.

b. 유닛 선택 기법에 따른 온도 변화

유닛 선택 기법에 따라 주 유닛(Primary Unit)인 주 부동소수점 가산기(FPAdd)와 보조 유닛(Secondary Unit)인 보조 부동소수점 가산기(FPAdd2) 유닛이 칩의 온도 변화와 함께 동작할 유닛을 선택하는 시점에 대한 모의 실험 결과가 (그림 20)부터 (그림 24)까지 이다.

SAM-1:1 기법의 모의 실험 결과인 (그림 20)를 보면, 일정한 비율로 주 유닛과 보조 유닛을 선택하면서 두 유닛의 온도가 발열과 냉각 과정을 반복하는 것을 볼 수 있다. 이 기법은 유닛을 활성화 하는 시간 간격에 따라 발열에 대한 성능 차이가 발생하는 문제가 있다. 활성화 시간 간격이 길어지면 하나의 유닛이 동작하는 시간이 길어지면서 발열하는 시간이 길어져서 유닛의 온도가 높아지는 현상이 발생할 수 있다. 이와 반대로 활성화 시간 간격이 짧으면 유닛이 냉각할 수 있는 시간이 부족하여 두 개의 유닛 모두 온도가 상승하는 문제가 발생할 수 있다.

DAM-PU 기법의 모의 실험 결과인 (그림 21)를 보면, 주 유닛의 온도가 임계 온도보다 높을 시점에 보조 유닛이 발열하고, 주 유닛은 냉각되는 것을 볼 수 있다. 이 시점을 보면, 주 유닛에서 보조 유닛으로 동적 이관된 것을 확인 할 수 있다. 이후, 보조 유닛의 온도가 임계 온도보다 높을 시점이 되면 다시 주 유닛이 동작하면서 주 유닛이 발열되고 보조 유닛은 냉각되는 것을 볼 수 있다. 그러나 주

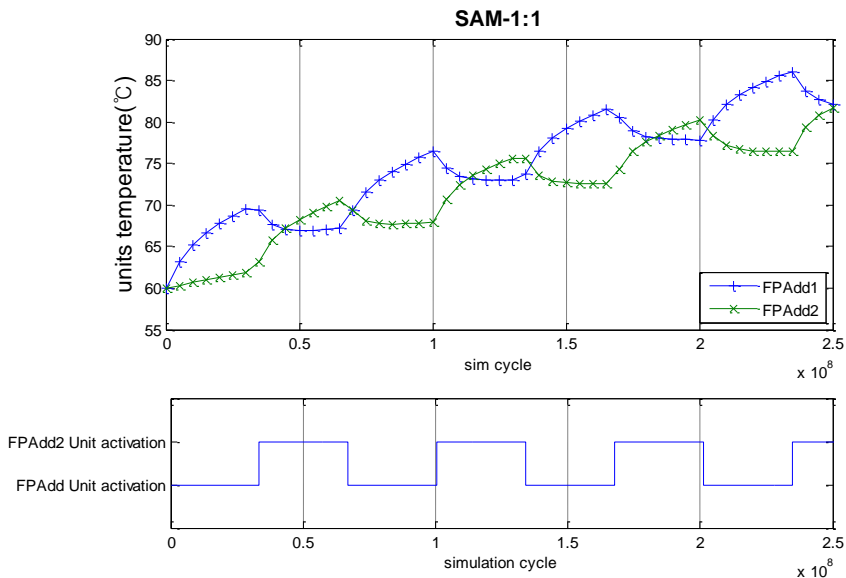
유닛과 보조 유닛의 온도 모두가 임계 온도보다 뜨거운 시점에는 냉각이 제대로 이루어지지 않아서 두 유닛이 상호 번갈아 동작하면서 두 유닛의 온도가 모두 상승하는 것을 확인할 수 있다. 이 기법은 보조 유닛의 발열 온도가 임계 온도보다 아래로 내려가더라도 보조 유닛이 활성화되면서 바로 보조 유닛의 온도가 발열로 인해 임계 온도를 바로 넘기 때문에 주 유닛이 냉각할 수 있는 시간이 부족하다는 문제점을 확인할 수 있다.

DAM-AU 기법의 실험 결과를 보면 두 유닛의 온도가 모두 임계 온도보다 높을 경우에는 상호 유닛이 번갈아 가면서 활성화가 되고, 이로 인해 두 유닛 모두 온도가 동반하여 상승하는 문제가 발생할 수 있다. (그림 22)을 보면 주 유닛의 온도가 임계 온도보다 높을 시점에 보조 유닛은 발열하고, 주 유닛은 냉각되는 것을 볼 수 있다. 이후, 보조 유닛의 온도가 임계 온도보다 높을 시점이 되면 다시 주 유닛이 동적 이관되어, 다시 주 유닛이 발열되고 보조 유닛은 냉각되는 것을 볼 수 있다. 그러나, 주 유닛과 보조 유닛의 온도 모두가 임계 온도보다 뜨거운 시점에는 냉각이 제대로 이루어지지 않아서 두 유닛이 상호 번갈아 동작하면서 두 유닛의 온도가 모두 상승하는 것을 확인할 수 있다. 따라서 보조 유닛의 발열 온도가 임계 온도보다 아래로 내려가더라도 보조 유닛이 동작하면서 바로 보조 유닛의 온도가 발열로 인해 임계 온도를 바로 넘기 때문에 주 유닛이 냉각할 수 있는 시간이 부족하다는 문제점을 확인할 수 있다.

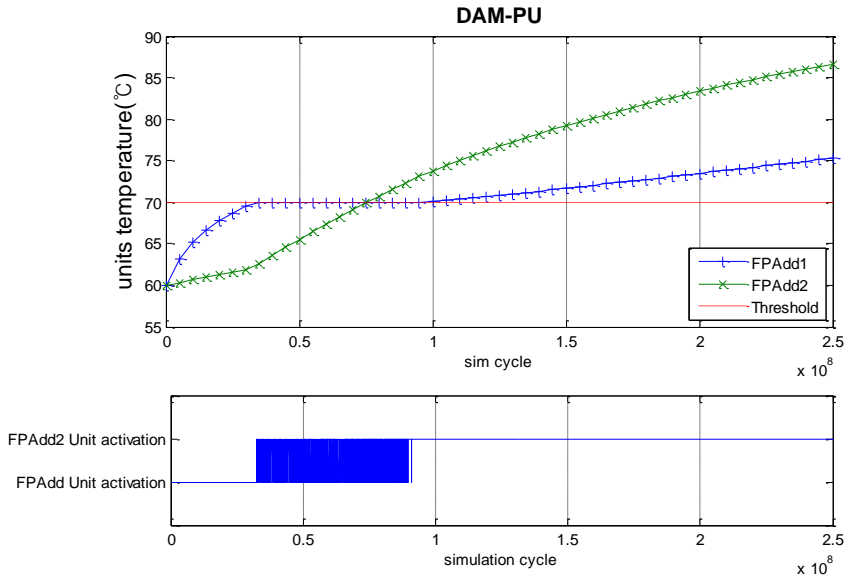
DAM-CT 기법의 실험 결과인 (그림 23)를 보면, 동작중인 유닛의 온도가 임계 온도보다 높더라도 동작하지 않는 유힬 유닛의 온도가 냉각 임계 온도만큼 냉각되는 시점에서만 동적 이관이 진행되는 것을 볼 수 있다. 문제는 동작하지 않는 유힬 유닛이라도 주변 유닛의 열전도로 인해 냉각 임계 온도만큼 온도가 떨어지지 않는 시점에서는 동적 이관을 진행할 수 없어 동작중인 유닛만이 계속 발열되는

문제점을 확인할 수 있다.

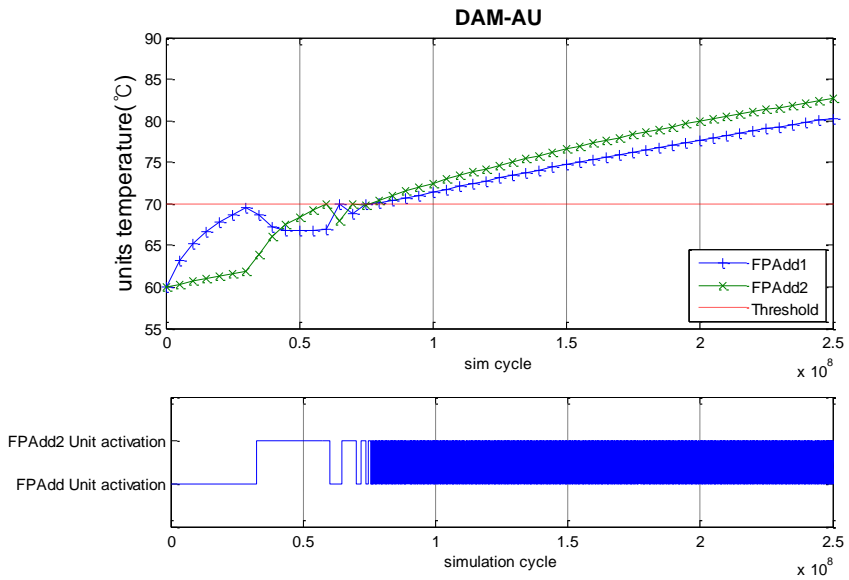
DAM-DT 기법의 실험 결과인 (그림 24)를 보면, 주 유닛의 발열 온도가 임계 온도를 넘더라도 두 유닛의 온도가 미리 설정한 온도 차이만큼 발생해야만 보조 유닛으로 동작 이관이 진행되어서 보조 유닛의 온도가 올라가는 것을 볼 수 있다. 이러한 유휴 유닛의 온도 차이를 이용하면 일정한 간격으로 유닛이 선택되는 것을 볼 수 있다. 그리고 냉각 임계 온도가 동적으로 변하기 때문에 두 유닛의 온도는 일정한 주기로 발열과 냉각을 반복하면서 유닛이 동적 이관이 되는 것을 확인할 수 있다.



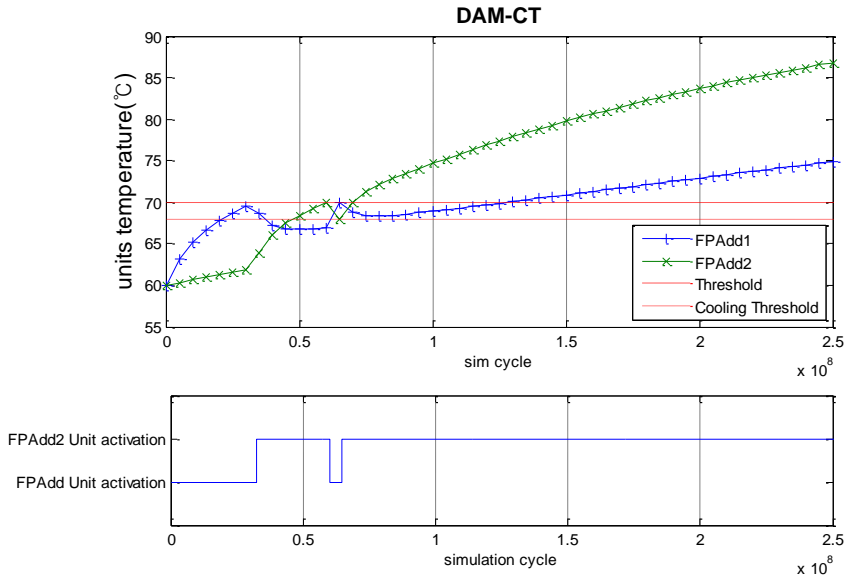
(그림 20) SAM-1:1 기법에서 유닛의 선택과 온도 변화



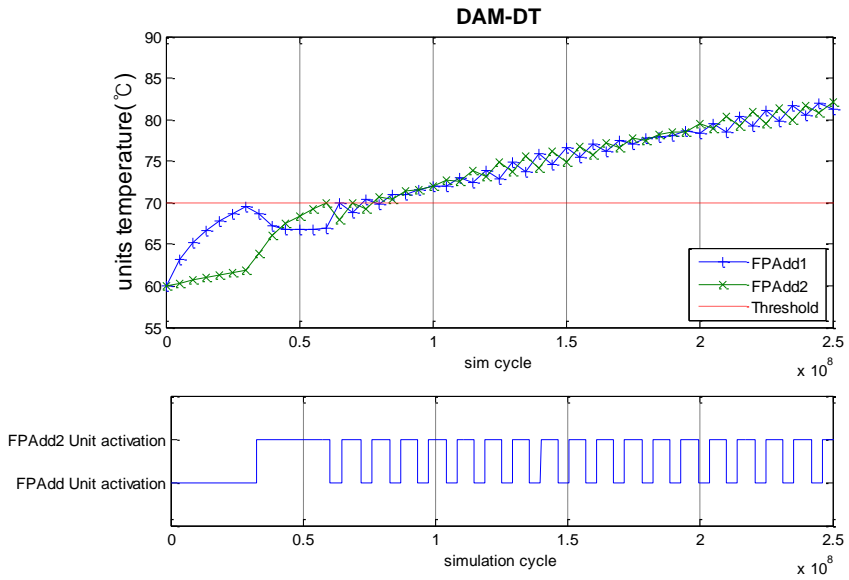
(그림 21) DAM-PU 기법에서 유닛의 선택과 온도 변화



(그림 22) DAM-AU 기법에서 유닛의 선택과 온도 변화



(그림 23) DAM-CT 기법에서 유닛의 선택과 온도 변화

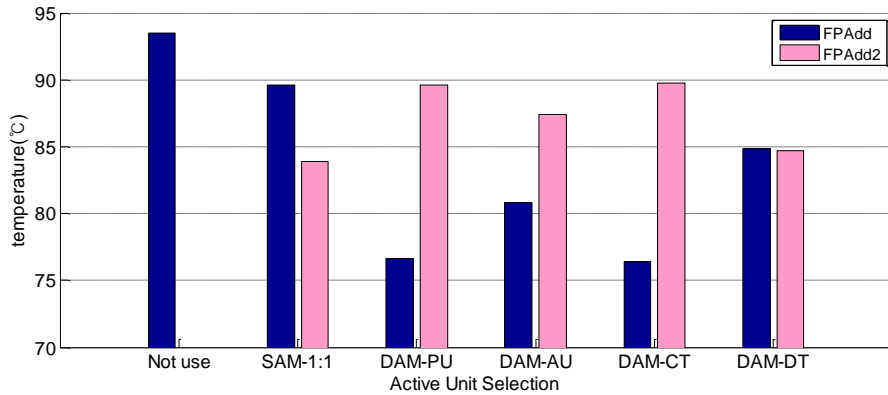


(그림 24) DAM-DT 기법에서 유닛의 선택과 온도 변화

c. 유닛 선택 기법에 따른 성능 비교

정적 이관 기법(SAM)과 동적 이관 기법(DAM)에서 유닛 선택 기법에 따른 주 유닛과 보조 유닛의 최고 온도에 대한 모의 실험 결과는 (그림 25)와 (표 7)와 같다. 모의 실험 결과를 보면, 동적 이관 기법 중에서 동작중인 유닛의 온도와 유티 유닛의 온도 차이를 기준으로 선택하는 DAM-DT 기법을 사용하는 것이 두 개의 유닛 중에서 최고 온도가 84.85°C로 가장 낮은 것을 확인할 수 있다. 또한 두 유닛(주 유닛과 보조 유닛)의 최고 온도 차이가 0.18°C로 각 유닛에 대한 냉각 효과가 매우 좋은 것을 확인할 수 있다. 1:1 비율로 유닛을 활성화 하는 정적인 클럭 주기 기법인 SAM-1:1 기법 역시 최고 온도가 89.63°C로 좋은 성능을 보여주고 있지만, 두 유닛의 최고 온도 차이는 5.70°C로 DAM-DT 기법보다는 발열 성능이 떨어지는 것을 볼 수 있다.

(표 8)은 긴급 단계 온도가 75°C로 설정한 동적 온도 관리(DTM) 기법을 사용할 때, 각 유닛 선택 기법에 따른 부동소수점 가산기들의 최고 온도와 CPI 결과이다[41]. (표 8)를 보면, DAM-DT 기법이 발열에 대한 안전성과 성능이 가장 우수하다는 것을 확인할 수 있다. 또한, 기존의 동적 온도 관리(DTM) 환경에서 듀얼 정수 레지스터 파일(IntReg)을 1:1과 1:2 비율로 동작할 유닛을 선택[8]하는 정적 이관 기법(SAM)보다는 DAM-DT 기법을 적용하는 것이 최고 온도가 평균 3.6°C 낮아지고, 성능은 평균 50% 향상된다는 연구 결과도 있다[42]. 따라서 본 논문은 유닛 선택 기법으로 발열 안전성과 성능이 우수한 DAM-DT 기법을 본 논문이 제안한 듀얼 부동소수점 가산기 구조에 적용한다.



(그림 25) 유닛 선택 기법에 따른 부동소수점 가산기 유닛의 최고 온도

(표 7) 유닛 선택 기법에 대한 온도 실험 결과

유닛 선택 기법	주 유닛 온도 (FPAdd)	보조 유닛 온도 (FPAdd2)	최고온도	온도차이
Not Use	93.49	-	93.49	-
SAM-1:1	89.63	83.93	89.63	5.70
DAM-PU	76.66	89.64	89.64	12.98
DAM-AU	80.80	87.43	87.43	6.63
DAM-CT	76.39	89.76	89.76	13.37
DAM-DT	84.85	84.67	84.85	0.18

(단위: °C)

(표 8) 동적 온도 관리 기법 적용시 유닛 선택 기법의 성능 비교

유닛 선택 기법	가산기 유닛의 최고 온도 (°C)	CPI
Not Use	76.92	0.68
SAM-1:1	75.21	0.50
DAM-PU	75.13	0.50
DAM-AU	75.04	0.50
DAM-CT	76.33	0.53
DAM-DT	74.39	0.50

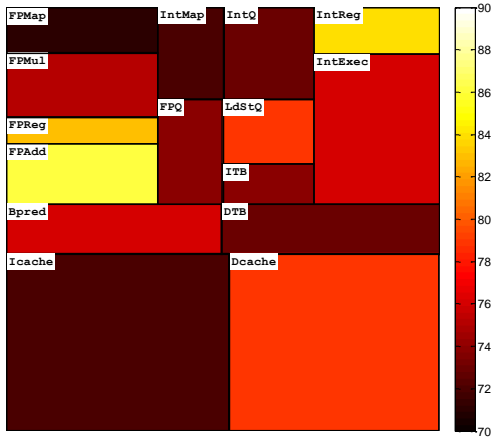
(긴급 단계 온도: 75°C)

2. 열섬 현상 분석

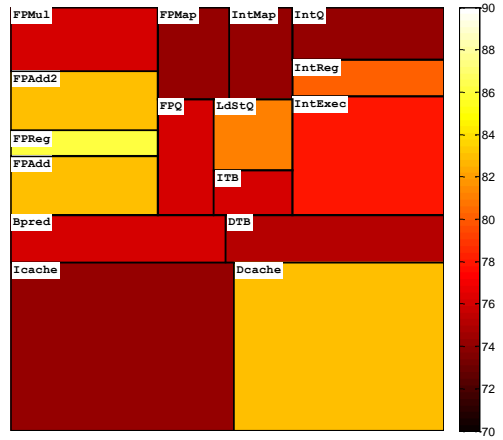
이번 장에서는 본 논문이 제안한 듀얼 부동소수점 가산기 구조와 열전도 지연 공간(DHTS)의 적용에 따라 칩 상의 유닛에 대한 열섬 현상을 분석하고자 한다. 실험 환경으로 동적 온도 관리 기법의 구동 단계 온도(Trigger Threshold Temperature)는 83°C, 긴급 단계 온도(Emergency Threshold Temperature)는 85°C, DAM-DT의 임계 온도(DAM-DT Threshold Temperature)는 80°C, 두 유닛의 온도 차이는 2°C로 설정한다. 플로어플랜 구성은 (그림 17)부터 (그림 19)까지 총 세 가지의 플로어플랜을 대상으로 실험한다. 실행 프로그램으로는 SPEC CPU2000 부동소수점 벤치마크(CFP2000) 중에서 mgrid 벤치마크를 중심으로 설명한다.

(그림 26)은 부동소수점 벤치마크인 mgrid 수행 결과에 따라 각각의 플로어플랜 상에 있는 유닛의 최고 온도 분포를 보여준다. 여기서 다른 유닛에 비해 유닛의 면적이 가장 넓고 최고 온도가 매우 낮은 L2 캐시는 그림에서 제외하였다. 제안하는 기법이 적용되지 않았을 경우의 결과인 (그림 26)의 (a)를 보면, 부동소수점 가산기(FPAdd)의 온도가 가장 높은 것을 볼 수 있다. 본 논문이 제안한 듀얼 부동소수점 가산기 구조를 적용한 (그림 26)의 (b)를 보면, 부동소수점 가산기(FPAdd)와 보조 부동소수점 가산기(FPAdd2)가 번갈아 동작한 결과, (그림 26)의 (a)보다 부동소수점 가산기의 온도가 낮아진 것을 볼 수 있다. 하지만 인접한 두 개의 부동소수점 가산기에서의 발생하는 발열로 인한 열전도 현상으로 부동소수점 레지스터 파일(FPReg)의 온도가 올라간 것을 볼 수 있다. 열전도 지연 공간을 추가로 배치한 구조인 (그림 26)의 (c)를 보면, 부동소수점 레지스터 파일(FPReg)과 정수 레지스터 파일(IntReg) 사이에 열전도 지연 공간(HCDS)을 배치한 결과, 모든 레지스터 파일의 온도가 낮아진 것을 확인할 수 있다. 특이한 사항으로 제안

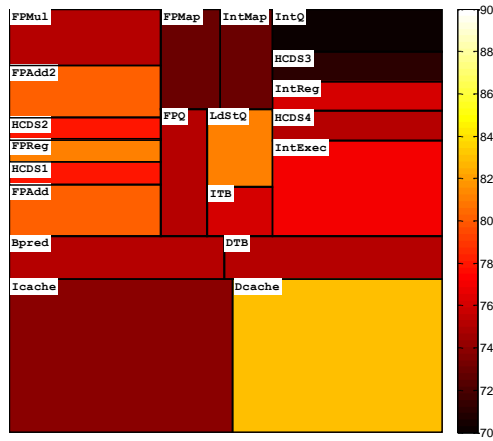
기법을 적용한 결과 목표로 하였던 부동소수점 레지스터 파일(FPReg)뿐만 아니라 정수 레지스터 파일(IntReg) 역시 온도가 낮아진 것을 볼 수 있다. 대신 보조 부동소수점 가산기가 추가되면서 데이터 읽기 접근 비율이 증가하여, 데이터 캐시(Dcache)와 적재/저장 큐(LdStQ) 유닛의 온도는 증가한다. 하지만 전체 유닛의 최고 온도보다는 낮기 때문에 발열 현상으로 인한 칩의 안정성에는 큰 영향을 미치지 않는다. 따라서 열전도 지연 공간(HCDS)을 배치하는 경우 유닛의 최고 온도가 낮아지면서 칩의 안정성을 높일 수 있음을 알 수 있다. 따라서 본 논문이 제안한 듀얼 부동소수점 가산기 구조와 열전도 지연 공간을 추가한 플로어플랜이 부동소수점 연산에 대해 유닛의 최고 온도를 낮출 수 있다는 사실을 확인할 수 있다.



(a) ev6



(b) ev6+FPAdd2



(c) ev6+FPAdd2+HCDS

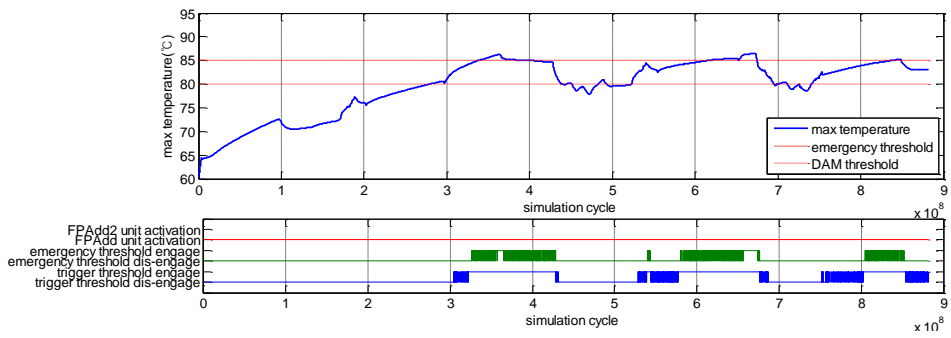
(그림 26) mgrid 벤치마크 실행 결과에 따른 유닛의 최고 온도 분포

다음은 DTM 사용 여부와 DAM-DT에서 따라 선택된 유닛을 비교 및 분석한다. mgrid 벤치마크를 실행하였을 때 최고 온도 변화에 따른 DTM과 DAM-DT의 모의 실험 결과는 (그림 27)과 같다.

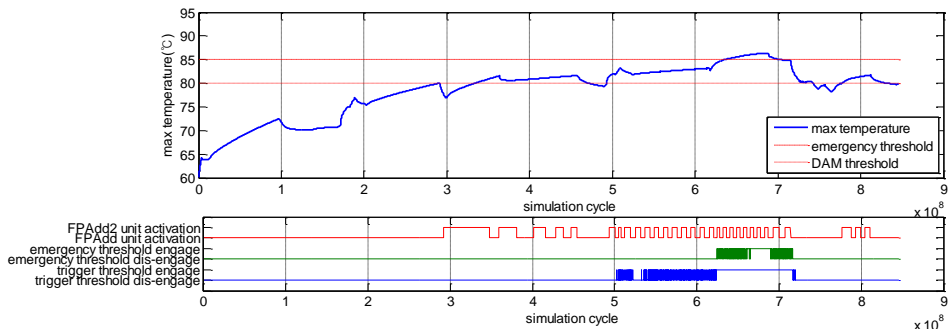
(그림 27)에서 위쪽에 있는 그래프를 보면 실선은 모의 실험이 진행되는 동안 각 유닛 중에서 가장 높은 온도를 의미한다. 일점 쇄선('•-')은 DAM-DT가 동작하는 DAM-DT 임계 온도(DAM-DT Threshold Temperature)를 의미하며, 쇄선('--')은 하이브리드 동적 온도 관리(HybrdDTM)가 긴급 단계(Emergency Threshold)로 동작하는 온도를 의미한다. (그림 27)의 아래에 있는 그래프를 보면, HybrdDTM를 사용하지 않으면 trigger threshold dis-engage를, 구동 단계(Trigger Threshold)로 사용하면 trigger threshold engage에 선을 나타낸다. 여기서 칩의 온도가 더 올라가 긴급 단계(Emergency Threshold)를 사용하면 emergency threshold engage에 선이 표시된다. 그리고 긴급 단계는 구동 단계와 같이 사용한다. 나머지는 DAM-DT 기법에 따라 선택되는 부동소수점 가산기(FPAdd와 FPAdd2) 유닛을 의미한다.

(그림 27)의 (a)를 보면, 칩의 온도가 구동 단계 온도로 접근하면 구동 단계로 동적 온도 제어가 사용되는 것을 볼 수 있다. 이 때에는 명령어 인출을 지연시키면서 칩의 온도가 올라가는 것을 억제한다. 그러다가 칩의 최고 온도가 긴급 단계 온도를 넘어가면 동적 온도 관리는 긴급 단계로 동작한다. 이렇게 되면 DTM은 강제로 마이크로프로세서의 공급 전압과 동작 주파수를 조절하면서 칩의 온도가 떨어지는 것을 볼 수 있다. 이후, 칩의 온도가 어느 정도 안정화되면 DTM은 동작하지 않으면서 조금씩 칩의 온도가 올라가는 현상을 볼 수가 있다. 이렇게 동적 온도 관리로 인해 칩의 온도를 조절하는 것을 볼 수 있다. (그림 27)의 (b)를 보면 DAM-DT 기법으로 보조 부동소수점 가산기(FPAdd2)가 선택되면서 칩이 냉각되

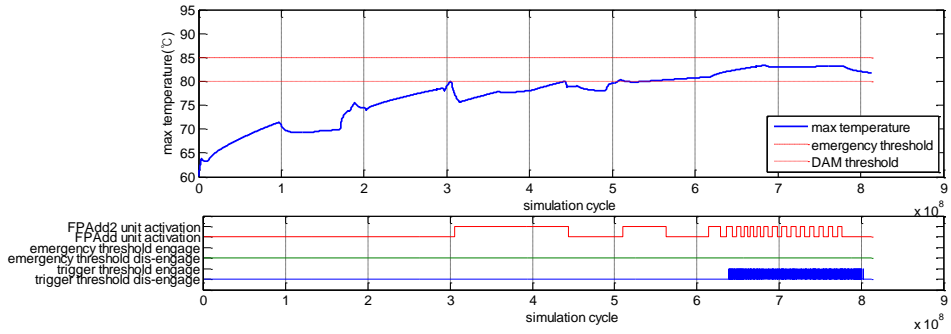
는 것을 확인할 수 있다. 또한, HybrdDTM의 적용 시간이 더 늦춰진 것을 확인할 수 있다. 여기에 (그림 27)의 (c)와 같이 열전도 지연 공간(HCDS)이 추가되면, 칩의 최고 온도가 긴급 단계 온도에 도달하지 않아서 HybrdDTM의 구동 단계가 적용되지 않는 것을 확인할 수 있다. 이는 온도 제어를 위해 동적 온도 관리 사용으로 인해 강제로 성능을 제한시키는 시간이 줄어들면서 성능이 향상될 수 있음을 의미한다.



(a) ev6



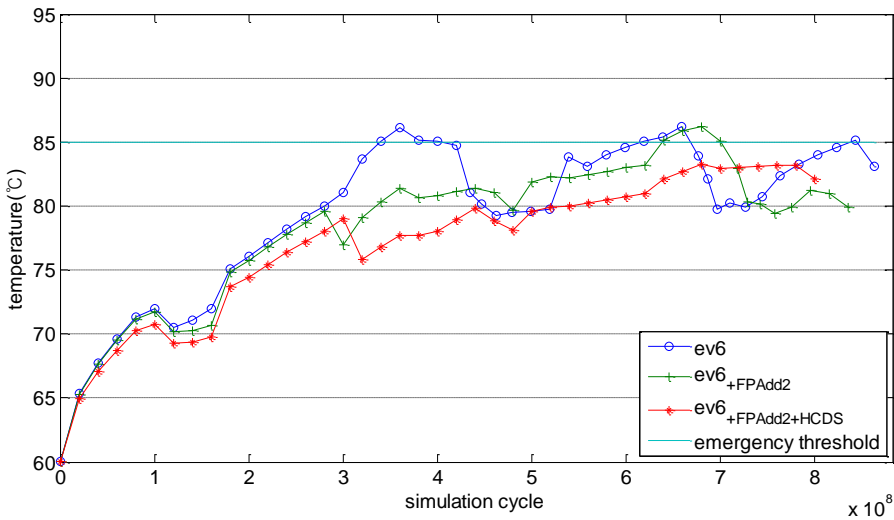
(b) ev6+FPAdd2



(c) ev6+FPAdd2+HCDS

(그림 27) mgrid 벤치마크 실행에 따른 온도 변화와 DTM, DAM-DT 수행 결과

(그림 28)은 mgrid 벤치마크 프로그램에 대한 모의 실험의 결과로, 'o' 기호는 ev6, '+' 기호는 ev6+FPAdd2, '*' 기호는 ev6+FPAdd2+HCDS 플로어플랜에서 모의 실험의 진행 시간에 따른 마이크로프로세서의 모든 유닛 중에서 최고 온도 변화를 보여준다. ev6 플로어플랜의 경우, 마이크로프로세서의 최고 온도가 긴급 단계(Emergency Threshold)보다 상승하면, 동적 온도 관리 기법의 사용으로 마이크로프로세서의 온도가 조금씩 낮아지는 것을 볼 수 있다. 보조 부동소수점 가산기인 FPAdd2를 추가한 ev6+FPAdd2 플로어플랜의 경우 ev6 플로어플랜과 비교하면, 부동소수점 가산기 유닛이 동적 이관 기법에 따라 번갈아 동작하면서 마이크로프로세서의 최고 온도가 낮아지는 것을 볼 수 있다. 여기에 열전도 지연 공간을 추가한 ev6+FPAdd2+HCDS 플로어플랜을 보면, 다른 두 개의 플로어플랜보다 최고 온도가 낮은 것을 볼 수 있으며, 모의 실험의 종료 시간이 단축된 것을 확인할 수 있다.



(그림 28) 플로어플랜 종류별 mgrid 벤치마크 결과

3. 발열 안정성 평가

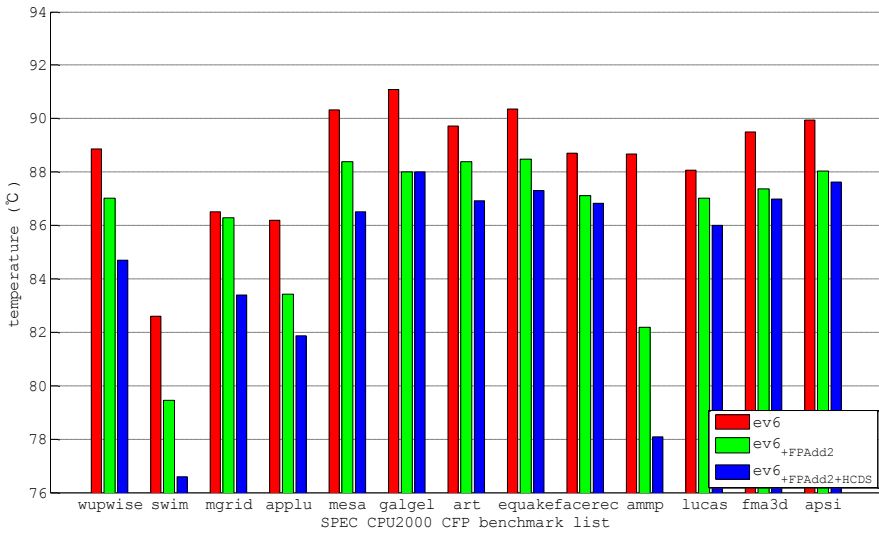
동적 온도 관리에서 긴급 단계의 온도를 85°C와 90°C로 설정한 후, 부동소수점 벤치마크 실행에 따른 유닛의 최고 온도 결과는 (그림 29)와 같다. 실험 결과, 본 논문에서 제안하는 기법인 $ev6_{+FPAdd2}$ 와 $ev6_{+FPAdd2+HCDS}$ 플로어플랜이 기존의 $ev6$ 플로어플랜보다 유닛의 최고 온도가 하강한 것을 확인할 수 있다. 듀얼 부동소수점 가산기 구조를 적용하는 경우 모든 벤치마크에서 유닛의 최고 온도가 하강하는 것을 볼 수 있다. (그림 29)의 (b)에서 $galge1(ev6_{+FPAdd2+HCDS})$ 벤치마크에서만 유닛의 최고 온도가 $ev6_{+FPAdd2}$ 플로어플랜보다는 높더라도 $ev6$ 플로어플랜 보다는 최고 온도가 낮으므로 발열 안정성이 개선되었음을 볼 수 있다.

(표 9)는 (그림 29)에서 $ev6$ 플로어플랜 기준으로 하여 $ev6_{+FPAdd2}$ 와 $ev6_{+FPAdd2+HCDS}$ 플로어플랜에 대한 온도차이 결과를 정리해서 보여준다. (표 9)에서 보이는 바와 같이 동적 온도 관리의 긴급 단계 온도를 90°C로 설정할 때, $ev6_{+FPAdd2}$ 플로어플랜을 적용하면 최대 6.7°C(평균 3.0°C) 온도가 하강하고, $ev6_{+FPAdd2+HCDS}$ 플로어플랜을 적용하면 최대 10.8°C(평균 5.3°C) 온도가 하강한다.

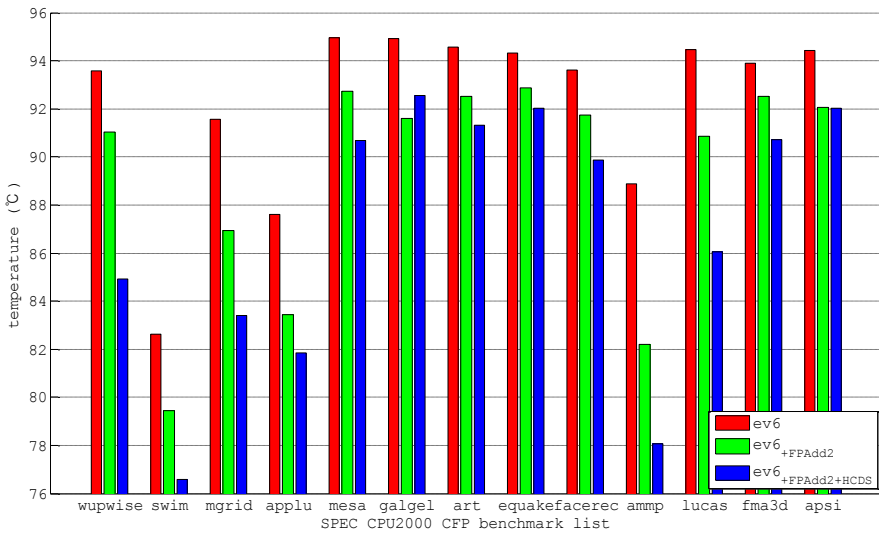
(표 9) $ev6$ 플로어플랜을 기준으로 한 온도 차이 결과

긴급 단계 온도	$ev6_{+FPAdd2}$		$ev6_{+FPAdd2+HCDS}$	
	최대 온도차	평균 온도차	최대 온도차	평균 온도차
85	- 6.5	- 2.3	- 10.9	- 3.8
90	- 6.7	- 3.0	- 10.8	- 5.3

(단위: °C)



(a) 긴급 단계 온도가 85°C일 때 최고 온도



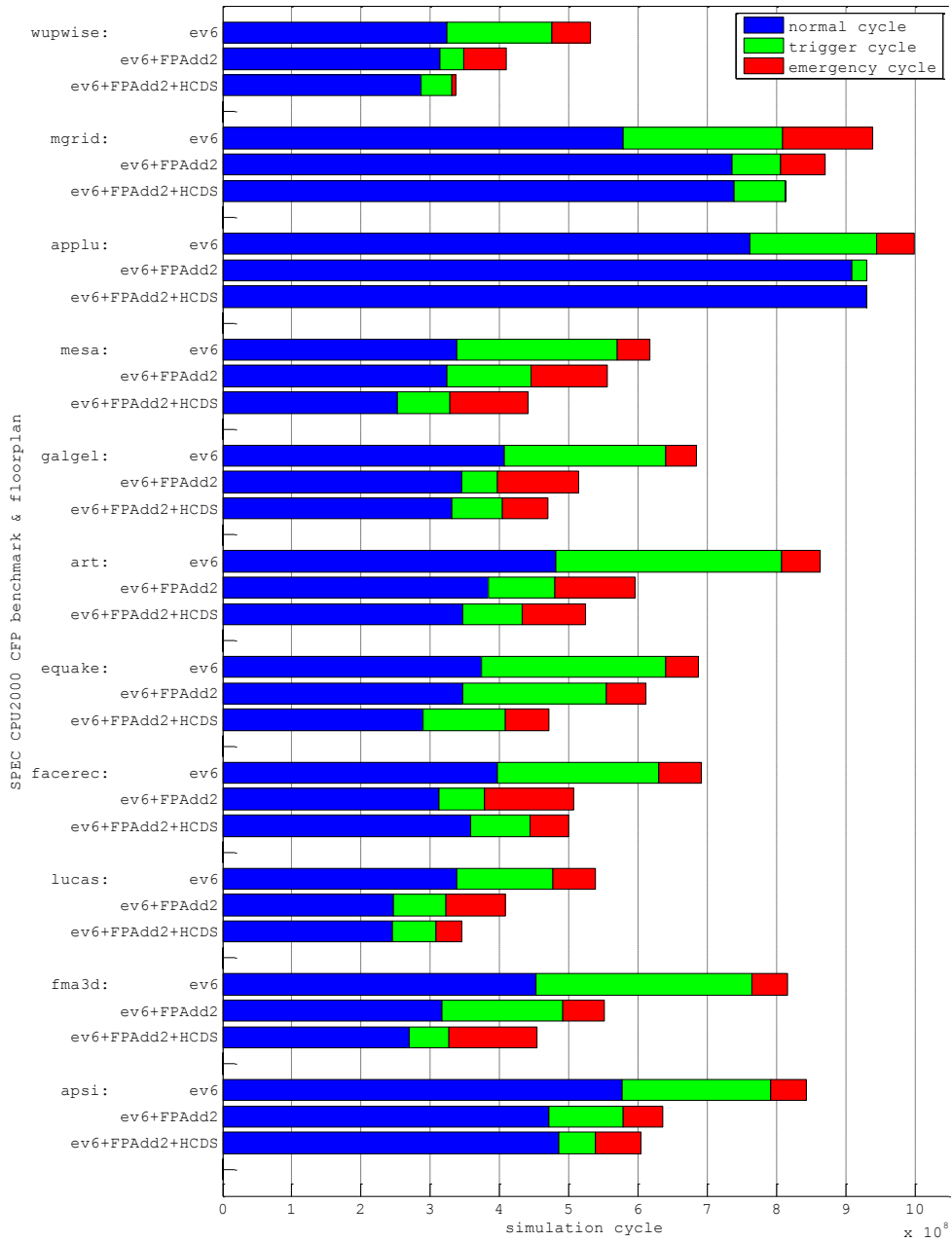
(b) 긴급 단계 온도가 90°C일 때 최고 온도

(그림 29) 동적 온도 관리 기법의 설정과 플로어플랜에 따른 마이크로프로세서의 최고 온도

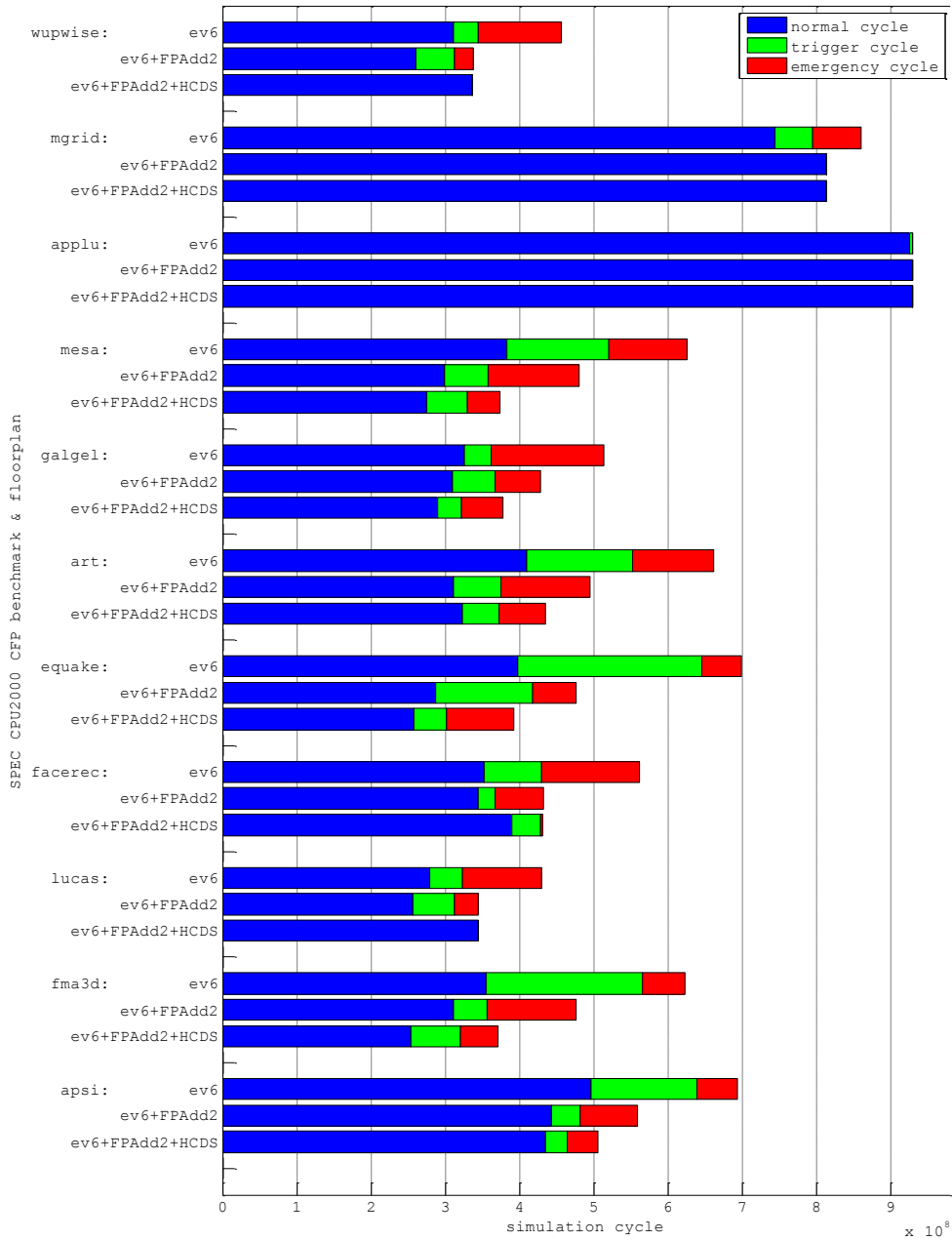
모의 실험 결과를 보면 극히 일부 부동소수점 벤치마크를 제외하고는 듀얼 부동소수점 가산기 구조와 열전도 지연 공간을 함께 적용하면 부동소수점 응용 프로그램에서 마이크로프로세서 내부의 최고 온도가 가장 효과적으로 낮아진다는 것을 확인할 수 있다. 이를 통해 본 논문에서 제안한 동적인 온도 관리 방법과 정적인 온도 관리 방법을 함께 사용한다면, 칩의 발열 안정성에 영향을 미치는 최고 온도를 낮추는데 매우 효과적이라는 것을 확인할 수 있다.

4. 마이크로프로세서 성능 평가

부동소수점 응용 프로그램의 실행 시간 중에 동적 온도 관리의 수행 시간이 짧을수록 프로그램의 실행 시간 역시 짧아진다. 즉, 동적 온도 관리의 개입 시간과 마이크로프로세서의 성능은 반비례한다. 동적 온도 관리 기법의 사용이 성능에 미치는 영향을 상세하게 분석하기 위하여, 모의 실험을 통해 벤치마크 프로그램의 수행 시간을 비교하였다. (그림 30)과 (그림 31)은 긴급 단계 온도가 85°C와 90°C일 때 부동소수점 벤치마크의 실행 시간을 보여준다. 그래프의 각 바는 벤치마크 프로그램 수행에 소요되는 사이클을 의미한다. 그래프에서 청색은 동적 온도 관리 기법을 사용하지 않는 시간을, 녹색과 적색은 동적 온도 관리 기법에 의한 구동 단계와 긴급 단계의 동작 시간을 의미한다. 마이크로프로세서의 성능 평가에서는 swim과 ammp 부동소수점 벤치마크는 제외하였다. 이 두 개의 부동소수점 벤치마크는 전체 유닛의 최고 온도가 설정한 구동 단계의 온도보다 낮아서 동적 온도 관리를 사용하지 않아서 제외하였다.



(그림 30) 동적 온도 관리 기법의 설정과 플로어플랜에 따른 부동소수점 벤치마크 실행 시간 (긴급 단계 온도는 85°C)



(그림 31) 동적 온도 관리 기법의 설정과 플로어플랜에 따른 부동산수점 벤치마크 실행 시간 (긴급 단계 온도는 90°C)

(그림 30)과 (그림 31)를 보면 $ev6_{+FPAdd2}$ 와 $ev6_{+FPAdd2++HCDS}$ 에서는 동적 온도 관리의 수행 시간이 줄어든다는 것을 확인할 수 있다. 대부분의 부동소수점 벤치마크 프로그램에서 동적 온도 관리 기능의 긴급 단계 수행 시간이 단축된 것을 볼 수 있다.

(표 10)은 $ev6$ 플로어플랜을 기준으로 $ev6_{+FPAdd2}$ 와 $ev6_{+FPAdd2++HCDS}$ 에 대한 성능 결과를 보여준다. (표 10)에서 긴급 단계 온도가 85°C일 때, $ev6_{+FPAdd2}$ 플로어플랜을 적용하면 성능이 최대 1.47배(평균 1.27배) 상승하고, $ev6_{+FPAdd2++HCDS}$ 배치를 적용하면 성능이 최대 1.80배(평균 1.45배) 향상된 것을 볼 수 있다.

(표 10) $ev6$ 플로어플랜을 기준으로 한 성능 개선 비교

긴급 단계 온도 (°C)	$ev6_{+FPAdd2}$		$ev6_{+FPAdd2++HCDS}$	
	최대 성능	평균 성능	최대 성능	평균 성능
85	+ 1.47	+ 1.27	+ 1.80	+ 1.45
90	+ 1.46	+ 1.25	+ 1.78	+ 1.40

(모의 실험 실행 시간은 $ev6$ 플로어플랜이 기준임.)

지금까지의 모의 실험 결과를 종합하면, 듀얼 부동소수점 가산기 구조를 적용한 $ev6_{+FPAdd2}$ 플로어플랜의 경우 모든 부동소수점 벤치마크 프로그램 수행 시 $ev6$ 플로어플랜보다 마이크로프로세서의 최고 온도가 낮아지면서 성능은 향상되었다. 여기에 열전도 지연 공간을 추가로 배치한 $ev6_{+FPAdd2++HCDS}$ 플로어플랜은 극히 일부 부동소수점 벤치마크 프로그램에서 마이크로프로세서의 최고 온도가 $ev6_{+FPAdd2}$ 보다 높았지만, 동적 온도 관리 기능만을 사용한 $ev6$ 플로어플랜 보다는

최고 온도가 낮으므로 발열 안전성은 좋아졌다고 할 수 있다. 또한, 대부분의 부동소수점 벤치마크 프로그램에서 마이크로프로세서의 최고 온도가 $ev6_{+FPAdd2}$ 플로우 플랜보다 더 내려갔으며, 성능은 보다 더 향상되었다. 따라서 듀얼 부동소수점 가산기 구조와 열전도 지연 공간을 배치한 구조가 부동소수점 응용 프로그램 수행 시 동적 온도 관리를 사용하는 경우에는 발열 안정성과 성능 모두를 향상시킬 수 있을 것으로 기대된다.

V. 결론 및 향후 연구

본 논문에서는 부동소수점 응용 프로그램 수행 시 마이크로프로세서의 성능 저하를 줄이면서 발열에 대한 안정성을 높이는 방법으로 듀얼 부동소수점 가산기 구조와 열전도 지연 공간을 이용한 유닛의 플로어플랜 방법을 제안하였다. 기존에는 저온도 마이크로프로세서 설계를 위해 정수 레지스터 파일의 열섬 현상을 해결하기 위한 연구가 대부분이었으나, 이러한 해결 방법은 부동소수점 응용 프로그램을 수행하는 경우에는 큰 효과를 볼 수 없었다. 본 논문에서는 듀얼 부동소수점 가산기 구조를 이용하여 부동소수점 응용 프로그램을 수행하는 경우 마이크로프로세서의 열섬 현상을 완화하였으며, 동적 온도 관리에 의한 성능 감소를 최소화 하였다. 또한 열전도 지연 공간을 레지스터 파일 주변에 추가로 배치하여 인접한 유닛의 열전도로 인해 발생하는 온도 상승 문제를 완화하였다. 그 결과, 마이크로프로세서의 최고 온도는 더욱 낮아졌으며, 이를 통해 동적 온도 관리의 개입으로 인한 성능 저하를 크게 완화하였다. 따라서 제안하는 기법들을 적용하면 부동소수점 응용 프로그램을 주로 수행하는 고성능 DSP나 멀티미디어 시스템에서 동작하는 마이크로프로세서의 안정성과 성능 향상에 도움을 줄 수 있을 것으로 기대된다.

본 논문에서 제안한 방법은 비록 작은 수치이지만 칩의 면적이 0.77%에서 1.62%까지 증가하는 단점이 있다. 따라서 동적 온도 관리에 따른 성능 저하가 다소 증가하더라도 칩의 면적이 증가하지 않는 방법에 대한 연구가 추후 필요하다. 또한 열전도 지연 공간의 배치는 칩의 면적을 증가시키는 커다란 원인이기에 가장 효율적인 공간 배치 방법에 대한 추가 연구가 필요하다.

참고 문헌

- [1] G. Moore, "Cramming more components onto integrated circuits," Proceedings of the IEEE, vol. 86, pp. 82–85, 1998.
- [2] Moore's Law 40th Anniversary, http://www.intel.com/pressroom/kits/events/moores_law_40th/
- [3] F. Pollack, "New microarchitecture challenges in the coming generations of cmos process technologies," International Symposium on Microarchitecture (MICRO–32) keynote speech, 1999
- [4] K. Skadron, M. R. Stan, W. Huang, V. Sivakumar, S. Karthik, and D. Tarjan, "Temperature–aware microarchitecture," Proceedings of International Symposium on Computer Architecture (ISCA '03), pp. 2–13, 2003
- [5] R. Mahajan, "Thermal management of CPUs: A perspective on trends, needs and opportunities," Proceedings of the 8th International Workshop on THERMal INvestigations of ICs and Systems, 2002
- [6] V. Narayanan and Y. Xie, "Reliability concerns in embedded system designs," COMPUTER, vol. 39, pp. 118–120, 2006.
- [7] D. Brooks and M. Martonosi, "Dynamic thermal management for high–performance microprocessors," Proceedings of International Symposium on High–Performance Computer Architecture (HPCA '01), pp. 171–182, 2001
- [8] 최진항, 공준호, 정의영, 정성우, "온도 인지 마이크로프로세서를 위한 듀얼 레지스터 파일 구조," 정보과학회논문지 : 시스템 및 이론, 제35권, 제12호, 540–551 페이지, 2008.
- [9] I. Yeo and E. J. Kim, "Hybrid dynamic thermal management based on statistical characteristics of multimedia applications," Proceeding of the thirteenth international symposium on Low power electronics and design, pp. 321–326, 2008
- [10] J. W. Sheaffer, K. Skadron, and D. P. Luebke., "Studying Thermal Management for Graphics–Processor Architectures," Proceedings of the 2005 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), pp. 54–65, 2005

- [11] Y. Inchoon, L. Heung Ki, K. Eun Jung, and Y. Ki Hwan, "Effective Dynamic Thermal Management for MPEG-4 decoding," International Conference on Computer Design (ICCD 2007), pp. 623-628, 2007
- [12] L. Wonbok, K. Patel, and M. Pedram, "GOP-Level Dynamic Thermal Management in MPEG-2 Decoding," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 16, pp. 662-672, 2008.
- [13] K. J. Lee and K. Skadron, "Using performance counters for runtime temperature sensing in high-performance processors," Proceedings of 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05), 2005
- [14] C. Isci and M. Martonosi, "Runtime power monitoring in high-end processors: methodology and empirical data," Proceedings of 36th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO '03), pp. 93-104, 2003
- [15] L. Jong Sung, K. Skadron, and C. Sung Woo, "Predictive Temperature-Aware DVFS," IEEE Transactions on Computers, vol. 59, pp. 127-133, 2010.
- [16] 공준호, 정성우, "최근 프로세서 발열 관리 연구 동향," 정보과학회지, 제27권, 제11호, 72-79 페이지, 2009.
- [17] L. Chee How, W. R. Daasch, and G. Cai, "A thermal-aware superscalar microprocessor," Proceedings of International Symposium on Quality Electronic Design (ISQED '02), pp. 517-522, 2002
- [18] H. Seongmoo, K. Barr, and K. Asanovic, "Reducing power density through activity migration," Proceedings of the 2003 international symposium on Low power electronics and design (ISLPED '03), pp. 217-222, 2003
- [19] K. Patel, W. Lee, and M. Pedram, "Active bank switching for temperature control of the register file in a microprocessor," Proceedings of the 17th great lakes symposium on Great lakes symposium on VLSI, pp. 231-234, 2007
- [20] H. Jang, E. Chung, and S. Chung, "Adopting the Banked Register File Scheme for Better Performance and Less Leakage," ETRI journal, vol. 30, pp. 624-626, 2008.

- [21] HotSpot, <http://lava.cs.virginia.edu/HotSpot/>
- [22] K. Sankaranarayanan, S. Velusamy, M. Stan, and K. Skadron, "A case for thermal-aware floorplanning at the microarchitectural level," *Journal of Instruction-Level Parallelism*, vol. 7, pp. 1–16, Oct. 2005.
- [23] Y. Han and I. Koren, "Simulated Annealing Based Temperature Aware Floorplanning," *Journal of Low Power Electronics*, vol. 3, p. 141, 2007.
- [24] V. Nookala, D. J. Lilja, and S. S. Sapatnekar, "Temperature-Aware Floorplanning of Microarchitecture Blocks with IPC-Power Dependence Modeling and Transient Analysis," *Proceedings of the 2006 international symposium on Low power electronics and design (ISLPED '06)*, pp. 298–303, 2006
- [25] C. Chun-Ta, Z. Xinyi, H. Lei, and J. Tom Tong, "Temperature aware microprocessor floorplanning considering application dependent power load," *Proceedings of IEEE/ACM International Conference on Computer Aided Design (ICCAD 2007)*, pp. 586–589, 2007
- [26] M. Mutyam, F. Li, V. Narayanan, M. Kandemir, and M. Irwin, "Compiler-directed thermal management for VLIW functional units," *Proceedings of ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems (LCTES 2006)*, p. 172, 2006
- [27] B. C. Schafer, L. Yongho, and K. Taewhan, "Temperature-Aware Compilation for VLIW Processors," *IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA 2007)*, pp. 426–431, 2007
- [28] M. Huang, J. Renau, S. Yoo, and J. Torrellas, "A framework for dynamic energy efficiency and temperature management," *Proceedings of International Symposium on Microarchitecture (MICRO 2000)*, pp. 202–213, 2000
- [29] K. Skadron, "Hybrid architectural dynamic thermal management," *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE '04)*, pp. 10–15, 2004

- [30] M. Goma, M. Powell, and T. Vijaykumar, "Heat-and-run: leveraging SMT and CMP to manage power density through the operating system," In Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'04), vol. 39, pp. 260–270, 2004.
- [31] A. Kumar, S. Li, P. Li-Shiuan, and N. K. Jha, "HybDTM: a coordinated hardware–software approach for dynamic thermal management," Proceedings of the 43rd annual Design Automation Conference 2006 (DAC '06), pp. 548–553, 2006
- [32] A. Kumar, S. Li, P. Li-Shiuan, and N. K. Jha, "System–Level Dynamic Thermal Management for High–Performance Microprocessors," IEEE transactions on Computer–Aided Design of Integrated Circuits and Systems, vol. 27, pp. 96–108, 2008.
- [33] J. Choi, C. Cher, H. Franke, H. Hamann, A. Weger, and P. Bose, "Thermal–aware task scheduling at the system software level," Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED'07), 2007
- [34] SIA, "Technology Roadmap for Semiconductors," 2005.
- [35] K. Skadron, M. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan, "Temperature–aware microarchitecture: Modeling and implementation," ACM Transactions on Architecture and Code Optimization (TACO), vol. 1, pp. 94–125, 2004.
- [36] SPEC(Standard Performance Evaluation Corporation) CPU2000, <http://www.spec.org/cpu2000/>
- [37] Z. Lu, J. Lach, M. Stan, and K. Skadron, "Improved thermal management with reliability banking," IEEE MICRO, pp. 40–49, 2005.
- [38] 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, 2000
- [39] T. Austin, E. Larson, and D. Ernst, "SimpleScalar: an infrastructure for computer system modeling," COMPUTER, vol. 35, pp. 59–67, 2002.

- [40] R. E. Kessler, "The Alpha 21264 microprocessor," *Micro, IEEE*, vol. 19, pp. 24–36, 1999.
- [41] 이병석, 김철홍, 이정아, "온도 인지 마이크로프로세서에서 연산 이관을 위한 유닛 선택 기법," *정보과학회논문지 : 컴퓨팅의 실제*, 제16권, 제2호, 212–216 페이지, 2010.
- [42] B.-S. Lee, M. Zeng, C. H. Kim, and J.-A. Lee, "Comparative Study of Static/Dynamic Computation Migration for Dual Integer Register File Architecture," *International Conference on Multimedia, Information Technology and its Applications (MITA 2009)*, pp. pp.218–219, 2009

저작물 이용 허락서

학 과	컴퓨터학과	학 번	10041056	과 정	박사
성 명	한글) 이 병 석 한문) 李 炳 錫 영문) Lee, Byeong-Seok				
주 소	전라남도 화순군 화순읍 계소리 오성 2 차 101 동 808 호				
연락처	e-mail : novrain@chosun.ac.kr				
논문제목	한글) 듀얼 FPU 와 열전도 지연 공간을 활용한 저온도 마이크로프로세서 구조 설계				
	영문) Thermal-Aware Microprocessor Design utilizing Dual FPU and Heat Conduction Delay Space				


본인이 저작한 위의 저작물에 대하여 다음과 같은 조건 아래 조선대학교가 저작물을 이용할 수 있도록 허락하고 동의합니다.

- 다 음 -

1. 저작물의 DB구축 및 인터넷을 포함한 정보통신망에의 공개를 위한 저작물의 복제, 기억 장치에의 저장, 전송 등을 허락함
2. 위의 목적을 위하여 필요한 범위 내에서의 편집과 형식상의 변경을 허락함(다만, 저작물의 내용변경은 금지함)
3. 배포·전송된 저작물의 영리적 목적을 위한 복제, 저장, 전송 등은 금지함
저작물에 대한 이용기간은 5년으로 하고, 기간종료 3개월 이내에 별도의 의사 표시가 없을 경우에는 저작물의 이용기간을 계속 연장함
4. 해당 저작물의 저작권을 타인에게 양도하거나 출판을 허락을 하였을 경우에는 1개월 이내에 대학에 이를 통보함
5. 조선대학교는 저작물 이용의 허락 이후 해당 저작물로 인하여 발생하는 타인에 의한 권리 침해에 대하여 일체의 법적 책임을 지지 않음
6. 소속 대학의 협정기관에 저작물의 제공 및 인터넷 등 정보통신망을 이용한 저작물의 전송·출력을 허락함

동의여부 : 동의(O) 반대()

2010 년 7 월

저작자 : 이 병 석 

조선대학교 총장 귀하