와이드 이슈 프로세서에서 이슈별 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형 예측기

전병찬*, 이영규**

요약

슈퍼스칼라 프로세서에서 명령어 수준 병렬성((Instruction Level Parallesim, :ILP)을 활용하여 명령들 사이에 존재하는 제어종속과 데이터 종속관계를 극복하는 것이 필수적이다. 최근의 값 예측기는 프로세서의 명령 이슈율이 커짐에 따라 예측 테이블의 갱신이 테이블의 참조 속도를 따라가지 못하여 예측기의 성능이 저하되는 경향이 있다. 본 논문에서는 이러한 성능저하를 줄이기 위해 명령의 결과가 나올 때까지 기다리지 않고 테이블 값을 모험적으로 갱신(speculative update)하는 명령어 이슈인 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형 예측기를 제안한다. 제안된 방식의 타당성을 검증하기 위해 SimpleScalar 시뮬레이터 상에 제안된 예측기를 구현하여 SPECint95 벤치마크로 시뮬레이션하고, 명령어 이슈(4,8,16)별로 예측률, 예측정확도, 성능향상을 평가한다.

A Hybrid Predictor using Stride Value Predictor Speculative Updates per Issue on Wide-Issue Processors

Byoung-Chan Jeon*, Young-Kyu Lee**

ABSTRACT

In a Superscalar Processor, it is essential to overcome the control dependency and data dependency between instructions by using the instruction level parallelism (ILP). The recent value predictors have a tendency to degrade performance according to the growth of the processor's instruction issue rate. This paper proposes a hybrid predictor that is instruction issue updating speculatively the value of the table without having to wait until the results of the instruction come out to reduce the performance degradation. To verify the validity of the proposed approach, we implement the predictor in SimpleScalar simulator and simulate it with SPECint95 benchmark. We also evaluate the prediction rate, the prediction accuracy, and the performace improvement using the instruction issue (4, 8, 16).

Key Words : Superscalar, ILP, prediction rate, prediction accuracy, performance improvement

* 청운대학교 방송영상학과(✉jbc66@chungwoon.ac.kr)
** 혜천대학교 사회복지과
· 제1저자(First Author) : 전병찬 · 교신저자(Correspondent Author) : 이영규
· 접수일(2012년 5월 2일), 수정일(1차 : 2012년 6월 4일), 게재확정일(2012년 6월 7일)

I. 서 론

슈퍼스칼라 프로세서는 ILP(Instruction Level Parallesim)[1,2]를 이용하여 다수의 명령을 동시에 이슈하고 처리하여 성능을 향상시키고 있다. ILP의 주요 장애요소는 명령어간의 종속관계인 이름종속(name dependency) 및 제어 종속(control dependency)과 데이터 종속(data dependency)관계에서 병렬처리를 방해한다. 이름종속관계는 레지스터나 메모리의 데이터 저장위치의 재사용으로 기인한 것으로 정적 또는 동적 재명명(renaming)기법으로 제거할 수가 있다. 제어종속관계는 분기의 결과에 따라 프로그램의 제어흐름이 변경되는 조건분기에 의해 발생한다. 데이터 종속관계는 현재 명령이 이전명령의 수행결과를 참조할 때 발생하고, 이전명령의 결과가 발생할 때까지 현재의 명령은 실행할 수가 없다. 최근에는 명령들 간의 데이터 종속관계를 극복하고 명령들을 조기에 공급, 이슈하기 위해 값 예측기가 개발 연구되고 있다[3]. 슈퍼스칼라[4] 프로세서에서 값 예측기는 한 명령어의 결과를 미리 예측하여 이 명령과 데이터 종속관계가 있는 명령들을 모험적으로 실행하여 성능을 향상시킨다. 최근의 값 예측기는 프로세서의 명령 이슈율이 커짐에 따라 예측 테이블의 갱신이 테이블의 참조 속도를 따라가지 못하여 예측기의 성능이 저하되는 경향이 있다[5,6,7,8]. 본 논문에서는 이러한 성능저하를 줄이기 위해 명령의 결과가 나올 때까지 기다리지 않고 테이블 값을 모험적으로 갱신하는 명령어 이슈(4,8,16)인 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형 예측기를 제안한다.

II. 기존 연구

값 예측(value prediction)을 위해 여러개의 기법들을 제안 하였다. 값 예측기에는 최근 값 예측기(last value prediction), 스트라이드 값 예측기(stride value prediction), 혼합형 값 예측기(Hybrid value prediction)을 구분할 수가 있다.

2.1 최근값 예측기(Last Value Predictor)

Lipasti에 의해 개발된 최근 값 예측기는 프로그램의 실행동안 대부분 명령 값의 변화가 적음을 주목하고 레지스터 값을 변경하는 명령들의 재사용되는 값의 국한성(locality)을 실험하여 이를 근거로 최근 값 예측기를 제안하였다. 최근 값 예측기는 값 예측장치를 위해 그림 1과 같은 2-단계 예측 구조를 갖는다. 첫 번째 레벨에서는 예측값을 생성하고 두 번째 레벨에서는 예측의 정확성 여부를 결정한다. CT(Classification Table)와 VPT(Value Prediction Table) 모두 예측되는 명령의 주소(PC)에 의해 인덱스되고 직접매핑 되어진다. CT의 엔트리는 유효 엔트리(valid entry)를 나타내는 1비트이거나 PC의 상위 비트를 구성된 태그를 저장하는 유효필드와 한 개 또는 여러개의 비트들로 구성된 포화카운터인 예측 히스토리(prediction history)로 구성된다.

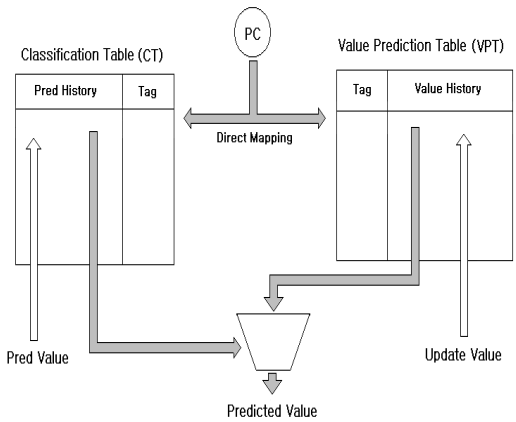


그림 1. 최근 값 예측기 구조
Fig 1. last value predictor

예측 히스토리는 예측 값에 증·감한다. CT는 특정 명령이 예측 가능한지를 결정한다. VPT는 유효필드와 LRU 정책에 따라 유지되는 하나이상의 32-64 비트 값으로 구성된 값 히스토리(value history) 필드로 구성된다.

2.2 스트라이드 값 예측기(Stride Value Predictor)

Gabbay 와Wang[9] 등은 상태 필드(state field), 값 필드(value field), 스트라이드 필드(stride field) 등이 저장된 VHT(Value History Table)로 구성된 스트라이드 값 예측기를 설계하였다. 스트라이드 값 예측기는 한 명령어의 연속적인 특징들의(instances) 결과 값들이 변화되어지는 스트라이드를 검사하여 이들 결과 값들이 일정한 간격으로 변하면 그 명령어의 장래 값들의 특징들의 결과 예측이 용이함을 착안하여 설계되었다. 이러한 스트라이드 값 예측기는 순환 명령이나 일정하게 배열의 값이 증감하는 프로그램과 데이터를 캐시로 선행페치하기 위해 주소를 생성하는데 매우 잘 수행된다. 스트라이드를 근거로 한 예측기에서는 한 명령에 대해 최근에 수행된 두 개의 결과 값의 차인 스트라이드를 구하고, 다음에 이 명령의 수행 시 이전의 결과 값에 스트라이드를 포함하여 예측한다.

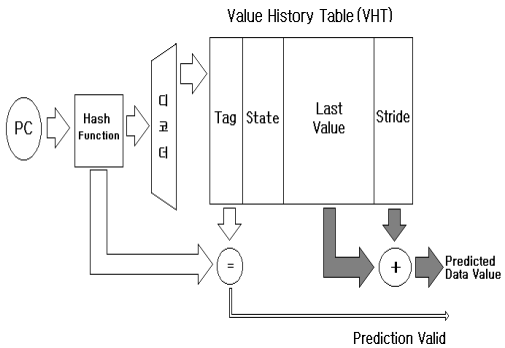


그림 2. 스트라이드 값 예측기 구조
Fig 2. stride value predictor

2.3 혼합형 예측기(Hybrid Value Predictor)

Wang[9]등은 벤치마크 프로그램 특성에 따라서는 스트라이드 값 예측기가 2-단계 값 예측기 보다 예측 정확도가 더 높은 사실에 주목하고 스트라이드 값 예측기와 2-단계 값 예측기를 결합한 혼합형 예측기도 제안되었다[10]. 2-단계 값 예측기와 혼합형 예측기는 일종의 간소화된 FCM(Finite Context Method) 예측기로 볼 수 있으며 하드웨어 적으로 구현 가능한 내용형 예측기이다. 그러나 VHT의 테이블의 각 엔트리에 대해 4개의 값이 저장되고, 다른 엔트리에 같은 값이 중복 저장되어 테이블의 크기가 크다는 문제점이 있다. 그림 3은 각각 혼합형 예측기의 엔트리와 구조를 보여준다.

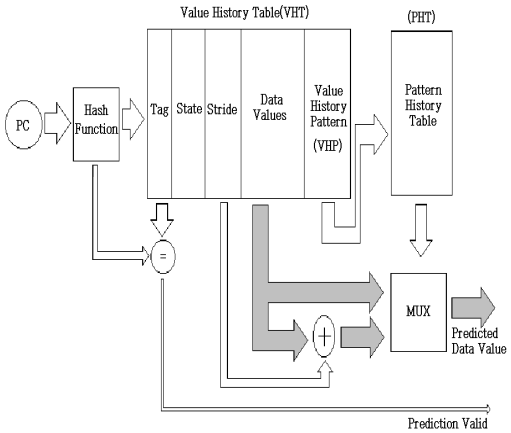


그림 3. 혼합형 데이터 값 예측기
Fig 3. hybrid data value predictor

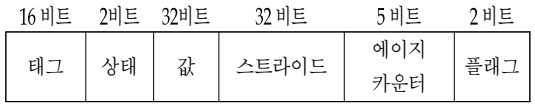
이러한 혼합형 예측기는 일정한 값으로 증감하는 명령을 예측하는 스트라이드 값 예측기와 2-단계 값 예측기를 결합한 것으로 확신 카운터(confidence counter)의 값에 의하여 값을 예측하는 방법이다. 명령어가 두 개의 예측기에서 모두 엔트리를 갖고 있다면, 카운터의 값이 높은 예측기의 예측 값을 선택한다. 따라서 혼합형 예측기는 스트라이드의 특성을 갖는 명

령과 다양한 패턴을 갖는 명령어를 모두 예측하게 됨으로써 높은 예측 정확도를 얻을 수 있지만 하나의 명령어가 두 개의 테이블 엔트리에 중복 저장되어 높은 하드웨어의 비용을 필요로 한다는 단점을 갖고 있다. 본 논문에서는 이러한 성능저하를 줄이기 위해 명령의 결과가 나올 때까지 기다리지 않고 테이블 값을 모험적으로 갱신하는 명령어 이슈(4,8,16)인 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형 예측기를 제안한다.

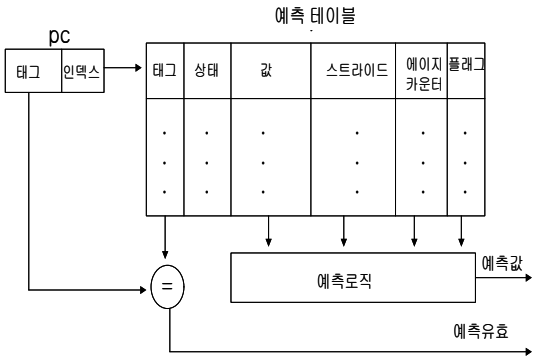
III. 스트라이드 값 예측기 모험적 갱신

3.1 스트라이드 예측기 구조 및 동작

그림 4는 스트라이드 값 예측기 모험적 갱신의 구성도이다[11,12,13].



(a)예측 테이블 엔트리의 필드

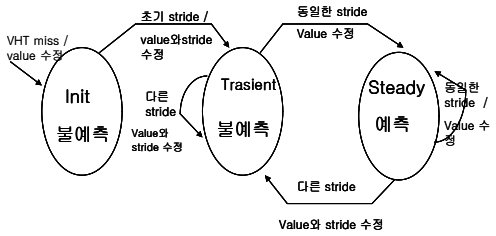


(b) 블럭도

그림 4. 스트라이드 예측기의 구성도
Fig 4. stride value predictor

그림 4의 (a)는 예측 테이블 엔트리의 필드 구성을 나타낸다. 태그(tag) 필드는 PC로 인덱스된 엔트리의 매치 여부를 검증하기 위한 필드로 테이블의 크기에 따라 가변인데 Simplescalar 머신 상에서 8K 앤트리를 갖는 테이블의 경우 16비트가 된다.

상태필드는 3가지의 예측 상태를 표시하는 필드로 초기상태, 전이상태, 안정 상태의 값을 표시하기 위해 2비트로 표현된다. 값 필드는 가장 최근의 명령 수행 결과 값을 저장하기 위해 32비트가 필요하고, 스트라이드 필드는 최근 두 명령의 결과 값의 차를 저장하기 위해 32비트로 표현된다. 또한, 에이지 카운터는 모험적 갱신을 위해 도입된 카운터이며, 5비트를 갖는다. 플래그 필드 모험적 갱신을 위해 테이블의 상태를 표시하는 2비트의 크기를 갖는다. (b)는 전체 예측기 블럭도를 보여주며, 명령이 페치될 때 PC를 이용하여 테이블을 인덱스 한다. 이때 태그를 비교하여 매칭되면 예측유효 비트를 1로 세팅하고, 테이블 미스인 경우에는 0으로 세팅하여 예측 값이 유효하지 않음을 표시한다. 모험적 갱신이 아니고 상태의 값이 안정상태인 정상적인 경우에는 테이블의 값 필드와 스트라이드 필드의 값을 더하여 인덱스 된 명령의 값을 예측한다. 스트라이드 예측 동작은 그림 5와 같다. 스트라이드 동작은 명령어를 만나게되는 첫 번째 시간에는 어떤 예측도 일어나지 않는다.



[스트라이드 예측을 위한 3-상태도]

그림 5. 스트라이드 예측동작
Fig. 5 stride prediction motion

그 명령어가 결과를 생성할 때 VHT에 앤트리가 할당되고 다음의 동작(action)이 발생하게 된다. 다음의 동작은 계속되며 결과 값은 앤트리의 값 필드에 저장되고 그 앤트리의 상태는 Iint(초기상태)로 설정된다. 초기상태에 있는 동안 동일 명령의 다른 인스턴스를 만날 때 예측은 일어나지 않고 스트라이드 시퀀스의 처음 데이터 값인 첫 번째 데이터(D1)이 생성되고 , 다음 예측될 스트라이드 S1은 D1-[VHT 앤트리내의 이전 값]으로 계산되며, D1와 S1인 값을 스트라이드 필드에 저장한다. 이때 상태(state)는 천이상태(Transient state)로 설정된다. 천이상태에 있는 동안 동일 명령의 다른 인스턴스를 만나면 예측은 일어나지 않고 그 인스턴스의 결과(D2)를 생성할 때 다음 동작이 발생한다. 따라서, 다음 스트라이드 S2는 D2-[VHT 내의 이전 값으로 계산되고 D2는 값 필드에 저장되고 S2가 이전 스트라이드와 같으면 상태필드는 안전상태(Steady state)로 설정이 되지만 그렇지 않으면 S2가 스트라이드 필드로 저장되고 여전히 상태필드인 천이상태로 남아 있게 된다. 만약 S1 와 S2가 같으면 “안전상태” 동안에는 값 와 스트라이드 필드를 함께 추가하여 다음의 예측이 만들어진다.

3.2 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형 예측기 설계

그림 6은 본 논문에서 제안한 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형 예측기의 하드웨어 구조로서 LVT, SVT, 2VT를 포함한 혼합형 예측기를 보여주고 있다. 그림에서 보여주듯이 LVT는 상수 시퀀스의 반복 패턴을 갖는 명령들의 값을 예측하기 위한 테이블로써 가장 최근 결과의 데이터 값 필드와 카운터 필드로 구성된다. 예측은 카운터 값이 임계값보다 크면 저장된 결과 값으로 예측을 한다. 명령의 실행후에 예측의 올바른 것으로 판명되면 키운터를 증가시키고, 그렇지 않으면 감소한다. SVT는 상태필드 ,스트라이드필드, 결과의 데이터 값 필드, 카운터필드, 에이지 카운터(age counter)필드로 구성되어 있다.

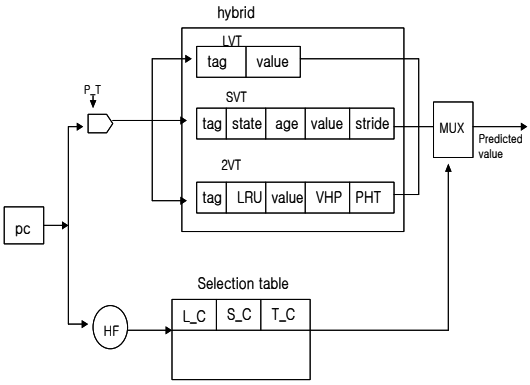


그림 6. 스트라이드 값 예측기 모험적 갱신를 적용한 혼합형 예측기
Fig 6. hybrid predictor using stride value predictor speculative updates

상태필드는 SVT의 현재 동작상태를 나타내며 그림 5와 같이 “Init", "Transient", "Steady" 상태를 갖는다. 스트라이드는 연속된 이전의 두 결과 값의 차이이며, 결과 값은 최근의 수행된 결과 값을 나타낸다. 에이지 카운터 필드는 스트라이드에서 지연갱신되는 테이블 앤트리를 추적하고 테이블의 값을 모험적으로 갱신하기 위해 사용되며, 테이블 참조시에는 증가하고 데이블 갱신시에는 감소하여 갱신의 지연을 감지한다. 2VT는 2-단계 데이터 값 예측기로 VHT와 PHT로 구성된다. 첫 번째 단계인 VHT는 과거의 실행결과를 기록하는 테이블로 최근에 실행된 4개의 데이터 값을 기록하는 데이터 필드와 명령의 마지막 P개의 실행을 추적하는 2 * p 비트 패턴으로 표시되어 과거패턴 필드로 구성된다. 두 번째 단계인 PHT는 VHT의 각 과거 패턴 필드(VHP)를 참조하여 PHT의 앤트리를 선택한다. 선택된 PHT의 앤트리 중에서 가장 큰 값을 갖는 카운터의 값이 임계 값을 초과할 경우 카운터와 대응하는 VHT의 데이터 값 필드에 저장된 값으로 예측한다. 만약 한계 값을 넘지 않으면 예측을 하지 않는다. 예측후

에 실행된 걸과 값에 따라 VHT의 과거 패턴은 2비트 왼쪽으로 시프트 시킨다. 또한 예측이 맞으면 선택된 PHT의 카운터는 증가되고, 다른 카운터는 감소된다.

또한, 반입된 명령어 결과 값 예측기를 Lookup 할 때 PC를 사용하여 HF(hashing function)에 적용하여 Selection table을index 하게 된다. Selection table entry구조는 L_C(last counter), S_C(stride counter), T_C(Two-level counter)로 되어 있다. 최대 값(maximum value)을 갖는 counter에 대응하는 혼합형 예측기의 해당 예측기로 예측을 하게 되는데 L_C가 최대(max)인 경우에는 LVT 예측을 하게 되며, S_C가 최대(max)인 경우에는 SVT로 예측하게 되며, T_C가 최대인 경우에는 2VT로 예측하게 된다.

IV. 결 론

4.1 실험방법

제안된 명령어 이슈인 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형 예측기를 비교 측정하기 위하여 4~16 이슈의 명령의 이슈 크기(Issue-width)에 대한 3가지 시스템의 기본적인 구조를 사용한다. 표 1에서는 기본구조에서 사용된 파라미터와 같이 사이클당 4~16 이슈를 기본 값으로 하였고, 분기 예측을 위해서 2-way의 2K 앤트리를 갖는 BTB(Branch Target Buffer)와 정확한 분기예측을 가정한 결합형 분기 예측기(Combine branch predictor)를 사용 하였다. 그리고 4K 크기를 갖는 명령 캐시(instruction Cache)와 데이터 캐시(Data cache)를 실험 하였다.

표 1. 실험에 사용된 기본 시스템 구성도
Table 1. basic system composition using experiment

	파라메터 (parameter)	값(value)	의미(Comments)
Processor Core	RUU size LSQ size Fetch width Decode width	256 64 4-16 inst/cycle 4-16 inst/cycle	instruction windows Load-store queue
	Issue width Commit width Function Unit	4-16 inst/cycle 4-16 inst/cycle 8 int alu(1) 4 fp add(2) 2 int mul(2) div(12) 2 fp mul(4) div(12)	( ) is latency
Branch Predictor	BTB Hubrid Branch Predictor RAS	2K,2-way gshare 16K 8bit history bomodal 16K 32	Branch target buffer Gshare + Bimodal Return addr, stack
Cache	L1 D- cache L1 I-cache L2 cache	512K, 2-way 32 byte blocks 4 ports 3-cycles latency 512K, direct-map, 32 byte blocks unified, 1M, 4-way, 64 blocks	Data cache Instruction cache Secondary cache

표 2는 스트라이드 값 모험적 갱신 예측기 테이블 크기는 각각 8K 앤트리로 구성된 4개의 히스토리를 포함하고 있는 VHT 와 8K 앤트리로 구성된 PHT의 테이블로 구성되어 있으며, 제안한 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형 예측기는 4K의 LVT와 1K의 SVT, 그리고 8K 앤트리로 구성된 2VT 테이블 크기를 갖는다. 테이블 크기는 예측된 명령어의 명령 비율로부터 구동된다.

표 2. 실험에 사용된 값 예측기
Table 2. value predictor using experiment

값 예측기	테이블 구성
Stride value predictor speculative update	8K SVT(Stride Value Table)
hybrid value predictor using stride value predictor speculative update	4K LVT(Last Value Table) 1K SVT(Stride Value Table) 8K 2VT(2-level Value Table) 32 queue size, 2 age counter

시뮬레이션에 사용한 SPECint95 벤치마크 프로그램은 표 3과 같이 시뮬레이션에 사용된 입력자료를 생성하기 의하여 GNU's gcc2.7.2를 UNIX 플랫폼상(platform)에 최적화 컴파일된 SPECint95 벤치마크 프로그램을 사용하였고, 수행 명령결과들의 평균은 레

지스터 기록(register-write)명령들이다.

표 3. 벤치마크 프로그램
Table 3. benchmark program

벤치마크 프로그램	입력자료	명령의 크기	기본 IPC		
			4-이슈	8-이슈	16-이슈
compress	10000 e 2231	35,261,720	2.1292	2.7738	2.8906
gcc	jump.i	38,772,653	1.4727	1.7692	1.8419
go	5 9	81,530,040	1.3631	1.5204	1.5588
ijpeg	tinyrose.ppm	63,409,713	2.2733	3.2256	3.6463
li	queen6.lsp	41,524,887	1.8413	2.5894	2.8712
m88ksim	dhry.big.100	240,738,821	3.2317	5.5756	6.7658
perl	scrabbl.in	40,353,113	1.8729	2.3622	2.5318
vortex	persons.250	66,740,617	1.9817	2.7003	3.0165
평 균		76,041,446	2.0207	2.8145	3.1403

4.2 성능분석

본 논문에서는 제안한 명령어 이슈인 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형 예측기인 명령어 이슈 길이(4,8,16)의 예측률, 예측정확도, 성능향상을 평가 분석한다.

4.2.1 예측률

예측률은 전체 실행된 명령어들 중에서 예측기에 예측되어질 명령들의 비율이다. 그림 7은 4-이슈 머신 예측률을 보여주고 있다.

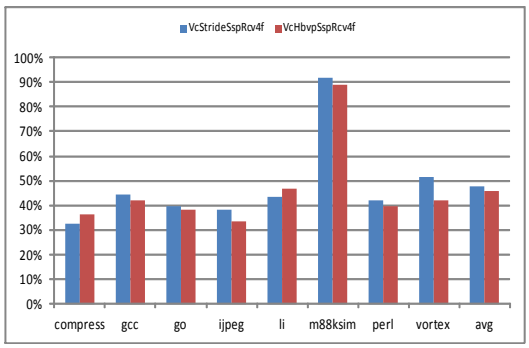


그림 7. 4-이슈 예측률
Fig. 7. 4-Issue prediction rate

그림 7에서 보여주는 것처럼 스트라이 값 예측기 모험적 갱신(VcstrideSspRcv), 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형(VcHbvpSspRcv) 예측기의 예측률은 각각 47.8%, 45.9% 임을 알 수 있다. 그림 8은 8-이슈 머신 예측률을 보여 주고 있다. 스트라이드 값 예측기 모험적 갱신 (VcstrideSspRcv), 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형 (VcHbvpSspRcv) 예측기의 예측률은 각각 49% , 47% 임을 알 수 있다.

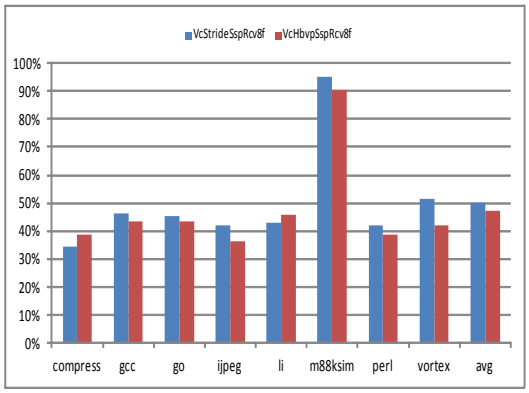


그림 8. 8-이슈 예측률
Fig. 8. 8-Issue prediction rate

그림 9는 16-이슈 머신 예측률을 보여 주고 있다. 16-이슈 일때도 마찬가지로, 스트라이드 값 예측기모험적 갱신(VcstrideSspRcv), 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형(VcHbvpSspRcv) 예측기의 예측률은 각각 51% , 48% 임을 알수 있다. 일반적으로, 예측률은 이슈별로 실험한 결과 스트라이드 값 예측기 모험적 갱신 (VcstrideSspRcv)이 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형 (VcHbvpSspRcv) 예측기 보다 예측률이 높음을 알 수 있었다.

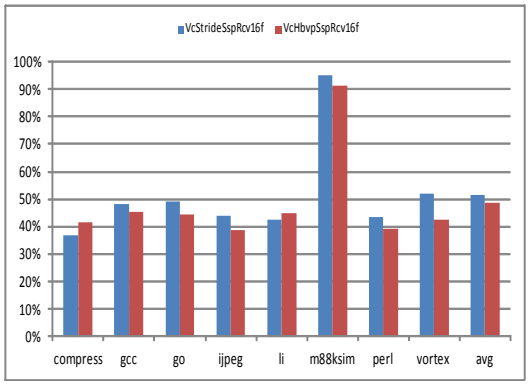


그림 9. 16-Issue 예측률
Fig.9 16-Issue prediction rate

4.2.2 예측 정확도

예측 정확도는 값 예측이 가능한 명령중에 안정상태에 도달하여 실제 값을 예측을 한 명령 중에 올바르게 값을 예측한 비율을 의미한다. 그림 10 는 4 이슈 일 때 예측 정확도를 보여주고 있다.

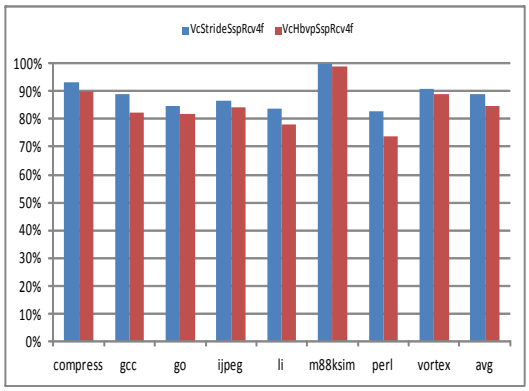


그림 10. 4-이슈 예측 정확도
Fig.10 4-Issue prediction accuracy

스트라이드 값 예측기 모험적 갱신 (VcstrideSspRcv), 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형(VcHbvpSspRcv) 예측기의 예측정확도는 각각 88.8%, 84%임을 알 수 있다. 그림 11은 8-이슈 예측정확도를 보여 주고 있다.

스트라이드 값 예측기 모험적 갱신 (VcstrideSspRcv), 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형(VcHbvpSspRcv) 예측기의 예측정확도는 각각 88.2%, 83.6%임을 알 수 있다.

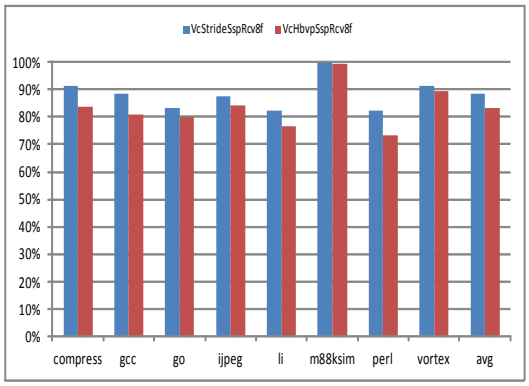


그림 11. 8-이슈 예측 정확도
Fig.11 8-Issue prediction accuracy

그림 12는 16-이슈 예측정확도를 보여 주고 있다. 스트라이드 값 예측기 모험적 갱신 (VcstrideSspRcv), 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형 (VcHbvpSspRcv) 예측기의 예측정확도는 각각 88.2.6%, 83.17% 임을 알수 있다.

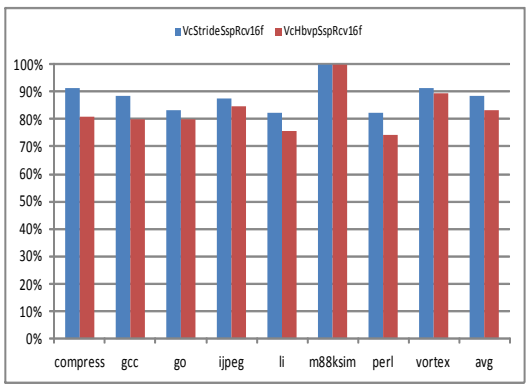


그림 12. 16-이슈 예측 정확도
Fig.12 16-Issue prediction accuracy

예측정확도는 이슈별로 실험한 결과 스트라이드 값 예측기 모험적 갱신(VcstrideSspRcv)이 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형 (VcHbvpSspRcv) 예측기 보다 예측률이 높음을 알 수 있었다.

4.2.3 성능향상

성능향상은 값 예측을 시도하지 않은 경우를 기준으로 하여 값 예측을 하였을 경우의 성능변화를 나타내는 것이다. 그림 13은 4-이슈 성능향상을 보여주고 있다.

스트라이드 값 예측기 모험적 갱신 (VcstrideSspRcv), 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형(VcHbvpSspRcv) 예측기의 성능향상은 각각 1.2%, 2.0% 임을 알 수 있다.

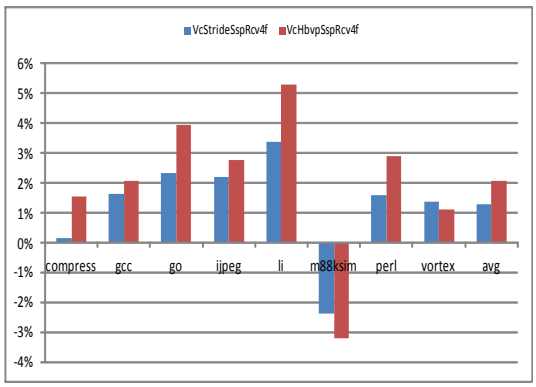


그림 13. 4-이슈 성능향상
Fig.13 4-Issue speedup

그림 14는 8-이슈 성능향상을 보여주고 있다. 스트라이드 값 예측기 모험적 갱신(VcstrideSspRcv), 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형 (VcHbvpSspRcv) 예측기의 성능향상은 각각 6.6%, 10.7% 임을 알 수 있다.

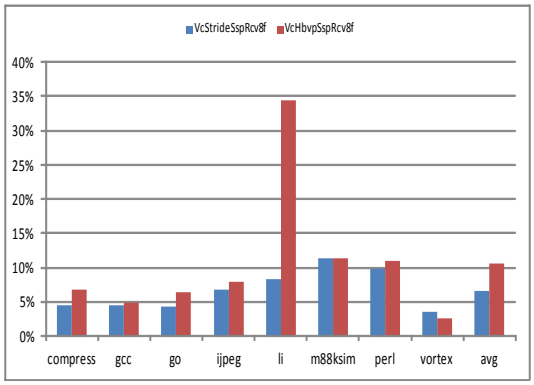


그림 14. 8-이슈 성능향상
Fig.14 8-Issue speedup

그림 15는 16-이슈 성능향상을 보여주고 있다. 스트라이드 값 예측기 모험적 갱신(VcstrideSspRcv), 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형 (VcHbvpSspRcv) 예측기의 성능향상은 각각 4.9%, 4.7% 임을 알 수 있다. 성능향상은 이슈별로 실험한 결과 4-8이슈에서는 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형(VcHbvpSspRcv) 예측기가 높음을 보였지만, 16-이슈인 경우는 스트라이드 값 예측기 모험적 갱신 이 높음을 알 수 있었다.

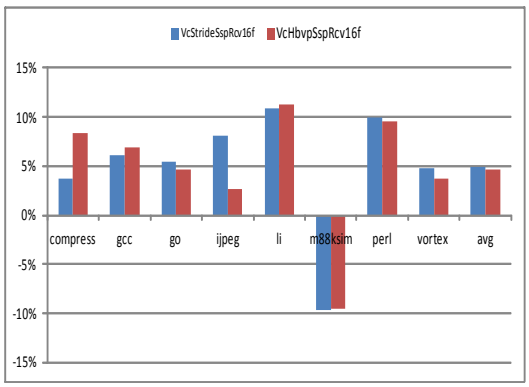


그림 15. 16-이슈 성능향상
Fig.15 16-Issue speedup

Ⅴ. 결론

본 논문에서는 슈퍼스칼라 프로세서에서 값 예측기의 예측 테이블 갱신 지연으로 인한 성능 저하를 위해 명령의 결과가 나올 때 까지 기다리지 않고 테이블 값을 모험적으로 갱신하는 스트라이드 값 모험적 갱신 예측기를 적용한 혼합형 예측기를 제안 하였다. 실험 결과 이슈별(4,8,16)로 보았을 때 예측률에서는 스트라이드 값 예측기 모험적 갱신이 높음을 알 수 있었으며, 예측정확도에서도 스트라이드 값 예측기 모험적 갱신이 높음을 알 수 있었다. 또한 , 성능향상은 4-8 이슈에서는 스트라이드 값 예측기 모험적 갱신이 높음을 보였지만, 16-이슈에서는 스트라이드 값 예측기 모험적 갱신을 적용한 혼합형 예측기가 높음을 알 수 있었다. 향후 연구과제로는 반입되는 명령어에 적합한 예측기를 선택하여 이슈별 하드웨어 비용을 줄이면서 예측 정확도나 성능향상을 더욱 향상시키도록 하는 연구가 필요하다.

참고문헌

[1] S.A.Mahlke, "Exploiting Instruction Level Parallelism in the Presence of Conditional Branches",Ph.D dissertation, Department of Electrical and Computer Engineering, University of Illinois,Urbana, IL, 1996.

[2] J.Gonzalez and A.Gonzalez, "The Potential of Data Value Speculation to Boost ILP", Proceedings of the International Conference on Supercomputing, 1998.

[3] F.Gabbay, and A.Mendelson, "The Effect of Instruction Fetch Bandwidth on Value Prediction", Proceedings of the 25th International Symposium on Computer Architecture(ISCA-25), p.272-281, 1998.

[4] D.Burger and T.Austin, "The Simplescalar Tool Set, Version2.0",TechnicalReport CS-RT-97-1342, University of Wisconsin, Madison, June 1997.

[5] T.Heil, Z.Smith, and J.Smith, "Improving Branch Predictors by Correlating on Data Values", Proceedings of the 32nd International Symposium

[6] M.Lipasti, and J.Shen, "Exceeding th Limit via Value Prediction",Proceedings of the 29th International Symposium on Microarchitecture (MICRO-29), Dec. 1996.

[7] Y.Sazeides, and J.Smith, "The Predicatability of Data Values",Proceedings of the 30th International Symposium on Microarchitecture (MICRO-30), Dec. 1997.

[8] K.Wang, M.Franklin, "Highly Accurate data value Predictions using Hybrid Predictor, "Proceedings of the 30th International Symposium on Microarchitecture (MICRO-30), Dec. 1997.

[9] T. Yeh and Y.Patt, "Two-Level Adaptive Branch Prediction,"Proceedings of the 24th International Symposium microarchitecture (MICRO-24), Nov. 1991.

[10] T.Sato, "Analyzing Overhead of Reissued Instructions on Data Speculative Processors",Workshop on Performance Analysis and its impact on Design Held in Conjunction with ISCA-25, 1998.

[11] K.Skadron, M.Martonosi, and D.Clark. "Speculative Updates of Local and Global Branch History" A Quantitative Analysis," Journal of Instruction-Level Parallelism, vol.2, Jan. 2000.

[12]Byoung Chan Jeon, Sang Jeong Lee,“Speculative Update of a Stride Value Predictor in Wide-Issue Processors",Journal of KIISE: Computing Practices and Letters Vol.11 No.28, 2001.

[13] Hong Jun Park, Young Ho Shin,Young Il Cho,“A Hybride Value Predictor using Speculative Update in Superscalar Processors”,Journal of KIISE: Computing Practices and Letters Vol.28 No.11, 2001.

감사의 글

이 논문은 2011학년도 청운대학교 학술연구조성비 지원에 의해 연구되었음.

저자소개

전 병 찬(Byoung-Chan Jeon)

1994년 수원대학교 전자계산학과 (공학석사)

2002년 순천향대학교 전산학과(공학박사)

2005년~현재 청운대학교 방송영상학과 조교수

※ 관심분야 : 컴퓨터구조, 홈 네트워크, 모바일

이 영 규(Young-Kyu Lee)

1999년 순천향대학교 전산학과(공학박사)

1997년~2001년 극동정보대학 정보처리과 교수

2001년~현재 혜천대학 사회복지과 교수

※ 관심분야 : 병렬처리, 데이타베이스