# 계산기 아키텍처의 과거, 현재, 미래

1Kbit DRAM, 최초의 마이크로세서 Intel 4004가 1970년대 처음으로 세상에 나온 지 약 30년이 경과되었다. 4004는 4비트 프로세서로 집적 트랜지스터 수 2,300개, 동작 주파수 750KHz, 16핀으로 내장되었다.
이 시점에서 누가 현재의 고성능 프로세서의 출현을 예측이나 했을까. 4비트에는 있는데, 하나의 프로세서가 한개의 칩에 탑재되어 실용화되어, 원칩 프로세서가 스타트를 끊었던 것이다. 오늘날의 DEC Alpha21264에는 1,520만개의 트랜지스터를 집적해서 600MHz, 587핀으로 내장되었다. 이러한 발전의 원동력은 반도체 집적회로 기술의 비약적인 발전은 물론, 계산기 아키텍처, 컴파일러, OS 등을 완수하는 역할도 대단히 크다. 본 원고에서는 주로 1990년대의 계산기 아키텍처의 변천을 살펴보고 향후 10년의 발전을 전망하려 한다.

도미타 신지(富田 眞治) 교토대학 정보학 연구과

## ■ 1980년대 : RISC의 시대

1970년대 VLSI의 요람기를 거쳐 1980년대에 들어서면 칩 상의 집적도는 그렇게 크지는 않지만 고기능 프로세서를 하나의 칩에 집적시킬 수 있게 되었다. 고급언어의 보급을 배경으로 컴파일러와의 협조설계가 그려져 RISC의 원리가 워크스테이션 등 과거의 프로그램 자산이 적은 분야에 보급되었다. 그 기본적인 생각은

① 고급언어 프로그램은 컴파일러를 통해 기계명령 프로그램으로 변환되기 때문에 컴파일러를 사용하여 단순하고 고속 실행할 수 있는 기계명령 세트로도 충분하다.

② CISC에서는 기계명령 실행을 위한 제어기억이 필요하고, 매우 큰 면적을 필요로 한다. RISC에서는 이것이 필요없기 때문에 메모리 관리 장치나 부동소수점 연산장치로 그 칩 면적을 나누어 고속화할 수 있다.

③ RISC는 단순하고, 생산 사이클이 짧기 때문에 최신의 기술을 언제고 새로운 제품에 반영할 수 있다.

위의 ②,③은 칩상에서의 집적도가 비약적으로 향상된 오늘날에는 적합하지 않지만 RISC의 세련된 파이프라인 구조와 메모리 계층구조의 설계법, 컴파일러와의 협조설계의 중요성을 지적하는 등 오늘날의 프로세서에 매우 큰 충격을 주었다.

## ■ 1990년대 : 명령 레벨 병렬 처리의 시대

가장 단순한 컴퓨터에서 기계명령은 하나씩 메모리에서 끄집어내고(IF), 해석하고(D), 오퍼랜드 어드레스가 계산하고(OA), 오퍼랜드에서 꺼내고(OF), 연산 실행해서(EX), 결과가 산출된다(S). 다음의 명령 어드레스의 계산이 이들의 실행과정 도중에 계산되어 기계명령 실행이 종료되면, 다음 명령에 대해서 위와 같은 과정을 반복한다.

1990년대 컴퓨터의 고속화는 한 마디로 말하면 기계명령 레벨 병렬처리의 원리에 기초한다. 이것을 아키텍처 측면에서 기본개념을 명확하게 하는 동시에 다음과 같은 원리를 채용하면서 발전해 왔다.

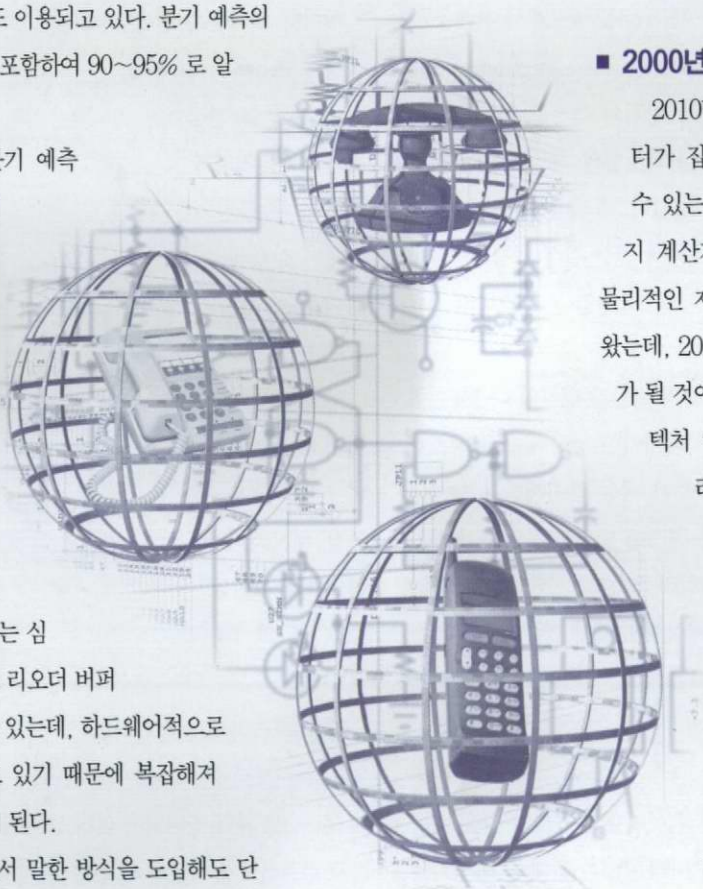
① 시간병렬성: 공장의 흐름 작업에 대응한 고안으로 명령 파이프라인 방식이라 불린다. 즉 위에서 설명한 명령실행 과정을 복수의 명령으로 오버랩하는 것이다. 스테이지(상기의 IF에서 S로 대응)수를 L로 하면 이상적으로는 L배 가속화 할 수 있다. 그러나 선행하는 명령과 후속하는 명령에는 데이터 의존관계, 제어 의존관계, 자본의 경합이 있어서 이상적인 경우처럼 명령들이 파이프라인상을 흐르지 않는다. 그래서 다음과 같은 여러가지 방법이 제안되어 실용화되었다.

② 난실행 : 선행명령이 종료되지 않아도 후속 명령이 그것과 무

관하고 실행시간이 짧으면 앞에 실행을 종료시키는 방식이 난(Out-of-Order)실행이다. 이처럼 하면 선행 명령과 후속 명령이 병렬로 작동할 수 있어 고속화시킬 수 있다.

③ 분기예측: 분기명령에는 한쪽으로 치우치는 것이 있다고 알려져 있다. 이 치우침은 에러 발생 체크를 위한 분기 명령처럼, 정적으로 정해져 있는(에러는 통상 발생하지 않기 때문에) 것도 있는데, 실행시에 변동하는 것도 다수 존재한다. 따라서 분기 명령이 TRUE(T)가 되는지, FALSE(F)가 되는지 실행 때에 예측하는 방식이 일반적으로 취급되었다. 그 분기 명령의 과거 여러차례 실행에서의 T/F 회수나 패턴 계열 정보, 혹은 그 분기 명령에 인접한 다른 분기 명령의 이력을 가미하는 등, 다양한 방식이 제안되어 실제 마이크로프로세서에도 이용되고 있다. 분기 예측의 히트률은 응용이나 방식을 포함하여 90~95% 로 알려져 있다.

④ 제어투기실행: 이 분기 예측을 더욱 진행시킨 것이 제어 투기실행 방식이다. 분기 예측을 한 명령예를 점차 실행해 버리는 방식이며, 예측이 성공하면 [훌륭하다]. 그러나 예측에 실패하면 레지스터나 메모리 상태를 원래의 상태로 복귀시켜야하는 심한 타격(패널티)을 입는다. 리오더 버퍼 등의 복구방법이 취급되고 있는데, 하드웨어적으로는 연상메모리를 이용하고 있기 때문에 복잡해져서 속도가 느려지는 요인이 된다.



⑤ 국소공간 병렬성: 앞서 말한 방식을 도입해도 단일 명령 파이프라인에서는 최대 L배의 성능 향상밖에 바라볼 수 없다. 따라서 프로그램카운터에서 가리키는 어드레스의 국소의 명령을 M개씩 동시에 패치해서, 파이프라인을 크게 하고(공간병렬), 다수의 명령을 병렬처리하는 방식이 채용되었다. 시간병렬과 조합시키면 M*L배 가속화된다. 슈퍼스칼라 방식과 VLIW(Very Long Instruction World)방식이 있다. 전자에서는 보통 기계명령을 동시에 실행하는 방식이고, 현재의 범용 프로세서 대부분에 도입되어 있다. 이것은 목적 프로그램의 레벨에서 호환성이 지켜지기 때문이다. 그러나 하드웨어는 다중도가 커지기 때문에 무척 복잡하다. 한편 VLIW방식은 컴파일러가 동시 실행 가능한 명령을 프로그램 안에서 수집하고 꽤나 긴 명령의 전용 제어 파일로 메우는 방식이다. VLIW 명령은 동시에 실행하는 것이 완전하므로 의존성의 해석 등을 실행 당시 할 필요가 없고, 하드웨어는 단순화 되어 동작 주파수를 크게 할 수 있는 특징이 있다. 그러나 호환성이나 NOP 조작의 증대 등과 같은 결점이 있어서, 지금까지는 미디어 프로세서나 슈퍼 컴퓨터의 스칼라(scalar : 하나의 수치만으로 완전히 표시되는 양) 프로세서 등 프로그램 자체에 병렬성이 있고, 전용 하드웨어화하는 데에는 표준화 등의 동향을 보고 유구조화할 필요가 있는 분야에 이용되어 왔다.

## ■ 2000년대 : 다양화의 시대

2010년에는 칩 상에 10억개의 트랜지스터가 집적되어 주파수 10GHz로 작동할 수 있는 시대가 될 것이라고 한다. 이제까지 계산기 설계자(아키텍트)는 여러 가지 물리적인 제약을 받으면서 시스템 설계를 해왔는데, 2010 년경에는 [뭐든지 가능한] 시대가 될 것이라고 한다. 그러면 2000년대 아키텍처 연구는 장미빛이라기 보다는 오히려 집적회로 디바이스 기술자와 응용기술자 사이에 틈이 있어서, 브래이크 스톨을 모색하는 등 어두운 폐쇄감이 존재하는 상태이다. 그 폐쇄감의 원인은 범용 고성능 프로세서 분야에서 Intel사의 절대적인 지배와 아키텍처 연구의 포화상태, 디바이스 기술의 밀림과 로드맵 지향, 요소기술을 경시하는 회사들의 제품개발 정책의 변화 등에 있다.

앞으로는 네트워크의 발전과 모바일 환경의 정비 등을 배경으로 해서, 고성능 대규모 서버(고학 기술 계산, 데이터 베이스, 멀티미디어(화상/CG)를 위한 슈퍼컴퓨터)를 중심으로 한 하이엔드 컴퓨터계, 퍼스널 컴퓨터나 워크 스테이션 등의 퍼스널 컴퓨터계, 게임이나 모바일 기기를 중심으로 한 로엔드 컴퓨터계로, 컴퓨터 시스템은 분명 다양하게 분화할 것이다.

## ■ 하이엔드 / 데스크 탑계

하이엔드계는 병렬처리가 보급돼서 그 요소 프로세서는 보통 마이크로세서로 된다(슈퍼 컴퓨터에서 백터 프로세서는 쇠퇴). 1990년대의 하이엔드계의 요소 프로세서와 데스크 탑은 다음과 같은 기술발전을 이루는데, 주요 기술은 메모리와 프로세서 사이의 속도괴리를 어떻게 메우는가가 초점이다.

① VLIW방식의 고속화와 범용화 : 앞서 말한 것처럼 슈퍼 스칼라에서는 하드웨어가 복잡하고 성능 향상이 한계에 도달하려 한다. 한편, 순수의 VLIW방식에서는 긴 명령 안에서의 NOP가 많아 효율적이지 않고, 또 호환성 면에서도 문제가 있었다. Intel사가 2000년에 Pentium시리즈 후속으로 그것들과 전혀 다른 명령 세트(IA-64)를 필요로 하는 프로세서 Merced를 발표해서 판매를 개시한다. Merced에서는 EPIC(Explicitly Parallel Instruction Computing)이 표방되고 있어 VLIW 방식의 특징도 포함하고 명령의 압축, NOP의 삭제, 호환성이나 확장성의 확보를 노리고 있다. Merced의 커다란 특징 중에 하나이다. 플레디 케이트 연산에서는 비교 조작 명령의 결과에 기초해서 연산 실행의 정상종료나 취소 제어를 할 수 있다. 성능이 저하되는 요인인 분기 명령의 삭제나 정적 분기 예측에 위력을 발휘한다.

② 초슈퍼 스칼라 방식 : 현재의 슈퍼 스칼라 방식의 다중도는 4정도이다. 이 다중도를 16 정도로 증가시키는 방식이 초슈퍼 스칼라 방식이다. 리오더 버퍼 등의 구조가 매우 복잡해짐과 동시에 TRUE 연속 분기 명령에서는 분기 예측이 맞더라도 명령 패치가 복수의 블록에 퍼지기 위한 시간이 걸려, 성능향상을 기대할 수 없다. 때문에 분기 패스상의 명령을 캐쉬 메모리의 연속 어드레스에 재배치하는 트레이스 캐쉬가 제안되고 있다. 32명령 다중까지 갈 수 있는 연구 그룹도 있다.

③ 대영역 공간 병렬성의 이용

· 공간 병렬 멀티 스레드 : 이 방식은 비교적 규모가 작은 슈퍼 스칼라에서 파이프라인을 여러 대 준비해 프로그램 내의 제어 의존이 없는 일련의 기계 명령계(스레드)를 병렬로 실행시키는 방식이다. 명령 레벨보다 좀 더 작은 단위의 큰 스레드 레벨에서의 병렬성을 추출해 고속화를 달성할 수 있는 가능성이 있기 때문이다. 다중도가 큰 슈퍼 스칼라보다 50% 정도 성능이 향상된다. DEC사의 차기 프로세서에서 채용되고 있다고 한다.

· 온 칩 멀티프로세서 : 스레드를 프로세스까지 크게 한 방식이다. 스누프캐쉬(캐쉬를 분기배치해서 코히런스 제어를 행하는 방식), 공유 캐쉬(캐쉬 메모리를 집중관리해 공유하는 방식), 레지스터를 공유하는 방식 등이 검토되고 있다.

위의 모든 방식들이 복수의 스레드나 프로세스를 실행할 수 있게 하는 캐쉬 메모리의 구성으로 열쇠를 쥐고 있어 칩 면적이 아직은 (?) 작으면 무척 곤란하다.

④ 데이터 투기실행과 재이용: 선행 명령의 결과를 후속 명령이 이용할 때, 후속 명령은 선행 명령이 종료되기까지 기다려야 한다. 그러나 선행 명령의 결과를 예측할 수 있는 경우가 있다. 예를 들면 배열 데이터 행방에 엑세스할 때, 어드레스는 일정한 활보로 가산된다. 이 방식에서는 선행명령과 후속명령을 동시에 실행시켜, 선행명령의 결과가 예측한대로 되면 고속화시킬 수 있다.

데이터 재이용은 선행명령의 실행에 즈음해서 그 환경을 조사해 그 환경이 전회와 마찬가지가 되면 선행 명령 자체의 실행을 생략하는 방법이다.

이들 방식에서는 많은 테이블 종류나 연상 메모리가 필요하다. 또 성능향상도 약 10 % 정도이다.

⑤ 레지스터레스 컴퓨터 : 오늘날 프로세서에서는 레지스터가 32 - 128개나 장비되어 있다. JAVA는 스타크 컴퓨터이기 때문에 레지스터는 없다. 또 슈퍼 스칼라 방식 등에서는 레지스터의 레네임 등에 연상 메모리가 필요하기 때문에 성능이 저하되는 요인이 된다. 과거의 데이터 프로 방식을 답습한 레스터가 없는 컴퓨터도 생각할 수 있다.

## ■ 로 엔드계

지금까지 아키텍트는 앞서 말한 초병렬 처리 등 무척 하이엔드한 시스템을 추구해 왔다. 거기에만 지적 호기심을 만족시키는 가치관이 있다고 해도 좋다. 그러나 휴대전화, 카나비, 모바일 기기 등의

커버를 열어보면 내장기술을 중심으로 한 학회의 논문지에는 투고되지 않은 믿을 수 없는 기술이 존재한다. 이와 같은 로 엔드 기기에서의 중심기술은 초절전화, 초경량화, 고신뢰성과 초자동화가 될 것이다. 전자식 탁상계산기의 가벼움과 소비전력으로 주머니에 둥글게 말아서 넣는 형상기업 합금처럼 주머니에서 밖으로 나오면 다시 원상태로 돌아가서 사용할 수 있는 컴퓨터가 실현된다면 매우 이상적일 것이다. 이와 같은 〔재미있는〕연구를 추진하기 위해서는 지금까지의 아키텍처 연구에 대한 가치관을 전환해서 로 앤드 기기 연구에 대한 업적평가에 대해서 다시 볼 필요가 있다. 대학 등에서 논문 지상주의는 먼저 버려야 한다. 로 앤드계에서의 아키텍처 관련 기술에는 아래와 같은 것이 있다.

① DRAM과 로직 혼재(混載) DRAM의 구조는 2차원 격자상으로 배치되어 있기 때문에 동일의 데이터로 동시 접근하는 것이 가능하다. 따라서 DRAM 중에 프로세서 등의 논리회로를 혼재하면 높은 바인드 폭과 낮은 레이텐시를 실현할 수 있어서, 곁에 붙은 회로가 적게되어 전력 소비가 줄어든다(미츠비시 전기의 32R/D에서 전력소비는 1/3정도가 된다). 화상처리, 그래픽스, 수치계산 등 정형적인 처리(연속 어드레스에 대한 데이터 접근처리)에 적용할 수 있다. 그러나 DRAM과 로직 제품 프로세스가 다르기 때문에 비용이 높다는 결점이 있다.

② 진화하는 컴퓨터 : FPGA(Field Programmable Gate Array)에서는 그 논리 구조를 변경할 수 있기 때문에 예를들면 실행빈도가 높은 연산 등에 대해서 병렬화 등으로 고속화할 수 있으리라 기대하고 있다. 1970년대에는 제어 기억장치를 각 문제마다 바꿔 쓸 수 있는 가변 구조형 컴퓨터가 제안되었는데, 실패로 끝났다. 진화한 컴퓨터의 보수나 프로그램 이식이 곤란하다는 것이 큰 이유일 것이다. EPGA에서는 어떻게 될지 흥미롭다.

③ 소프트/하드 협조설계 : 멀티미디어 시대가 되어 화상처리나 그래픽스가 친근하게 되었다. 화상처리에서는 화상의 부호화, 부호화에는 100GOPS, 복호화에는 10GOPS(OPS : Operations Per Second) 정도의 성능을 필요로 한다. 가전제품으로 많이 사용되기 때문에 소비전력도 적게 0.5-1.5W 정도일 필요가 있다.

이와 같은 멀티미디어 처리에서는 표준화가 무척 빠르게 변화한다는 난점 있다. VLIW 방식의 프로세서나 DSP 등의 범용성이 있는 기능장치에서 구성하는 것인지, 일찍이 주목받은 바 있는 시스토릭아레이 등의 전문장치로 구성하는 것인지, 또는 그런 것들의 기능분담을 어떻게 하는지가 문제이다. 성능, 소비전력, 비용 등의 균형을 해결할 수 있는 소프트/하드와의 협조설계 등이 뛰어난 CAD 도구의 실용화가 필요하다. 또 슈퍼컴퓨터 등의 하이엔드계에서 얻어온 각종 기술 게임기 등을 로 엔드계로 기술이전 하는 것도 중요시될 것이다.

## ■ 역사에 남는 아이디어

범용 컴퓨터에서의 가장 큰 브레이크스톨은 IBM 360의 개발, Xerox의 Alto로 대표되는 워크스테이션과 Ethernet 개발, 소프트웨어 위기를 반영한 고급언어 계산기의 안티테제로서 VLSI의 개발을 배경으로 제안된 RISC 프로세서의 개발, 슈퍼 스칼라, VLIW 등의 기계 명령 레벨 병렬처리 등이 있다. 어쨌든 매우 큰 충격이 컴퓨터 기술과 더 나아가서는 사회의 변혁을 가져다 주었다.

오늘날 프로세서 기술의 중신을 보면, 많은 혁명적인 아이디티가 1970년대(JAVA에서 이용된 스타크 컴퓨터, RISC나 난 실행 등은 1960년대)부터 1980년에 걸쳐 연구되고 있다. 몇 년 동안 짧은 타임스팬에서의 연구개발도 중요하지만, 10년, 20년 후의 아키텍처의 연구를 지도에 올릴 필요도 있다.

현재의 실리콘 VLSI에는 물리적 한계가 오는 것은 명확하니만큼 양자 컴퓨터 등의 새로운 방식이 검토되고 있다. 양자 컴퓨터는 물질이 가진 파의 성질을 이용한 컴퓨터이며, 기본적인 알고리즘이나 기본회로의 제안 등이 나오고 있다. 현 상태에서 실용성은 전혀 미지수이지만 성과를 기대한다. 컴퓨터 아키텍트에는 컴퓨터 공학 관련 분야의 넓은 지식뿐만 아니라, 세계의 정치 경제 구조나 사회생활양식의 변화에 민감한 후각을 가진, 자신을 포함한 유저의 요구를 파악해서 질적인 기술비약을 가능하게 하는 독창적인 발상력을 지닌 것들이 강력하게 요구된다.