목 차

목 차 1

1. INTRODUCTION AND ACCOUNT 4

2. ARCHITECTURE OF HIGH-PERFORMANCE COMPUTERS 7

2.1. THE MAIN ARCHITECTURAL CLASSES 7
2.2. SHARED-MEMORY SIMD MACHINES 9
2.3. DISTRIBUTED-MEMORY SIMD MACHINES 12
2.4. SHARED-MEMORY MIMD MACHINES 14
2.5. DISTRIBUTED-MEMORY MIMD MACHINES 16
2.6. CCNUMA MACHINES 19
2.7. CLUSTERS 20
2.8. PROCESSORS 22
2.8.1. AMD Opteron 23
2.8.2. HP/Compaq Alpha EV7 24
2.8.3. Hewlett-Packard PA-RISC 8800 26
2.8.4. IBM POWER4+ 29
2.8.5. IBM POWER5 32
2.8.6. IBM BlueGene Processor 33
2.8.7. Intel Itanium 2 34
2.8.8. Intel Pentium 4/Xeon Nocona 36
2.8.9. MIPS R16000 38
2.8.10. Sun UltraSPARC IV 39
2.9. NETWORKS 42
2.9.1. Infiniband 43
2.9.2. Myrinet 44
2.9.3. QsNet 45
2.9.4. SCI 46

3. RECOUNT OF THE (ALMOST) AVAILABLE SYSTEMS 49

3.1. SYSTEM DESCRIPTIONS 49
3.1.1. The Bull NovaScale 49
3.1.2. C-DAC PARAM Padma 50
3.1.3. The Cray MTA-2 51
3.1.4. The Cray SX-6 53

3.1.5. The Cray Inc. X1. 53
3.1.6. The Cray Inc. XD1 55
3.1.7. Fujitsu/Siemens PRIMEPOWER 57
3.1.8. The Hitachi SR11000 58
3.1.9. The HP 9000 Superdome 60
3.1.10. The HP Integrity Superdome 61
3.1.11. The HP/Compaq GS series 63
3.1.12. The Compaq AlphaServer SC45 Series 64
3.1.13. IBM eServer p690 66
3.1.14. The IBM BlueGene/L 67
3.1.15. The NEC SX-6 69
3.1.16. The NEC TX-7 series 70
3.1.17. The Quadrics Apemille 72
3.1.18. The SGI Altix 3000 series 73
3.1.19. The SGI Origin3900 75
3.1.20. The Sun Fire E25K 76
3.1.21. The Unisys Server ES7000 78

4. SYSTEMS DISAPPEARED FROM THE LIST 79

4.1. DISAPPEARED MACHINES 80

5. SYSTEMS UNDER DEVELOPMENT 93

5.1. CRAY INC. 93
5.2. HEWLETT-PACKARD/INTEL 94
5.3. IBM 94
5.4. SGI 95
5.5. SRC 95

6. GLOSSARY OF TERMS 96

참고자료 102

찾아보기 105

이 문서는 최신 슈퍼컴퓨터에 대한 이해를 돕고자 Top500 Supercomputer sites의 Overview of recent supercomputers(2004) 자료를 번역한 것이다. 원 저자는 네덜란드 Utrecht 대학의 Aad J. van der Steen과 미국 University of Tennessee의 Jack J. Dongarra이다.

Abstract

본 문서는 2004년 현재 이용되고 있거나 가까운 시일 내에 이용 가능한 고성능 컴퓨터들에 대한 개략적 자료이다. 2004년 현재 시점에서 아직 연구 단계에 있는 시스템들에 대해서는 고려하지 않았다. 시스템들은 SIMD와 MIMD에 대해 각각 공유 메모리, 분산 메모리 시스템 아키텍처로 나눠 아키텍처 클래스로 구분해 기술하였다. 각 시스템에 대한 기술은 가능한 간단하게 했으며, 간혹 시스템 성능보다 더 알기 어려운 가격정보는 제외했다. 본 자료에서는 각 시스템 정보에 대한 보다 나은 평가를 위해 고성능 시스템 아키텍처에 대한 일반적인 정보들도 다루고 있으며, 아울러 시스템에서 사용하고 있는 다양한 프로세서에 대해서도 기술하였다.

1. Introduction and account

이 문서는 현재 업무용으로 사용되고 있거나 수 개월 내에 사용될 예정인 병렬 시스템과 벡터 시스템에 대한 개략서로 14번째 버전이다. 병렬과 벡터 시스템 시장은 매우 빠르게 변화하여 어떤 시스템이 금방 나타났다가 금방 사라지곤 하는 것이 다반사이다. 때문에 이 자료상의 정보들은 근사적으로 참조되는 것이 좋다. 그렇지만, 고성능 처리를 위해 시도되는 여러 가지 수단들에 대한 일반적인 지식을 얻고자 한다면 이 자료의 내용이 많은 도움이 될 것이다.

본 자료에서는 가급적 최신의 정보를 간략하게 나타내고자 했다. 지금 이순간에도 새로운 시스템들이 나타나는 것에 맞춰 기존의 시스템들은 시장에서 사라지고 있다. 이것은 통합된 시스템 스펙트럼에서 새로운 시스템이 나타나면서 일부가 빠져 나가는 식으로 이해할 수 있다. 최근에는 대형 클러스터의 사용이 매우 빠르게 증가하고 있다. 거시적인 아키텍처 관점과 프로세서 관점 모두에 있어서 컴퓨터 아키텍처에 대한 관심이 새롭게 유발됨에 따라 향후 수년 동안 많은 새로운 시스템들이 나타날 것으로 기대되고 있다. 아키텍처에 대한 새로운 관심이 증폭된 것은 일본의 Earth Simulator 시스템이 소개 되면서이다. 이 시스템은 처음 발표된 이후 아직 세계에서 가장 빠른 시스템의 자리를 차지하고 있다[42]. ASCI[3] 프로젝트로 인해 나타난 시스템들의 이론 최고 성능과 실제 프로그램 성능 사이의 차이가 꾸준히 증가 하면서 해당 시스템 사용자들이 더 이상 참을 수 없는 수준에 이르자 미국 내에서는 새로운 시스템 아키텍처를 찾자는 의견이 나타났다. DARPA-funded HPCS 프로그램과 같은 프로그램들이 메모리 접근과 노드간 네트워크 대역폭에서의 성능향상 없이 동일한 SMP-clustered RISC 프로세서 기술에 머무는 것으로는 이뤄질 수 없는 이런 동향을 억제해야 한다.

여전히 대부분의 시스템들은 고속의 네트워크로 연결된 RISC 기반 SMP 클러스터라는 동일한 틀에서 약간의 변화를 준 것으로 보인다. Culler 등[8]은 이것을 자연스런 아키텍처의 진화라고 하지만, ASCI 프로그램에서 공식화된 요구가 시스템들을 이러한 방향으로 몰고 간다는 것과 이것이 위에서 설명한 이유로 인해 수년 내에 변화하게 될 것이라는 것은 논의할 만한 일이다.

첫 등장이래 지난 수년간 리스트에서 놀라운 비율을 차지하고 있는 Beowulf 클러스터들을 보듯이 슈퍼컴퓨터 시장은 매우 동적이다. 클러스터를 제작해 판매하는 업체들이 매우 많으며 또한 업체와 시스템이 나타나고 사라지는 속도가 지나치게 빨라 각 클러스터에 대한 성능을 여기서 다루는 것은 어렵다. 여기서는 클러스터의 특성과 그들이 다른 슈퍼컴퓨터에 대해 상대적으로 어떤 위치를 차지하고 있는가 정도를 간략하게 다룰 것이다. 그러나, tightly-coupled 또는 통합 병렬 시스템에 대해서는 인기 있고 현재 나오고 있는 아키텍처들의 주된 경향을 따라 이 문서에서 다룰 수 있을 것이다. 그리고, 시스템의 핵심이 되는 프로세서와 네트워크에 대해서는 따로 다룰 것이다.

이곳에서 다루게 될 시스템의 선정은 다음과 같은 규칙을 따른다. 이 자료를 작성하는 현재,

또는 6개월 이내에 상업적으로 가용한 시스템을 다룬다. 이 규칙으로 Sandia, Los Alamos, Lawrence Livermore National Laboratory 등에 있는 ASCI 시스템들(실제 성능이 6 ~ 19 Tflop/s)과 약 40 Tflop/s 정도의 성능을 보이는 일본의 Earth Simulator 시스템 등과 같이 관심을 끄는 연구 시스템들이 제외된다.

마케팅 목적으로 지나치게 일찍 발표되어 적당한 기간 내에 사용자들에게 나타날 수 없는 시스템들을 제외하기 위해 적어도 6개월 내에 가용될 시스템을 기준으로 한다. 아직 사용되고 있는 오래된 시스템들은 가급적 내용에 포함시키지 않았다. 그래서 IBM SP1, Cray T90 등의 시스템들은 아직 사용되고 있지만 제외되었다. 엄밀히 말하자면, 비교적 최근에 시장에 나온 시스템 그리고 6개월 내에 시장에 나올 시스템들을 기준으로 작성하였다. 본문에서는 해당 시스템에 대한 정보를 더하기 위해 각 업체들의 웹 주소를 인용해 두었다.

SGI의 Origin3000 시리즈와 Fujitsu의 PRIMEPOWER와 같이 일부 업체에서 제공하는 시스템들은 노드의 클럭 수를 제외하고 모든 것이 동일한 경우가 있다. 이런 경우 대부분에게 가장 많은 관심을 받게 되는 제일 빠른 클럭 수를 가진 모델에 대해 다루었다.

아키텍처 별로 시스템을 구분해 나열하는 것이 일상적이지만 이런 방식이 점점 더 인위적이 되어 가는 문제로 여기에서는 모든 시스템을 단순히 철자 순으로 나열하였다. 각 시스템에 대한 해설 첫 머리에 시스템 타입을 언급해 둔다. 대부분 곧 무의미해지는 가격 정보는 생략한다. 가능하다면 이론 최고 성능 대신에 사용자의 경험에 기반한 성능 정보를 제공한다. 이때 가능하다면 측정 성능치의 최고 값을 인용해서 이론적인 최고 성능보다 더 현실적인 성능 상한을 제공하도록 한다. 그렇지만 슈퍼컴퓨터 속도의 범위는 매우 광범위해서 최고로 측정된 시스템 성능이 사용자가 실행할 프로그램의 실제 성능을 반영하지는 않을 것이다. 실제로, HPC Linpack 테스트를 통해 측정된 성능과 비교해 대부분 평균적인 사용자들이 체험하는 성능은 훨씬 더 느릴 것이다.

사용자들에게 새로운 시스템들의 외양이 점점 더 비슷해져 가고 있지만 이러한 외양에 근거한 아키텍처 클래스를 되짚어 보는 것은 유용한 일이다. 이를 통해 고성능을 얻게 되는 다양한 방식과 왜 시스템들이 그처럼 작동되는 지에 대해 감을 잡을 수 있게 된다. 관련하여 보다 자세한 내용은 아키텍처 섹션에서 다룰 것이다.

이미 시장에서 사라져 버린 시스템들에 대한 정보가 다수의 독자들에게 여전히 관심거리이고, 지난 시간 동안의 슈퍼컴퓨터 역사에 대한 정보를 줄 수 있기 때문에 해당 시스템들을 Systems disappeared from the list에서 다루었다. Systems under development에서는 개발중인 시스템들을 다루었고 많은 전문 용어를 포함하는 프로세서 섹션이 있기에 용어 설명을 위해 glossary 섹션을 따로 구성해 두었다.

이 자료에서는 시스템의 계산 능력에 초점을 맞추고 있다. 여기에다 현재의 고성능 컴퓨터가 가지고 있는 모든 장점들을 충분히 판단하기 위해서는 해당 시스템의 I/O 성능과 연결 능력 또한 논의되어야 한다. 그러나 어떤 시스템의 한 모델에 대해서 조차 가능한 설정 조합이 지나치게 많아 이 자료에서 그 모든 것을 다루기 어려운 면이 있다. 그래서 시스템의 모든 특성에 대해 이곳에서 다루지는 못할 것이다. 그리고 이곳에서는 데이터 베이스 관리

나 실시간 컴퓨팅, 가시화 등과 같이 특정 분야에 국한되어 사용되는 고성능 시스템들은 제외하였다. 일반적인 과학 기술 계산 분야에서 사용되는 고성능 시스템에 대한 개요를 제공하고자 하므로, 주로 데이터베이스 검색을 위한 수단으로 사용되는 AT&T GIS 시스템이나 Concurrent Computing Systems 같이 배타적으로 실시간 사용자 커뮤니티에 집중하는 시스템 등은 제외시켰다. 그리고 실제 성능은 전체적으로 다른 것들에 영향 받을 수 있을지라도 단일 CPU 자체의 성능이 2 Gflop/s를 넘어서는 일이 이제는 흔한 일이 되었기에, 이곳에서는 이론최고성능이 적어도 80 Gflop/s의 이상이 되는 시스템들에 대해서만 다루었다.

2. Architecture of high-performance computers

2.1. The Main Architectural Classes

시스템 자체에 대한 설명 이전에 시스템의 성능 향상을 위해 사용된 메커니즘을 먼저 생각해보는 것도 중요할 것이다. 하드웨어 구조 또는 아키텍처는 CPU 하나의 성능, 그 이상을 얻기 위해 어떤 것이 가능하고 불가능한 지를 넓은 범위에서 결정한다. 하드웨어와 조화되는 다른 중요한 요소는 해당 하드웨어 플랫폼에 적합하도록 효율적인 실행코드를 만들어 낼 수 있는 컴파일러의 능력이다. 많은 경우에 있어 하드웨어와 소프트웨어의 영향을 명확히 구별하는 것은 어려운 일이다. 어떤 영향에 대한 결과를 해석하는 데 있어 그것이 하드웨어 특성에 의한 것인지, 소프트웨어에 의한 것인지 아니면 둘 모두에 의한 것인지 그 구분을 조심스럽게 할 필요가 있다. 이 곳에서는 특히 하드웨어 아키텍처에 대해 집중적으로 알아본다.

여러 해 동안 Flynn[13]의 구분법이 고성능 컴퓨터를 구분하는 데 유용하게 이용되어 왔다. Flynn의 구분은 명령어와 데이터 흐름을 처리하는 방법을 기반으로 네 개의 주요 아키텍처 클래스를 구별한 것이다. 우선 이곳에서는 네 개의 클래스에 대해 간단히 기술하고 향후 각 클래스 별로 따로 알아볼 때 좀 더 자세한 내용을 다루도록 한다.

SISD machines: 하나의 CPU를 가지고 순차적으로 실행되는 하나의 명령어 흐름을 조절할 수 있는 전통적인 시스템이다. 현재는 많은 대형 메인 프레임들이 하나 이상의 CPU를 가지고 있으나, 각각은 서로 연관되지 않은 명령어 흐름을 실행시킨다. 따라서 그와 같은 시스템들은 아직도 다른 데이터 공간에서 동작하는 SISD 시스템으로 간주한다. 예로 들자면 DEC, HP, IBM, SGI 등의 대부분 워크스테이션들이 모두 SISD 시스템이다. 시스템 구분 측면에서 SISD 시스템의 정의를 이곳에서 하고 있지만, 실제 본문에서 이와 같은 형태의 시스템을 다루지는 않을 것이다.

SIMD machines: 대부분 1,024 ~ 16,384개에 이르는 많은 수의 연산처리 유닛을 가지는 시스템으로 이 유닛들은 서로 다른 데이터에 대해 동일한 연산을 동시에 수행한다(lock-step). 즉 하나의 명령어가 많은 데이터를 병렬로 처리하는 것이다. 이런 시스템의 예로 CPP Gamma II와 Quadrics Apemille 등이 있으나 약 1년 전부터 더 이상 시장에 나오지 않고 있다.

SIMD 시스템의 다른 서브클래스로 벡터 프로세서 시스템이 있다. 벡터 프로세서는 특별하게 구조화된 CPU를 이용해, 단일 데이터보다는 유사한 데이터의 배열에 대해 동작한다. 데이터가 이러한 벡터 유닛에 의해 처리될 수 있을 때 처리결과는 클럭 사이클당 하나에서 두 개 또는 특별한 경우 세 개의 비율로 전송 가능하다. 그래서 벡터 프로세서는 벡터모드로 동작할 때 해당 데이터에 대해 거의 병렬로 실행된다. 이 경우, 전통적인 스칼라 모드로 동

작하는 것과 비교해 몇 배씩 빠른 성능을 보여준다. 벡터 프로세서 시스템은 거의 대부분 SIMD 시스템으로 간주된다. 벡터 시스템의 예로 NEC SX-6i가 있다.

MISD machines: 이론적으로 이러한 형태의 시스템은 여러 명령어가 하나의 데이터 흐름에 작용한다. 이와 같은 방식으로 실용화된 시스템은 아직 없다.

MIMD machines: 이러한 시스템에서는 여러 명령어 흐름이 병렬로 서로 다른 데이터에 작용한다. 앞서 언급한 다중 프로세서를 가지는 SISD 시스템과 다른 점은 명령어와 데이터가 실행되어야 하는 동일한 작업의 다른 부분을 나타내므로 서로 연관되어 있다는 것이다. MIMD 시스템은 주 작업이 실행되는 시간을 줄이기 위해 많은 부분 작업을 병렬로 실행하는 것이 가능하다. 매우 다양한 MIMD 시스템이 있으며 그런 이유로, 특별히 이 클래스에서만큼은 Flynn의 구분 방식이 시스템을 완전하게 구분하는 데 부족한 것으로 보여진다. 4개의 프로세서를 가지는 NEC SX-6와 1000개의 프로세서를 가지는 IBM p690은 매우 다르게 동작하는 시스템이지만 두 시스템 모두 MIMD 클래스에 포함된다.

Shared memory systems: 공유 메모리 시스템은 동일한 주소공간을 공유하는 다수의 CPU를 가지는 시스템이다. 이것은 모든 CPU가 동일한 조건하에 하나의 메모리에 접근 가능하므로 데이터의 저장 위치 정보에 대해 사용자가 관심을 가질 필요가 없다는 것을 의미한다. 공유 메모리 시스템은 SIMD 또는 MIMD일 수 있다. 단일 CPU를 가지는 벡터 프로세서 시스템은 SIMD(SM-SIMD)이고 다중 CPU 모델은 MIMD(SM-MIMD)의 예가 된다.

MIMD memory systems: 각 CPU는 자기 고유의 메모리 공간을 가진다. CPU들은 서로 네트워크로 연결되고 필요할 때 각자의 메모리에 있는 데이터를 교환할 수 있다. 공유 메모리 시스템과 비교해 사용자는 어느 지역 메모리에 필요한 데이터가 있는지 알고 있어야 하고 필요할 때 데이터 이동과 분산을 명시적으로 수행해야 한다. 분산 메모리 시스템은 SIMD 또는 MIMD중 하나의 형태가 된다. 일렬로 동시 동작(lock-step)하는 SIMD 시스템은 모두 각 프로세서에 연관된 분산 메모리를 가진다. 이제 곧 보게 되겠지만, 분산 메모리 MIMD 시스템은 매우 다양한 형태의 네트워크 연결 토폴로지를 보여준다. 이 토폴로지의 상세한 부분은 사용자에게 많은 부분 감춰져 있어 응용 프로그램의 이식성 측면에 도움을 준다. 공유 메모리 시스템과 마찬가지로 분산 메모리 시스템도 DM-SIMD와 DM-MIMD 시스템의 두 부분으로 분류될 수 있다.

이미 언급한대로, 공유 메모리 시스템과 분산 메모리 시스템의 차이는 분명해 보이지만 사용자 관점에서 봤을 때 항상 그런 것은 아니다. 예를 들자면, 이전의 Kendall Square Research 시스템은 하드웨어 수준에서 '가상 공유 메모리' 개념을 사용했다. 이런 가상 공유 메모리는 프로그래밍 수준에서도 구현될 수 있다. 1993년[19]에 발표된 High Performance Fortran(HPF)은 컴파일러 지시어를 이용해 데이터를 가용 프로세서들에게 분

배할 수 있다. 따라서 HPF를 사용하면 해당 시스템은 사용자에게 공유 메모리 시스템으로 보여지게 된다. HP와 SGI 같은 기타 다른 MPP 시스템 업체들에서는 물리적인 분산 메모리 시스템을 전체적으로 통합된 주소 공간으로 어드레스할 수 있는 독자적인 가상 공유 메모리 모델을 지원한다. 그래서 그런 시스템들은 전체 메모리를 하나의 글로벌 주소 공간으로 대응시킨다. ccNUMA 섹션에서 이런 시스템들의 구조에 대해 좀 더 자세히 알아볼 것이다. TreadMarks[1] 같은 패키지들은 NOW(networks of workstations)와 같은 클러스터에 대한 분산 공유 메모리 환경을 제공하고 있다. 이와 관련된 시스템들에 대한 개요를 자료[11]에서 참조할 수 있을 것이다.

분산처리는 DM-MIMD의 개념을 한 걸음 더 나아가게 했다. 하나 또는 몇 개의 시스템에 통합된 많은 프로세서를 두는 대신 워크스테이션, 메인 프레임 등이 (Gigabit) Ethernet, FDDI 등에 의해 연결되어, 같은 프로그램의 작업들을 동시에 수행하도록 설정되었다. 이것이 개념적으로는 DM-MIMD 컴퓨팅과 별로 다를 것이 없지만 프로세서들 사이의 통신이 많이 느려질 수 있다. 분산 컴퓨팅을 실현한 많은 패키지들을 이용할 수 있는데, 이런 패키지의 예로는 PVM[14]과 MPI[22,23] 같은 것들이 있다. 메시지 패싱 모델 프로그래밍 스타일을 많이 사용하게 되면서 PVM과 MPI는 거의 모든 분산 메모리 MIMD 시스템 업체에서 채택되었고, 호환성 문제로 공유 메모리 MIMD 시스템에서도 채택되어 사용되었다. 최고의 계산 파워를 얻기 위해 HiPPI 채널을 이용해 공유 메모리 시스템을 클러스터로 구성하는 경향이 있는데, NEC의 SX-6, Cray의 SV1ex 등이 이와 같은 구조를 가진다. 그래서 클러스터의 노드 내에서는 공유 메모리 프로그래밍 스타일이 이용되고 클러스터 사이에는 메시지 패싱이 이용되어야 한다. 현재 PVM은 그다지 많이 사용되지 않고 있으며, MPI가 사실상의 표준으로 많이 사용되고 있음을 밝혀둔다.

SM-MIMD 시스템 환경에서는 코드 내에 지시어(Fortran)나 pragma(C/C++)를 넣어 Fortran이나 C/C++ 프로그램을 병렬화할 수 있는 OpenMP가 있다. OpenMP는 공유 메모리 시스템을 위해 잘 만들어진 표준이 되고 있으며, 빠른 속도로 주요 업체들에 의해 채택되어 사용되고 있다.

2.2. Shared-memory SIMD machines

이 분류에 속하는 시스템들은 실질적으로 단일 프로세서 벡터 시스템이다(그러나, VLIW 시스템[31] 같은 다른 시스템들도 있다.). 아래 그림은 벡터 아키텍처의 일반적인 모델이다.

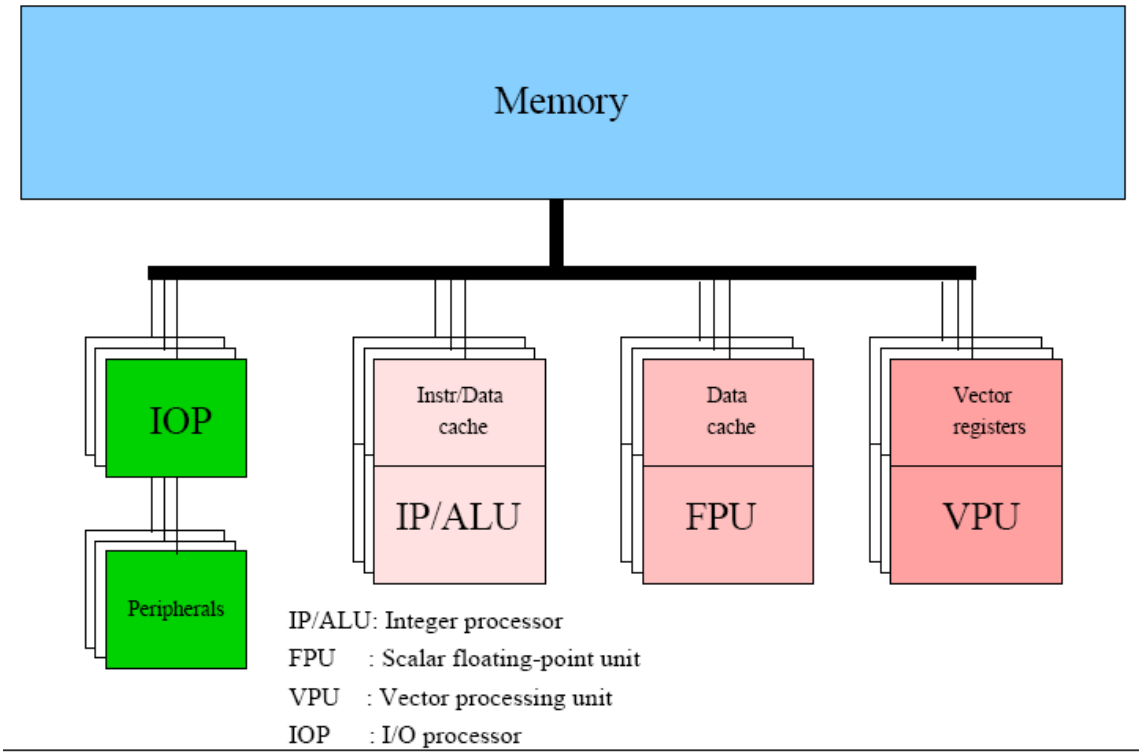


그림 1: Block diagram of a vector processor

단일 프로세서 벡터 시스템은 위의 그림과 같은 벡터 프로세서를 하나 가지며, 벡터 프로세서를 통해서 처리 가능한 스칼라 부동소수 처리기능을 따로 가질 수도 있다. VPU에는 캐시가 없다는 것을 주의해 보자. 벡터 시스템 대부분은 캐시를 가지고 있지 않다. 많은 경우에 있어 벡터 유닛은 캐시의 효과를 볼 수 없고 오히려 잦은 캐시 오버플로우로 인해 실행 속도에 나쁜 영향을 끼칠 수 있기 때문이다.

메모리에서 직접 데이터를 읽어 들이고, 계산 즉시 그 결과를 메모리로 저장할 수 있는 벡터 시스템들이 있어 왔지만(CDC Cyber 205, ETA-10), 현대의 모든 벡터 시스템들은 벡터 레지스터를 사용하고 있다. 통상적으로 벡터 레지스터를 사용하는 것이 연산 속도에 나쁜 영향을 주지 않으면서 연산수를 취합하고 중간 결과를 처리하는 데 있어 더 많은 유용성을 제공한다.

위의 그림에서는 VPU와 메모리 사이의 세부적인 연결방식이 나타나 있지 않지만, 효과적인 벡터 연산 속도를 위해 연결방식은 매우 중요하다. 만약 메모리와 VPU사이의 대역폭이 지나치게 작다면 VPU가 연산하기 전에 데이터를 기다려야 하고 결과를 저장하기 전에 기다려야 하므로 VPU의 장점을 완전하게 살릴 수 없게 될 것이다. 이와 같은 상황을 보상할 수 있을 만큼 load/store에 대한 연산의 비가 충분히 크지 않다면 심각한 성능 저하를 가져올 수 있다.

다음 그림은 이항 벡터 연산 c = a + b (a, b, c,는 벡터)에서 load/store 경로 수의 영향을 나타내고 있다.

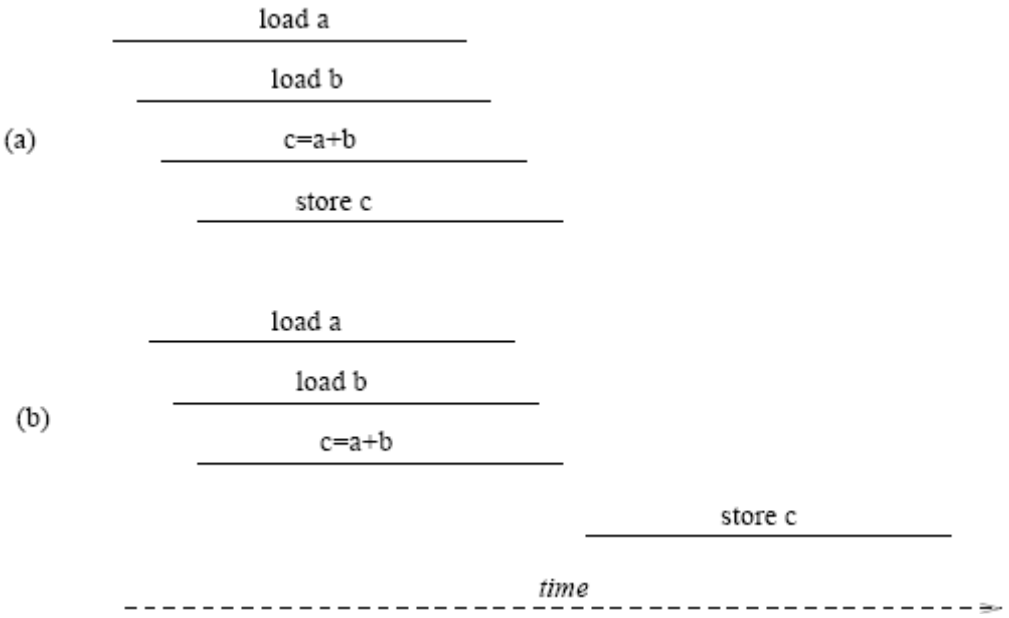


그림 2: Schematic diagram of a vector addition. Case (a) when two load- and one store pipe are available; case (b) when two load/store pipes are available.

메모리와 VPU 사이에 위와 같은 데이터 경로를 구현하는 비용 문제로 인해 종종 다른 방법이 모색되고 있고, 전체 요구 대역폭 (full required bandwidth: 동시에 2개의 load, 1개의 store 연산)을 가지는 시스템 수는 제한적이다. 사실상, 현재 시장에 나오는 벡터 시스템에서 이러한 높은 대역폭은 더 이상 구현되지 않는다. 오히려, 업체들은 대역폭 손실을 감추기 위해 추가적인 캐시 등과 같은 다른 방법을 사용한다.

그림 1에서 VPU는 하나의 블록으로 표현되어 있지만, VPU의 구조는 매우 다양하다. 모든 VPU는 그 내부에서 하나 또는 그 이상의 기능을 수행할 수 있는 다수의 벡터 Functional 유닛(또는 파이프)으로 구성된다. 모든 VPU는 메모리 접근 기능을 수행하도록 디자인된 파이프를 가지고, 이를 통해 연산수를 연산 파이프로 적시에 전달하고 연산 결과를 다시 메모리로 저장하도록 한다. VPU는 정수/논리 연산, 부동소수 덧셈, 곱셈 연산과 덧셈과 곱셈이 조합된 연산(compound operation) 등과 같은 다수의 연산 수행 유닛을 가진다. 나눗셈은 반복되는 프로시저나 table look-up, 또는 덧셈과 곱셈 파이프를 이용한 양쪽의 조합을 통해 수행된다. 거기에 연산 벡터의 일부 원소에만 연산이 적용되도록 하는 마스크 파이프가 항상 있다. 마지막으로 이런 벡터 파이프 집합은 하나의 VPU내에 복제(2 ~ 16회 복제 가능)될 수 있다. 이상적으로는 메모리 대역폭이 충분하다면 복제된 수만큼 VPU당 성능이 증가할 것이다.

2.3. Distributed-memory SIMD machines

프로세서 배열(Processor-array) 시스템으로도 알려져 있다[17]. 이것은 이 시스템들의 모든 프로세서가 동시에 똑 같은 명령어를 (다른 데이터에) 적용하기 때문이다. 이때 따로 동기화 과정이 필요하지 않은데 이로 인해 시스템 설계가 매우 단순해진다. 제어 프로세서가 프로세서 배열에 있는 프로세서들에서 실행될 명령어를 보내게 된다. 현재 사용되고 있는 모든 DM-SIMD 시스템들은 데이터 경로를 통해 제어 프로세서에 연결되는 프론트-엔드 프로세서를 이용한다. 프로세서 배열이나 제어 프로세서에 의해 실행될 수 없는 연산들은 프론트-엔드 시스템으로 내려진다. 예를 들어 I/O는 프론트-엔드 시스템을 통할 수 있다. 아래 그림은 DM-SIMD 시스템의 일반적인 모델을 보여주고 있으며, 실제 모델들은 이와 약간씩 차이가 있을 수 있다.

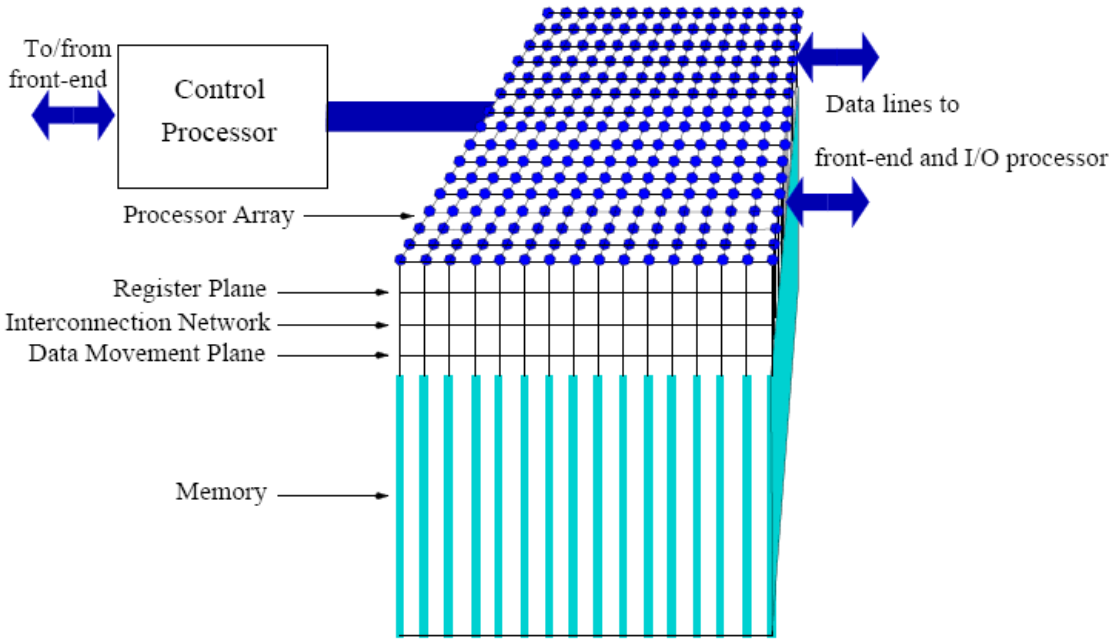


그림 3: A generic block diagram of a distributed memory SIMD machine.

위 그림에서 모든 프로세서들은 2-D 그리드로 연결되도록 나타나고 있는데, 실제로 이와 같은 형태의 시스템에서는 항상 2-D 그리드를 내부 연결 토폴로지로 포함하고 있다. 각 그리드 라인의 반대편 끝은 마치 토러스(torus)처럼 다시 토폴로지로 연결된다. 일부 시스템들은 이와 같은 연결 이외에 항상 3-D로 연결되기도 하고, 대각선으로 연결되거나 좀더 복잡한 구조로 연결되는 경우도 있다.

프로세서 배열에서 일부 프로세서가 어떤 논리적인 조건하에 특정 명령어 실행에서 제외되는 것이 가능하다. 그러나 명령어가 실행되는 동안 그 프로세서들은 대기 상태에 놓여지므로 이로 인한 성능 손실이 발생하게 된다. 성능 손실을 가져오는 또 다른 원인은 프로세서

i에서 필요한 데이터가 프로세서 j의 메모리에 있는 경우이다(실제로, 이와 같은 상황이 모든 프로세서에 동시에 발생하기 때문에 이것은 데이터가 전체 프로세서에 걸쳐 순환되어야 함을 의미한다.). 프로세서 j에 있는 데이터에 접근하는 것은, 해당 데이터가 프로세서에 의해 꺼내져서 네트워크를 경유하여 프로세서 i로 보내어지는 것을 의미한다. 이로 인해 시간이 많이 소비된다. 이런 두 가지 이유로 인해 DM-SIMD 시스템은 사용자가 시스템의 전체 병렬성을 모두 사용하고자 하는 특별한 경우에 주로 사용된다. 일반적으로는 디지털 신호와 영상 처리, 프로세서 사이에 데이터 교환이 불필요한 Monte Carlo Simulation에 적절히 사용될 수 있으며, 시스템에서 처리하기 적합한 크기를 가지는 다량의 데이터 집합에 똑 같은 연산을 적용해야 하는 경우에도 DM-SIMD 시스템이 적합할 것이다.

어느 정도 지능적이라 할 수 있는 제어 프로세서는 프로세서 배열에서 실행될 명령어 시퀀스를 꺼낸다. 이때 발생되는 최악의 경우는(자율성이 떨어지는 제어 프로세서에서 이런 경우가 발생할 확률이 높다) 프로세서 배열에서 실행되기에 적합하지 못한 명령어를 실행하고자 하는 경우이다. 예를 들자면, 단순한 프린트 명령어 같은 경우를 들 수 있는데, 이런 명령어가 제어 프로세서에서 실행되는 것보다 훨씬 느린 프론트-엔드 프로세서로 내려가게 될 수 있다. 보다 자율적인 제어 프로세서는 이런 상황의 발생을 미리 피해서 프론트-엔드 프로세서와 제어 프로세서 사이의 인터럽트 발생을 줄여준다. 대부분 DM-SIMD 시스템들은 I/O를 프론트-엔드 프로세서와 독립적으로 처리할 수 있다. 이로 인해 좋아지는 것이 프론트-엔드와 백-엔드 시스템 사이의 통신이 줄어드는 것뿐만이 아니다. 프로세서 배열 시스템의 특별한 I/O 장치는 일반적으로 프로세서 배열의 메모리에 필요한 데이터를 직접적으로 제공하기에 훨씬 효율적이다. 레이더와 영상 처리같이 데이터 처리가 많은 프로그램에서 이런 I/O 시스템은 대단히 중요하다.

이 시스템 고유의 특징은 때때로 프로세서들이 대단히 단순한 bit-serial 타입이라는 것이다. 이것은 프로세서들이 데이터 아이템에 대하여 타입과 상관없이 bitwise 연산을 한다는 것으로 가령 소프트웨어 루틴에 의해 수행되는 정수에 대한 연산은 적어도 연산수의 길이만큼 사이클 수가 소요되는 이러한 단순한 bit-serial 프로세서를 통해 처리된다. 그래서 32비트 정수는 64비트 정수보다 두 배 빠르게 처리된다. 부동소수 연산에서도 비슷한 이야기를 할 수 있는데 다만 연산에 소비되는 사이클 수가 정수연산에서 필요한 것보다 몇 배 더 필요해진다. 이런 형태의 시스템들은 대부분 프로세서 개수가 많아서 1,024개 또는 그 이상의 프로세서를 가진다(Quadrics Appemille는 예외). 때문에 부동소수에 대한 느린 연산 속도는 프로세서 개수에 의해 보상될 수 있는데, 프로세서 당 비용은 부동소수 전용 프로세서와 비교해 매우 낮다. 그렇지만, 경우에 따라 부동소수 보조처리기(coprocessor)가 프로세서 배열에 추가될 수도 있다. 이때 보조처리기의 수는 비용 때문에 bit-serial 프로세서의 수의 8분의 1에서 16분의 1정도가 된다. bit-serial 프로세서의 장점은 어떤 길이의 연산수에 대해서도 연산을 수행할 수 있다는 것이다. 이것은 random number 발생과 신호처리에 특히 유리한데, 그 이유는 두 가지 경우 모두 1 ~ 8비트의 연산수가 많기 때문이다. bit-serial 시스템의 실행속도는 연산수의 길이에 비례하므로 1 ~ 8비트 연산수가 많은 경우 현저한 성

능 향상을 얻을 수 있다.

2.4. Shared-memory MIMD machines

앞서 그림 1에서 이 분류에 속하는 형태의 시스템을 하나 소개했었다. 실제로, 앞에서 논의된 단일 프로세서 벡터 시스템은 좀더 일반적인 형태에 속하는 한가지 특별한 예라고 할 수 있다. 그 그림에서는 하나의 시스템에 하나 이상의 FPU와 VPU가 가능함을 보여주고 있다. 공유 메모리 시스템에서 직면한 주된 문제점 한 가지는 CPU와 CPU 그리고 CPU와 메모리 사이의 연결이다. 이상적으로는 더 많은 CPU가 추가된다면 전체 메모리 대역폭은 프로세서 개수에 선형적으로 증가해야 한다. 반면에 각 프로세서는 중간 단계에서 메모리를 사용함으로써 성능이 저하되는 방식을 사용하지 않고 직접적으로 다른 모든 프로세서와 통신하는 것이 좋다. 문제는 이와 같은 전체 연결 비용이 매우 높다는 것이다. 프로세서 개수가 O(n)으로 증가할 때 그 전체 연결 비용은 $O(n^2)$로 증가한다. 그래서 여러 가지 대체 방안이 연구되었고 지금도 연구되고 있으며, 그들 중 일부가 아래 그림에 있다.

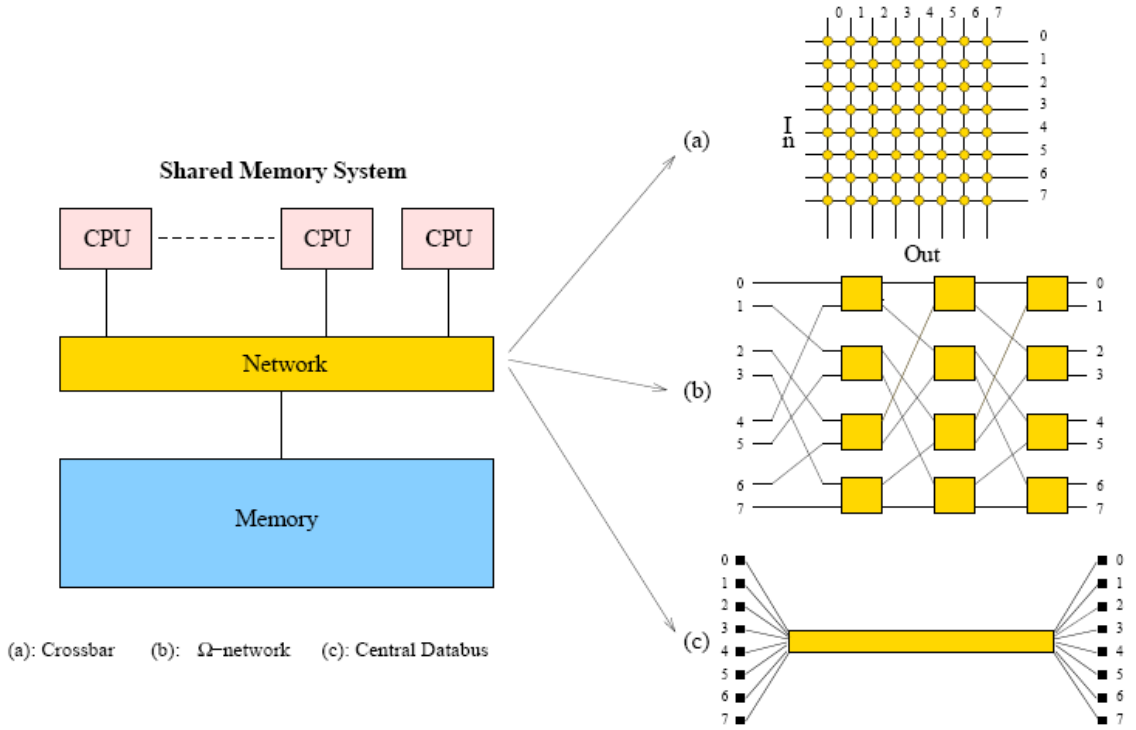


그림 4: Some examples of interconnection structures used in shared-memory MIMD systems.

위의 그림 4에서 크로스바를 통한 n^2 연결, Ω-network를 통한 $n\log_2 n$ 연결, 그리고 중앙

버스(central bus)를 통한 하나의 연결 등을 볼 수 있다. 이러한 연결 구조들은 다른 형태의 연결 구조와 비교해 각 연결 경로의 사용에 차이가 있는데 크로스바의 경우 각 데이터 경로(data-path)가 직접적이고 다른 요소들과 공유되지 않는다. Ω-network의 경우 $\log_2 n$ 스위칭 단계가 있고 많은 데이터들이 경로확보를 위해 경쟁해야 하는 경우가 발생할 수 있다. 중앙 데이터 버스는 모든 데이터가 동일한 버스를 공유해서 n개 데이터가 항상 경쟁해야 한다.

버스 연결은 그 비용이 최소가 되지만, 버스 경쟁이 발생해 전체적인 성능 저하가 발생한다는 분명한 결점을 가진다. 버스 소통량(traffic)을 최소화하기 위해 CPU와 연관된 캐시를 사용하는 여러 가지 복잡한 전략이 고안되어 왔다. 그러나 이로 인해 버스 구조가 복잡해졌고 따라서 비용 역시 증가되었다. 프로세서의 속도가 매우 빠르게 증가해 왔고 이로 인해 만족할 만한 성능을 얻을 수 있는 프로세서의 최대 연결 개수를 버스가 제한하는 사정에 비추어 사실상 충분히 빠른 속도를 얻는 버스를 설계하기란 매우 어려운 일로 알려져 있다. 버스로 인해 실제 최대 프로세서 연결 개수는 10-20개를 넘지 않는다. 1992년에 내부 시스템 요소 또는 외부 시스템을 연결하는 고속 버스에 대한 새로운 표준(IEEE P896)이 제정되었다. 이것이 SCI(Scalable Coherent Interface)인데 200 ~ 1,000 Mbyte/s 정도의 점대점 대역폭을 제공한다. 실제로 SCI는 HP Exemplar 시스템과 SCALI에서 제공하는 클러스터에서 사용되고 있다. SCI는 단순한 버스 그 이상이며 분산 컴퓨팅을 위한 하드웨어 네트워크 프레임워크로 사용될 수 있다[20].

다단(multi-stage) 크로스바는 로그함수적인 복잡성을 가지는 네트워크로 잠재적인 용량과 비용 측면에서 크로스바와 버스 중간 정도에 위치하는 구조이다. 앞서 그림 4에서 보았던 Ω-network가 하나의 예가 된다. IBM의 eServer p690, SGI Origin3000과 최근의 Cenju-4와 같은 상용 시스템에서 이 네트워크 구조를 사용하고 있으며, 그 외 다수의 실험적 시스템에서도 이와 같은 네트워크 또는 이와 유사한 종류의 연결 구조를 사용하고 있다. virtual shared-memory MIMD 시스템처럼 동작하는 BBN TC2000은 유사한 형태의 네트워크(a Butterfly-network)를 사용하였고 프로세서 개수가 증가됨에 따라 새로운 시스템에서도 그것을 사용할 수 있을 것이라 생각된다. 많은 수의 프로세서가 연결되는 시스템에서 $n\log_2 n$ 연결 구조는 크로스바에서 사용되는 n^2 연결 구조보다 빠른 속도로 더 많은 주목을 받게 되었다. 물론 중간 단계에서 사용되는 스위치는 요구 대역폭을 처리할 수 있도록 충분히 빨라야 한다. 구조뿐 아니라 프로세서들 사이의 연결 폭 역시 분명히 중요하다. 즉, 16 비트 병렬 연결을 사용하는 네트워크는 같은 구조에서 순차 연결을 하는 네트워크에 비해 16배 높은 대역폭을 가져야 하는 것이다.

현재 모든 다중 프로세서 벡터 시스템에서는 크로스바를 사용하고 있다. 한 시스템에서 최대 32개 정도의 프로세서를 연결하고 있으므로 아직까지는 이 연결 구조가 적합해 보인다. 그러나 프로세서 개수가 더 많아지게 되면 기술적인 문제가 생길 수 있다. 프로세서 개수가 더 많아짐에 따라 그에 맞춰 충분한 속도를 내는 크로스바를 개발하는 것이 더 어려워지면서, 또 일반적으로는 프로세서 자체의 속도가 더욱 증가함에 따라 그에 요구되는 대역폭에

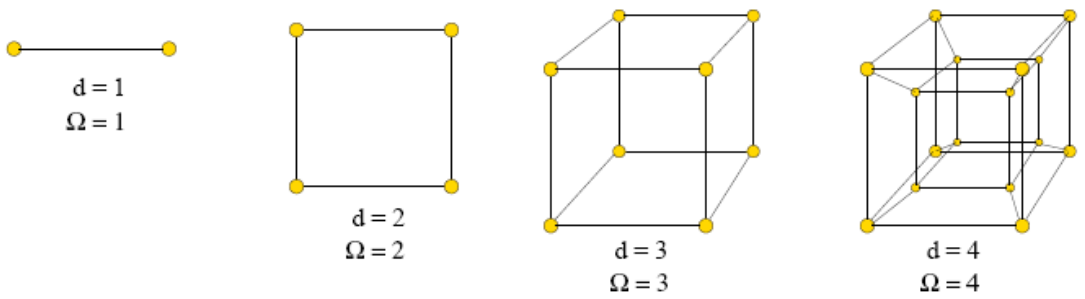
맞춰 크로스바의 속도를 올리는 것까지 고려하면 문제는 더 커지게 된다.
사용되는 네트워크가 무엇이든지 프로세서 타입은 원칙적으로 연결 토폴로지에 무관하지만, 실제로는 버스 구조를 가지는 시스템은 벡터 프로세서를 사용하지 못한다. 이것은 프로세서의 속도가 합당한 비용으로 구성 가능한 어떠한 버스 구조와도 많은 차이가 있기 때문이다. 사용되는 모든 버스 지향적(bus-oriented)인 시스템은 RISC 프로세서를 사용한다. RISC 시스템에서 데이터 접근이 캐시에서 이루어지면 메모리에 접근할 필요가 없어지므로, 각 프로세서의 로컬 캐시를 통해 대역폭 문제를 줄일 수 있다.
MIMD 타입 시스템은 서로 다른 작업을 다른 프로세서를 이용해 동시에 실행시킬 수 있다. 이때 작업들 사이의 동기화가 필요한 경우가 많다. 그래서 다시 연결 구조가 중요해 진다. 대부분의 벡터 프로세서 시스템은 CPU 내부에 통신 레지스터를 두고 있다. 이것을 통해 동기화가 필요한 다른 프로세서와 직접적으로 통신할 수 있다. 시스템은 공유된 메모리를 경유해 동기화할 수도 있다. 이 방법은 많이 느리지만, 동기화가 상대적으로 적게 필요한 경우 충분히 사용될 수 있다. 버스 기반 시스템에서의 통신은 물론 버스를 통하게 되는데 동기화 속도를 최대로 보장할 수 있도록 대부분 데이터 버스와는 별개의 버스를 사용한다.

2.5. Distributed-memory MIMD machines

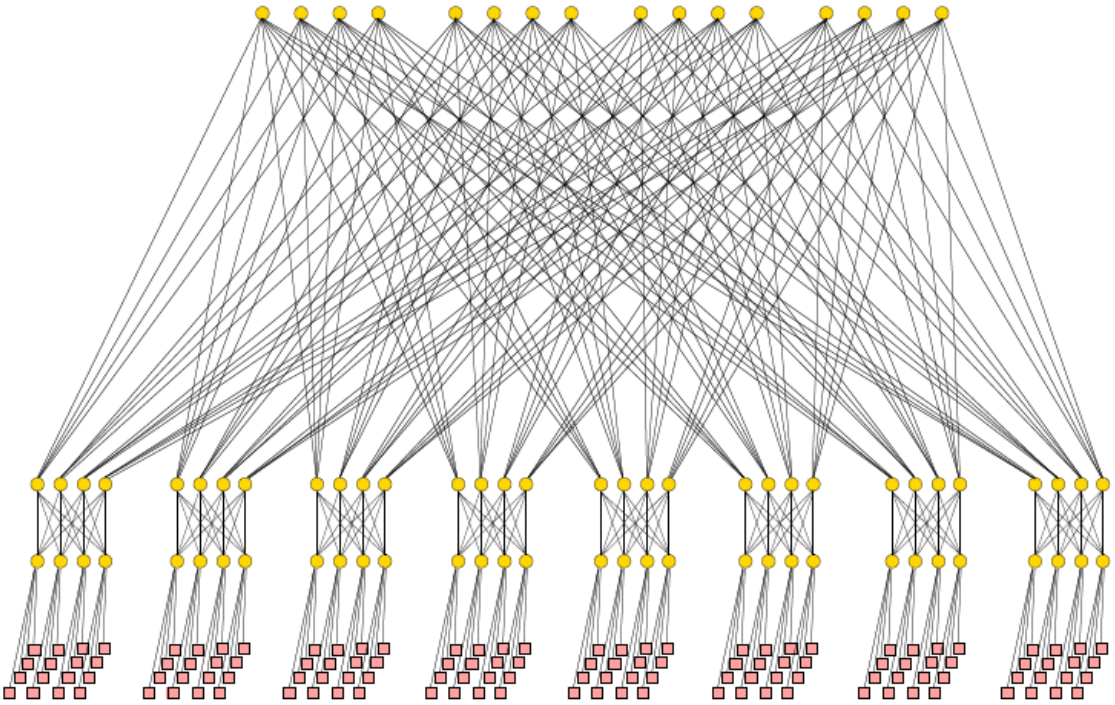
DM-MIMD 시스템은 공유 메모리 시스템이나 DM-SIMD 시스템보다 다루기 어렵지만, 현재 고성능 컴퓨터 분야에서 가장 빠르게 성장하고 있다. DM-SIMD 시스템은 프로세서 배열 시스템으로 시스템 소프트웨어에 의해 자동으로 프로세서에 올려지는 벡터와 다차원 배열 등과 같은 자료구조에 대해 병렬화를 시도한다. 공유메모리 시스템에서도 사용자가 데이터 분배를 직접 처리하지 않아도 된다. 그러나 DM-MIMD 시스템에서는 사용자가 프로세서에게 데이터를 분배해야 하고 프로세서 사이의 데이터 교환도 항상 사용자에 의해 명시적으로 이루어져야 한다. 초창기 DM-MIMD 시스템 사용을 꺼려하게 되던 여러 원인들이 점차 줄어 들고 있다. 부분적으로 이것은 통신 소프트웨어[14,22,23]에 대한 표준이 이제는 존재하기 때문이기도 하고, 또 다른 부분적인 이유로는 적어도 이론적으로는 이 종류의 시스템이 다른 어떤 시스템보다 더 나은 성능을 보여줄 수 있기 때문이기도 하다.
프로세서가 많아 질수록 자동으로 대역폭이 확장되므로 공유 메모리 시스템에서 자주 볼 수 있는 대역폭 문제를 피할 수 있다는 것은 DM-MIMD 시스템의 분명한 장점이다. 게다가 공유 메모리 시스템의 성능 결정에 중요한 역할을 하는 메모리 속도는 DM-MIMD 시스템에서는 상대적으로 덜 중요해 진다. 앞서 언급한 대로 DM-MIMD 시스템에서는 대역폭 문제 없이 더 많은 프로세서를 연결할 수 있기 때문이다. 공유 메모리 시스템에서 DM-MIMD 시스템과 비교되는 최고의 성능을 얻기 위해서는 매우 빠른 프로세서와 그에 맞는 빠른 메모리가 갖춰져야 한다.
DM-MIMD 시스템의 단점도 몇 가지가 있다. SM-MIMD 시스템과 비교해 프로세서 사이의

통신이 많이 느리다. 그래서 통신이 필요한 작업에서 공유 메모리 시스템에 비해 동기화 부하가 많이 커지게 된다. 또, 어떤 프로세서의 로컬 메모리에 없는 데이터에 대한 접근은 논로컬 메모리에서 얻어야 한다. 대부분의 시스템에서 이것은 로컬 데이터 접근과 비교해 매우 느리다. 문제 자체가 프로세서들 사이에서 잦은 데이터 교환이 필요하고 프로세서 동기화가 많다면 시스템이 가지는 이론 최고치에 훨씬 못 미치는 성능을 보일 수도 있다. 앞서 설명한대로 대부분 데이터나 작업의 분배가 사용자에 의해 명시적으로 수행되어야 한다는 것도 하나의 단점이 된다.

DM-MIMD 시스템 역시 데이터 경로들의 속도와 토폴로지가 전체 시스템 성능에 중요한 영향을 미친다. SM-MIMD 시스템처럼 연결 구조를 얼마나 많이 할 것이냐를 비용을 고려해 결정해야 한다. 생각할 수 있는 많은 연결 구조들 중에서 일부만이 실질적으로 사용되고 있는데 그 중 하나가 아래 그림 (a)에서 볼 수 있는 hypercube 토폴로지다.



(a) Hypercubes, dimension 1-4



(b) A 128-way fat tree

그림 5: Some often used networks for DM machine types

hypercube 토폴로지의 장점은 2^d개 노드를 가지는 하나의 hypercube에서 임의의 두 노드는 많아야 d개의 경로를 거쳐 연결된다는 것이다. 그래서 노드의 개수에 따라 네트워크 차원은 로그함수적으로 증가한다. 게다가 이론적으로는 트리, 링, 2-D와 3-D 메쉬 등 hypercube의 다른 어떤 토폴로지라도 모사(simulate)하는 것이 가능하다. 실제로는, hypercube에 대한 정확한 토폴로지는 더 이상 중요하지 않으며, 그 이유는 현재 시장에 나와 있는 모든 시스템들이 "wormhole routing"을 사용하고 있기 때문이다. 이 방식에서는 노드 i에서 노드 j로 메시지가 전송될 때 우선 헤더메시지가 먼저 전송되어 두 노드 사이의 직접연결이 이뤄진다. 연결이 이뤄지면, 데이터는 중간에 있는 노드들의 연산에 의해 방해받지 않고 이 연결을 통해 보내어진다. 노드 사이의 연결 설정을 위해 소비되는 약간의 시간을 제외하면, 통신 시간은 두 노드 사이의 거리에 거의 영향을 받지 않는다. 그렇지만, 작업부하가 많이 걸린 네트워크에서 일부 메시지들이 같은 경로를 두고 경쟁해야 할 때 모든 프로세서 사이에 직접적인 연결이 이뤄지지 않은 네트워크에서는 대기 시간으로 인해 성능에 영향을 받게 된다. 이럴 경우 busy links를 피하기 위해 rerouting을 시도하게 된다.
비용 효율적으로 많은 수의 프로세서를 연결하는 또 다른 방법은 fat tree를 사용하는 것이다. 원칙적으로 간단한 네트워크 트리 구조는 한 컴퓨터 시스템 내의 모든 노드를 연결하기 충분하다. 그러나 실제로 트리의 root 근처에서 정체가 발생하게 되는데 이는 메시지가 목적지 노드까지 내려가기 위해 우선 상위 단계를 거쳐야 하므로 root 근처에서 메시지 집중 현상이 발생하기 때문이다. fat tree는 트리의 상위 구조 대역폭을 확장함으로써 단순 트리 구조의 이런 단점을 보완한다. 이때 대부분은 다중 연결을 통해 대역폭을 확장시키게 된다. N-ary fat tree는 상위 단계 노드와의 연결 수가 하위 단계 노드와의 연결 수의 N배가 된다. 최상위 단계에서 대역폭이 하위 단계 대역폭의 4배가 되는 quarter-nary fat tree가 위의 그림 5 b에 있다.
다수의 초병렬 DM-MIMD 시스템은 2-D 또는 3-D 메쉬(토러스)구조를 많이 사용한다. 이유는 대부분의 large-scale physical simulation들이 이와 같은 토폴로지에 효과적으로 대응되기 때문이기도 하고 더 많은 연결 구조의 구현이 어렵기 때문이기도 하다. 그러나 일부 시스템에서는 데이터 분배와 검색에서의 병목현상을 처리하기 위해 메쉬 이외에 추가적인 네트워크를 가지는 경우도 있다[18].
DM-MIMD 구조를 가지는 상당수의 시스템에서는 크로스바를 사용한다. 상대적으로 적은 수(64개 정도)의 프로세서를 가지는 경우 직접 또는 1단 크로스바를 사용하며 많은 수의 노드를 가지는 시스템에서는 다단 크로스바를 사용한다. 다단 크로스바는 토폴로지 상에서 보다 멀리 있는 노드 사이의 연결을 직접적으로 하지 않고 level 1 크로스바를 level 2 크로스바에, level 2는 다시 level 3에 연결하는 식으로 노드와 노드 사이의 연결을 구성하는 것이다. 이와 같은 방식으로 몇 개의 스위칭 단계를 이용해 수천 개의 노드를 연결하는 것이 가능하다. hypercube 구조 이외에 butterfly, Ω, 또는 shuffle exchange 네트워크 등과 같

이 로그함수적인 복잡성을 가지는 것들도 종종 사용된다.
SM-MIMD 시스템처럼, 하나의 노드에는 계산이나 transaction 처리를 위해 스칼라나 벡터 어떤 형태의 프로세서도 가능하다. 프로세서들은 캐시가 있던 없던 로컬 메모리를 가지며, 대부분의 경우, 이웃 노드와 연결되는 독립적인 통신 프로세서를 가진다. 최근엔 노드 프로세서들 대부분이 이미 생산되어 있는 RISC 프로세서들을 사용하며, 경우에 따라서 성능 향상을 위해 벡터 프로세서가 이용되기도 한다. 노드간 통신 성능의 향상 없이 노드 프로세서를 업그레이드 할 때, 통신과 계산 성능이 조화되지 못하는 DM-MIMD 시스템 고유의 문제가 발생한다. 이로 인해, computational-bound한 문제가 communication-bound한 문제로 변할 수도 있다.

2.6. ccNUMA machines

최근의 클러스터는 상대적으로 적은 수(16개 정도)의 RISC 프로세서를 가지는 SMP 노드를 이용해 구성되는 경향을 보인다. 한 노드 내에 있는 프로세서들은 대부분 1단 크로스바로 연결되며, 클러스터를 구성하는 노드들은 좀더 저렴한 네트워크를 이용해 연결된다. 이와 같은 시스템의 구조를 나타낸 것이 아래 그림이며, 여기서 클러스터 내의 모든 프로세서들은 메모리를 공유하고 있음을 주의해 보자.

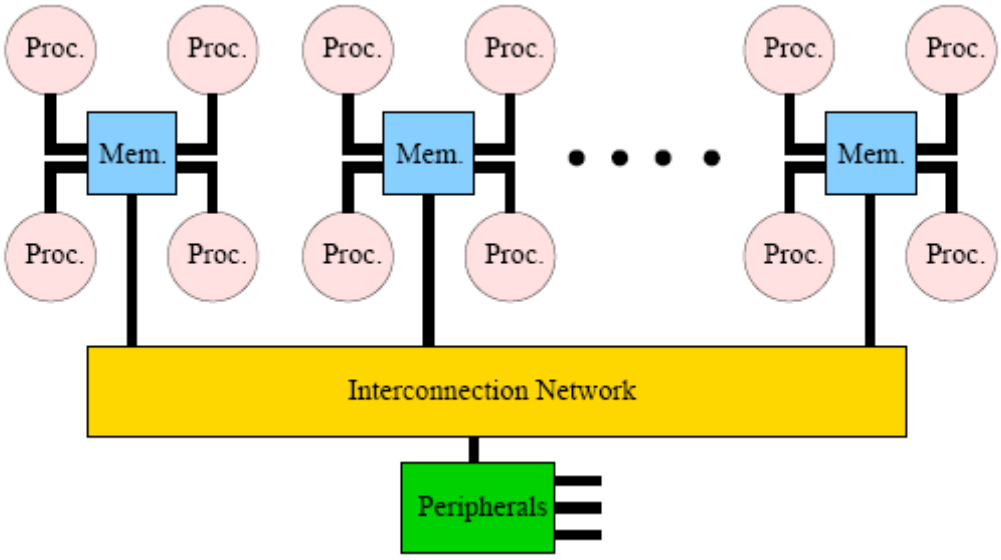


그림 6: Block diagram of a system with a “hybrid” network: clusters of four CPUs are connected by a crossbar. The clusters are connected by a less expensive network, e.g., a Butterfly network.

이것은 앞서 언급됐던 대형 벡터 프로세서 앙상블과 비슷한 형태이지만, 중요한 차이점은 모든 프로세서들이 필요하다면 모든 주소 공간에 접근할 수 있다는 것이다. 이런 시스템들은 SMP 노드들이 메모리를 공유하는 방식에 따라 S-COMA(Simple Cache-Only Memory)와 ccNUMA(Cache Coherent Non-Uniform Memory Access)의 두 가지 형태로 나눌 수 있다. 이와 같은 형태의 시스템들은 SM-MIMD로 볼 수 있지만, 메모리가 물리적으로 분산되어 있으므로, 데이터 접근 연산이 항상 같은 시간에 처리된다는 것을 보장할 수 없다. S-COMA 시스템에서 로컬 노드의 캐시 계층구조는 다른 노드의 메모리로 확장된다. 그래서 로컬 노드의 메모리에 없는 데이터가 요청될 때 해당 데이터가 저장된 노드의 메모리에서 검색된다. 이런 개념이 ccNUMA에서는 더욱 확장되어서 시스템의 모든 메모리를 대상으로 데이터를 검색하게 된다. 그래서 데이터는 물리적으로 로컬이 아닐지라도, 논리적으로 하나인 공유 주소 공간에 속하게 된다. 데이터는 물리적으로 여러 노드에 분산되어 있을 수 있기 때문에 다른 데이터에 대한 접근 시간이 달라질 수 있고 이로 인해 non uniform data access라는 표현을 사용한다. Cache Coherent의 의미는 변수들이 모든 CPU에 대해 항상 일관성 있는 값을 가져야 한다는 것이다. 이런 관점에서 시스템은 해당 변수들을 저장하는 캐시들이 일관성 있게 값을 가지도록 보장한다. CPU의 캐시 일관성을 보장하는 다양한 방법이 있으며, 그 중 한 가지가 스누피 버스 프로토콜(snoopy bus protocol)이다. 이것은 캐시가 시스템내의 CPU들로 전송되는 모든 변수들을 체크하여 그에 해당하는 변수를 현재 가지고 있다면 그것을 갱신하는 방법이다. 다른 방법으로는 directory memory를 사용하는 것이며, directory memory는 변수들의 모든 복사본과 그 유효성을 추적할 수 있는 메모리의 특별한 부분이다.

현재 S-COMA scheme을 쓰는 상용 시스템은 없으며, 많이 사용되는 Bull NovaScale, HP Superdome, SGI Altix3000 등과 같은 시스템들은 상용 ccNUMA 시스템들이다. NUMA 시스템의 중요한 특성으로 NUMA factor가 있다. 이 factor는 로컬 메모리에 있는 데이터에 접근하는 latency와 논-로컬 메모리에 있는 데이터에 접근하는 latency의 차이를 나타낸다. 그런데 시스템의 연결 구조에 의존하는 NUMA factor는 시스템 부분마다 달라질 수 있다. 이웃 노드 데이터에 접근하는 경우가 다수의 크로스바 단계를 거쳐야 하는 먼 거리에 있는 노드의 데이터에 접근하는 것 보다 더 빠를 것이다. 그래서 NUMA factor가 언급될 때는 시스템에서 가장 먼 거리만큼 떨어져 있는 프로세서 사이의 latency를 기준으로 한다.

실용적인 목적에 의해 ccNUMA 시스템을 SM-MIMD 시스템으로 분류할 수 있으며, 또한 메모리가 물리적으로 분산되어 있더라도 단일 시스템 이미지를 만들기 위해 특별한 하드웨어/소프트웨어(directory memory 같은)의 지원이 같이 어우러져야 하기 때문이기도 하다.

2.7. Clusters

로컬 네트워크로 연결된 워크스테이션/PC의 집합인 클러스터는 1994년 Beowulf 클러스터

가 처음 소개된 이후 시장에서 놀라운 성장속도를 보이고 있다. 그 매력은 저렴한 하드웨어와 소프트웨어의 비용과 더불어 클러스터 개발자 또는 사용자가 쉽게 그들의 시스템을 제어할 수 있는 데 있다. 정례적으로 클러스터 컴퓨팅에 대한 현재의 상황을 리뷰하는 IEEE TFCC(Task Force on Cluster Computing)의[43] 활발함에서 클러스터에 대한 높은 관심을 엿볼 수 있으며, 클러스터를 어떻게 구성하고 관리하는 가에 대한 책들이 최근에 높은 인기를 얻고 있다는 것을 봐도 알 수 있다[33,41]. 클러스터 시장이 성숙하고 매력적인 시장이 되면서 거대 HPC 업체들뿐 아니라 많은 창업 회사들이 이 분야에 뛰어들어 맨손으로 클러스터를 구성하기를 원치 않는 그룹들에게 준비된 클러스터 솔루션을 제공하고 있다.

클러스터 설정을 판매하는 업체의 수가 지나치게 많아 이곳에서 모두를 언급할 수는 없다. 게다가 클러스터를 사용하는 용도는 통합된 시스템들과는 일반적으로 많은 차이가 있다. 클러스터는 대부분 capability computing에 많이 사용되지만, 통합 시스템들은 주로 capacity computing에 사용된다. capability computing이란 하나 또는 소수의 프로그램만을 실행하여 높은 컴퓨팅 성능을 얻기 위해 동시에 많은 수의 프로그램이나 많은 사용자가 시스템을 사용하지 않도록 하는 것을 말하며, capacity computing이란 많은 프로그램과 사용자에 의해 시스템의 가용 사이클을 대부분 사용하여 시스템이 가진 성능을 최대로 사용하도록 하는 것이다. 전통적으로 대형 슈퍼컴퓨터 업체들은 자신들 시스템의 주요 자원이 가능하면 효과적으로 사용되도록 해서 capacity computing을 제공하도록 해왔다. 이와 대조적으로 Beowulf 클러스터에서 대부분 사용하는 운영체제인 리눅스에는 capacity computing을 위한 툴들이 없거나 있더라도 상대적으로 부족하였다. 그러나 클러스터가 평균적으로 더 커지고 보다 안정적이 되면서 capacity 서버에서도 클러스터를 사용하는 경향이 생기고 있다. 이를 위해 필요한 클러스터 관리 툴들과 배치 시스템 등의 조건들을 참고자료 [39]에서 잘 다루고 있다. 아울러 이 문서에서는 RISC(Compaq Alpha) 기반 시스템과 Intel Pentium III 기반 시스템에 대해 응용 프로그램 작업 부하의 성능을 비교 평가하고 있다. 중요하지만 그다지 놀랍지만은 않은 이 문서의 결론은 네트워크의 성능이 대부분의 경우, 특히 compute bound 프로그램에 있어 매우 중요하다는 것이다. 또 다른 주목할 만한 결과는 CPU 하나 보다는 여러 개를 가진 계산 노드를 사용하는 것이 클러스터를 compact하게 구성하고 에너지와 냉각 효율을 고려한 측면에서 유리하지만, 이때 많은 CPU가 공통된 노드의 메모리를 끌어 써야 한다는 측면에서 심각한 성능저하를 가져올 수도 있다는 것이다. 이런 경우, 메모리 집중적인 프로그램에서 노드의 대역폭은 중요한 조건이 아니다.

클러스터에서 사용 가능한 통신 네트워크로 최근에 사용되는 몇 가지가 있다. 경제적인 이유로 매력적이지만, 높은 latency(≒100 μs) 결점을 가지는 Gigabit Ethernet 또는 100 Mb/s Ethernet은 언제나 사용 가능하다. 더불어 Myrinet[24,25], Infiniband[32], SCI[20] 같은 user space에서 동작하는 네트워크들이 있다. Myrinet과 Infiniband의 경우 최대 200 ~ 850 MB/s의 대역폭과 7 ~ 9 μs의 latency를 가진다. SCI는 이론적으로 400 ~ 500 MB/s의 대역폭을 가지며 3 μs 이하의 latency를 가진다. SCI는 상대적으로 많은 비용이 들지만, 일부 클러스터에서 사용되고 있다. Myrinet과 QsNET, SCI의 네트워크 속도는 일부 통합

병렬 시스템과 비슷한 성능을 보여준다. 그래서 DM-MIMD 슈퍼컴퓨터 업체에서 제공하는 소프트웨어와 프로세서의 속도와는 별개로, 클러스터와 DM-MIMD 시스템들 사이의 차이는 많이 줄었고 얼마 있지 않아서는 그 차이가 더 줄어들 것이 분명하다.
클러스터 컴퓨팅에 대한 보다 최신의 내용들은 TFCC[43] 백서에서 볼 수 있다. 하드웨어와 소프트웨어의 가용 제품들, 제시된 질문들과 이에 대한 현재의 연구 초점들 등에 대한 조언을 이 문서에서 볼 수 있을 것이다.

2.8. Processors

10년 전과 비교해 출시되는 프로세서는 매우 많이 변했다. 1980년에서 1990년 사이에는 벡터 프로세서와 같은 독점적인 프로세서들이 그 당시 슈퍼컴퓨터 시장의 견인차였다. 현재는 일상적인 RISC 프로세서들과 인텔 x86 호환 프로세서들이 그 역할을 하고 있다. 벡터 시스템을 생산하는 업체는 사실상 두 개만 남았고, 다른 모든 시스템들은 RISC CPUs 또는 x86-like (Cray MTA-2 제외)를 기반으로 한다. 이곳에서는 현재 슈퍼컴퓨터에 사용하는 주 프로세서들에 대해 간략히 알아보고 곧 나오게 될 프로세서들에 대해 미리 살펴본다.
지난해에 RISC 프로세서의 출시가 대폭 줄어들었다. 비록 HP/Compaq EV7과 HP PA-RISC 프로세서가 여전히 생산되고 있지만 이보다 앞선 칩들을 사용하는 시스템들은 HP가 이 프로세서들을 Itanium 라인으로 교체하기로 결정한 2006년까지만 시장에 나올 것이다. PA-RISC 8800은 Superdome 9000 시스템을 위해 아직 생산되지만, 곧 사라지게 될 것이다. 공식적으로는 PA-RISC 8900이 계속 나오게 될 것이다.
아직까지는 RISC 프로세서들이 주요 역할을 담당하고 수행하고 있지만, 많은 RISC 프로세서들이 곧 사라지게 될 것이다. Alpha, PA-RISC와 MIPS 프로세서를 장착한 시스템들은 모두 사라지고 Itanium 프로세서 시스템으로 대체될 것으로 발표되었다. RISC 프로세서 군이 사라지는 경향은 걱정과 기대를 모두 갖게 한다. 프로세서의 다양성이 사라지고 이로 인해 해당 프로세서를 채택하는 시스템 선택의 다양성이 사라진다는 측면에서 우려의 목소리가 있다. 한편으로는 보통의 프로세서를 가지는 시스템의 성능을 향상시키려는 경향이 있는데 특별한 목적을 가지고 부가적인 프로세서를 두는 방법을 사용한다. 미리 설정된 FPGAs나 DSPs의 형태로 부가 프로세서를 두게 되는데 성능과 가격, 사용의 용이성 등에서 이들의 가능성은 여러 응용분야에서 관심을 가질 정도의 많은 발전이 있었다.
일반적으로 RISC 프로세서들은 인텔 Pentium 3/4 프로세서나 AMD의 프로세서보다 낮은 클럭 주파수를 가지지만, 부동소수 연산이 많은 프로그램의 속도를 빠르게 하는 여러 가지 특성을 가진다. 우선 모든 RISC 프로세서들은 하나의 클럭 사이클에 2개 이상의 64비트 부동소수 연산 결과를 낼 수 있다. 두 번째로 모든 RISC 프로세서는 out-of-order 명령 실행 (out-of-order instruction execution) 특성을 가지는데, 이를 통해 사이클당 처리 가능한 명령어의 수를 늘이게 된다. 최신의 AMD 프로세서는 한 번에 2개의 부동소수 명령어를 처

리하고 out-of-order 실행을 하지만 인텔 x86 명령어를 지원함으로 인해 어느 정도 제한을 받는다. 캐시 실패가 발생할 때 프로세서와 메모리 사이의 대역폭은 인텔(또는 호환) 프로세서들 보다 더 크지만 AMD Opteron의 경우는 예외다. 그것은 Opteron의 on-chip 메모리 컨트롤러와 메모리와 CPU를 연결하는 HyperTransport 버스 때문이다. 여러 가지 RISC 프로세서들 사이의 공통점에도 불구하고 명령어 latency나, 처리되는 명령어의 수 등에 있어 차이점도 많이 있다.

2.8.1. AMD Opteron

오랫동안 코드네임 Hammer로 알려진 Opteron은 Athlon의 뒤를 이어 나온 AMD의 최신 프로세서이다. 첫 제품은 2003년 초에 발표되어 사용되었다. Athlon이 그랬던 것처럼 Opteron도 인텔의 x86 ISA(Instruction Set Architecture)를 따르는 clone으로, 클러스터 환경에서 자주 사용될 것이다. 따라서, Opteron이 현재 통합 병렬 시스템에서 사용되고 있지는 않지만 이곳에서 알아보도록 다.

Opteron은 현재의 RISC 프로세서가 가진 많은 특성들을 가지고 있다. out-of-order 실행을 지원하고, 다중 부동소수 유닛을 가지고, 동시에 9개의 명령어를 처리할 수 있다. Opteron 프로세서 코어는 사실상 Athlon의 그것과 매우 유사하다.

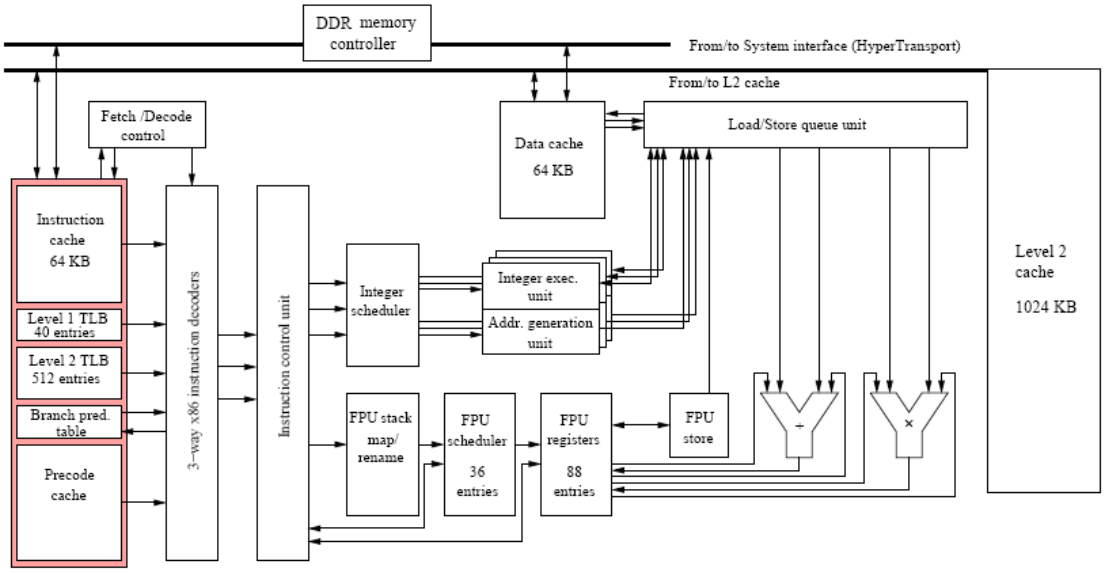


그림 7: Block diagram of AMD Opteron processor

위의 그림에서 프로세서는 정수와 주소 계산을 관리하는 24-entry Integer Scheduler를 경유하는 Integer Execution Unit과 Address Generation Unit을 세 쌍 가진다. Integer Scheduler와 Floating-Point Scheduler 모두, 명령어 해석기로부터 해석된 명령어를 받는 96-entry Instruction Control Unit를 거친 명령어를 받게 된다. Opteron의 특징 중 관심을 끄는 것은 Pre-decode 캐시에 저장 가능한 고정 길이 매크로 연산에서 x86 명령어에 대

한 pre-decoding 이다. 이를 RISC Operations (ROPs)라 부른다. 이것이 명령어 해석기로 보다 빠르고 지속적인 명령어 흐름을 가능하게 한다. RISC 프로세서처럼 분기 예측을 지원하는 Branch Prediction Table도 있다.

부동소수 유닛은 FPU Stack Map & Rename 유닛을 경유한 명령어의 out-of-order 실행을 지원한다. 부동소수 유닛은 Instruction Control Unit으로부터 부동소수 명령을 받아 그것을 FPU Scheduler에게 넘기기 전에 필요하다면 순서를 바꾼다. Floating-Point Register File은 RISC 프로세서에서 가용한 레지스터 수에 가까운 88개의 구성요소를 가진다. x86 명령어를 위해 인텔 아키텍처에서 흔히 사용하는 레지스터 스택을 두는 대신 하나의 Flat 레지스터 파일에 16개의 레지스터를 두고 있다.

프로세서의 부동소수 부분은 세 개의 유닛을 가진다. Floating Store 유닛은 Load/Store Queue Unit과 Floating Add and Multiply 유닛의 처리 결과를 저장한다. Load/Store Queue Unit과 Floating Add and Multiply 유닛은 슈퍼스칼라 모드로 동작할 수 있고 그 결과 클럭 사이클 하나당 두 개의 부동소수 처리가 가능하다. 인텔의 Pentium 3 프로세서와의 호환성 문제 때문에 부동소수 유닛은 항상 인텔의 SSE2/3 명령어와 AMD의 3DNow! 명령어를 실행할 수 있다. 그러나 그런 명령어들이 Fortran90 이나 C(++)와 같은 고급 언어를 통해 접근할 수 없다는 문제점을 가지고 있다. 두 명령어 집합은 가시화 데이터의 대규모 처리를 위해 나타났고 32비트 정밀도만 허용하고 있다.

구성요소들이 축소되면서 Opteron 칩은 8 사이클의 latency를 가지는 1M 크기의 2차 캐시와 메모리 컨트롤러를 가지게 되었다. 이것과 더불어 대역폭이 6.4 GB/s에 이르는 향상된 메모리 버스 등으로 인해 이전의 메모리 시스템과 비교해 많은 성능 향상을 가져왔다. AMD가 HyperTransport라 부르는 Opteron의 메모리 버스는 인증된 Compaq 기술에 근거하고 있고 Compaq의 EV7 프로세서에 채택된 것과 유사하다. 이것은 매우 낮은 메모리 latency를 가진 다중 프로세서 시스템을 만들기 위해 몇몇 프로세서의 “glueless” 연결을 허용한다. 1.8 ~ 2.2 GHz 정도의 클럭 수를 가지는 Opteron은 현재 가용되는 RISC 프로세서의 대용으로 관심을 끌고 있다. 특별히 HyperTransport 연결의 발전성이 SMP-type 클러스터를 구성하는 데 있어 높은 관심을 모으고 있다. 11,000개 프로세서 이상을 연결하는 대형 Opteron 클러스터가 Cray에 의해 Sandia 국립 연구소에 곧 구축될 것이다. 여기서 HyperTransport 기능을 사용하게 될 것인데 이는 HyperTransport가 Cray 네트워크를 이용한 연결을 보다 간단하게 하는 개방 표준이기 때문이다.

2.8.2. HP/Compaq Alpha EV7

AlphaServerSC, GS1280과 다양한 클러스터 같은 Compaq 시스템들은 2003년 시작 이래로 Alpha EV7 프로세서에 맞춰 졌다. 최근의 제품으로는 0.13 μm 기술로 제작된 EV79가 있다. EV7 프로세서의 코어는 EV6x 아키텍처와 거의 동일하다(아래 그림 참조).

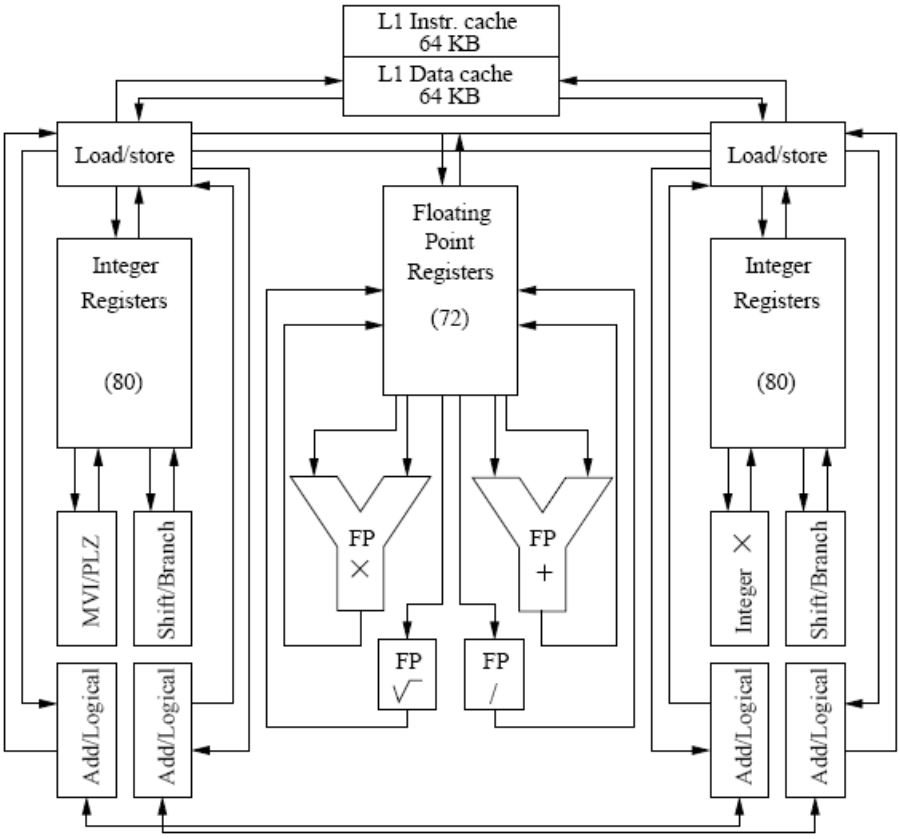


그림 8a: Block diagram showing the functional units in a Alpha EV7 processor

여기서 주목할만한 사실은 각각 80 entry를 가지는 두 개의 정수 레지스터 파일이 있다는 것이다. 정수 레지스터 파일은 각각은 Compaq에서 cluster0, cluster1이라 부르는 Integer Functional 유닛 집합을 지원한다. 필요하다면 네 개의 Integer Add/Logical 유닛이 1 사이클 만에 값을 교환할 수 있다. 비록 그림에 나타나 있지는 않지만 정수의 곱셈은 완전히 파이프라인화된다. 두 개의 정수 clusters와 두 개의 부동소수 유닛은 명령어를 동시에 6개까지 처리할 수 있다. 두 개의 load/store 유닛은 각각 2-way 집합 연관성을 가지는 64 KB 크기의 명령어 캐시와 데이터 캐시에 의존한다. 4개의 명령어가 추정 실행(speculative processing)을 통해서 처리될 수 있다. 80개의 정수 레지스터와 72개의 부동소수 레지스터에 대해 각 41개의 레지스터 파일들이 추정 결과를 가질 수 있다. 명령어의 out-of-order 처리는 길이가 20인 정수 큐(queue)와 15 entry를 가지는 부동소수 큐를 경유하여 지원된다. 그러나 정수 처리 클러스터가 똑같은 Functional 유닛을 가지고 있지 않아서 정수 명령어를 꺼내는 것은 모두가 동적으로 스케줄 될 수는 없다. 특정 유닛에서 실행될 필요가 있는 명령어들(예를 들어 정수 곱셈은 cluster0에서만 가능)은 정적으로 스케줄된다. 명령어가 꺼내지거나 잘못된 예측에 의해 제거되자마자 그 명령어는 큐에서 삭제되고 다른 명령어로 대체될 수 있다. 명령어를 꺼내는 것은 분기 예측기 (branch predictor)에 의해 조정된다.

이 하드웨어는 글로벌 예측 테이블과 로컬 예측 테이블 그리고 Branch History Table(BHTs)을 가지고 있는데, 이를 통해 최적의 명령어를 명령어 캐시와 레지스터로 꺼내기 위해 예측기를 학습시키게 된다.
EV7 프로세서 에서는 0.18 μm대신 0.13 μm feature density를 사용함으로써 1.75 MB의 2차 캐시를 둘 수 있게 되었다. 이 2차 캐시는 7-way 집합 연관성과 12-사이클 latency를 가진다. 칩에는 2개의 메모리 컨트롤러가 있다. 이전과 가장 큰 차이점은 칩과 이웃 칩을 연결하는 4개의 dual 채널(North, East, South, West)이 있다는 것이다. HP에서 "switchless SMP processing"이라고 부르는 단일 채널 하나는 6.4 GB/s의 대역폭을 가진다. 그리고 이 채널은 그 이름이 의미하듯 낮은 메모리 latency를 가지는 SMP 노드를 구성하는 데 아주 적합하다. 완전한 칩의 레이아웃을 그림 8b에서 볼 수 있다.

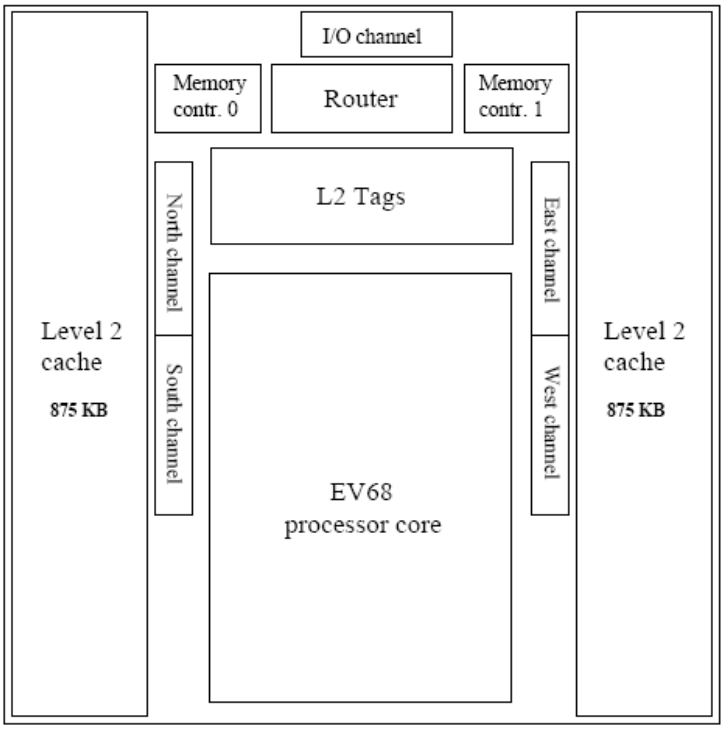


그림 8b: Chip layout for the Alpha EV7 processor

시스템이 6 GB/s(총 12 GB/s) 대역폭을 가지는 RAMBUS 메모리에 맞춰져 있어 메모리 경로는 4 x 5 RAMBUS link로 구현된다. 칩에서 나가는 direct I/O dual link 또한 1.6 GB/s 대역폭을 가진다. 가장 빠른 EV79 제품은 1.3 GHz의 클럭을 가지고 있으며 차후 조금 더 클럭이 높아질 것으로 기대된다.

2.8.3. Hewlett-Packard PA-RISC 8800

Superdome과 같은 Hewlett Packard 시스템의 계산 성능은 PA-8800 칩에서 나온다. 올해

2월 이후로 이 칩은 PA-RISC 8700+를 대체하고 있다. 8700+는 PA-8700과 비교해 동일한 프로세서지만 클럭 수가 750 MHz에서 875 MHz로 높아져 이름에 "+"를 붙이고 있다. PA-8800 프로세서는 다르다. PA-8800은 하나의 칩에 두 개의 CPU를 넣는 최근의 경향을 따르고 있다. 이것은 180 nm 대신 130 nm를 사용하는 고밀도 기술에 의해 가능해 졌다. 또 이로 인해, 클럭 사이클을 1 GHz까지 올리는 것이 가능하게 됐다. 게다가 칩 하나당 대역폭이 1.6 GB/s에서 6.4 GB/s로 대폭 증가되었다. 이것은 Itanium 2의 프론트-사이드 버스 속도와 같은 것으로 HP는 PA-RISC 칩을 Itanium 2 칩으로 교체하려고 한다. 칩에 있는 CPU 코어들은 거의 PA-8700(+)정도로 작아졌다. 차이점이 하나 있는데 데이터 캐시와 마찬가지로 두 부분에서 L1 명령어 캐시 크기가 750 KB에서 1.5 MB로 두 배가 되었다.
하나의 칩에 두 개의 CPU가 있어서 CPU당 순 대역폭은 2배로 증가되었다. Superdome과 같은 더 큰 시스템들은 4개 프로세서 셀로 구성되는데, 여기서 칩들은 보통 셀 안의 시스템 메모리와 연결되고, 하나의 셀을 SMP 노드로 구성하고 다른 셀과의 통신을 담당하는 셀 컨트롤러와 연결된다. 다음 그림은 프로세서 코어를 나타내고 있다.

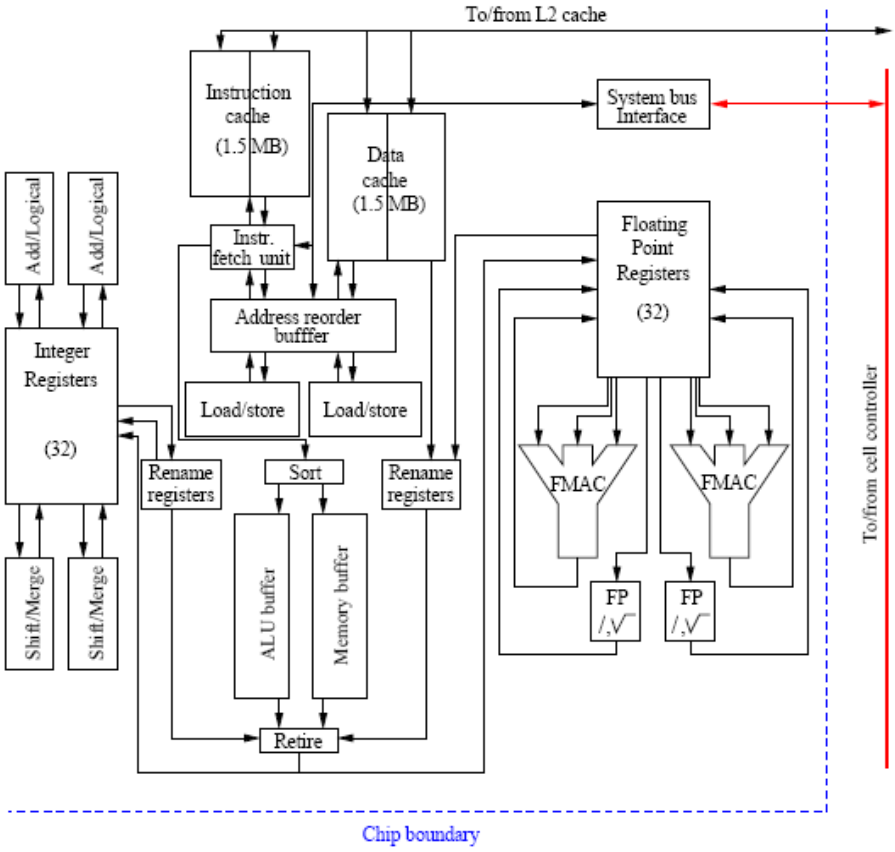


그림 9a: Block diagram of an HP PA-RISC 8800 processor core

두 코어의 레이아웃과 칩에 있는 다른 중요한 장치들을 다음 그림 9b에서 볼 수 있다.

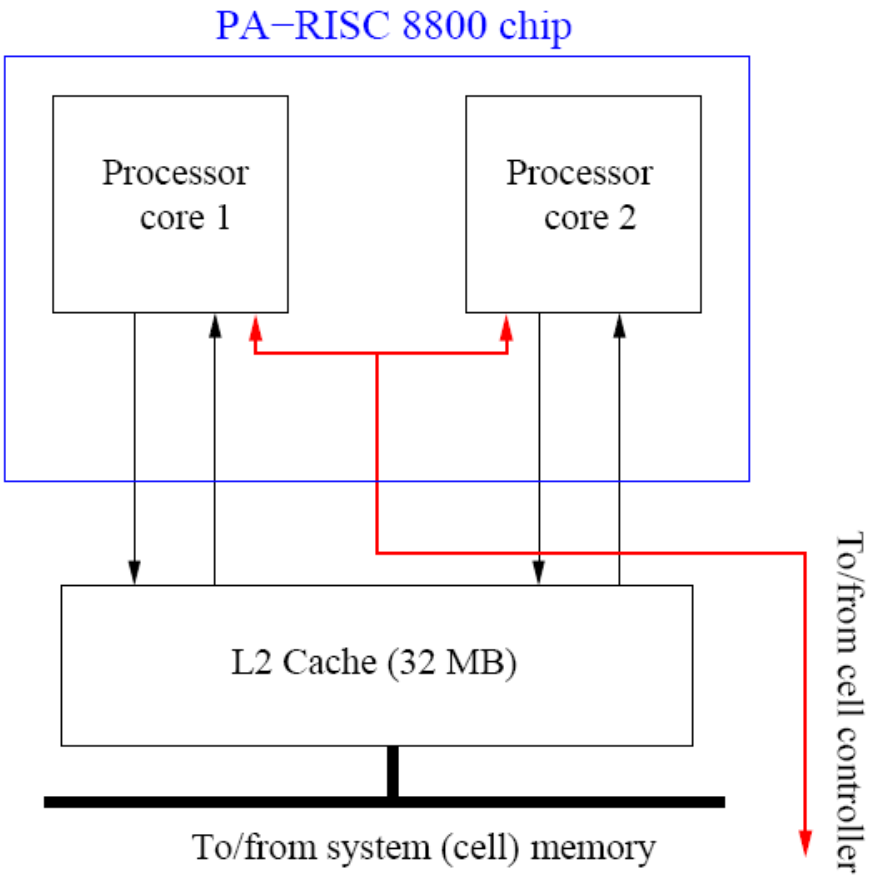


그림 9b: Chip layout of an HP PA-RISC 8800 CPU

PA-8x00 칩의 특성은 2차 캐시가 없다는 것이다. 이로 인한 손실은 큰 L1 캐시를 두어서 보상된다. L1 캐시는 1.5 MB 데이터 캐시와 1.5 MB 명령어 캐시로 구성된다. 둘 모두 4-way 집합 연관성을 가진다. L2 캐시가 없는 문제는 8800에서 수정되었다. 8800은 칩 밖에 32 MB나 되는 큰 L2 캐시를 가진다.

PA-8600부터 logic의 축소로 인해 L1 캐시를 칩에 둘 수 있게 되었다. 캐시의 latency는 2 사이클이다. 매 사이클마다 데이터가 레지스터로 옮겨질 수 있도록 하기 위해 load/store 유닛들은 "out-of-phase"로 동작한다. 이것은 하나의 유닛이 데이터 캐시의 반을 load할 때 나머지 반을 다른 유닛이 load 하는 것이다. Address Reorder Buffer는 load들에 우선순위를 정하고 매 사이클에 반씩 교차하여 load 하도록 한다.

다른 모든 진보된 RISC 프로세서들처럼 PA-8700(+)는 out-of-order 실행을 한다. 여기서 명령어의 순서는 Instruction Reorder Buffer (IRB)에 의해 결정되는데 IRB는 Computational Functional 유닛을 구동하는 ALU 버퍼와 load/store 유닛들을 제어하는 메모리 버퍼를 가지고 있다. 추정 분기 (speculative branch)가 추정을 잘못한 경우, 이에 의존하는 명령어들은 IRB에서 내려지고 새로운 후보 명령어들로 채워진다. 분기 예측은

branch history table(BHT)를 통해 제어되지만, 이것은 동적 분기 예측이고 정적인 분기 예측은 컴파일러 수준에서 또는 프로그램의 이전 실행에 대한 실행 추적(execution trace)에 의해서 수행될 수 있다. PA-8600 이전에는 BHT의 크기가 작았었는데, 이제는 더 나은 추정 결과를 얻기 위해 많이 커져서 2048 entry를 가진다. 더 효과적인 address translation을 위해 Translation Look-aside Buffer (TLB)도 160 entry까지 커졌다. 또 PA-8800에는 데이터 캐시로부터의 pre-fetch 기능도 있다.

앞서 그림 9a에서 보았듯이 2개의 부동소수 연산 유닛이 있다. 이 부동소수 연산 유닛 하나는 연산이 axpy(x=x+ ay)형태일 때, 한 사이클에 2회의 부동소수 연산이 가능한데, 이것을 HP에서는 Floating Multiply Accumulate (FMAC)라고 부른다. 이에 의해 1 GHz의 클럭에서 이론 성능 최대치가 4 Gflop/s/CPU, 8 Gflop/s/chip에 이른다. 그러나, 연산 순서나 연산 구성이 달라지게 되면 1 flop/cycle/floating-point 유닛은 더 낮은 flop rate으로 실행될 수 있다.

HP의 로드맵에 따르자면 앞으로 최소 하나이상의 새로운 PA-8x00가 계획되어 있다. PA-8900은 IA-64 Itanium Montecito와 함께 시장에 나올 것인데, 그런 측면에서 보면 좀 혼란스럽다. PA-RISC family는 가장 최신의 PA-8800 chip이 마지막이 될 것이라는 예측도 있는데 이렇게 생각하는 것이 더 맞을 것 같다.

2.8.4. IBM POWER4+

POWER4+ 을 잇는 POWER5가 IBM의 다른 서버 라인에서 이미 사용 중이지만, IBM의 슈퍼컴퓨터 라인에서 p690시리즈 노드들은 아직도 POWER4+ 칩을 계산 엔진으로 사용하고 있다. 의심할 여지없이 POWER5가 p690 타입 시스템에서도 사용될 것이 분명하지만, 아직은 그렇지 않다. 그래서 이곳에서는 POWER4+ 와 이를 대신할 POWER5 칩 둘 다에 대해 알아볼 것이다.

이 자료가 작성되는 현재 시점에 POWER4+ 의 클럭 수는 1.5 ~ 1.9 GHz 정도이다. 이것은 POWER4 칩의 클럭을 증가시킨 것이고 클럭 수가 증가한 것 외에 다른 새로운 특성은 없다. 아래 그림 10a에서 볼 수 있듯 IBM은 하나의 칩에 두 개의 프로세서 코어를 넣어 칩의 크기는 많이 커졌다. 칩에는 0.5 MB 크기의 세 개의 모듈로 구성된 1.5 MB 크기의 2차 캐시가 있다.

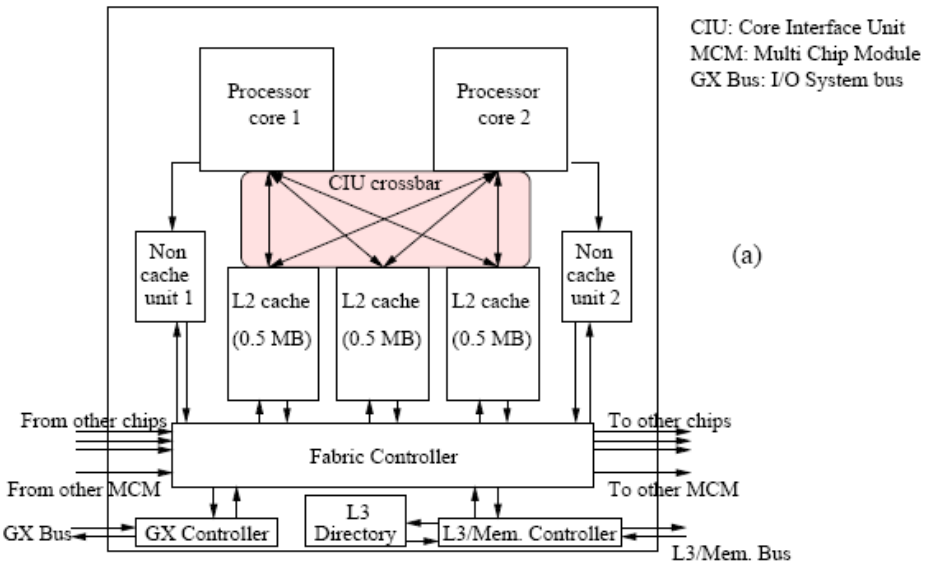
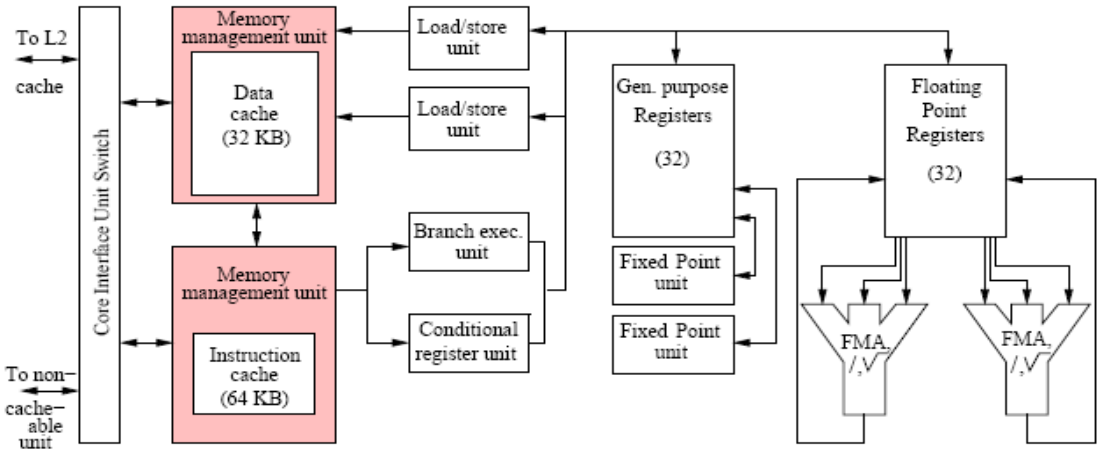


그림 10a: Diagram of the IBM POWER4+ chip layout

L2 캐시 모듈은 Core Interface Unit (CIU) 스위치에 의해 프로세서와 연결된다. 이 스위치는 포트당 40 Byte/cycle의 대역폭을 가지는 2x3 크로스바로, 각 프로세서의 L1 명령어 캐시나 데이터 캐시로 32 바이트를 올릴 수 있고, 동시에 8 바이트 값을 저장할 수 있다. 또한 각 프로세서에는 Fabric Controller와 연결되어 non-cacheable 연산을 담당하는 Non-cacheable Unit이 있다. Fabric Controller를 통해 MCM 내에 있는 세 개의 다른 칩들과 통신(대역폭 13.6 GB/s: 1.7 GHz) 하며, L3 캐시와의 통신(대역폭 9.0 GB/s: 1.7 GHz)과 다른 MCM들과의 통신(대역폭 6.8 GB/s: 1.7 GHz)에서도 Fabric Controller를 이용하게 된다. 칩에는 다양한 디바이스들이 있는데 L3 캐시 디렉터리, off-chip latency를 현저하게 낮추는 L3와 메모리 컨트롤러, GX 버스의 소통량을 담당하는 GX 컨트롤러 등이 있다. GX 버스는 시스템으로(부터) 데이터를 전송하고 실질적으로 I/O에 사용된다. L3 캐시의 최대 크기는 32 MB나 된다. 그림 10a에는 logging과 에러를 검출하는 logic과 performance monitor와 같은 일부 통합 장치들이 빠져 있다.



(b)

그림 10b: Block Diagram of the POWER4+ processor core

POWER4+ 프로세서 코어는 여러 가지 면에서 POWER3 프로세서와 유사하다. 3개가 아닌 2개의 Integer Functional 유닛(IBM에서는 Fixed Point Unit이라고 함)이 있고 POWER4 코어는 Fused Branch/Dispatch Unit 대신 분리된 Branch and Conditional Register Unit을 가져, 모두 8개의 실행 유닛을 가진다. 특이한 것은 명령어 캐시의 크기가 데이터 캐시 크기의 2배인 것이고(명령어 캐시: 64 KB 직접 사상, 데이터 캐시: 32 KB 2-way 집합 연관) 모든 실행 유닛은 그들과 관련된 명령어 큐(queue)를 가진다는 것이다. 이 명령어 큐는 다양한 단계에서 200개에 이르는 명령어를 out-of-order 처리할 수 있다. 이렇게 많은 명령어를 동시에 처리하는 것은 매우 정밀한 분기 예측 능력을 요구한다. 차례대로 분기 예측 logic에 의해 영향을 받는 Instruction Fetch Address Register의 제어 하에 명령어 캐시에서 명령어가 꺼내어 진다. 이것은 각각 16K entry를 가지는 로컬과 글로벌 Branch History Table(BHT)과 하나의 선택 테이블(selector table)로 구성된다. 선택 테이블은 특정한 상황에서 최적의 예측을 얻게 되는 BHT를 기억해 둔다. 이는 앞으로 유사한 경우가 나타날 경우에 있어 BHT의 예측 우선순위를 두기 위한 것이다.

POWER3와 달리 fixed point 유닛은 한 사이클에 완료될 수 있는 정수 연산뿐 아니라 곱셈이나 나눗셈과 같이 여러 사이클이 걸리는 정수 연산도 수행한다. 나눗셈이나 제곱근 계산과 같이 많은 사이클이 소비되는 연산을 위해 따로 준비된 부동소수 연산 유닛은 없다. 모든 부동소수 연산은 FP 유닛에 의해 처리되고 PA-8800처럼, axpy 형태의 연산처리에 적합한 명령어가 있다. 이것을 IBM에서는 Fused Multiply Add (FMA)라고 하며 이를 통해 매 사이클 마다 2개의 부동소수 연산처리가 가능하다. 이로 인해 1.9 GHz 클럭을 가지는 프로세서의 이론 최고성능이 7.6 Gflop/s에 이른다. HP 프로세서처럼, 부동소수 연산의 구성 자체가 FP 유닛이 FMA를 수행할 수 있는 형태여야 하며 그렇지 않다면 성능은 절반으로 떨어진다.

여기서 언급하고 있는 이중 코어를 가지는 프로세서가 일반적으로 많이 사용되지만, HPC

분야에서는 단일 코어를 가지는 프로세서도 권장되고 있다. 그 이유는 L2 캐시로부터의 대역폭이 CPU들 사이에 공유될 필요가 없어서 108.8 GB/s에 이르는 경합없는(contention-free) 전송을 할 수 있기 때문이다. 이에 반해 이중 코어 버전에서는 163.2 GB/s의 최대 대역폭을 두 개의 CPU에서 같이 사용해야 한다.
패키징이나 그 구현 등의 방법이 조금 다르긴 하지만 현재 세 업체(AMD, Compaq, IBM)에서 프로세서를 glueless coupling하는 기술을 가지고 있다. 세 업체 모두 구현방식에서는 다수의 프로세서를 가지는 low-latency의 SMP 노드 구성을 가능하도록 하고 있어, 최근의 SMP 노드 기반 병렬 시스템 구축에 대한 경향을 촉진하고 있다.

2.8.5. IBM POWER5

앞서 언급했듯 POWER5는 얼마 있지 않아 POWER4+를 대신할 것이므로 이곳에서 POWER5의 특성에 대해 알아본다. 현재 POWER5는 가장 빠른 POWER4+(1.9 GHz)와 같은 클럭을 가지고 있다. POWER4(+)와 같이 POWER5도 하나의 칩에 두 개의 프로세서를 가지며, 이 CPU 코어들은 그림 10a에서 본 것과 유사한 구조를 가진다. 그러나, POWER4와 POWER5 사이에는 분명한 차이가 있다. POWER4+에서는 180 nm 공정을 POWER5에서는 130nm 공정을 사용해 칩의 집약도가 높은 POWER5에 더 많은 장치가 올라갈 수 있다. 우선 이웃하는 두 개 칩에 L2 캐시가 연결되고 L3 캐시는 L2 캐시와 직접적으로 연결된다. 두 캐시의 크기는 모두 POWER5가 POWER4보다 크다(POWER4의 L2와 L3는 각각 1.5 MB, 32 MB, POWER5의 L2와 L3는 각각 1.875 MB, 36 MB). 거기에 L3 캐시의 속도가 80 사이클(POWER4)에서 120 사이클(POWER5)로 증가되었다. 또한 각 캐시의 연관성에 있어서도 L1 캐시의 연관성이 2-way에서 4-way로 L2 캐시는 8-way에서 10-way로 L3 캐시는 8-way에서 12-way로 향상되었다. 또 다른 큰 차이점은 메모리에서 칩으로의 대역폭이 향상된 것이다. POWER4+에서 4 GB/s의 대역폭이 POWER5에서는 약 16 GB/s로 증가되었다.

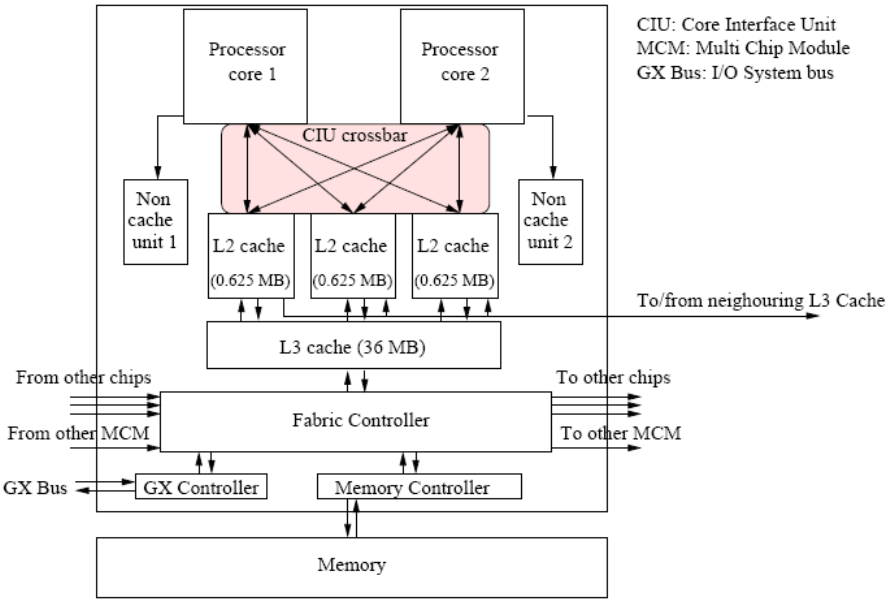


그림 11: Diagram of the IBM POWER5 chip layout

더 나은 캐시 특성이 규칙적인 데이터 접근에 대해 대기시간을 줄이게 된다. 고차 다항식의 계산과 행렬의 곱셈에서 POWER5는 최대 성능의 90% 혹은 그 이상을 보여주는데 반해 POWER4+에서는 65 ~ 75% 정도를 나타낸다. 규칙적이지 않은 데이터 접근을 하는 프로그램에 대해서도 성능 향상을 얻을 수 있는 POWER5의 또 다른 특성은 바로 Simultaneous Multithreading(SMT)이다. POWER5 CPU는 동시에 동작하는 두 개의 프로세스 스레드를 가질 수 있다. Functional 유닛들은 두 스레드의 어느 쪽으로부터든 해당 Functional 유닛을 사용하는 명령어를 받는다. 두 스레드는 모두 해당 Functional 유닛으로 꺼내어질 명령어 워드로 slot을 채울 수 있다. 이런 방식으로 Functional 유닛의 더 많은 부분이 실행되도록 할 수 있어 전체적인 효율을 높이게 된다. 그렇지만, 아주 규칙적인 계산을 하는 경우는 단일 스레드 모드가 더 나을 수 있다. 이것은 SMT 모드에서 두 개의 스레드가 캐시내의 entry를 두고 경쟁하게 되면, 규칙적 데이터 접근의 경우 캐시 thrashing이 발생하기 때문이다. 이런 측면에서 SMT는 보통의 다중 스레딩(multi-threading)과는 다르다. 다중 스레딩에서 어떤 원인으로 지연된 스레드는 멈춰지고 이 순간 실행 대기 중인 다른 프로세스의 스레드로 대체된다. 이때 시간이 소비되고, 이 소비시간은 이어지는 스레드에 의해 메워져야 한다. 이것은 두 번째 스레드가 적정 사이클(적어도 수백 사이클) 동안 실행되어야 한다는 것을 의미한다. SMT에는 이런 단점이 없지만, 두 스레드의 명령어 스케줄링이 아주 복잡하다.

2.8.6. IBM BlueGene Processor

사실상 이 프로세서는 변형된 PowerPC 400 프로세서로 IBM BlueGene 패밀리를 위해 특별히 만들어졌고 현재 700 MHz의 속도를 가지고 있다. 그 특별한 속성으로 인해 뒤에서

BlueGene 시스템을 설명할 때 이 프로세서에 대해 기술하도록 하겠다.

2.8.7. Intel Itanium 2

Itanium 2는 인텔의 2세대 IA-64 64비트 프로세서 패밀리를 대표한다. 이전의 Itanium은 2001년에 출시되었으나 그다지 많이 사용되지 않았는데, 주된 이유는 Itanium의 성능보다 2배나 향상된 Itanium 2가 곧이어 나왔기 때문이다. 처음 나온 Itanium 2 제품은 0.8 ~ 1.0 GHz의 속도를 보였으나 얼마 있지 않아 shrink 기술(code name Madison)을 적용해 클럭 수가 1.3 ~ 1.6 GHz에 이르는 현재의 Itanium 2가 나왔다.
Itanium 계열의 프로세서는 앞에서 다뤘던 RISC 칩들과는 다른 특성을 가지고 있다.

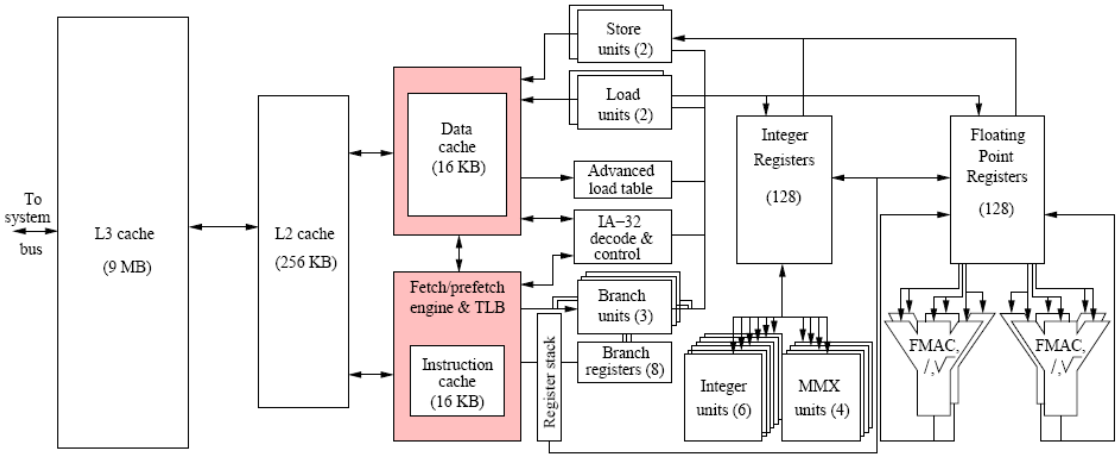


그림 12: Block diagram of the Intel Itanium 2.

이곳에서는 제일 큰 9 MB의 L3 캐시를 장착해 가장 빠른 Itanium 2에 대해 살펴본다. 그림 12에서 항상 실행 중에 있어야 하는 많은 Functional 유닛들을 볼 수 있다. 이 많은 Functional 유닛들이 항상 실행 중에 있도록 하기 위해 크기가 큰(128 비트) 명령어 워드를 사용한다. 이 명령어 워드는 3개의 41 비트 명령어와 명령어를 해석하고 조정하는 5비트 template 하나를 가진다. 이것은 약 10년 전에 시장에 나왔던 Very Large Instruction Word(VLIW) 시스템에서 사용하던 아이디어이다. 두 개의 load/store 유닛이 사이클당 두 개의 명령어 워드를 꺼내서 결과적으로 사이클당 여섯 개의 명령어가 처리된다. Itanium의 명령어 스케줄링은 RISC와는 다르게 프로그램 실행 중에 동적으로 진행되지 않고 컴파일러에 의해 처리되는데 이 역시도 VLIW 시스템과 같은 방식이다. VLIW-like 연산은 예측된(predicated) 실행으로 향상되었다. 일반적으로 분기 테스트의 결과를 기다려야 하는 명령어들의 실행을 예측된 실행을 통해 병렬로 실행하는 것이 가능하다. 이 향상된 VLIW 모드 연산을 인텔에서는 Explicit Parallel Instruction Computing (EPIC)이라고 부른다. load 명령어들을 옮기고 그곳에서 코드를 바꿔 load된 값을 분기나 저장에 앞서 사용할 수 있게 하는 것은 그 연산에 대한 유효성을 판단하는 것으로 가능하다. Advanced load를 확인하기

위해 Advanced Load Address Table(ALAT)에 그것들을 기록해 둔다. Advanced load에 의존하는 연산의 유효성이 검사될 때 ALAT가 검색되며, 이때 해당 entry가 없다면, 그 연산 체인은 무효가 되고 적절한 fix-up 코드가 실행된다. 이 코드는 컴파일 단계에서 만들어진 코드이고 이와 같은 추정 실행을 위해 제어 추정 하드웨어는 필요하지 않다. 이것이 어떤 순간 동시에 수행될 수 있는 많은 Functional 유닛들을 매우 복잡하게 한다.

그림 12에서 볼 수 있듯 Fused Multiply Accumulate(FMAC) 연산 수행이 가능한 4개의 부동소수 연산 유닛이 있다. 그렇지만, 이들 중 두 개는 Itanium 프로세서의 내부 표준인 82비트 정밀도를 완전히 사용하는 데 반해, 나머지 두 개는 32비트 정밀도 연산만 수행할 수 있다. 64 비트 정밀도 작업에서 1 GHz 클럭을 가지는 Itanium 프로세서는 4 Gflop/s의 이론 최대 성능을 보여준다. 32 비트 부동소수 연산에서는 최대성능이 2배가 된다. 1세대 Itanium에는 정수 연산과 정수나 문자 처리를 위한 4개의 정수 유닛이 있었는데 정수 성능이 그다지 뛰어나지 않았기 때문에 2개의 정수 유닛이 Itanium 2에 더 추가되었다. 그리고, 멀티미디어 연산처리를 위해 Intel Pentium 프로세서 계열에서 사용되는 MMX 유닛을 4개 가지고 있다. 또, Pentium 계열과의 호환성을 위해 IA-32 해석(decode) 유닛과 제어(control) 유닛을 특별히 가지고 있다.

정수와 부동소수를 위한 레지스터 파일은 각각 128 entry를 가지지만, 첫 32 entry만 고정되고 33 ~ 128 entry는 레지스터 스택으로 구현된다. 주 데이터 캐시와 명령어 캐시는 4-way 집합 연관성을 가지고 상대적으로 크기가 작아 각 16 KB 이다. 이것은 이전의 Itanium과 같다. 그러나 L1 캐시의 속도는 2배가 되어 데이터와 명령어는 매 클럭마다 레지스터로 전송될 수 있다. 게다가 L2 캐시는 96 KB에서 256 KB로 크기가 커졌고 8-way 집합 연관성을 가진다. 거기다, 9 MB에 이르는 L3 캐시가 칩 위로 옮겨졌다. 이와 같은 캐시 구조가 프로세서 코어로의 대역폭을 평균적으로 3배정도 향상시켰다. 이 사실은 클럭 수를 800 MHz에서 1.6 GHz로 높인 것보다 더 나은 성능 향상을 가져온다. 또한, 메모리와 캐시 사이의 대역폭도 3배 이상 증가되었다. 128 비트로 넓어진 버스는 400 MHz의 클럭으로 동작해서 전체적으로 6.4 GB/s의 대역폭을 가진다. 이전 Itanium은 2.1 GB/s의 대역폭을 가졌었다. 프론트사이드 버스가 더 커질 것이 아니기 때문에, 프로세서의 속도가 증가되면서 이 속도가 더 빨라지지 않을 것이고, 따라서 얼마 있지 않아서는 전체 성능에 방해가 될지도 모르겠다.

처음 Itanium의 발표가 차일피일 미뤄지면서 HPC 분야에서 그에 대한 관심이 많이 줄어들었었지만, 2002년 하반기에 Itanium 2가 발표된 이후 이 프로세서를 채택하는 비율이 빠르게 증가되었다. HP/Compaq 말고도, Bull, Fujitsu, NEC, SGI와 Unisys 등이 Itanium 2를 장착한 다중 프로세서 시스템을 공급하고 있다. HP/Compaq, Fujitsu, SGI 등에서 각각 채택해 사용하고 있는 Alpha, PA-RISC, SPARC, MIPS 등의 프로세서들은 머지않아 사라지게 될 것이다.

2.8.8. Intel Pentium 4/Xeon Nocona

Pentium 프로세서가 비록 통합된 병렬 시스템에서 사용되고 있지는 않지만 Beowulf 클러스터의 계산 노드에서와 같이 클러스터 분야에서 주요한 역할을 하고 있다. 그런 이유로 Pentium 프로세서에 대해 간략히 알아본다. Xeon 프로세서가 Pentium 4 보다 더 큰 캐시를 수용할 수 있어 다중 프로세서 환경에서는 대부분 Xeon 프로세서가 더 적합하다. 거의 대부분의 인텔 기반 클러스터에서는 2-프로세서 노드를 사용하므로 Xeon 프로세서가 가장 적절하다. 이곳에서 기술하는 차이점을 제외한다면 Pentium 4와 Xeon 아키텍처는 동일하다. 프로세서에 대해 부족한 기술적 정보를 제공하는 곳은 인텔이 유일하다. 이로 인해 아래 P4 프로세서의 그림은 여러 정보를 취합해 그린 대략적인 것이다.

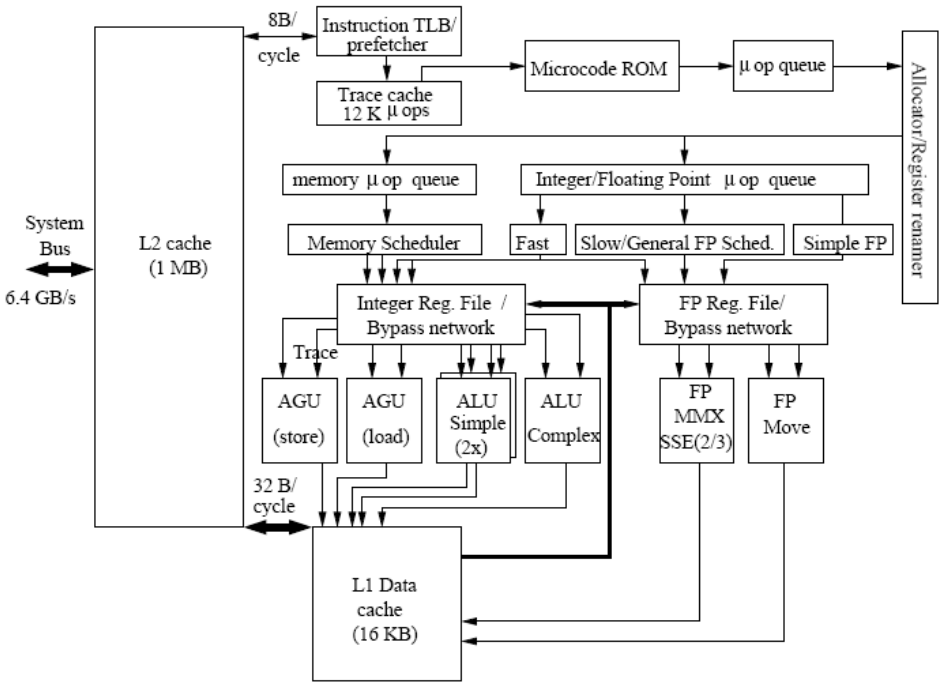


그림 13: Block diagram of Intel Xeon Nocona

진보된 90 nm 기술로 제작되었기에 큰 2차 캐시를 가지는 것과 더불어 여러 가지 부가적인 특성을 가지고 있는 Nocona 칩에 대해 알아 본다. Xeon은 초기의 Pentium 세대와 달리 많은 차이점을 가지고 있다. 프로세서의 성능을 높이기 위해 사용하는 두 가지 주된 방법은 클럭 수를 높이는 것과 사이클당 처리 명령어 회수(IPC)를 높이는 것이다. 이 두 가지 접근법은 서로 상충해서 IPC를 높이고자 할수록 그 칩은 더욱 복잡하게 된다. 칩이 더 복잡해지면 같은 클럭 사이클 동안 더 많은 일들이 정리되고 실행되어야 하므로 클럭 수에 나쁜 영향을 주게 된다. 아주 드물지만, 칩 설계자들이 클럭과 IPC를 같이 올리는 것에 성공하는 경우도 있다. 그렇지만 Pentium 4에서는 이와 같은 성공을 거두지 못했고, 그래서 인텔이 선택한 것은 IPC 감소(10 ~ 20%)를 감수하고 클럭 수를 높이는 것이었다(동일한 제작 기술

을 가지고 Pentium 3 대비 40% 이상 증가). 비록 프로세서에 다른 변화가 있었지만, 클럭 수 증가를 통해 전체적인 성능 향상을 얻고 있다. 현재 약 3.8 GHz에 이르는 높은 프로세서 클럭 수를 유지하기 위해서는 아주 깊은 명령어 파이프라인이 필요하다. Pentium 3의 명령어 파이프라인 깊이가 10인데 반해 P4에서는 그 깊이가 31단계에 이른다. 이것이 높은 클럭 수에 적합하기는 하지만, 분기 예측 실패와 같은 파이프라인 실패로 인한 성능 손실이 더 커지게 되는 단점이 있다. 그래서 인텔은 Branch Target Buffer의 크기를 0.5에서 4 KB로 높여 분기 예측 성능을 더 향상시켰다. Pentium 4에는 실행 추적 캐시 (execution trace cache)가 있다. 여기에는 이전 실행 추적에서 일부분 해석된 명령어를 저장해 두게 된다. 이렇게 저장된 명령어가 이후에 실행된다면, 결국 명령어 해석이 미리 처리되어버린 상황이라 명령어 파이프라인에 구멍(hole)이 생길 수 있다. 할당자는 해석된 마이크로 명령어들을 적절한 μop(micro operation) 큐에 넣는데, 하나는 메모리 연산을 위한 것이고 다른 것은 정수와 부동소수 연산을 위한 것이다.

두 개의 정수 Arithmetic/Logical Unit은 클럭속도의 두 배로 실행할 수 있도록 하기 위해 단순하게 구성된다. 거기에 한 사이클에 처리될 수 없는 복잡한 정수 연산을 위해 하나의 ALU를 두고 있다. 사이클마다 하나의 결과를 내는 부동소수 유닛이 단 하나 있다. 그러나 보통의 부동소수 유닛 이외에 Streaming SIMD Extensions 2와 3(SSE2/3) 명령어 집합을 실행하는 추가적인 유닛들이 더 있다. SSE2/3는 144개의 명령어를 가지는 명령어 집합으로 주로 멀티미디어와 3차원 가시화 프로그램을 위해 쓰인다. 이 유닛들을 위한 연산수의 길이는 128 비트이다. 인텔 컴파일러는 SSE2/3 유닛을 어드레스하는 기능을 가지는데 이를 통해 부동소수 연산 성능을 2배 정도 더 높일 수 있다.

Xeon Nocona는 일부 환경하에서 프로세서가 두 개의 스레드를 동시에 실행할 수 있는 Hyper-threading 기술을 자랑한다. 예를 들어 이 기능은 IF 분기의 예측실행을 위해 사용될 수 있다. 실험을 통해 다양한 코드에서 평균 약 30% 정도의 성능 향상을 얻을 수 있다고 알려졌지만, 실제로 3 ~ 5% 정도의 성능 증가를 볼 수 있다.

주 캐시는 최근의 표준에 비춰 아주 작았는데(8 KB) Prescott 제품에서 16 KB로 두 배가 되었다. 문제는 Functional 유닛으로 데이터를 보내는 latency가 이전에 2사이클이던 것이 3사이클로 증가된 것이다.

그러나, 역시 이전 프로세서와 가장 큰 차이점은 64비트 코드를 실행할 수 있다는 것이다. 이 때문에 AMD에서도 Opteron과 Athlon 프로세서에서 동일한 접근 방식을 써서 64 비트 지원을 하고 있다. 이 기술을 인텔에서는 Extended Memory 64 Technology(EM64T)라고 부른다. 이것은 명령어의 64 비트 버전이 실행되어야 하는지 신호로 알리기 위해 x86 명령어 집합의 명령어 워드에서 “사용되지 않는 비트”를 사용한다. 물론 64 비트 모드로 동작하기 위해 일부 추가적인 장치가 더 필요하며, 여기에는 8개의 새로운 general purpose registers(GPRs), SSE2/3를 지원하기 위한 8개의 새로운 레지스터, 그리고 64 비트 GPRs와 명령어 포인터 등이 포함된다.

Nocona 프로세서가 제공하는 모든 특성의 장점을 제대로 살릴 수 있기 위해서는 컴파일러

성능이 매우 중요하다. 인텔에서는 약 30% 이상의 성능 개선을 얻을 수 있다고 하는 데, 만약 그렇다면 이 프로세서를 기반으로 HPC 클러스터를 구축해 보는 것도 흥미로운 일이 될 것이다.

2.8.9. MIPS R16000

MIPS R1x000 시리즈 프로세서의 핵심은 맨 처음 R10000이 발표된 이후 변화한 것이 없다. 현재는 R16000이 SGI Origin3000 시리즈의 핵심으로 사용되고 있다. R16000은 이전의 R14000과 클럭 사이클 말고는 대부분 비슷하다. 현재 R16000의 클럭 수는 800 MHz로 HPC 시스템에서 사용 중인 모든 RISC 프로세서들 중에서 가장 낮은 수치를 가진다.

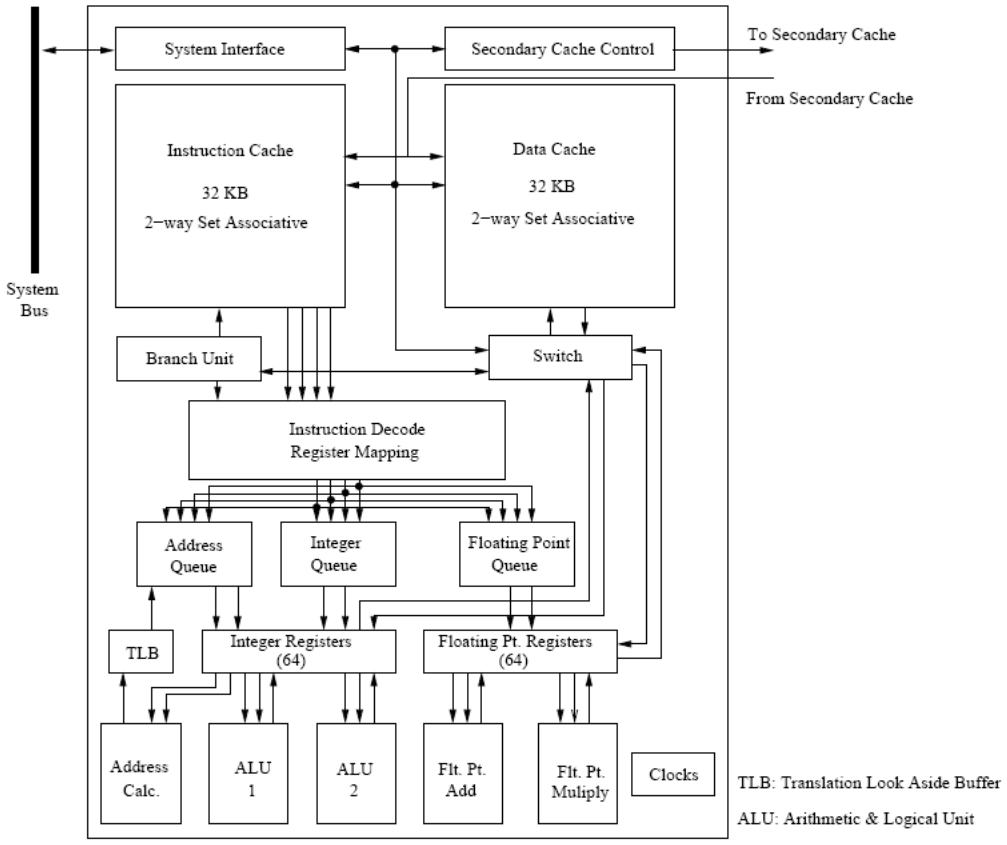


그림 14: Block diagram of the MIPS R16000 processor

R16000은 out-of-order와 추정 실행을 하는 오늘날 전형적인 RISC 프로세서를 대표한다. Compaq Alpha 프로세서와 같이 덧셈과 곱셈을 위한 독립적인 두 개의 부동소수 유닛이 있고 추가적으로 부동소수 나눗셈과 제곱근 계산을 수행하는 두 개의 유닛을 더 가진다(그림 14에는 표시되어 있지 않다.). 그러나 이 부가적인 부동소수 유닛은 파이프라인되어 있지 않으며 상대적으로 느린 20 ~ 30 사이클의 latency를 가진다. 모두 5개의 파이프라인된 functional 유닛이 있는데, 주소 계산과 데이터와 명령어 loading/storing을 수행하는 주소

계산 유닛, 일반적인 정수 계산을 위한 두 개의 ALU 유닛과 앞서 언급한 부동소수 덧셈과 곱셈을 수행하는 유닛 등이 그것이다.
L1 명령어 캐시와 데이터 캐시는 32 KB로 적절한 크기를 가지며, 2-way 집합연관성을 가진다. 이에 비해 2차 캐시는 매우 커서 16 MB에 이른다. 정수와 부동소수 레지스터 두 가지 모두 64 entry의 물리적 크기를 가지지만, 그 중 32개가 소프트웨어에 의해 접근가능하고 나머지 반은 레지스터 re-mapping을 위해 CPU에 의해 직접 제어된다.
MIPS R1x000 프로세서의 클럭 수는 대체로 낮은 편이다. 처음 R10000이 180 MHz의 클럭을 가졌던 반면, 새로운 R16000은 700 MHz를 가져 기간에 비해 클럭 수 증가 정도가 그다지 크지 않다. 초기 700 MHz 클럭 프로세서의 이론 최대 성능은 1.4 Gflop/s 이다. Fused multiply-add 기능이 없는 독립적인 부동소수 유닛 덕분에 최고성능에 가까운 결과를 얻을 수 있는 경우가 종종 있다. 앞선 칩들에 비해 개선된 부분들로는 우선 버스 속도가 100 MB/s에서 200 MB/s로 두 배가 되었고, R12000 프로세서에서 최대의 2/3정도를 얻을 수 있었던 L1 캐시 속도가 R14000부터는 최대 속도로 동작하게 되었다.
R16000 프로세서는 0.11 μm 기술로 제작되며, 현재 800 MHz의 클럭을 가지고 전력소비가 겨우 20 W에 달해 매우 낮은 수치를 보여준다. 작은 공간에 많은 프로세서를 설비할 수 있는 밀집도가 높은 시스템을 위해 SGI는 의도적으로 클럭 수를 낮게 하고 있다.
최근 Origin3000 시리즈를 Itanium 기반의 Altix 계열로 완전히 대체하면서 SGI가 MIPS 플랫폼을 버릴 것이라는 것은 분명해 보인다. 따라서 R16000을 이어 계획대로 2004년 중반에 R18000, R20000이 계속해서 나오게 될지 의심스럽고 실제로 그 실현성은 낮아 보인다.

2.8.10. Sun UltraSPARC IV

UltraSPARC IV는 UltraSPARC 계열의 4세대 프로세서이다. HP PA-RISC 8800과 IBM POWER4/5와 같이 Sun은 항상 하나의 칩에 두 개의 코어를 두고 있다. CPU 코어는 UltraSPARC III 프로세서와 조금 달라졌다. 현재 클럭 수 1.2 GHz로 0.13 μm CMOS 기술로 제작된다. UltraSPARC 프로세서는 워크스테이션부터 규모가 큰 Fire 3800-15K 서버에 이르기까지 모든 SUN 제품에서 사용된다. 아래 그림은 UltraSPARC IV 칩에 들어가는 프로세서 코어를 보여준다.

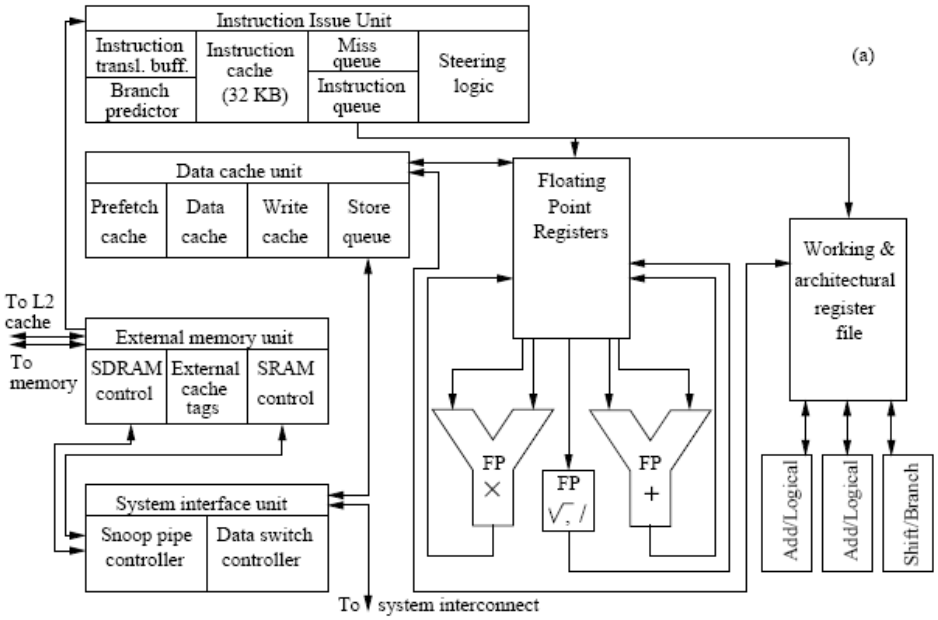


그림 15a: Block diagram of the UltraSPARC IV processor core

그림에서 볼 수 있듯 이 프로세서의 특징은 다양하고 크기가 매우 큰 캐시에 있다. Data Cache Unit(DCU)는 64 KB 크기의 4-way 집합 연관성 캐시와 더불어 2 KB 크기의 쓰기(write) 캐시와 선인출(pre-fetch) 캐시를 가진다. 선인출 캐시는 데이터 캐시와 독립적이고 적절하다고 판단될 때 데이터를 load할 수 있다. 쓰기 캐시는 L2 캐시로의 쓰기를 연기시켜 전체 캐시라인이 갱신되어야 할 때까지 각각의 바이트를 불필요하게 쓰기 하는 것을 막는다. Instruction Issue Unit(IIU)에는 SUN에서 Instruction Translation Buffer라고 하는 명령어 TLB와 더불어 32 KB 크기의 4-way 집합 연관 명령어 캐시가 있다. 또 IIU에는 이른바 실패 큐(miss queue)가 있다. 실패 큐에는 분기 예측이 실패한 경우 실행 유닛에서 즉시 사용할 명령어를 넣어 둔다. 분기 예측은 UltraSPARC III에서 완전히 정적이었고, 크기 때문에 파이프라인 처리되는 IIU에서 1 6KB의 테이블로 구현된다.

Integer Execution Unit(IEU)은 두 개의 Add/Logical Units과 하나의 분기 유닛을 가진다. 정수의 덧셈과 곱셈은 파이프라인 처리되지만 나눗셈은 그렇지 않다. 나눗셈은 ALUs에 파이프라인 부담을 주지 않는 Arithmetic Special Unit(그림에 나타나 있지는 않다.)에 의해 수행된다. 정수 레지스터 파일은 효과적으로 두 개로 나뉘어져 SUN에서는 Working과 Architectural Register File이라고 각각 부른다. 연산수가 접근되고 그 결과는 working 레지스터에 저장된다. 예외가 발생했을 때 작업(working) 레지스터에서 취소 처리되는 결과는 아키텍처 파일에 있는 것들로 겹쳐 쓰기된다. 오리지널 UltraSPARC III 디자인에 대하여 향상된 한 가지는 쓰기 캐시를 위한 해시 인덱싱이 추가된 것이다. 이것이 쓰기 실패 회수를 줄여서 저장이 필요한 결과에 대해 쓰기 저장 대역폭이 더 커지게 된다. 부동소수 유닛(FPU)은 덧셈과 곱셈을 파이프라인 처리하는 두 개의 독립된 유닛을 가지고, 20 ~ 25 사이

클 정도가 소비되는 부동소수 나눗셈과 제곱근 계산을 위해 파이프라인 처리되지 않는 유닛을 하나 더 가진다. 또 FPU에는 파이프라인 처리되는 adder와 multiplier를 일반적인 64 비트 계산들과 공유하는 그래픽 하드웨어가 있다(그림 15a에는 나타나 있지 않다.). 1.2 GHz 클럭을 가지는 칩의 이론 최대 성능은 프로세서 코어당 2.4 Gflop/s 이다. UltraSPARC IV 기술이 조금이라도 더 클럭 수를 높이는데 걸리는 시간을 줄일 수 있을 것으로 기대된다. UltraSPARC IV에서 FPU는 IEEE754 부동소수 오류에 대한 처리를 지원하는 하드웨어를 가지고 있어 향상된 성능을 보여준다. 이 오류는 적절히 처리되지 않으면 손실이 클 수 있다.

L2 캐시의 크기는 2x8 MB이고 각각은 프로세서 코어 중 하나를 대상으로 한다. L2 캐시라인은 데이터 contention을 줄이기 위해 512 바이트였던 것이 UltraSPARC IV에서는 128 바이트로 짧아졌다.

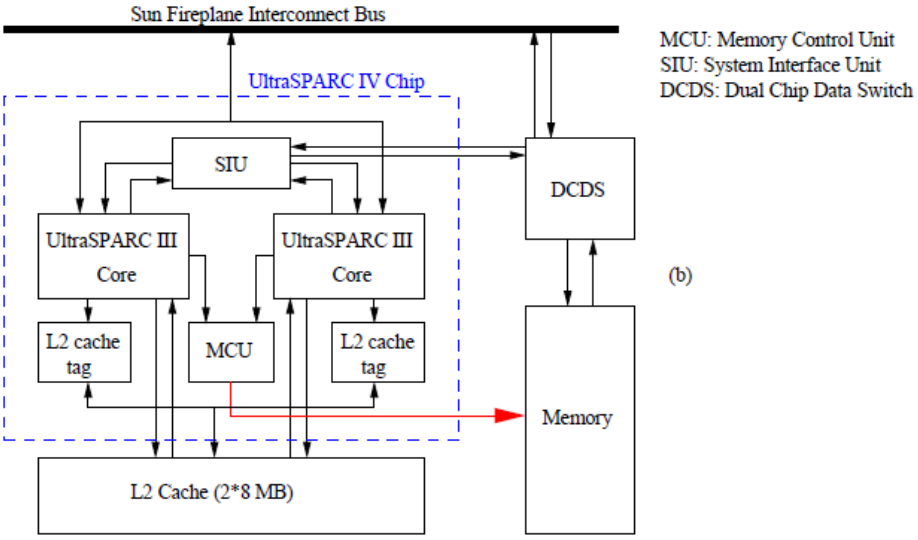


그림 15b: Chip layout of the UltraSPARC IV processor

위의 그림 15b에서 Memory Control Unit(MCU)와 L2 캐시 컨트롤러 그리고 L2 캐시 tags까지 칩에 포함되어 있다는 것을 분명히 볼 수 있다. 이것이 양쪽 메모리 레벨에서 접근하는 latency를 줄여준다. 게다가, System Interface Unit(SIU)와 통신하는 두 컨트롤러들 역시 칩 안에 있는데 이것은 SIU에 있는 스눕(snoop) 파이프 컨트롤러와 연결을 유지하기 위한 것이다. UltraSPARC 프로세서는 처음부터 멀티 프로세싱을 염두에 두고 개발되었고 스눕 컨트롤러는 캐시 일관성을 유지하기 위해 시스템 전체에서 전달되는 데이터를 감시(snooping)하게 된다.

UltraSPARC IV는 2004년 2월 이래로 많이 사용되고 있다. Sun은 하나의 칩에 두 개의 프로세서 코어를 가진 점과 하나의 실행 스레드가 그 둘 각각을 Chip Multithreading(CMT)으로 실행할 수 있음을 이야기 한다. 우리가 이해하고 있는 다중 스레딩은 프로세서 코어당

여러 개의 실행 스레드가 동작하는 것이어서 CMT는 보통의 다중 스레딩과는 다르다. 그런 점에서 CMT라는 용어는 혼란스러움을 준다.
SPARC 프로세서 계열이 앞으로 어떻게 될지 지금으로선 명확하지 않다. 한편으론 Fujitsu-Siemens와 힘을 합친 Sun이 Fujitsu-Siemens의 SPARC64 제품을 장착한 새로운 시스템을 내놓을 것으로 보인다. 표준 RISC 디자인을 따른다는 점에서 SPARC64 라인은 "고전적"이다. SPARC64는 2004년 말에 듀얼 코어 제품이 시장에 나올 것이다. 또 다른 한편으로, Sun은 2004년 4월까지 UltraSPARC V와 VI 프로세서를 내놓겠다는 계획을 보류시켰다. 이것은 8개 이상의 프로세서 코어를 가지고, 각 코어에서는 여러 실행 스레드를 처리할 수 있는 새로운 프로세서를 계획하고 있기 때문이다. Niagara로 이름 붙여진 이 프로세서는 현재 실험 단계에 있으며, 하나에 8개의 코어를 가지고 각 코어에서는 4개 스레드까지 실행이 가능하다. 아마도 2005년 하반기에 시장에 나올 수 있을 것으로 보인다. Sun은 "throughput processing"으로 이름 붙여진 이와 같은 종류의 프로세싱을 원하고 있다. 그리고 다음에는 더 많은 프로세서 코어와 코어당 더 많은 스레드 실행이 가능한 이른바 Rock 프로세서를 계획 중이다. 많은 수의 실행 스레드를 최적으로 관리하기가 어렵기 때문에 코어 내에 특별한 무엇인가가 도입될 것으로 보여진다.

2.9. Networks

빠른 프로세서와 더불어 프로세서와 프로세서를 연결하는 빠른 네트워크는 통합 병렬 시스템과 클러스터 모두에게 아주 중요한 요소다. 초창기 클러스터에서 프로세서 사이의 통신은 대부분 Ethernet을 사용했었고, 이로 인해 부족한 네트워크 대역폭과 높은 latency가 프로그램의 확장성에 좋지 않은 영향을 미쳤다.
이제 상황은 확 달라졌다. 놀랍게 성능이 좋아진 네트워크에 대해서 앞서 살펴본 프로세서와 같은 관점으로 차례차례 살펴 볼 것이다. 고성능 네트워크의 일부는 통합된 병렬 시스템에서도 적용되고 있다.
Gigabit Ethernet은 폭 넓은 범위에서 사용 가능하고 125 MB/s에 이르는 최대 대역폭을 가지고 어떤 식으로든 latency-bound하지 않은 일부 응용 프로그램에서 유용한 역할을 수행할 수 있다. 그러나, 100 μs 보다 작은 latency와 35 ~ 45 μs 정도의 in-switch latency를 가지고 네트워크 환경에 사용되기에는 많은 제한이 있다. 여기서는 IBM과 SGI의 그것처럼 특정 시스템을 위한 독점적인 네트워크로 시장에 나와있는 네트워크는 다루지 않을 것이다. 이와 같은 네트워크는 해당 시스템과 함께 다루도록 할 것이다. 네트워킹 분야는 워낙 빠르게 변하기 때문에 이곳에서 특정 네트워크에 대해 완전한 것을 다루지는 않을 것이다. 그보다는 HPC 분야를 훑어볼 때 만나게 될법한 주요 발전 부분에 대해 알아볼 것이다.
클러스터에서 사용하는 고속 네트워크를 복잡하게 하는 것은 노드와 노드 사이의 연결이다. 통합된 병렬 시스템에서 노드에 접근하는 것은 개별화(customizing)되어서 노드내의 내부

대역폭과 네트워크 대역폭이 잘 맞지만, 클러스터에서는 PC기반 노드에 있는 PCI 버스 연결을 이용해 구현해야 한다. 32 비트(33 MHz)에서 64 비트(66 MHz) 범위를 가지는 PCI 버스의 타입이 네트워크를 통해 데이터가 얼마나 빨리 노드 안팎으로 전송되는 가를 결정하며, 이는 곧 노드간 통신의 최대 대역폭을 결정하게 되는 것이다. 실제 사용 가능 대역폭은 110 ~ 480 MB/s 정도 이다. 1999년 PCI-X가 나온 이후, 처음 1 GB/s이던 대역폭이 PCI-X 2.0에 와서는 2에서 4 GB/s에 이르게 되었다. 클러스터의 노드로 사용되는 PC노드에서 PCI-X를 사용하는 것은 이제는 일상화되었다. 최근엔 PCI Express가 사용 가능하게 되었는데 이것은 데이터 레인당 200 MB/s의 대역폭을 제공한다. 데이터 레인은 1x, 2x, 4x, 8x, 12x, 16x, 32x 다중 데이터 레인이 지원되어 어떤 통신 네트워크 업체의 호스트 버스 어댑터와도 충분히 연결될 수 있다. 그래서 다음부터 기술하는 네트워크들에 대해 종종 다른 대역폭이 인용될지 모른다. 그것은 대역폭이 도달할 수 있는 PCI 버스 타입과 지원 칩셋에 따라 다르기 때문이다. 따라서 참조되는 속도는 항상 호스트 노드의 PCI 버스가 충분히 넓고 빠르다고 가정한 것이다.

2.9.1. Infiniband

Infiniband가 노드간 네트워크로 널리 사용되는데 그다지 많은 시간이 걸리지 않았다. 2001년 6월에 스펙이 나왔고, 2002년부터 많은 업체들이 Infiniband 표준에 기반하는 제품을 생산하기 시작했다. Infiniband는 다양한 시스템 구성요소를 하나의 시스템으로 연결하도록 하고 있다. Infiniband 구조는 Host Channel Adapters(HCAs)를 거쳐 프로세서간 네트워크에 사용될 수 있고, I/O 서브시스템들의 연결이나 Gigabit Ethernet 스위치 같은 다중 프로토콜 스위치와의 연결 등에 사용될 수 있다. 이런 다양성으로 인해 시장에서는 프로세서간의 네트워크에만 제한시켜 사용하지 않았으며, 이것이 많은 양의 소비를 불러일으켜 상대적으로 Infiniband의 가격이 낮아지게 되었다. Infiniband는 대역폭이 높은 다중 연결을 위해서 사용할 수 있다. Infiniband 표준에서는 구리선과 광 섬유 연결 모두에 대한 제품 정의를 하고 있고, 스위치와 라우터 특성을 정의하고 있다. 또한, 패킷 내에서 메시지를 쪼개고 다시 모으는 방법에서부터 라우팅, 우선순위결정(prioritizing)과 오류 처리(error handling) 등이 모두 표준에 기술되어 있다. 이와 같은 특성들이 Infiniband를 특정 기술에 대해 독립적으로 만들고 있고, 그 완전성으로 인해 그 위에 MPI와 같은 통신 라이브러리를 구축하는 데 있어 좋은 기반이 된다.

Infiniband는 시스템과 연결되는 커넥터로 Host Channel Adapters (HCAs)와 Target Channel Adapters(TCAs) 이렇게 두 가지 형태를 인식한다. 앞서 언급된 HCA가 프로세서 사이의 통신에 사용되는 커넥터로 많이 사용되는 반면에, TCAs는 전형적으로 I/O 서브시스템과의 연결에 사용된다. Infiniband는 2.5 GB/s(312.5MB/s)의 기본적인 연결속도와 이것의 4배(1.25 GB/s), 12배(3.75 GB/s)에 이르는 연결속도를 정의하고 있다. 또 HCAs와 TCAs는 독립적인 다중 포트를 가질 수 있어 더 높은 속도와 신뢰도를 보장한다.

하나의 HCA/TCA로부터 다른 HCA/TCA로 보내는 메시지는 Remote Memory Direct

Access (RDMA)를 기반으로 해서 전송될 수 있는데, 이것은 하나의 HCA/TCA가 다른 HCA/TCA의 메모리로 읽기/쓰기가 허용된다는 것이다. 이로 인해 일단 권한과 쓰기/읽기 위치가 정해지면 매우 빠른 전송이 가능해 진다. HCA/TCA와 더불어, 포트 하나가 128비트 헤더를 가지는 메시지를 제공한다. 헤더는 IPv6를 따르고 cut-through wormhole 라우팅을 거쳐 목적지로 전송하는 데 직접 사용된다. cut-through wormhole 라우팅은 각 스위칭 단계에서 다음 단계로 보내는 라우팅이 해석되고 전송된다. 32 바이트의 짧은 메시지들은 제어 메시지들에 대한 협상 시간(negotiation time)을 줄이기 위해 제어 메시지에 포함될 수 있다.

HPC를 위한 Infiniband 스위치는 8 ~ 128 포트와 1.25 GB/s의 속도를 제공한다. 이 스위치는 어떤 형태의 토폴로지로도 설정 가능하지만 실제로는 fat tree 토폴로지가 많은 경우에 있어 선호되고 있다. 실제속도가 스펙상의 속도에 얼마나 근접하게 되느냐는 Infiniband의 상위에 위치하는 MPI 구현에 따라 달라지는 것이 분명하다. Infiniband 기반의 클러스터에서 Ping-Pong 실험을 한 결과 약 850 MB/s의 대역폭을 보였고 크기가 작은 메시지에 대해 MPI latency는 7 μs보다 작은 결과를 보였다. 전형적으로 in-switch latency는 약 200 ns 정도가 된다.

현재 시장 점유율이 가장 높은 Myrinet과 비교했을 때 포트당 가격은 여전히 조금 높다. 그러나 Infiniband가 점유율을 높이면서 가격은 많이 떨어질 것이고 그렇게 되면 Myrinet의 강력한 경쟁 상대가 될 수 있을 것이다.

현재 PCI Express의 가용성으로, 12-wide Infiniband 연결이 기술적으로 장점을 가지고 있다. 그렇지만, 그것이 상업적으로 어떨지는 좀더 지켜 봐야 하겠다.

2.9.2. Myrinet

Myrinet은 현재 고속 클러스터 네트워크 시장에서 가장 많이 사용되고 있다. Myrinet을 판매하는 Myricom사는 클러스터 노드를 연결하는 Ethernet을 대체할 Myrinet 구현제품을 가지고 1994년에 시작되었다[25]. 그 당시 약 100 MB/s에 이르는 상대적으로 높은 대역폭을 가지고 있었고 다른 무엇보다 완전히 user space에서 구동된다는 장점을 가지고 있었다. 완전히 user space에서 동작함으로써 운영체제의 간섭을 피해 이로 인한 성능 손실을 막을 수 있었고 이로 인해 크기가 작은 메시지에 대한 latency를 약 10 ~ 15 μs까지 낮출 수 있었다. Latency와 대역폭은 그 당시 Convex, IBM, SGI 등에서 생산되던 통합 병렬 시스템의 고유 네트워크와 비교해 상당히 좋았다. 비록 Myrinet을 사용하기 위한 비용이 무시할 수 없는 정도지만, 많은 경우에 있어 Ethernet으로 연결된 시스템이나 혹은 더 많은 비용이 드는 통합 병렬 시스템에게 있어서도 Myrinet으로의 이전은 해볼 만한 것으로 평가되었다.

지속적으로 하드웨어와 소프트웨어가 업그레이드되고 개선되면서 클러스터 구성에 사용되는 네트워크로 Myrinet이 많이 선택되었고 최근까지도 빠르고 latency가 낮은 네트워크로 Myrinet을 대신할 만한 것을 찾기 어려웠다.

Infiniband처럼 Myrinet에서도 효율적인 네트워크 사용을 위해 cut-through 라우팅을 사용

한다. 그리고 마찬가지로 다른 호스트 어댑터 카드(Lanai 카드라고 함)의 원격 메모리로 쓰기/읽기를 위해 RDMA를 사용한다. 이 카드들은 그들이 부착된 호스트의 PCI-X 버스와 인터페이스하고 있다. 최신의 Myrinet 제품은 신호 전달자(signal carrier)로 fiber만 사용한다. 이것이 연결에서의 flexibility를 높여주고 신호의 속도에 많은 headroom을 준다. 그러나 fiber 케이블과 커넥터는 깨지기 쉬워 클러스터 노드에 손상을 주기 쉽다는 단점이 있다.
Myrinet은 8 ~ 256 포트 기성품 스위치를 제공한다. 8 포트와 16 포트 스위치는 풀 크로스바이다. 이 스위치들을 가지고 Clos 네트워크 토폴로지를 이용해 모든 대형 네트워크들을 구성하는 것이 원칙이다. 아래 그림은 64 포트 시스템에 대한 예를 보여준다.

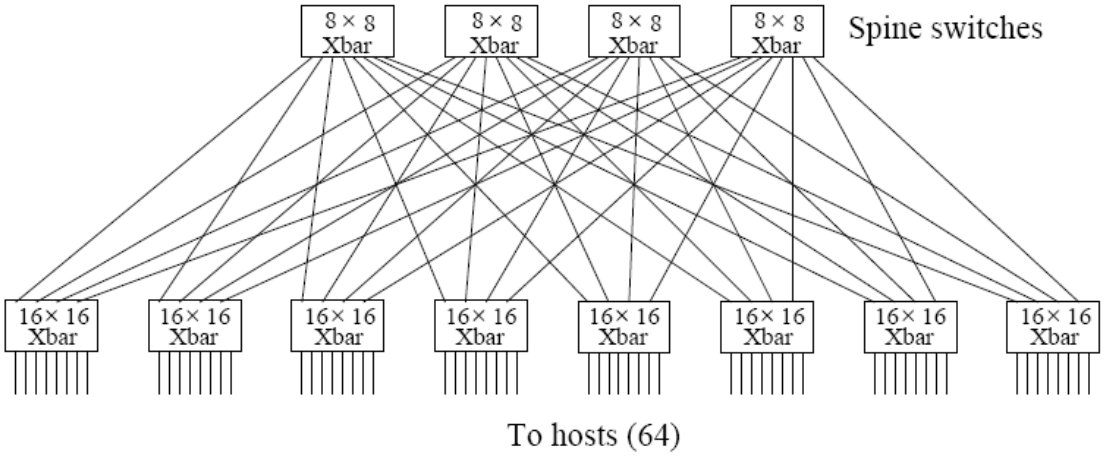


그림 16: An 8x16 Clos network using 8 and16 port crossbar switches to connect 64 processors.

Clos 네트워크는 종점의 최대 bisectional 대역폭을 가지는 로그함수적 네트워크이다. 16x16 크로스바 스위치의 4개 포트는 사용되지 않지만 다른 설정은 더 많은 스위치나 더 많은 연결 또는 둘 모두를 필요로 한다.
Myricom사에서 제공하는 네트워크 벤치마크 측정치[24]를 보면 최신의 Myrinet2000 스위치가 아주 좋은 결과를 보여주고 있다. Ping-Pong 실험에서 대역폭이 약 230 MB/s, Myricom사에서 네크워크와 같이 배포하는 MPI 버전을 이용한 결과 작은 메시지에 대해 latency가 7 μs 정도이다.

2.9.3. QsNet

1990년대 Computer Surface 1과 2로 불리던 병렬 시스템을 생산하던 영국의 Meiko라는 회사로 시작된 Quadrics사의 주력이자 유일한 생산품이 QsNet이다. 당시 CS-2에서는 아주 훌륭한 로그함수적 네트워크를 사용했었고, Meiko가 문을 닫은 뒤 이 네트워크는 독립적인 상품으로 만들어 졌다. 이후 네트워크 부분은 이탈리아 회사인 Alenia에서 인수했고,

지금은 Quadrics라는 독립회사 체제로 QsNet이 개발되고 있다. QsNet의 성공은 Digital/Compaq이 자신들의 대형 HPC 시스템인 AlphaServer SC 라인을 위한 네트워크로 QsNet을 사용하면서 시작되었다. 이 시스템은 Pittsburgh Supercomputing Center와 French CAE 등과 같은 곳에 대형으로 제작되어 도입되었다. QsNet은 매우 빠르고 신뢰성 있는 네트워크로 인정받았고 이후 약 2년 동안 클러스터 시스템에도 도입되었다. Infiniband와 Myrinet이 그런 것처럼 QsNet도 두 부분을 효과적으로 가진다. Infiniband의 Host Bus Adapters 또는 Myrinet의 Lanai 인터페이스 카드에 해당하는 ELAN 인터페이스 카드와 Infiniband 스위치/라우터 또는 Myrinet 스위치에 해당하는 Elite 스위치가 그것이다. 토폴로지는 대부분의 Infiniband 스위치처럼 (quaternary) fat tree를 사용한다(그림 5b 참조). ELAN 카드는 호스트 컴퓨터의 PCI-X 포트와 인터페이스 한다.

클러스터를 위한 기성품 Elite 스위치는 16 포트와 128 포트를 가지는 것 두 가지만 있다. 물론 포트 수가 다른 스위치를 같이 사용할 수는 있지만 속도와 간결성 측면에서 부담이 된다. 다른 스위치들과의 차이점은 연결당 두 개의 가상 양방향 채널(virtual bi-directional channels)을 제공하는 것이다. 2003년 말 이후 Quadrics사는 2 세대 QsNet인 QsNet II를 판매하고 있다. 이것은 이전의 QsNet과 구조와 프로토콜 등이 유사하지만 속도가 매우 빠르다. MPI Ping-Pong 실험에서 QsNet이 300MB/s의 속도를 보인 반면[30], QsNet II는 1.3 GB/s의 연결속도를 보여준다. 이렇게 빠른 연결속도를 가지는 경우, PCI-X 버스 속도가 전체 성능에 영향을 줄 수 있는데 QsNet II에서 PCI-X 버스 속도는 약 900 MB/s 이상의 속도를 가진다. 그러나, PCI Express가 출현하면서 PCI가 병목이 되는 문제는 해결되었고 작은 메시지에 대한 latency도 약 5 μs에서 3 μs으로 개선되었다. 거기에다 메시지 패킷의 균등한 분배에 크게 기여하는 두 개의 priority 단계를 스위치가 지원하고 있다. 이것은 Myrinet보다는 많지만 Infiniband의 16 priority 단계보다는 작은 것이다. In-switch latency는 약 35 ns 정도로 매우 낮다.

Infiniband처럼 QsNet(II)도 ELAN 카드상에서 원격 메모리 영역으로(또는 영역으로부터) 읽기/쓰기를 할 수 있는 RDMA 기능을 가진다. 그러나 QsNet에서는 이것이 호스트 프로세서 자체의 메모리 영역으로 확장될 수 있다. 그래서 원리상으로는 QsNet으로 연결된 시스템을 가상 공유메모리 시스템으로 볼 수 있다. 그렇지만, 어떤 시스템(통합 병렬 시스템 또는 클러스터)에서도 아직까지 현실화 되지는 않았다. 그럼에도 불구하고 ThreadMarks[1]처럼 느린 메모리 페이지 기반의 가상 공유 메모리 시스템을 대체할 수 있을 것으로 기대된다. get과 put 연산을 이용해 일방통신을 할 수 있는 Cray 스타일의 shmem 라이브러리가 제공된다.

2.9.4. SCI

이곳에서 알아보고 있는 네트워크들 중에서 가장 오래된 SCI는 Scalable Coherent Interface의 약자이다. SCI는 1992년 10월에 IEEE/ANSI 표준이 되었다. 그 당시 버스 기반의 다중 프로세서 시스템이 가진 한계를 극복하기 위한 해법을 찾기 위해 여러 업체와 대

학들이 힘을 모아 노력한 결과로 나온 것이 SCI이다.
SCI 설계의 주요 관심사였던 버스의 한계를 피하기 위해 SCI에서 선택한 방법은 호스트와의 연결을 링 구조로 하는 것이었다. 사실 여러 쌍의 링들은 송신과 수신 호스트 사이에 역신호를 가능하게 한다. 이것은 잡음 간섭(noise interference)없이 매우 높은 신호율(signal rate)이 가능하도록 신호는 지속적인 흐름을 유지해야 한다는 SCI의 설계 원리 때문이다. 이러한 설계 원리의 결론으로 링은 항상 작동 중 이어야 하고, 만약 전송되어야 할 메시지가 없을 때는 길이가 0인 payload 메시지를 보내게 되는 것이다. 패킷 내의 정보 길이는 0, 16, 64, 256 바이트로 그 크기가 고정되어 있으며 헤더의 크기는 16 바이트 또는 32 바이트 이다. 한편으로, 패킷은 2바이트 크기의 오류 수정 CRC 코드에 의해 차단된다. 패킷의 끝에 오류 수정 코드를 가짐으로써 데이터가 깨졌는지 아닌지를 바로 알 수 있고 이에 따라 즉각적인 조치가 가능해진다. 제한된 패킷 포맷은 패킷에 대한 빠른 수신과 검사를 가능하게 한다.
SCI의 주요 특성 중 한가지는 연결하는 프로세서들의 캐시를 일관성 있게 유지한다는 것이다. 이전의 버스 기반 다중 프로세서 시스템에서는 스누피 버스 프로토콜을 이용하여 프로세서의 캐시를 최신으로 유지하였다. 이런 버스를 대체하기 위한 SCI는 자연적으로 캐시 일관성을 보장하는 특성을 가지게 된 것이다. 그래서, QsNet처럼 SCI를 이용한 가상 공유 메모리 구현이 가능하다. 그렇지만 QsNet과는 다르게 SCI를 이용한 실제 가상 공유 메모리 구현이 진행되어 오고 있으며 Data General이나 이전의 Convex Exemplar 등에서 노드 연결을 위해 SCI를 이용했었다. 현재는 이전 Sequent사(지금은 IBM으로 넘어갔다.)의 NUMA-Q 시스템에서 메모리 연결을 위해 SCI를 사용한다. 클러스터에서 SCI는 항상 노드간 네트워크로 사용되는데, 이때 네트워크는 1-D, 2-D, 또는 3-D 토러스로 연결된다. 이것은 SCI의 링 구조 때문이다.

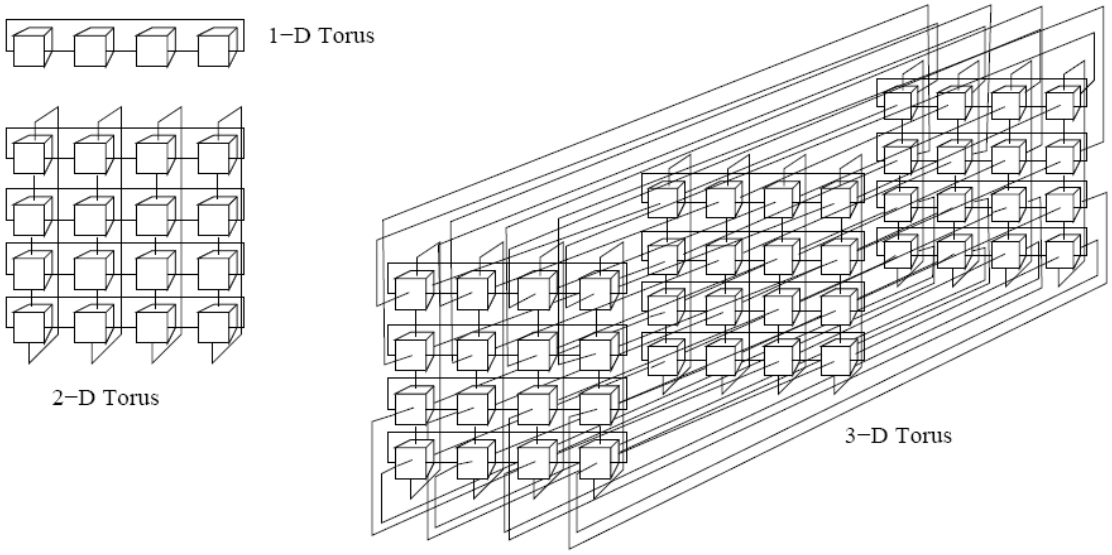


그림 17: SCI networks arranged as 1-D, 2-D, and 3-D toruses

토러스 네트워크는 fat tree나 Clos 같은 다른 네트워크에 비해 위험요소를 갖고 있다. 그 중 하나는 노드 어댑터 하나가 실패되면 그 노드가 있는 링을 공유하는 모든 노드가 같이 죽어 버리는 것이다. 그래서 SCI 업체 중 한 곳에서는 그와 같은 상황에서 최소한의 노드만 불능이 되도록 토러스를 재설정하는 소프트웨어를 제공하고 있다(Dolphin 네트워크). 거기에다 토러스 토폴로지로 인해 클러스터에 임의의 수만큼의 노드를 빼거나 추가하는 것이 불가능하다는 단점도 있다.

SCI 기반 클러스터에 대해 보고된 대역폭은 MPI의 Ping-Pong 실험에서 약 320 MB/s에 이른다. 작은 메시지에 대한 latency는 1 ~ 2 μs로 아주 작다.

3. Recount of the (almost) available systems

앞서 언급된 것으로 현재 시장에 나와있거나 6개월 내에 나오게 되는 모든 형태의 시스템들에 대해 이곳에서 보다 자세히 알아본다. Cray 시스템들 중에서 X1과 XD1은 아주 다른 구조를 가지고 있다. 이처럼 한 업체에서 한 가지 형태 이상의 시스템을 시장에 내놓았다면 이들을 따로따로 구분해서 알아 볼 것이다.
현재 Compaq은 HP로 완전히 합병되었다. 그래서 Compaq에 뿌리를 둔 Alpha 기반의 생산라인과 HP에서 시작된 PA/RISC 기반의 라인을 구별하기 위해 초기 Compaq 시스템들은 "HP/Compaq" 부분으로 넣어 두었다.

3.1. System descriptions

시스템 나열은 알파벳 순이고, "Machine type" 사항은 앞에서 논의된 Processor Array, ccNUMA, 등과 같은 시스템 타입 특성을 나타내는 것이다.

3.1.1. The Bull NovaScale

Machine type: ccNUMA system.
Models: NovaScale 5080, 5160.
Operating system: Linux, WindowsServer 2003, GCOS 8
Connection structure: Full crossbar.
Compilers: Intel's Fortran 95, C(++).
Vendors information Web page: www.bull.com/novascale/
Year of introduction: 2002.

System parameters:

Model	NovaScale 5080	NovaScale 5160
Clock cycle	1.5 GHz	1.5 GHz
Theor. peak performance	48 Gflop/s	96 Gflop/s
No. of processors	8	16
Communication bandwidth		
Point-to-point	6.4 GB/s	6.4 GB/s
Aggregate peak	12.8 GB/s	25.6 GB/s

Remarks:
Itanium 2의 효용성이 전통적으로 HPC 사업을 하지 않던 일부 업체들을 이 분야에 뛰어들게 하고 있다. 노드 수가 16개에 이르는 NovaScale ccNUMA SMPs를 시장에 내놓은 Bull

이 그런 업체들 중 하나이다. NovaScale 시스템은 표준 인텔 Quad Building Blocks(QBBs)로 구성되고 각각의 QBB는 Itanium 2 프로세서 4개와 메모리 일부로 이루어진다. 이 QBB들은 차례대로 FAME Scalability Switch(FSS)에 의해 연결된다. Bull 고유의 이 스위치는 25.6 GB/s의 aggregate 대역폭을 제공한다. 신뢰성을 높이기 위해 NovaScale 5160에서는 2 개의 FSS를 가지고 있으며, 이로 인해 QBB와 스위치 사이의 연결이나 스위치와 스위치 사이의 연결이 실패했을 때도 시스템이 정상적으로 동작한다. 각 FSS는 8개의 포트를 가지며 이들 중 6개만이 5160 시스템 하나에 포함된다. 나머지 포트를 이용해 두 개의 5160 시스템을 하나로 묶을 수 있고 이것이 32개 프로세서를 가지는 하나의 ccNUMA 시스템이 된다. 이렇게 연결(coupling)된 시스템들을 QsNet II로 묶어 보다 큰 시스템을 구성할 수 있다. Bull사에서는 매우 효율적이라고 알려진 MPI 구현을 자체적으로 제공하고 있다.
NovaScale 시스템이 가지는 차별화된 특성은 분할(partition)이 가능해서 이를 통해 서로 다른 노드에서 서로 다른 운영체제를 구동할 수 있다는 것이다. 이와 같은 분할은 동적으로 재분할 될 수도 있다. 이런 특성이 특별히 HPC 사용자를 끌어들이는 것은 아닐지라도 Bull 고유의 GCOS 운영체제를 사용하는 고객을 아직 확보하고 있듯이 다른 분야에서 관심을 끌 수도 있을 것이다.
4개의 프로세서를 가진 좀 더 작은 시스템인 NovaScale 4040은 departmental 서버로 사용할 수 있다. Bull이 Itanium 2를 사용하면서 자동적으로 인텔의 Fortran 95와 C 컴파일러도 사용할 수 있게 되었다. Bull의 문서에는 사용 가능한 다른 HPC 소프트웨어에 관한 정보는 없지만, Itanium 2 플랫폼에 맞는 모든 third-party 소프트웨어를 사용할 수 있을 것이다.

Measured Performance:
1.3 GHz 프로세서 16 개를 장착한 NovaScale 5160에 대해 EuroBen Benchmark를 이용한 다소 광범위한 벤치마크 실험이 2004년 봄에 수행되었다. EuroBen Benchmark를 이용한 dense matrix-vector 곱셈의 MPI 버전이 16개의 프로세서를 사용해 13.3 Gflop/s로 나왔다. 크기 N=1000인 dense linear system과 크기 N=65356인 1-D FFT를 풀이하는 데 관측된 속도는 3.3 ~ 3.4 Gflop/s였다.

3.1.2. C-DAC PARAM Padma

Machine type: RISC-based Distributed-memory multi-processor.
Models: C-DAC PARAM Padma.
Operating system: AIX(IBM's UNIX flavour), Linux.
Connection structure: Clos network.
Compilers: Fortran 77/90, C, C++.
Vendors information Web page: www.cdac.in/html/parampma.asp
Year of introduction: 2003.

System parameters:

Model	PARAM Padma
Clock cycle	1 GHz
Theor. peak performance	
Per Proc.	4 Gflop/s
Maximal	1.13 Tflop/s
Memory	500 GB
No. of processors	248
Communication bandwidth	
Aggregate	4 GB/s
Point-to-point	312 MB/s

Remarks:

인도의 C-DAC에서 만든 최신 시스템 PARAM Padma는 4개 프로세서를 가지는 54개의 SMP 노드와 32개 프로세서를 가지는 하나의 노드로 그 구성이 비대칭적이다. 모든 노드에서는 1 GHz IBM POWER4 프로세서를 사용한다. C-DAC 고유의 PARAMnet-II를 연결 네트워크로 사용하며 이것은 2.5 Gb/s(312 MB/s)의 최대 대역폭을 가지고 짧은 메시지에 대해 약 10 μs의 latency를 제공한다. 네트워크는 16 포트를 가지는 PARAMnet-II 스위치로 구축되고 Myrinet에서 사용된 구조와 매우 유사한 Clos64 토폴로지를 가진다. 이 네트워크를 이용한 MPI 실험 결과는 없다.

C-DAC은 이미 오랫동안 병렬 시스템을 구축해 왔고, 그에 맞는 고유의 소프트웨어를 항상 제공해 왔다. 그래서 Padma 시스템에는 그에 맞는 Fortran90, C(++), MPI와 Parallel File System이 있다.

Measured Performances:

이론 성능이 992Gflop/s에 이르는 62노드 Padma는 크기 N=224000인 linear system에 대한 HPC Linpack Benchmark에서 532Gflop/s(효율 53.6%)에 이르는 성능을 보였다.

3.1.3. The Cray MTA-2

Machine type: Distributed-memory multi-processor.

Models: MTA-2x, x = 16, 32, …, 256

Operating system: Unix BSD4.4 + proprietary micro kernel.

Compilers: Fortran 77/90, ANSI C, C++.

Vendors information Web page: www.cray.com/products/systems/mta/

Year of introduction: 2001.

System parameters:

Model	MTA-x
Clock cycle	—
Theor. peak performance	
Per Proc. (64-bits)	≥ 750 Mflop/s
Maximal (64-bits)	≥ 192 Gflop/s
Main Memory	≤ 1 TB
No. of processors	16{256

Remarks:

Cray MTA에 대한 데이터가 비밀에 부쳐져 있어 MTA 시스템의 정확한 최대 속도는 클럭으로 주어질 수 없다. 최대 성능의 하한 정도만이 일부 알려져 있다. 200 MHz와 227 MHz 클럭을 가지는 시스템이 있지만 227 MHz를 최대 클럭 수로 할 수 없는 이유는, 이것이 Cray 문서에 명시된 속도의 약 1/3정도 밖에 되지 않기 때문이다.

아키텍처 특성을 보자면, MTA의 메모리는 물리적으로 분산되어있지만 시스템은 공유 메모리 시스템(non-uniform access time)으로 강조되어 제안되고 있다. 메모리 참조에 소비되는 latency는 다중 스레딩으로 보완된다. 다중 스레딩은 여러 개의 프로그램 스레드가 어떤 순간에도 실행되도록 하는 것이다. 예를 들자면, load 명령어가 메모리 latency로 인해 처리될 수 없을 때 해당 스레드는 이 명령어가 지연되도록 요청하게 되고 명령어가 처리될 수 있는 다른 스레드가 실행되도록 교환이 일어나는 것이다. 프로그램 스레드 사이의 교환에 1 사이클이 걸린다. 128개 명령어 흐름과 8개의 메모리 참조가 앞선 결과를 기다리지 않고 issue될 수 있어서 1024 사이클의 latency를 줄이는 효과가 있다. 지연된 참조는 retry pool로부터 재시도된다. 이와 유사하게 동작하는 구조를 Stern Computing Systems의 SSP 시스템에서도 볼 수 있다.

연결 네트워크는 p개의 프로세서를 가지는 3-D cube로 연결된다. 이로서 모든 노드는 6 개 중 4개의 이웃과 연결된다. p개 프로세서 시스템은 평균 $2.25p^{1/3}$ 사이클, 최악의 경우 $4.5p^{1/3}$ 사이클의 latency를 가진다. 모든 노드에는 I/O포트가 하나씩 있다. 각 네트워크 포트는 사이클당 64비트 워드 하나를 송신과 수신할 수 있고 포트당 대역폭은 5.33 GB/s에 이른다. 실패가 감지되면 네트워크 포트들은 시스템의 인터럽트 없이 우회될 수 있다.

MTA는 예전의 Fortran 프로그램들을 당연히 실행시킬 수 있어야 한다. 이들 프로그램에서 다중 스레드를 이용할 수 있는 부분이 발견되면 바로 병렬성을 이용할 수 있지만 프로그램의 병렬성을 명시적으로 제어하고 알려진 데이터 지역성을 이용하는 것이 더 나을 수도 있다. 이를 위해 MTA에서는 정의된 스레드 그룹에 대한 synchronization, barrier, reduction 연산 등을 포함하는 라이브러리 루틴을 제공하고 있다. 한 프로그램에서 제어된 병렬성과 제어되지 않은 병렬성은 자유롭게 섞어 사용할 수 있다. 게다가 각 변수는 별다른 부하없이

병렬성과 동기화를 제어하는 데 사용할 수 있는 full/empty 비트를 하나 가진다. DNA sequence alignment에서 이런 특성을 흥미롭게 사용하였다.
28개의 프로세서(보통 32개)를 가진 첫 MTA 시스템은 2002년 미국의 Naval Research Laboratory에 납품되었다.

Measured Performance:
Cray가 San Diego Supercomputing Center에 납품 설치한 16 프로세서 시스템은 4.4 ns(처음 계획은 3 ns)의 클럭 사이클로 가동되고 프로세서 하나의 최대 성능은 450 Mflop/s이다. EuroBen Benchmark에서는 800 matrix-vector 곱에 대해 388Mflop/s의 결과로 86%의 효율을 보여 주었다. 성분이 백만 개에 이르는 1-D FFT에 대해 프로세서 한 개로 106 Mflop/s의 속도가 나왔다. 프로세서 네 개로도 이와 비슷한 결과가 나왔는데 이것은 병렬 스레드 효용성이 충분하지 못해서이다.
NAS parallel benchmark[27]를 이용해 얻은 제한된 성능데이터가 참고자료[5]에 실려있다. 이곳을 참조하자면 CG, FT, MG 커널에서 700 ~ 750 Mflop/s의 속도를 보이는 4개의 프로세서로 약 90%가 넘는 효율을 보인다.
참고자료[5]에는 더 최근의 성능 실험과 응용 프로그램 코드에 대한 scalability에 대해서도 언급하고 있다.

3.1.4. The Cray SX-6

Cray SX-6는 미국 시장에 NEC SX-6로 나와 있어 NEC SX-6 섹션에서 다룰 것이다.

3.1.5. The Cray Inc. X1.

Machine type: Shared-memory multi-vector processor.
Models: X1 (cluster).
Operating system: UNICOS (Cray Unix variant).
Connection structure: Crossbar.
Compilers: Fortran 90, C, C++, Co-Array Fortran, UPC.
Vendors information Web page: www.cray.com/products/x1/
Year of introduction: 2002.

System parameters:

Model	Cray X1
Clock cycle	800 MHz
Theor. peak performance	
Per Proc. (64 bits)	3.2/12.8 Gflop/s
Maximal	819 Gflop/s

Memory	≤ 1 TB
No. of processors	16-64 (MSP, see below)
Memory bandwidth	
Memory-Cache	34.1 GB/s
Cache-CPU	76.8 GB/s

Remarks:

Cray X1의 하드웨어 구조는 이전의 Cray SV1ex (Disappeared Systems 참조)과 거의 유사하다. 클럭 수가 500에서 800 MHz로 올랐고 메모리와 프로세서로의 대역폭이 증가되었다. 각 프로세서 보드는 사이클당 최대 4개의 부동소수 연산을 처리할 수 있는 4개의 CPU를 가진다. CPU당 이론 최대 성능은 3.2 Gflop/s에 이른다. 그러나 이른바 Multi Streaming Processor (MSP)라는 설정을 통해 여러 CPU보드에 걸쳐 4개까지 CPU를 연결할 수 있고 그 결과로 한 프로세싱 유닛의 이론 최대 성능이 12.8 Gflop/s에 달하는 효과를 가져온다.

단일 CPU 조합, 기존 MSP의 재설정 등은 작업부하에 따라 동적으로 수행될 수 있다. 단일 CPU에 대한 벡터 시작 시간은 MSP의 그것보다 작다. 그래서 작은 벡터에 대해서는 단일 CPU가 낫고 긴 벡터를 처리하는 프로그램의 경우는 MSP가 유리하다고 할 수 있다. MSP 모드가 표준 연산모드가 되어 Cray의 제원에는 MSP를 기준으로 프로세서 개수가 주어진다. 하나의 cabinet에 최대 64개의 MSP가 포함된다는 것은 256개의 SSP 프로세서를 포함할 수 있다는 의미가 된다. Cray의 최적화 문서에 의하면 현재 Cray Programming Environment가 아직 SSP 프로세싱에 대해 완전하게 최적화 되어있지 않다고 한다. 그렇다고 이것이 적절한 프로그램이 SSP 모드에서 효율적으로 실행되지 않는다는 것을 의미하는 것은 아니다.

메모리에서 CPU 보드로 그리고 캐시에서 CPU로의 상대 대역폭은 양쪽 모두 SV1ex에 비해 많이 향상되었다. 메모리에서 CPU 보드로 5.3개 8 바이트 연산수가 전송될 수 있고 캐시에서 CPU로는 최대 대역폭에서 12개 8 바이트 연산수 전송이 가능해 2항 연산 수행이 충분하다. 캐시 구조는 오히려 복잡해져 하나의 보드에 있는 4개 SSP는 각각 16KB 2-way 집합 연관성 L1 데이터 캐시와 명령어 캐시를 가진다. L1 데이터 캐시는 오직 스칼라 데이터만 저장한다. L2 캐시의 크기는 2 MB이고 CPU 보드의 SSP 프로세서들이 서로 공유한다. 사용자들에게 잘 드러나지 않는 특성으로 IEEE 754 부동소수 표준 연산을 따르는 것과 새로운 벡터 명령어 집합을 들 수 있다. 이 새로운 벡터 명령어를 통해 캐시, 원격 노드에 대한 동기화와 주소지정성 등과 같은 새로운 특성을 더 잘 사용할 수 있다. 이것은 특히 모든 cabinet이 클러스터의 한 노드로 간주될 수 있기 때문에 중요하다. 클러스터의 한 노드로 인식되는 cabinet은 수정된 2-D 토러스 토폴로지를 통해 최대 64개까지 설정할 수 있다. Cray는 4개의 MSP를 가지는 하나의 보드를 한 노드로 간주한다. 각 노드는 외부와 2개의 연결 통로를 가진다. 홀수와 짝수 노드들이 쌍으로 연결되고 보드의 다른 연결은 스위치를 거쳐 다른 보드와 연결된다. 그래서 캐비닛 안의 다른 MSP와 연결하기 위하여 두 개의

hops만 거치면 된다. 꽉 찬 캐비닛 하나의 aggregate 대역폭은 400 GB/s이다. 다중 캐비닛 설정은 과거의 Cray T3E에서 그랬던 것처럼 2-D 토러스를 3-D 토러스 구조로 확장하는 것이다. Point-to-point 통신에 대한 latency와 대역폭 데이터는 나와 있는 것이 없으나 MPI 환경에서 다양하게 측정되어 왔다(아래 Measured Performances 참조).
4-SSP CPU 보드 하나에서 OpenMP를 사용할 수 있다. 다른 CPU 보드들에 접근할 때는 Cray의 shmem 라이브러리를 one-sided 통신에 사용할 수 있고 MPI, Co-Array Fortran 등도 사용 가능하다.

Measured Performances:
Oak Ridge National Laboratory에서 504개 프로세서(MSP)를 이용해 494,592차 linear system을 푸는데 5895Gflop/s의 속도가 나왔다[42]. 이것은 효율이 91%에 달하는 것이다. 참고자료 [10]에 ORNL 시스템에 대한 보다 광범위한 실험결과가 나와있다. 이곳을 보면, 4-MSP 한 노드에서 point-to-point 대역폭이 13.9 GB/s, 노드와 노드 사이의 대역폭은 11.9 GB/s이고, 각각에 대해 작은 메시지에 대한 MPI latency는 8.2, 8.6 μs이다. shmem과 Co-Array Fortran에서의 latency는 겨우 3.8, 3.0 μs이다.

3.1.6. The Cray Inc. XD1

Machine type: Distributed-memory multi-processor.
Models: XD1.
Operating system: Linux (kernel 2.4.21 with Cray HPC enhancements).
Connection structure: Variable (see remarks).
Compilers: Fortran 90, C, C++.
Vendors information Web page: www.cray.com/products/xd1/
Year of introduction: 2004.

System parameters:

Model	Cray XD1
Clock cycle	2.2 GHz
Theor. peak performance	
Per Chassis (see remarks)	52.8+ Gflop/s
Per Rack (see remarks)	663+ Gflop/s
Memory	
Per Chassis	96 GB
Per Rack	1.2 TB
No. of processors	
Per Chassis	12

Per Rack	144
Communication bandwidth	
Point-to-point	≤ 2.9 GB/s
Aggregate, per chassis	96 GB/s

Remarks:

지금은 Cray로 인수되고 없는 Octigabay사에서 지금의 Cray XD1을 처음 개발하였다. Cray XD1이 처음의 Octigabay 시스템과 다른 점은 알고리즘 가속을 위해 시스템의 계산 보드에 FPGAs를 추가할 수 있다는 것이다. 알고리즘 가속은 대규모 FFTs나 DNA sequence alignments 등의 대규모 계산을 수행해야 하는 사용자들이 특히 관심을 기울일 만하다. 위의 system parameter 리스트에서 이론 최대 성능 값에 '+' 기호를 붙인 이유가 바로 알고리즘 가속 기능 때문이다. Cray는 계산 보드와 노드를 연결하는 특별한 통신 네트워크를 추가해 시스템을 구성했다. Cray 고유의 Rapid Array Network로 구축된 이것을 섀시(chassis)라고 한다.

XD1의 일반적인 구조는 다음과 같다. 섀시 하나에 6개의 계산 카드가 있고 각 계산 카드는 2개의 AMD Opteron(2.2GHz) 프로세서와 통신을 수행하는 하나 또는 두 개의 RapidArray Processors(RAPs)로 구성된다. 두 개의 Opteron 프로세서는 3.2 GB/s의 대역폭을 가지는 AMD의 HyperTransport를 거쳐 연결되어 2-way SMP를 구성한다. HyperTrandport 버스의 대역폭이 높아 하나의 보드에서 두 개의 프로세서를 사용하는 것으로 인한 메모리 접근 문제점은 없다. 한 노드에 2개의 프로세서를 가지는 대부분의 클러스터들은 메모리 접근에 문제점을 가지고 있다. 선택적으로 응용 프로그램 가속 프로세서(FPGA)가 계산 보드에 올려질 수 있다. 보드당 두 개의 RAPs를 가지며 보드와 보드 사이의 대역폭은 8 GB/s(4 GB/s bi-directional)로 RapidArray 스위치를 거치게 된다. 이 스위치는 48개의 연결이 있는데, 그 중 반은 섀시 내의 계산 보드에 위치한 RAPs와의 연결에 사용되고 나머지 반은 다른 섀시와의 연결에 사용된다. 표준 랙(rack) 하나에는 12개의 섀시가 적합하다. RapidArray 스위치마다 있는 자유 연결(free links)의 수 때문에 두 개의 랙에 있는 섀시가 직접 연결된다. 물론, 3-D 토러스나 fat tree처럼 좀 더 성긴 네트워크 연결구조를 통해 더 큰 시스템으로의 설정이 가능하다.

RAPs는 Opteron 프로세서의 통신 부하를 덜어주고, MPI, Cray-style shmem과 Global Arrays(가상 공유 메모리 시스템) 등에 대해 하드웨어 수준에서 지원을 한다. Cray에서 발표한 내용을 보면, RapidArray 네트워크를 이용한 MPI 통신 특성은, 크기가 큰 메시지에 대해 2.9 GB/s, 작은 메시지에 대해서는 1.6 μs로 아주 우수하다.

Cray-enhanced Linux OS의 추가적인 특성 하나로, 시스템에서 실행되는 작업들에 대한 동기화 기능이 있다. 시스템 내에서 무작위로 스케줄링 되는 작업들이 latency를 증가시켜 MPI 성능을 감소시킬 수 있는데, 작업 동기화로 이런 문제를 해결할 수 있다.

Measured Performance:

Cray XD1은 아주 최근의 시스템이라 아직 성능 실험된 결과가 나와있지 않다.

3.1.7. Fujitsu/Siemens PRIMEPOWER

Machine type: RISC-based shared-memory multi-processor.

Models: PRIMEPOWER 900, 1500, 2500.

Operating system: Solaris (Sun's Unix variant).

Connection structure: Crossbar.

Compilers: Parallel Fortran 90, OpenMP, C, C++.

Vendors information Web page: primepower.fujitsu.com/en/index.html

Year of introduction: 2002.

System parameters:

Model	PRIMEPOWER 1500	PRIMEPOWER 2500
Clock cycle	1.89 GHz	1.82 GHz
Theor. peak performance		
Per Proc. (64-bits)	3.78 Gflop/s	3.64 Gflop/s
Maximal	121.0 Gflop/s	466 Gflop/s
Main Memory		
Memory/node	≤ 4 GB	≤ 4 GB
Memory/maximal	≤ 128 GB	≤ 512 GB
No. of processors	4 - 32	8 - 128
Communication bandwidth		
Point-to-point	—	—
Aggregate	17.2 GB/s	133 GB/s

Remarks:

900 모델은 프로세서 개수가 최대 16개까지만 가능하다는 것을 제외하고는 1500, 2500 모델과 동일한 구조를 가지기 때문에 이곳에서는 PRIMEPOWER 1500과 2500에 대해서만 알아본다. PRIMEPOWER는 여러 가지 측면에서 SUN Fire E25K와 유사하다. 프로세서는 Fujitsu의 SPARC 64 V를 사용하는 데, 이것은 SUN의 SPARC 프로세서를 계승한 것으로 64비트 아키텍처를 가지며, SUN의 모든 제품과 호환된다. Fujitsu의 SPARC64 프로세서는 모든 CPU 성분과 캐시, 레지스터, 실행 유닛 등에 확장 RAS 특성을 가져 신뢰도를 높였다. 1500 모델에서는 1.89GHz 프로세서를 2500 모델에서는 1.82GHz 프로세서를 각각 사용한

다. PRIMERPOWER 시스템에서 프로세서 사이의 연결은 Fire E25K와 같은 것으로 모든 프로세서를 동일하게 연결하는 크로스바를 이용한다. 즉, 이 시스템은 NUMA 시스템이 아니다.
안타까운 것은 Fujitsu의 웹 사이트에서 제공하는 정보 외에 이 시스템에 대한 다른 기술적인 정보를 찾을 수 없다는 것이다. Fujitsu에서 제공하는 데이터에는 point-to-point, bi-sectional, aggregate 등의 대역폭과 관련된 거의 모든 정보가 생략되어 있다. 다만, PRIMEPOWER 2500 모델의 aggregate 대역폭이 유일하게 나와있는데, 이 대역폭은 두 개의 66.5 GB/s 크로스바를 통해 133 GB/s이라는 높은 값을 가진다. 나와 있는 정보를 가지고 판단컨대, 1500 모델은 HPC보다는 상업적 용도의 서버에 맞춘 것 같고, 2500 모델은 HPC 시장을 목표로 하는 것으로 보인다. Fujitsu는 4x4 GB/s에 이르는 초고속 optical 연결을 통해 많은 수의 PRIMEPOWER 2500 시스템을 연결해서 거대 시스템을 구축할 수 있도록 하고 있다.

Measured Performance:
1.3 GHz 프로세서를 장착한 완전한 PRIMEPOWER 2500 시스템을 18대 연결해 구성된 클러스터를 테스트한 성능 정보가 참고자료[42]에 나와 있다. 이에 따르면 N = 658,800인 linear system의 해를 구하는 데, 2,304개 프로세서에서 45%의 효율에 해당하는 5.4 Tflop/s의 결과가 나왔다.

3.1.8. The Hitachi SR11000

Machine type: RISC-based distributed-memory multi-processor.
Models: SR11000 H1.
Operating system: AIX (IBM's Unix variant).
Connection structure: Multi-dimensional crossbar (see remarks).
Compilers: Fortran 77, Fortran 95, Parallel Fortran, C, C++.
Vendors information Web page: www.hitachi.co.jp/Prod/comp/hpc/SR_e/11ktop_e.html
Year of introduction: 2003.

System parameters:

Model	SR11000 H1
Clock cycle	1.7 GHz
Theor. peak performance	
Per Proc. (64-bits)	109 Gflop/s
Maximal	27.8 Tflop/s
Main Memory	
Memory/node	≤ 64 GB

Memory/maximal	16.4 TB
No. of processors	4 - 256
Communication bandwidth	
Point-to-point	12 GB/s (bidirectional)

Remarks:

SR11000은 Hitachi의 4세대 분산 메모리 병렬 시스템으로 이전의 SR8000(Systems Disappeared from the List) 시스템을 대체하고 있다. SR8000이 서로 다른 프로세서 클럭과 대역폭에 따라 4개의 모델이 있었던 반면, SR11000은 현재 하나의 모델(SR11000 H1)만 나와있다.

기본 노드 프로세서는 IBM의 1.7 GHz POWER4+이다. 이전의 SR2201과 SR8000에서는 Hitachi의 Pseudo Vector Processing에 맞도록 프로세서를 수정했으나, SR11000에서는 이와 같은 수정 없이 POWER4+ 프로세서를 사용한다. Hitachi의 Pseudo Vector Processing은 캐시에 없는 데이터에 접근할 때 일반적으로 발생하는 손실효과 없이 길이가 매우 긴 벡터를 처리할 수 있는 기술이다. 이제 Hitachi는 진보된 데이터 선인출(prefetching)을 통해 같은 효과를 얻고 있다.

SR11000의 기본 프로세서당 최대 성능(또는 IP)은 2개의 simultaneous multiply/add 명령어를 통해 6.8Gflop/s에 이른다. 그러나 사용자에게 기본적인 계산 단위가 되는 것은 16개의 기본 프로세서를 묶어 모든 메모리를 공유하도록 구성한 하나의 프로세싱 노드이다. 이 프로세싱 노드의 이론 최대 성능은 108.8Gflop/s에 달한다. Hitachi에서는 이와 같은 노드 설정을 Co-operative Micro-Processors in single Address Space(COMPAS)라고 한다. 사실 이것은 앞선 the main architectural class와 ccNUMA 시스템에서 살펴본 SMP 클러스터링의 일종이다. 대부분의 SMP 클러스터와 다른 점은, 클러스터 노드를 구성하는 각 프로세서에 사용자가 접근할 수 없다는 것이다. 그리고, 모든 노드에는 하나의 SP가 있다. SP는 시스템 작업을 수행하고, 다른 노드와의 통신과 일련의 I/O 장치를 관리하는 시스템 프로세서이다.

SR11000은 12 GB/s의 single-directional 연결속도를 가지는 다중차원 크로스바를 가지고 있다. 그리고 IBM Federation Switch fabric이라는 IBM 기술을 사용하고 있다. 이 IBM 기술은 IBM이 p690 서버에서 사용하는 것과는 다른 토폴로지이다. 노드가 4 ~ 8개 이상에서 네트워크 cross-section에 1hop이 있다. 16 ~ 64개 노드를 가질 경우 2 hop이고 128 노드 이상의 시스템은 3 hop가 있다.

Cray XD1, AlphaServer SC, 이전의 NEC Cenju-4 같은 다른 시스템들처럼, SR11000 시스템에서 프로세서는 다른 원격 프로세서의 메모리에 직접 접근할 수 있다. 이때 고속의 하드웨어 기반 장벽 동기화를 사용하게 되는데 이 동기화는 매우 낮은 병렬 부하로 분산 프로그램의 쓰기를 지원해야 한다.

당연히 PVM이나 MPI 같은 통신 라이브러리가 제공된다. MPI를 사용하면 노드 내에서 개

별 IP들에 접근하는 것이 가능하다. 거기에 한 노드에서 OpenMP를 이용하는 것도 가능하다. 대부분의 경우 Hitachi의 컴파일러를 이용해 자동병렬화를 하는 것이 더 낫지만, 원하는 성능 이득에 도달하기 위해 OpenMP를 이용해 coarse-grained 태스크 병렬성을 이용할 수 있다. Hitachi는 dense, sparse linear system과 FFT 등을 풀기 위한 고유의 수치계산 라이브러리를 제공한다. NAG나 IMSL과 같은 수치계산 라이브러리를 사용할 수 있는지 여부는 아직 알려지지 않았다.
2005년 초에 POWER5를 기반으로 하는 SR11000 시스템이 시장에 나올 것으로 보인다.

Measured Performances:
SR11000은 2003년 말에 소개되었다. 그사이 일본에서 몇 대의 시스템이 판매되었으나 아직 발표된 성능 결과는 나와있지 않다.

3.1.9. The HP 9000 Superdome

Machine type: RISC-based ccNUMA system.
Models: HP 9000 Superdome.
Operating system: HP-UX (HP's usual Unix flavor).
Connection structure: Crossbar.
Compilers: Fortran 77, Fortran 90, Parallel Fortran, HPF, C, C++.
Vendors information Web page:
www.hp.com/products1/servers/scalableservers/superdome/
Year of introduction: 2000, 2004 with PA-RISC 8800.

System parameters:

Model	HP 9000 Superdome
Clock cycle	1 GHz
Theor. peak performance	
Per Proc. (64-bits)	8 Gflop/s
Maximal	512 Glop/s
Main Memory	
Memory/node	≤ 64 GB
Memory/maximal	1 TB
No. of processors	≤ 64
Communication bandwidth	
Aggregate (global)	64 GB/s
Aggregate (cell-backplane)	8 GB/s
Aggregate (within cell, see below)	16 GB/s

Remarks:

Superdome은 이제는 물러난 HP의 Exemplar V2600(Systems Disappeared from the List 참조)을 대체하고 있다. Superdome의 연결 구조는 이전의 2600 시스템의 구조를 획기적으로 발전 시킨 것이다. Superdome은 4개 프로세서를 가지는 cell 안의 레벨과 크로스바 backplane으로 cell을 연결하는 또 하나의 레벨까지 2-level 크로스바를 가진다. 모든 cell은 8 GB/s의 속도로 backplane에 연결되어서 전체 aggregate 대역폭은 64 GB/s에 이른다. Superdome은 4-processor cell을 기본으로 해서 구성되며 cell 내부에서의 모든 데이터 소통은 Cell Controller(10-port ASIC)에 의해 제어된다. Cell Controller는 16 GB/s로 4개의 로컬 메모리를 연결하고, 8 GB/s로 backplane 크로스바를 연결한다. 그리고 두 개의 포트를 연결하는 데 한 포트가 6.4 GB/s의 속도로 두 개의 프로세서를 지원한다. 각 프로세서가 두 개의 CPU 코어를 가져서 CPU 코어당 가용 대역폭은 1.6 GB/s가 된다. SGI Altix 시스템과 같이 Superdome에서도 캐시 일관성을 위해 directory memory를 사용한다. HP의 발표에 의하자면 64개 프로세서를 완전히 가지는 시스템의 NUMA factor는 겨우 1.8이다.

PA-RISC 8800 프로세서는 1 GHz 클럭으로 동작한다. 각 프로세서가 두 개의 프로세서 코어를 가지고 각 코어는 부동소수 곱셈과 덧셈을 동시 처리하는 명령어를 실행할 수 있는 두 개의 부동소수 유닛을 가져 최적의 상황에서 8 flops/cycle을 얻을 수 있다. 이로 인해 프로세서 하나의 이론 최고 성능이 8 Gflop/s에 이른다. 따라서 64개 프로세서를 완전히 가지는 시스템의 최대 속도가 512Gflop/s에 달하게 된다.

공유 메모리 병렬 프로그래밍 모델이 전체 시스템에 걸쳐 지원되므로 OpenMP를 이용해 64개 프로세서(128 CPU 코어)를 모두 사용할 수 있다.

Superdome은 Itanium 2와 같은 다른 프로세서를 이용해 실행되는 서로 다른 복합체(complexes)들로 나눌 수 있다. 이 경우 같은 backplane을 사용하지만 cell들은 서로 다른 타입이 된다. 이론적으로 프로세서가 혼합된 HP 9000 Superdome과 Integrity Superdome을 가질 수 있다.

Measured Performances:

듀얼 코어 PC-RISC 8800 프로세서를 장착한 모델은 2004년 4월 시장에 발표된 새로운 시스템이라 HPC영역에서 알려진 성능 결과가 없다. 크기가 알려지지 않은 완전 linear system의 해를 얻는데 756 Gflop/s의 속도가 나온 것을 참고자료[42]에서 확인할 수 있다. 이 결과는 총 512개의 875 MHz PA-RISC 8700+ 프로세서를 가지는 이전의 8-way coupled 시스템을 이용해 얻은 것이다. 이 결과는 사용한 클러스터의 이론 최고 성능이 1,792 Gflop/s인 것을 감안하면 42%의 효율을 보이는 것이다.

3.1.10. The HP Integrity Superdome

Machine type: RISC-based ccNUMA system.

Models: HP Integrity Superdome.
Connection structure: Crossbar.
Operating system: HP-UX (HP's usual Unix flavour), Linux
Compilers: Fortran 77, Fortran 90, Parallel Fortran, HPF, C, C++.
Vendors information Web page:
ww.hp.com/products1/servers/integrity/superdome_high_end/
Year of introduction: 2004.

System parameters:

Model	Integrity Superdome
Clock cycle	1.5 GHz
Theor. peak performance	
Per Proc. (64-bits)	6 Gflop/s
Maximal	768 Glop/s
Main Memory	
Memory/node	≤ 128 GB
Memory/maximal	≤ 1 TB
No. of processors	≤ 128
Communication bandwidth	
Aggregate (global)	64 GB/s
Aggregate (cell-backplane)	8 GB/s
Aggregate (within cell, see below)	16 GB/s

Remarks:
Integrity Superdome은 미래의 하이-엔드 서버를 위해 HP가 투자하고 있는 시스템이다. 몇 년 지나지 않아 이 시스템이 PA-RISC 기반의 HP 9000 Superdome을 대체하게 될 것으로 여겨지는데, HP는 cell들 사이의 통신을 위해 cell이 backplane 크로스바에 연결되는 거시적 구조를 동일하게 가지게 함으로써 미리부터 이를 예지시켜 왔다. cell 안에 포함된 프로세서가 PA-RISC인지 Itanium인지는 backplane에게 중요한 것이 아니다. Integrity Superdome의 거시적 구조는 앞서의 HP 9000 Superdome을 참조하면 될 것이다. 또한 cell 내부 구조도 매우 유사해서 로컬 메모리 접근, I/O 요청, backplane을 거치는 다른 cell과의 통신 등을 제어하는 중심 허브의 역할을 HP 9000 Superdome과 동일한 Cell Controller ASIC가 담당한다.
1.5 GHz Itanium 2 프로세서를 사용한다는 것이 Integrity Superdome이 HP 9000 Superdome과 다른 점이다. HP는 2개의 Itanium을 하나의 소켓에 장착할 수 있는 특수한 프로세서 소켓을 개발해 내었다. 그래서 총 8개의 Itanium이 cell 보드에 위치한 4개의 소

켓에 장착될 수 있다. 각 Itanium이 부동소수 덧셈과 곱셈을 한꺼번에 처리하는 명령어를 가진 부동소수 유닛을 2개씩 가져서 최적의 상황에서 4 flops/cycle의 성능을 얻을 수 있다. 이에 따라 프로세서당 이론 최대 성능은 6Gflop/s에 이르고 완전한 시스템(128 프로세서)에서 최대 속도가 768 Gflop/s에 이르게 된다.
두 Superdome 시스템 사이의 유사성이 워낙 커서 Integrity Superdome도 HP 9000 Superdome과 동일한 ccNUMA 특성을 가진다. 최대 128개 프로세서에 걸쳐 OpenMP를 사용할 수 있다. Integrity Superdome은 Itanium 2 프로세서를 기반으로 하기 때문에 Linux OS를 사용할 수 있다. 실제로는 시스템이 분할이 가능해, 같은 시스템 내에서 서로 다른 복합체(complexes)를 구성해 Linux와 HP-UX를 모두 구동할 수 있다. HP 9000 Superdome에서 언급했듯이 다른 복합체의 cell은 다른 타입의 프로세서를 가질 수 있어서, 이종 Integrity Superdome과 이종 HP 9000 Superdome 시스템을 구성하는 것이 가능하다.

Measured Performances:
크기가 알려지지 않은 완전 linear system의 해를 얻는데 1,049 Gflop/s의 속도가 나온 것을 참고자료[42]에서 확인할 수 있다. 이 결과는 총 220개의 프로세서를 가지는 시스템에서 측정된 것이고 이 클러스터 시스템의 이론 최대 성능이 1,320 Gflop/s이므로 79%의 효율을 보여주는 것이다.

3.1.11. The HP/Compaq GS series

Machine type: RISC-based SMP system.
Models: AlphaServer GS80, GS160, GS320, GS1280.
Connection structure: 2-D torus.
Operating system: Tru64 Unix (Compaq's avour of Unix), OpenVMS.
Compilers: Fortran 77, Fortran 90, HPF, C, C++.
Vendors information Web page: h18002.www1.hp.com/alphaserver/sc_gs.html
Year of introduction: 1999.

System parameters:

Model	GS1280
Clock cycle	1.3 GHz
Theor. peak performance	
Per Proc.	2.6 Gflop/s
Maximal	166 Gflop/s
Memory	≤ 512 GB
No. of processors	≤ 64
Memory bandwidth	

Processor/memory	12 GB/s
Communication bandwidth	
Point-to-point	6.4 GB/s
Aggregate/processor	25.6 GB/s
Aggregate (global)	1638 GB/s

Remarks:

GS 시스템 패밀리의 여러 가지 모델 중에 가장 빠른 한 가지 모델에 대해서만 알아본다. 이것은 기타 다른 모델들은 처음에 언급한 최소 시스템 속도(80 GB/s)을 넘어 가지 못하기 때문이다. GS 시리즈는 ccNUMA 서버 패밀리로 현재 가장 빠른 EV7z(1.3 GHz Alpha EV7) 프로세서를 장착하고 있다. 이 시스템은 4개의 듀얼 프로세서 카드를 가지는 drawer로 구성된다. 완전히 구성된 GS1280 시스템은 8개의 drawer를 가진다.

칩에 4개(개당 6.4 GB/s 대역폭)의 직접 통신 연결을 가지는 EV7 프로세서를 기반으로 하면서 2-D 토러스 구조로 프로세서들을 연결하는 것이 자연스런 방법이 되었다. 실질적인 측정을 통해 얻은 결과, 16개 프로세서를 가지는 시스템의 NUMA factor가 겨우 1.6이다 [21]. 1.3 GHz EV7 프로세서의 이론 최대 성능은 2.6 Gflop/s 이다. 최대 성능이 다른 프로세서들과 비교해 상대적으로 떨어질지라도 메모리 대역폭이 아주 높아 많은 프로그램에서 더 높은 효율을 기대할 수 있다. 메모리의 최대 용량은 512 GB(프로세서당 8 GB)로 적절한 개수의 프로세서로 하이-엔드 과학기술 응용 프로그램에 사용하기에 충분하다.

시스템의 ccNUMA 특성 때문에 사용자들은 OpenMP를 이용해 공유 메모리 병렬화가 가능하다. 이때 GS1280 시스템에서 최대 64개까지 프로세서 사용이 가능하다. 물론 MPI를 사용할 수도 있다.

Measured Performance:

EV7을 기반으로 하는 GS1280 시스템에 대해 아직 성능 결과가 나와 있는 것이 없다.

3.1.12. The Compaq AlphaServer SC45 Series

Machine type: RISC-based SMP-clustered DM-MIMD system.

Models: AlphaServer SC45.

Operating system: Tru64 Unix (Compaq's flavour of Unix).

Compilers: Fortran 77, Fortran 90, HPF, C, C++.

Connection structure: Fat Tree.

Vendors information Web page: h18002.www1.hp.com/alphaserver/sc/

Year of introduction: 2002.

System parameters:

Model	AlphaServer SC45
Clock cycle	1.25 GHz
Theor. peak performance	
Per Proc. (64-bit)	2.5 Gflop/s
Maximal (64-bit)	10 Tflop/s
Main Memory	≤ 8 TB
No. of processors	≤ 4096
Memory bandwidth	
Processor/memory	1.33 GB/s
Between cluster nodes	280 MB/s

Remarks:

AlphaServer SC(SC는 SuperComputer를 의미)는 HP AlphaServer 라인의 가장 하이-엔드 시스템이다. 이 시스템은 전형적인 SMP 클러스터 시스템으로 기본 SMP 노드는 Compaq ES45이다. ES45는 4개의 1.25 GHz Alpha 21264a(EV68) CPU을 가지는 SMP 시스템이다. 노드의 내부 네트워크로 5.2 GB/s의 aggregate 대역폭(프로세서당 1.33 GB/s)을 가지는 크로스바를 사용한다. 이 네트워크를 통해 노드 내의 각 프로세서는 동시에 클럭 사이클당 1.064 바이트의 데이터 전송이 가능하다.

노드 하나는 공유 메모리 시스템으로 이곳에서 OpenMP등을 이용한 공유 메모리 병렬 처리가 가능하다. 만약 프로세서를 네 개보다 더 많이 사용해야 한다면 MPI, PVM, HPF등과 같은 메시지 패싱 프로그래밍 모델을 사용해야 한다. 고유의 HPF 컴파일러를 제공하는 회사가 그리 많지 않은데 HP/Compaq은 아직 고유의 HPF 컴파일러를 제공하고 있다.

SMP 노드 사이의 통신을 위해 SC는 QSW사에서 제작되는 QsNet을 사용한다. 앞서 나왔었지만 QsNet은 예전 Meiko CS-2 시스템에서 사용하던 네트워크의 후속이다. PCI 기술을 기반으로 하는 QsNet은 fat tree 구조를 가지며 280 MB/s의 point-to-point 대역폭을 제공한다. fat tree 구조로 인해 네트워크의 상위레벨에서 대역폭은 340 MB/s로 유지된다. 제작사 자료에 의하면 최대 대역폭은 서버당 500 MB/s로 나와 있다. 매우 큰 대역폭이기는 하지만 별다른 설명이 추가되어 있지 않아 정보로서의 가치가 떨어져 보인다. 또 QSW는 이 시스템이 MPI 메시지에 대해 매우 낮은 latency(5 μs)를 보인다고 주장한다.

Measured Performances:

참고자료[42]에 8,192개의 프로세서를 가지는 AlphaServer SC45s 2-way 클러스터 시스템의 실제 성능이 13.88 Tflop/s로 보고되어 있다. 이는 633,000차의 linear system을 풀이한 것으로 68.0%의 효율을 보여주는 것이다.

3.1.13. IBM eServer p690

Machine type: RISC-based distributed-memory multi-processor.

Models: IBM eServer p690.

Operating system: AIX (IBM's Unix variant), Linux.

Connection structure: Ω-switch.

Compilers: XL Fortran (Fortran 90), (HPF), XL C, C++.

Vendors information Web page: www-1.ibm.com/servers/eserver/pseries/hardware/highend/p 690.html.

Year of introduction: 2002 (16/32-CPU POWER4+ SMP).

System parameters:

Model	eServer p690
Clock cycle	1.9 GHz
Theor. peak performance	
Per Proc. (64-bits)	7.6 Gflop/s
Per 16-proc. HPC node	121.6 Gflop/s
Per 32-proc. Turbo node	243.2 Gflop/s
Maximal	124.5 Tflop/s
Main Memory	
Memory/node	≤ 1 TB
Memory/maximal	512 TB
No. of processors	8-16,384
Communication bandwidth	
Node-to-node (bi-directional)	2 GB/s

Remarks:

eServer p690은 RS/6000 SP의 다음 시스템이다. RS/6000 SP는 다중 CPU 노드들을 하나의 프레임 안에서 전용 스위치나 switched Ethernet과 같은 다른 수단으로 연결한 구조를 가지는데 이와 같은 전체적인 구조를 eServer p690에서도 그대로 유지하고 있다. 그렇지만, 앞서 POWER4+ 프로세서 소개에서 살펴봤듯 노드의 구조는 많이 바뀌었다.

Federation 스위치는 p690 시리즈를 위해 만들어진 4 세대 고성능 연결장치로 이전과 마찬가지로 Ω-switch 구조를 가진다. 이 스위치는 bi-directional 연결 속도가 2 GB/s에 달하고, 5 ~ 7 μs의 MPI latency를 가진다. 고성능 스위치는 p690에서 통신을 위해 사용할 수 있는 가장 빠른 방법이지만, Gigabit Ethernet과 같은 기타 다른 장치를 이용한 통신도 가능하다.

병렬 프로그램들은 MPI나 PVM을 사용해 실행될 수 있다. IBM은 고유의 버전과 Portland

Group의 컴파일러 버전 모두에 대해 High Performance Fortran을 지원해 왔으나, 아직도 지원하고 있는지는 확실치 않다. IBM은 데이터 포맷 변환기 XDR을 제거한 고유의 PVM 버전을 사용한다. 또 MPI 구현환경인 MPI-F가 p690 기반 시스템에 맞춰 최적화되었다. 시스템을 구성하는 노드들이 실질적으로 공유 메모리 SMP시스템이어서 각 노드에서는 OpenMP를 이용해 공유 메모리 병렬화를 수행할 수 있으며, 필요하다면 MPI와 같이 사용될 수도 있다. IBM의 Unix OS인 AIX뿐 아니라 일부 Linux 배포판(RedHat과 SuSe의 프로페셔널 버전을 p690에서 사용할 수 있다.)도 지원한다.
시장에 나와있는 상업적 표준 모델은 최대 128개의 노드로 구성되지만, 특별한 용도로 사용되는 시스템에 대해 최대 512개 노드까지 가능하다. 아직 128 노드를 초과하는 시스템이 판매되지는 않았지만, 위의 System parameters 자료는 최대 가능한 설정인 512 노드를 기준으로 작성한 것이다. POWER5 기반의 p690 시스템이 곧 시장에 나올 것이지만 아직 정확한 일정을 알 수는 없다.

Measured Performance:
크기 N=355,000인 dense linear system을 푸는데 1600개의 1.7 GHz 프로세서를 이용해 6,188 Gflop/s의 결과를 얻었으며[42], 이는 약 57%의 효율을 보이는 것이다. 8개의 Turbo 노드를 가진 시스템에서 크기가 285,000인 linear system을 푸는데 73 7Gflop/s의 결과를 보인다. 이것은 해당 시스템의 이론 최고 성능이 1,331 Gflop/s인 것을 감안하면 55%의 효율을 보여주는 것이다. 이와 같은 형태의 프로그램들은 주로 L1 캐시를 이용해 실행되므로 대부분 유사한 효율이 나올 것으로 생각된다.

3.1.14. The IBM BlueGene/L

Machine type: RISC-based distributed-memory multi-processor.
Models: IBM BlueGene/L.
Operating system: Linux.
Connection structure: 3-D Torus, Tree network.
Compilers: XL Fortran (Fortran 90), XL C, C++.
Vendors information Web page: www-1.ibm.com/servers/deepcomputing/.
Year of introduction: 2004.

System parameters:

Model	BlueGene/L
Clock cycle	700 MHz
Theor. peak performance	
Per Proc. (64-bits)	2.8 Gflop/s
Maximal	367/183.5 Tflop/s

Main Memory	
Memory/card	≤ 512 MB
Memory/maximal	≤ 16 TB
No. of processors	≤ 2 x 65,536
Communication bandwidth	
Point-to-point (3-D torus)	175 MB/s
Point-to-point (Tree network)	350 MB/s

Remarks:

초병렬 컴퓨팅(very massively parallel computing)을 위해 IBM에서 개발된 새 세대 시스템의 첫 주자가 BlueGene/L이다. BlueGene/L은 고밀도 패키징과 프로세서당 전력 소비를 낮추기 위해 프로세서의 속도를 희생하였다. 시스템에서 사용하는 기본 프로세서는 700 MHz 클럭을 가지는 수정된 PowerPC 400이다. 하나의 칩에는 4 MB의 L3 캐시를 공유하는 두 개의 프로세서가 들어간다. 2 KB 크기의 L2 캐시는 각 프로세서에 따로 존재한다. 프로세서에는 L2 캐시와 연결되는 두 개의 load 포트와 하나의 store 포트가 있다. 각 포트는 8 bytes/cycle의 속도를 가지는데 이것은 두 개의 FPU에서 요구되는 대역폭의 반이다. 각 CPU는 32 KB의 명령어 캐시와 같은 크기의 데이터 캐시를 가지고 있다. 두 개의 FPU가 덧셈-곱셈을 동시 처리하는 명령어를 실행할 수 있으므로 최적의 상황에서 하나의 CPU는 2.8 Gflop/s의 최대 성능을 가진다. 색다른 것이 L2 캐시의 크기가 L1캐시보다 작다는 것인데 이와 같은 구조를 통해 보다 빠른 처리를 하게 된다.

시스템 패키징은 다음과 같다. 512 MB의 메모리를 가지는 하나의 compute 카드에 두 개의 칩이 들어 간다. 이 compute 카드 16개가 하나의 노드 보드에 들어가고 노드 보드 32개로 하나의 캐비닛을 구성한다. 그래서 하나의 캐비닛에는 1,024개의 칩 즉, 2,048개의 CPU가 있게 되는 것이다. 최대 64개의 캐비닛을 묶어 하나의 시스템(65,356 chips/130,712 CPUs)을 구성할 수 있다. 정상적인 동작 모드에서 하나의 칩에 있는 두 개의 CPU 중 한 CPU만이 계산을 위해 사용되고 나머지 하나는 통신 작업에 사용된다. 이 모드에서 시스템의 이론 최고 성능은 183.5 Tflop/s가 된다. 그러나 통신 요청이 아주 작을 경우 두 개의 CPU를 모두 계산을 위해 사용하는 것이 가능하고, 이때 최고 성능은 두 배가 된다. 그래서 System parameters 자료에 최고 속도가 두 가지 표기되어 있다. IBM은 360 Tflop/s를 영업 자료로 사용한다.

BlueGene/L에는 5개의 네트워크가 있다. 그 중 3-D 토러스 네트워크와 트리 네트워크 두 개는 프로세서 간의 통신을 담당한다. 토러스 네트워크는 대부분의 일반적인 통신 패턴에 사용되고 트리 네트워크는 broadcasting이나 reduction과 같은 집합통신 패턴에 자주 사용된다. 트리 네트워크의 하드웨어 대역폭은 연결당 350 MB/s로 토러스 네트워크(175 MB/s)의 두 배이다. 지금 이 문서가 작성되는 시점에 아직 완전히 설정된 시스템이 없으나 올 해 말에 Lawrence Livermore Laboratory에 한 대가 설치될 예정이다. 조금 더 작은 시스템

(최고 34 Tflop/s)이 네덜란드의 천문 연구기관인 ASTRON에 설치될 것이다.

Measured Performance:
최근 IBM은 HPC Linpack 벤치마크에서 36.01 Tflop/s의 속도가 측정되었다고 발표했다. 이 같은 결과를 얻은 linear system의 크기나 BlueGene 시스템의 크기에 대해 알려진 바는 없다.

3.1.15. The NEC SX-6

Machine type: Shared-memory multi-vector processor.
Models: SX-6i SX-6A, SX-6xMy.
Operating system: Super-UX (Unix variant based on BSD V.4.3 Unix).
Compilers: Fortran 90, HPF, ANSI C, C++.
Connection structure: Multi-stage crossbar (see Remarks)
Vendors information Web page: www.hpce.nec.com/468.0.html
Year of introduction: 2002.

System parameters:

Model	SX-6i	SX-6A	SX-6xMy
Clock cycle	500 MHz	562.5 MHz	562.5 MHz
Theor. peak performance			
Per Proc. (64-bits)	8 Gflop/s	9 Gflop/s	9 Gflop/s
Maximal			
Single frame:	8 Gflop/s	72 Gflop/s	—
Multi frame:	—	—	9.2 Tflop/s
Main Memory (per frame)	4-8 GB	32-128 GB	≤ 16 TB
No. of processors	1	4-8	8-1024

Remarks:
SX-6 시리즈에는 다양한 모델이 있지만 적은 수의 동종 프로세서를 가지는 작은 프레임으로 구성되어 있는 것이 대부분이다. 이곳에서는 본질적으로 서로 다른 모델들에 대해 알아볼 것이다. 모든 모델은 동일한 8-way replicated vector 프로세서를 기반으로 하고 있다. 이 프로세서의 각 벡터 파이프 set(8개)에는 logical, mask, add/shift, multiply, division 파이프가 있다(SM-MIMD systems 참조). 곱셈과 덧셈이 연계되어 있어 500 MHz 한 파이프 set의 최대 성능은 1 Gflop/s이다. 8-way replication으로 인해 CPU 하나의 최대 성능은 8 Gflop/s가 된다. 하나의 스칼라 프로세서가 벡터 유닛을 보완하고 있다. 562.5 MHz의 이 스칼라 프로세서는 4-way 슈퍼스칼라로 이론 최고 성능이 1.125 Gflop/s이다. CPU당 최대

대역폭은 64 Bytes/cycle로, 이것은 8개의 8바이트 연산수를 양방향으로 전송할 수 있고 하나의 연산수를 각각의 replicated 파이프 set에 제공할 수 있는 대역폭이다.
단일 프로세서의 최대 성능이 SX-6의 이전 시스템인 SX-5에서 10 Gflop/s이던 것이 8Gflop/s로 떨어진 것은 흥미롭다. 그 이유는 이전 버전의 CPU는 다중 칩이 필요했지만, SX-6의 CPU는 단일 칩에 올라가기 때문이다. 같은 이유로 SX-5에서는 16이었던 replication factor가 SX-6에서는 8로 1/2이 되었다.
SX-6i는 단일 CPU 시스템으로 desk side 모델로 제공되지만, 두 개의 시스템을 하나의 랙에 넣은 랙 모델도 있다. 이때 랙에 설치된 두 시스템은 서로 연결되지 않는다.
SX-6A 모델의 단일 프레임에는 같은 클럭 수를 가지는 CPU가 8개까지 들어간다. 프레임 내에서 CPU들은 단일 CPU 시스템과 같은 대역폭(36 GB/s/port)을 가지는 1단 크로스바로 연결된다. 완전히 설정된 프레임은 최대 72 Gflop/s에 이르는 속도를 가질 수 있다.
SX-6xMy는 멀티프레임 모델로 여기서 x(= 8,…,1024)는 총 CPU 수를 나타내며 y(= 2,…,128)는 프레임의 수를 나타낸다. SX-6 프레임을 멀티프레임으로 묶는 방법은 2가지가 있다. NEC에서는 다양한 프레임을 연결하기 위해 IXS 크로스바라는 풀 크로스바를 제공한다. 이 크로스바는 8 GB/s의 속도로 프레임 외부와의 point-to-point unidirectional 통신이 가능하며 최대 1,024 GB/s bisectional 대역폭을 제공한다. 그리고 낮은 비용과 속도로 프레임 간의 통신을 위해 HiPPI 인터페이스 사용이 가능하다. IXS 크로스바 solution을 이용해 전체 멀티프레임 시스템을 하나의 주소공간으로 묶어 NUMA 시스템으로 구성할 수 있다. 그러나 성능문제로 인해 MPI를 이용하는 분산 메모리 모드에서 시스템을 사용할 것을 권장한다.
CMOS 기술을 통해 fabrication 비용과 전력 소비를 크게 낮추었다(CMOS는 이전의 Fujitsu VPP5000과 Cray SV1 그리고 최근의 Cray X1에서도 사용되고 있다.). 모든 모델은 공랭식이다.
분산 메모리 컴퓨팅을 위해 HPF 컴파일러가 있고 메시지 패싱을 위해 최적화된 MPI(MPI/SX)를 사용할 수 있다. 그리고 공유 메모리 병렬성을 위해 OpenMP도 사용할 수 있다.

Measured Performance:
31개 프레임에 248개의 프로세서를 장착한 SX-6/248M31 시스템에 대한 결과가 [42]에 나와있다. 크기가 220,224인 linear 시스템에 대해 2,155 Gflop/s의 성능을 얻었고 이것은 97%의 효율을 보여주는 것이다.

3.1.16. The NEC TX-7 series

Machine type: Shared-memory SMP system.
Models: TX-7 i9010, i9510.
Operating system: Linux, HP-UX (HP's Unix variant).

Connection structure: Crossbar.
Compilers: Fortran 90, HPF, ANSI C, C++.
Vendors information Web page: www.hpce.nec.com/465.0.html
Year of introduction: 2002.

System parameters:

Model	i9010	i9510
Clock cycle	1.5 GHz	1.5 GHz
Theor. peak performance		
Per Proc. (64-bits)	6 Gflop/s	6 Gflop/s
Maximal	96 Gflop/s	192 Gflop/s
Main Memory (per frame)	≤ 64 GB	≤ 128 GB
No. of processors	16	32

Remarks:
TX-7 시리즈는 4가지 모델이 있지만 여기서는 그 중 두 가지 상위 모델에 대해서 알아본다. TX-7은 최근 시장에 나온 Itanium 2 기반의 서버들(Bull NovaScale, SGI Altix3000, Unisys ES7000 등) 중 하나이다. 현재 제공되는 모델 중 가장 대형은 TX-7/i9510으로 32개의 1.5GHz Itanium 2 프로세서를 가진다. NEC는 이미 16개 프로세서를 가지는 Itanium 1 서버(AsuzA)를 생산적이 있어서 TX-7 시스템은 2 세대 시스템으로 볼 수 있다.
프로세서는 플랫 크로스바로 연결된다. NEC는 TX-7 시스템에 3 ~ 6 MB의 L3 캐시를 가지는 Intel 프로세서를 클럭 수(1.3, 1.4, 1.5 GHz)에 따라 선택할 수 있도록 하고 있다(Itanium 2 section 참조).
Itanium 2를 채택해 사용하고 있는 다른 업체들과는 다르게, NEC는 HPF 컴파일러를 비롯한 고유의 컴파일러를 제공하고 있다. TX-7 같은 공유 메모리 시스템에서 별로 쓰일 일이 없을 것 같은 HPF 컴파일러를 제공하는 이유는 아마도 NEC SX-6 소프트웨어와의 호환성 때문일 것이다. MPI와 OpenMP도 제공한다. Linux외에 HP-UX를 운영체제로 제공하고 있어 HP에서 개발된 프로그램을 손쉽게 TX-7으로 옮길 수 있도록 하고 있다.

Measured Performance:
2004년 봄, EuroBen Benchmark를 이용한 광범위한 벤치마크 테스트가 1.5 GHz 프로세서 16개를 가진 TX-7 i9010 시스템에 수행되었다. EuroBen 벤치마크를 이용한 dense matrix-vector 곱셈의 MPI 버전에서 14.5 Gflop/s가 나왔고, 크기 N = 2,000인 linear system과 크기 N = 16,384인 1-D FFT를 푸는데 3.8 ~ 4.1 Gflop/s의 속도가 나왔다[40].

3.1.17. The Quadrics Apemille

Machine type: Processor array.

Models: Quadrics Apemille.

Front-end: Almost any Unix workstation.

Operating system: Internal OS transparent to the user, Unix on front-end.

Connection structure: 3-D mesh (see remarks).

Compilers: TAO - a Fortran 77 compiler with some Fortran 90 and some proprietary array extensions.

Vendors information Web page: www.quadrics.com

Year of introduction: 1999.

System parameters:

Model	Apemille
Clock cycle	267 MHz
Theor. peak performance	
Per Proc. (32-bits)	533 Mflop/s
Maximal (32-bits)	1 Tflop/s
Main memory	≤ 64 GB
No. of processors	8-2048
Communication bandwidth	
Per Proc.	—
Aggregate local	—
Aggregate non-local	—

Remarks:

Apemille은 Italian National Institute for Nuclear Physics에서 추진한 APE-1000 프로젝트의 상업적 후속 판이자 APE-100을 잇는 시스템이다. 이 시스템은 8개의 프로세서 노드를 가지는 보드를 단위로 구성된다. 최대 16개의 보드(128 노드)가 하나의 crate을 구성하는 데 한 시스템에는 최소 하나에서 최대 16개의 crate까지 가능하다. Quadrics의 내부연결 토폴로지는 반대편과 연결되는 3-D 그리드 구조로 실질적으로는 3-D 토러스가 된다. 8 노드를 가지는 FPB(Floating Point Board)는 point-to-point 통신과 global control distribution 을 제공하는 crate backplane에 접속된다. FPB는 2^3 cube로 구성되어 다른 보드로 연결되며 전체적으로 3-D 그리드 구조를 가지게 된다.

MAD 칩으로 불리는 기본적인 부동소수 프로세서는 레지스터 파일에 128개의 레지스터를 가진다. 이 레지스터의 첫 두 개는 영구적으로 0 또는 1의 값을 가지며 이것은 혼합된 덧셈-곱셈 연산 명령어를 이용해 덧셈 또는 곱셈을 표현할 수 있도록 한다. 가령, 곱셈은 ax

b+0으로 덧셈은 ax1+b로 표현하는 것이다. 최적의 상황에서 프로세서는 사이클당 두 개의 부동소수 연산이 가능하다. 명령어들은 컨트롤러에 의해 2 사이클당 하나의 비율로 issue된다.
통신은 crate의 backplane에 위치한 Memory Controller와 Communication Controller에 의해 제어된다. Memory Controller가 주소를 발생시키면 그 주소는 Communication Controller에 의해 해석된다. 논-로컬 접근이 필요한 경우, Communication Controller가 필요한 데이터 전송을 제공할 것이다. 프로세서당 메모리 대역폭과 논-로컬 통신에 대한 대역폭 모두 문서에 밝혀져 있지 않다. 안타깝게도 Quadrics는 로컬 통신이나 글로벌 통신의 속도에 대해서도 자세한 내용을 제공하지 않고 잇다.
Apemille는 PCI 어댑터 카드를 거쳐 프론트-엔드 시스템과 통신한다. 이때 통신 대역폭은 약 100 MB/s일 것으로 보여 지지만, 실제 속도에 대해서는 알려져 있지 않다. 통신 인터페이스는 노드 메모리와 I/O 컨트롤러에 쓰기와 읽기가 가능하고 8.5 GB/s에 이르는 대역폭을 가지는 것으로 알려져 있다.
TAO 언어는 Quadrics의 SIMD 특성을 사용하기 위해 일부 확장속성을 가진다. 우선 부동소수는 그것을 가지고 있는 프로세서에게 로컬 변수로 취급되고, 정수는 글로벌 변수로 취급된다. 로컬 변수는 글로벌 변수로 올려질 수 있다. 다른 확장속성은 글로벌 테스팅과 제어에 사용될 수 있는 ANY, ALL과 WHERE/END, WHERE 키워드들이다. 이를 이용해 글로벌 조건에 해당되지 않는 프로세서들은 그 조건과 연관된 명령어들을 건너뛰게 할 수 있다. 최인접 위치에 대한 참조를 쉽게 하기 위해 LEFT, RIGHT, UP, DOWN, FRONT, BACK과 같은 특별한 상수들이 제공된다. 거기에 새로운 데이터 타입과 이와 연관된 연산자들 그리고 연산자 overloading 등이 지원된다. 이를 통해 특정 계산들을 매우 간략하게 구현할 수 있다.

Measured Performance:
이 시스템에 대해 알려진 성능 데이터가 아직 없다.

3.1.18. The SGI Altix 3000 series

Machine type: RISC-based ccNUMA system.
Models: Altix 3300, Altix 3700.
Operating system: Linux.
Connection structure: Crossbar, hypercube (see remarks).
Compilers: Fortran 95, C, C++.
Vendors information Web page: www.sgi.com/servers/altix/.
Year of introduction: 2003.

System parameters:

Model	Altix 3700
Clock cycle	1.5 GHz
Theor. peak performance	
Per Proc. (64-bits)	6 Gflop/s
Maximal	1.5 Tflop/s
Main Memory	
Memory	≤ 512 GB
No. of processors	4-256
Communication bandwidth	
Point-to-point	1.6 GB/s
Aggregate peak/64 proc. frame	44.8 GB/s

Remarks:

Altix 3700 시스템의 구조는 SGI Origin의 구조와 흡사하다. 3700과 거의 동일한 Altix 3300은 이곳에서 다루지 않는다. Origin 시스템과 마찬가지로 Altix도 C-bricks를 가지며, 이곳에는 4개의 Itanium 2 프로세서와 2개의 메모리 모듈, 2개의 I/O 포트 그리고, SHUBs로 불리는 2개의 ASICs를 포함하는 보드가 들어 간다. 각 SHUB는 하나의 메모리 모듈과 하나의 I/O 포트 그리고 두 개의 프로세서가 공유하고 있는 하나의 경로에 연결된다. 그리고 두 SHUBs는 6.4 GB/s의 대역폭으로 서로 연결된다. 메모리 모듈과 I/O포트가 SHUB에 연결되는 대역폭은 각각 10.2와 2.4 GB/s 이다. 다른 bricks와의 연결을 위해 Origin 3000에서와 동일한 라우터와 네트워크가 사용된다. 이른바 Numalink3 네트워크라는 것인데 25.6 GB/s의 bi-section 대역폭을 가진다. Origin처럼 Altix도 주소 공간이 모든 프로세서에게 공유되는 ccNUMA 시스템이다. (프로세서들이 물리적으로 분산되어 있어 메모리 접근이 uniform 하지 않다.) 보드 상에서는 새로운 Numalink4 기술이 사용되어 노드 내에서의 대역폭이 보드 밖으로의 연결과 비교해 대역폭이 크다.

SGI는 고유의 컴파일러를 제공하지 않고 Itanium 프로세서를 위해 인텔 컴파일러를 제공하고 있다. 운영체제도 SGI 고유의 UNIX인 IRIX를 사용하지 않고 대신 Linux를 사용한다. SGI는 Linux에서 구동되는 클러스터 파일 시스템 CXFS를 개발 중이고 곧 사용하게 될 것이다.

64개 프로세서를 가지는 프레임이 다시 Numalink3를 통해 연결되어 클러스터를 구성한다. 4개 프레임까지 단일 시스템 이미지로 구성할 수 있어 프로세서는 256개, 최고 성능이 1.5 Gflop/s에 이르는 시스템이 된다. 즉 OpenMP 병렬화를 통해 최대 256개의 병렬 작업 실행이 가능해 진다. Numalink가 원격 어드레싱을 지원하므로 더 큰 시스템을 구성할 수 있고 이때는 MPI뿐 아니라 일방통신을 위해 Cray의 shmem 라이브러리도 사용할 수 있다. SGI는 머지 않아 더 많은 수의 프로세서로 단일 시스템 이미지를 구성할 수 있을 것으로 보인다.

Measured Performance:

Altix의 64개 프로세서 프레임을 8개 연결한 시스템에서 252,960 크기의 linear system을 푸는데 2,439 Gflop/s가 측정되었다[42]. 이는 79%의 효율에 해당되는 것이다.

3.1.19. The SGI Origin3900

Machine type: RISC-based ccNUMA system.

Models: Origin3900.

Operating system: IRIX (SGI's Unix variant).

Connection structure: Crossbar, hypercube (see remarks).

Compilers: Fortran 77, Fortran 90, HPF, C, C++, ADA, Pascal.

Vendors information Web page:

www.sgi.com/products/servers/origin/3000/overview.html.

Year of introduction: 2003 (With new R16000-based nodes).

System parameters:

Model	Origin3900
Clock cycle	800 MHz
Theor. peak performance	
Per Proc. (64-bits)	1.6 Gflop/s
Maximum (64-bits)	819 Gflop/s
Main Memory	
Memory	≤ 1 TB
No. of processors	16-512
Communication bandwidth	
Point-to-point	1.6 GB/s
Aggregate peak	717 GB/s

Remarks:

Origin3200, Origin3400, Origin3800으로 구성된 새로운 Origin3000 시리즈가 Origin2000 시리즈를 대신하게 된 것은 2000년 7월이었다. 이제 Origin3000 시리즈에는 보다 작은 중간급 모델로 Origin350 미드레인지 서버가 있고 하이-엔드 분야에는 Origin3900만이 남아 있다. Origin2000의 많은 특성들이 그대로 이어졌는데 그 중 가장 중요한 것은 ccNUMA 특성이다. 현재 사용하는 프로세서는 MIPS R16000으로 Origin2000 시스템에서 사용되던 R14000의 후속 프로세서이다. 600 MHz 클럭이 800 MHz가 되었다는 것을 제외하면 R16000은 R14000과 동일하다.

SGI는 Origin3900을 이전 모델보다 더욱 모듈화 했다. 시스템에는 C-bricks라 부르는 CPU 보드가 있고 여기에는 2 ~ 4개의 프로세서와 라우터 칩이 들어간다. 라우터 칩은 프로세서의 온-보드 메모리와 R-bricks, I-bricks를 연결한다. R-bricks는 라우터 보드로 프로세서는 여기를 거쳐 시스템의 나머지 부분과 통신하게 된다. I-bricks는 디스크, PCI 확장 슬롯 등을 가지며 시스템의 I/O 서브 시스템을 구성하게 된다. 최근에 4U 높이의 brick에 16개의 R16000을 넣은 고집약(untra-dense) Cx-brick이 나왔다. 이것은 MIPS 프로세서의 전력소비가 아주 작기 때문에 가능한 것이다. 그래서, Cx bricks를 사용하면 시스템이 차지하는 공간을 대폭 줄일 수 있다.

C-brick 내에서 기본적인 하드웨어 대역폭은 라우터 칩에서 한 쌍의 CPU 사이가 1.6 GB/s, 메모리에서 라우터 칩 사이가 3.2 GB/s(2x1.6 GB/s full duplex) 이다. 동일한 대역폭이 노드 간의 통신에서 유효하다. off-board I/O 대역폭은 2.4 GB/s(2x1.2 GB/s full duplex)이다. R-brick은 16개의 C-bricks와 연결될 수 있고 8개의 포트를 통해 다른 R-bricks와 연결된다. 그래서 이 같은 방법으로 최대 128개의 C-bricks 또는 512개의 프로세서가 연결 가능하다.

Origin3900은 대표적인 ccNUMA 시스템이다. 메모리는 여러 노드 보드에 걸쳐 물리적으로 분산되어 있지만 하나의 시스템 이미지를 가진다. 시스템의 구조 때문에 시스템의 bi-sectional 대역폭은 8개의 프로세서에서 210 GB/s로 일정하다. 이는 Origin2000의 bi-sectional 대역폭이 82 GB/s 이었던 것을 감안하면 크게 개선된 것이다.

병렬화는 컴파일러(Fortran 또는 C)에 의해 자동으로 수행될 수도 있고 사용자가 지시어를 이용해 명시적으로 수행할 수도 있다. 모든 동기화 등의 작업이 메모리를 거쳐야 하므로 잠재적으로는 이것이 큰 병렬부하가 될 수 있다. SGI 버전으로 최적화된 PVM과 MPI 그리고 SGI/Cray-specific shmem 라이브러리를 이용한 메시지 패싱 모델 병렬화도 가능하다. 이와 같은 방식으로 구현된 프로그램들이 매우 효율적으로 실행될 것이다.

Origin이 가지는 큰 장점은 프로세스가 필요로 하는 데이터를 갖고 있는 노드로 해당 프로세스를 옮길 수 있다는 것이다. 이와 같은 방법을 통해 데이터 전송에 의한 부하를 최소화시킨다. 이 기법은 예전의 Kendall Square Systems를 기억나게 한다. SGI는 논-로컬 메모리 참조 시간이 로컬 메모리 참조 시간보다 평균적으로 약 2배 정도 더 크다고 하며, 이는 Origin2000 시리즈와 비교해 50% 정도 개선된 것이다.

Measured Performances:

800 MHz 프로세서 기반의 시스템에 대해서 보고된 성능 측정 결과는 없다. 그러나 참고자료[42]에는 초기의 600 MHz R14000를 512개 장착한 시스템에서 크기가 알려지지 않은 linear system을 푼 성능이 405.6 Gflop/s로 나와있다. 이는 66%의 효율에 해당된다.

3.1.20. The Sun Fire E25K

Machine type: RISC-based shared-memory multiprocessor.

Models: Sun Fire E25K.
Operating system: Solaris (Sun's Unix flavour).
Connection structure: Crossbar (see remarks).
Compilers: Fortran 77, Fortran 90, C, C++.
Vendors information Web page: www.sun.com/servers/highend/sunfire_e25k/index.xml
Year of introduction: 2004.

System parameters:

Model	Sun Fire E25K
Clock cycle	1.2 GHz
Theor. peak performance	
Per Proc. (64-bits)	4.8 Gflop/s
Maximal (64-bits)	345.6 Gflop/s
Main Memory	≤ 576 GB
No. of processors	≤ 72
Communication bandwidth	
Aggregate	≤ 172.8 GB/s

Remarks:
E25K는 3800-15K의 뒤를 이어 나온 시스템으로 1.2 GHz 클럭을 가지는 최신의 UltraSPARC IV 프로세서를 사용한다. 시스템의 구조는 3800-15K와 완전히 동일하다. 실제로는 backplane이 동일하고 프로세서 보드를 바꿔 다른 시스템으로 바꿀 수 있다. 프로세서 보드와 메모리 보드는 backplane(18x18 플랫 크로스바)에 삽입된다. 각 보드는 4개의 1.2 GHz UltraSPARC IV 프로세서와 최대 32 GB 메모리를 가진다. 그래서 보통의 시스템에서 프로세서 개수는 최대 72개까지 가능하다. 이전의 3800-15K 시스템에서는 I/O 용량을 희생해서 컴퓨팅 파워를 확장하기 위해 여분의 2-CPU 보드를 삽입하도록 추가적인 크로스바의 슬롯을 사용하는 것이 가능하다. 그러나 E25K 시스템은 이런 기능을 지원하지 않는다. 플랫 크로스바로 인해 메모리 접근이 일정하며, 크로스바의 aggregate 대역폭은 172.8 GB/s 이른다. 이것은 프로세서당 2.4 GB/s 또는 2 Byte/cycle과 동일한 것이다. 따라서, 8 바이트의 연산수가 프로세서에 올려지기 위해서는 4사이클이 필요하다.
앞선 3800-15K 시스템과 비교해 가장 크고 유일한 차이점이라면, E25K가 듀얼-코어 UltraSPARC IV 프로세서를 장착했다는 것이다. 이것이 프로세서당 최대 성능을 공식적으로 두 배로 만든다. 그렇지만 대역폭 제한으로 인해 실제 유효 성능은 두 배까지 되지 않을 것이다. Sun에서는 하나의 UltraSPARC IV 칩에 두 개의 프로세서 코어가 있어 각각 독립적으로 동작하는 것을 칩 멀티-스레딩(CMT)이라고 하며, 이는 일반적인 의미의 다중 스레딩

과는 좀 다른 의미로 혼동을 이다. 두 개의 코어에서 프로세스 스레드 사이의 스위칭을 할 수 없다는 점에 있어서 그렇다.
E25K는 전형적인 SMP 시스템으로, 시스템 내의 모든 프로세서에 걸쳐 OpenMP를 이용한 공유 메모리 병렬화가 가능하다.

Measured Performances:
총 672개의 프로세서를 가진 7-way 클러스터에서 크기가 알려지지 않은 dense linear system을 푸는데 891.4 Gflop/s의 속도가 참고자료[42]에 보고되어 있다. 이는 효율이 74%에 달하는 것이다.

3.1.21. The Unisys Server ES7000

Machine type: RISC-based shared-memory multiprocessor.
Models: ES7000 Orion 440.
Operating system: Microsoft Windows Server 2003, UnitedLinux.
Connection structure: Crossbar.
Compilers: Fortran 90, C, C++.
Vendors information Web page:
www.unisys.com/products/es7000__linux/hardware/orion__440.htm
Year of introduction: 2004.

System parameters:

Model	Orion 440
Clock cycle	1.5 GHz
Theor. peak performance	
Per Proc. (64-bits)	6.0 Gflop/s
Maximal (64-bits)	192 Gflop/s
Main Memory	≤ 512 GB
No. of processors	≤ 32

Remarks:
Orion 440을 여기에서 다루는 것은 단지 완전한 구성을 위해서이다. 일반적으로 Unisys는 HPC 시장에 그다지 관심이 없었으나 최근 들어 조금씩 관심을 보이고 있다. 위에 정리된 것처럼 Orion 440 시스템은 Windows OS, Linux와 더불어 공급되고 있으며, Bull NovaScale, NEC TX-7과 매우 유사한 플랫 크로스바를 사용한다. 성능 측정결과를 비롯해 더 이상의 알려진 정보는 없다.

4. Systems Disappeared from the List

이미 introduction에서 언급했듯이, 시스템 리스트는 완전한 것이 아니다. 이것은 시장에 나와있지만, 위에서 언급한 시스템들과 매우 유사한 시스템들을 다루지 않았기 때문이다. (예를 들자면, 리스트에 없는 Volvox 시스템은 리스트에 있는 C-DAC 시스템과 매우 유사하지만 똑같지는 않다. 그리고 이와 같은 예는 많다.) 한편으론 전 세계에서 아직 가동중인 시스템들 중에서 많은 양의 시스템들이 다른 원인으로 인해 리스트에서 제외되었다. 가장 중요한 두 가지 이유는 다음과 같다.

1) 시스템이 시장에서 사라졌기 때문으로 일반적으로 두 가지 이유가 있다.
   A. 생산 회사가 파산한 경우.
   B. 생산 회사가 시스템을 다른 타입으로 바꿨거나 같은 타입의 새로운 모델로 대체한 경우.
2) 동일한 타입의 다른 시스템들과 비교해 기술적으로 뒤처져 있어서 더 이상 리스트에 남아있는 것이 무의미 하기 때문이다.

위에서 언급한 두 가지 경우 중 하나에 해당되는 시스템들을 아래에서 정리해 보았다. 이것은 해당 분야의 최신 시스템은 아니지만 여전히 사용되고 있는 시스템들을 찾고 있는 사람들을 위해 의미 있는 일일 것으로 생각된다. 또 최신의 시스템들과 비교해 어느 정도의 성능을 가지는지 그들을 어느 위치에 둘 수 있는지 알아보는 것도 최소한 흥미로운 일이 될 것이다.

위에서 언급한 이유들로 인해 리스트에서 사라져버렸지만 여전히 존재하고 있고 실제로 팔리고 있는 시스템들을 알아보는 것은 좋은 일이다. 그러나 그러한 시스템들이 리스트에서 사라진 보다 근본적인 원인은 더 이상 HPC 분야에서 중요한 역할을 하지 못하기 때문이라고 할 수 있다.

아래에서 정리할 리스트 역시 완전한 것은 아니고 어느 정도 임의적인 것이 사실이다. 데이터는 매우 함축된 형태로, 시스템 명, 시스템 타입, fully configured 시스템의 이론 최대 성능과 사라지게 된 이유 등을 정리해 두었다. 어떤 시스템을 리스트에 넣을 것인가에 대한 판단 여부는 그 시스템이 여전히 관심을 가질만한 것인가 아닌 가에 의해 임의적으로 결정하였다.

도입된 시기와 퇴출된 시기를 알 수 있는 시스템들은 그 시기들도 데이터에 명시해 두었다. 이 시간 간격을 통해서 컴퓨터 산업에서도 아주 동적인 이 분야를 좌우하는 시스템 변천에 대한 실마리를 잡을 수 있을 것이다.

4.1. Disappeared machines

Machine: The Alex AVX 2.
Year of introduction: 1992.
Year of exit: 1997.
Type: RISC-based distributed memory multi-processor.
Theoretical Peak performance: 3.84 Gflop/s.
Reason for disappearance: System is obsolete, there is no new system planned.

Machine: Alliant FX/2800.
Year of introduction: 1989.
Year of exit: 1992.
Type: Shared memory vector-parallel, max.28 processors
Theoretical Peak performance: 1120 Mflop/s.
Reason for disappearance: Manufacturer out of business.

Machine: Avalon A12.
Year of introduction: 1996.
Year of exit: 2000.
Type: RISC-based distributed memory multi-processor, max. 1680 processors.
Theoretical Peak performance: 1.3 Tflop/s.
Reason for disappearance: Avalon is not in business anymore.

Machine:
Year of introduction:
Year of exit:
Type:
Theoretical Peak performance:
Reason for disappearance:

Machine: The AxilSCC.
Year of introduction: 1996.
Year of exit: 1997.
Type: RISC-based distributed-memory system, max. 512 processors.
Theoretical Peak performance: 76.8 Gflop/s.
Reason for disappearance: System is not marketed anymore by Axil.

Machine: BBN TC2000.
Year of introduction: …
Year of exit: 1990.
Type: Virtual shared memory parallel, max. 512 processors.
Theoretical Peak performance: 1 Gflop/s.
Reason for disappearance: Manufacturer has discontinued marketing parallel computer systems.

Machine: Cambridge Parallel Processing DAP Gamma.
Year of introduction: 1986.
Year of exit: 1995.
Type: Distributed-memory processor array system, max. 4096 processors.
Theoretical Peak performance: 1.6 Gflop/s (32-bit).
Reason for disappearance: replaced by newer Gamma II Plus series.

Machine: Cambridge Parallel Processing DAP Gamma II Plus.
Year of introduction: 1995.
Year of exit: 2003.
Type: Distributed-memory processor array system, max. 4096 processors.
Theoretical Peak performance: 2.4 Gflop/s (32-bit).
Reason for disappearance: system became too slow, even for its specialized tasks.

Machine: C-DAC PARAM 9000/SS.
Year of introduction: 1995.
Year of exit: 1997.
Type: RISC-based distributed-memory system, max. 200 processors.
Theoretical Peak performance: 12.0 Gflop/s.
Reason for disappearance: replaced by newer OpenFrame series (see below).

Machine: C-DAC PARAM Openframe series.
Year of introduction: 1996.
Year of exit: 1999.
Type: RISC-based distributed-memory system, max. 1024 processors.
Theoretical Peak performance: Unspecified.
Reason for disappearance: The system is not actively marketed anymore by C-DAC.

Machine: C-DAC PARAM 10000 Openframe series.
Year of introduction: 2000.
Year of exit: 2002.
Type: RISC-based distributed-memory system, max. processors: Unspecified.
Theoretical Peak performance: Unspecified.
Reason for disappearance: The system is replace by the newer C-DAC PARAM Padma.

Machine: Convex SPP-1000/1200/1600.
Year of introduction: 1995 (SPP-1000).
Year of exit: 1996 (SPP-1600).
Type: Distributed memory RISC based system, max. 128 processors.
Theoretical Peak performance: 25.6 Gflop/s
Reason for disappearance: replaced by newer HP 9000 Superdome. series.

Machine: Cray Computer Corporation Cray-2.
Year of introduction: 1982.
Year of exit: 1992.
Type: Shared memory vector-parallel, max. 4 processors.
Theoretical Peak performance: 1.95 Gflop/s
Reason for disappearance: Manufacturer out of business.

Machine: Cray Computer Corporation Cray-3.
Year of introduction: 1993.
Year of exit: 1995.
Type: Shared memory vector-parallel, max. 16 processors.
Theoretical Peak performance: 16 Gflop/s
Reason for disappearance: Manufacturer out of business.

Machine: Cray Research Inc. APP.
Year of introduction: 1994.
Year of exit: 1996.
Type: Shared memory RISC based system, max. 84 processors.
Theoretical Peak performance: 6.7 Gflop/s
Reason for disappearance: Product line discontinued, gap was expected to be filled by the Cray J90 (see below).

Machine: Cray T3D.

Year of introduction: 1994.

Year of exit: 1996.

Type: Distributed memory RISC based system, max. 2048 processors.

Theoretical Peak performance: 307 Gflop/s

Reason for disappearance: replaced by newer Cray T3E (see below).

Machine: Cray T3E Classic.

Year of introduction: 1996.

Year of exit: 1997.

Type: Distributed-memory RISC based system, max. 2048 processors.

Theoretical Peak performance: 1228 Gflop/s.

Reason for disappearance: replaced Cray T3Es with faster clock (see below).

Machine: Cray T3E 1350.

Year of introduction: 2000.

Year of exit: 2003.

Type: Distributed-memory RISC based system, max. 2048 processors.

Theoretical Peak performance: 2938 Gflop/s.

Reason for disappearance: Cray does not market the system anymore.

Machine: Cray J90.

Year of introduction: 1994.

Year of exit: 1998.

Type: Shared-memory vector-parallel, max. 32 processors.

Theoretical Peak performance: 6.4 Gflop/s.

Reason for disappearance: replaced by newer Cray Inc. SV1ex (see below).

Machine: Cray Research Inc. Cray Y-MP, Cray Y-MP M90.

Year of introduction: 1989 (Cray Y-MP).

Year of exit: 1994 (Cray Y-MP M90)..

Type: Shared memory vector-parallel, max. 8 processors.

Theoretical Peak performance: 2.6 Gflop/s

Reason for disappearance: replaced by newer C90 (see below).

Machine: Cray Y-MP C90.

Year of introduction: 1994.

Year of exit: 1996.

Type: Shared memory vector-parallel, max. 16 processors.

Theoretical Peak performance: 16 Gflop/s

Reason for disappearance: replaced by newer T90 (see below).

Machine: Cray T90.

Year of introduction: 1995.

Year of exit: 1998.

Type: Shared-memory vector-parallel, max. 32 processors.

Theoretical Peak performance: 58 Gflop/s.

Reason for disappearance: replaced by newer Cray Inc. SV1ex (see below).

Machine: Cray SV-1(ex).

Year of introduction: 2000.

Year of exit: 2004.

Type: Shared-memory vector-parallel, max. 32 processors.

Theoretical Peak performance: 64 Gflop/s.

Reason for disappearance: replaced by newer Cray Inc. X1.

Machine: Digital Equipment Corp. Alpha farm.

Year of introduction: ---

Year of exit: 1994.

Type: Distributed memory RISC based system, max. 4 processors.

Theoretical Peak performance: 0.8 Gflop/s

Reason for disappearance: replaced by newer HP/Compaq GS series.

Machine: Digital Equipment Corp. AlphaServer 8200 & 8400.

Year of introduction: …

Year of exit: 1998.

Type: Distributed-memory RISC based systems, max. 6 processors (AlphaServer 8200) or 14 (AlphaServer 8400).

Theoretical Peak performance: 7.3 Gflop/s, resp. 17.2 Gflop/s.

Reason for disappearance: replaced by newer HP/Compaq AlphaServer SC45.

Machine: Fujitsu AP1000.

Year of introduction: 1991.

Year of exit: 1996.

Type: Distributed memory RISC based system, max. 1024 processors.

Theoretical Peak performance: 5 Gflop/s.

Reason for disappearance: replaced by the Fujitsu AP3000 (see below).

Machine: Fujitsu AP3000.
Year of introduction: 1996.
Year of exit: 2003.
Type: Distributed memory RISC based system, max. 1024 processors.
Theoretical Peak performance: 614 Gflop/s.
Reason for disappearance: Fujitsu does not market the system anymore in favour of its PrimePower series.

Machine: Fujitsu VPP500 series
Year of introduction: 1993.
Year of exit: 1995.
Type: Distributed memory multi-processor vectorprocessors, max. 222 processors.
Theoretical Peak performance: 355 Gflop/s
Reason for disappearance: replaced by the VPP300/700 series (see below).

Machine: Fujitsu VPP300/700 series.
Year of introduction: 1995/1996.
Year of exit: 1999.
Type: Distributed-memory multi-processor vectorprocessors, max. 256 processors.
Theoretical Peak performance: 614 Gflop/s.
Reason for disappearance: replaced by the VPP5000 series (see below).

Machine: Fujitsu VPP5000 series.
Year of introduction: 1999.
Year of exit: 2002.
Type: Distributed-memory multi-processor vectorprocessors, max. 128 processors.
Theoretical Peak performance: 1.22 Tflop/s.
Reason for disappearance: Fujitsu does not market vector systems anymore, this machine line is replaced by the PRIMEPOWER series.

Machine: Fujitsu VPX200 series
Year of introduction: ---
Year of exit: 1995.
Type: Single-processor vectorprocessors.
Theoretical Peak performance: 5 Gflop/s
Reason for disappearance: replaced by the VPP300/700 series (see above).

Machine: Hitachi S-3800 series.
Year of introduction: 1993.
Year of exit: 1998.
Type: Shared-memory multi-processor vectorprocessors, max. 4 processors.
Theoretical Peak performance: 32 Gflop/s.
Reason for disappearance: Replaced by the newer SR11000 system.

Machine: Hitachi S-3600 series.
Year of introduction: 1994.
Year of exit: 1999.
Type: Single-processor vectorprocessor.
Theoretical Peak performance: 2 Gflop/s
Reason for disappearance: Replaced by the newer SR11000 system.

Machine: Hitachi SR2001 series.
Year of introduction: 1994.
Year of exit: 1996.
Type: Distributed memory RISC based system, max. 128 processors.
Theoretical Peak performance: 23 Gflop/s
Reason for disappearance: Replaced by the successor SR2201 (see below).

Machine: Hitachi SR2201 series.
Year of introduction: 1996.
Year of exit: 1998.
Type: Distributed-memory RISC based system, max. 1024 processors.
Theoretical Peak performance: 307 Gflop/s.
Reason for disappearance: Replaced by the newer SR8000 (see below).

Machine: Hitachi SR8000 series.
Year of introduction: 1998.
Year of exit: 1998-2003 (different models).
Type: Distributed-memory RISC based system, max. 512 processors.
Theoretical Peak performance: 7.3 Tflop/s.
Reason for disappearance: Replaced by the newer SR11000.

Machine: HP/Convex C4600.
Year of introduction: 1994.
Year of exit: 1997.
Type: Shared memory vector-parallel, max. 4 processors (C4640).
Theoretical Peak performance: 3.2 Gflop/s

Reason for disappearance: The C4600 is not marketed by HP/Convex anymore.

Machine: The HP Exemplar V2600.
Year of introduction: 1999.
Year of exit: 2000.
Type: Distributed-memory RISC based system, max. 128 processors.
Theoretical Peak performance: 291 Gflop/s.
Reason for disappearance: Replaced by the HP 9000 Superdome.

Machine: IBM ES/9000 series.
Year of introduction: 1991.
Year of exit: 1994.
Type: Shared memory vector-parallel system, max. 6 processors.
Theoretical Peak performance: 2.67 Gflop/s
Reason for disappearance: IBM does not pursue high-performance computing by this product line anymore.

Machine: IBM SP1 series.
Year of introduction: 1992.
Year of exit: 1994.
Type: Distributed-memory RISC based system, max. 64 processors.
Theoretical Peak performance: 8 Gflop/s.
Reason for disappearance: Replaced by the newer RS/6000 SP series (see below).

Machine: IBM RS/6000 SP series.
Year of introduction: 1999.
Year of exit: 2001.
Type: Distributed-memory RISC based system, max. 2048 processors.
Theoretical Peak performance: 24 Gflop/s.
Reason for disappearance: Replaced by the newer eServer p690.

Machine: Intel Paragon XP.
Year of introduction: 1992.
Year of exit: 1996.
Type: Distributed memory RISC based system, max. 4000 processors.
Theoretical Peak performance: 300 Gflop/s.
Reason for disappearance: Except for a non-commercial research system (the ASCI Option Red system at Sandia National Labs.) Intel is not in the business of high-performance computing anymore.

Machine: Kendall Square Research KSR2.
Year of introduction: 1992.
Year of exit: 1994.
Type: Virtually shared memory parallel, max. 1088 processors.
Theoretical Peak performance: 400 Gflop/s
Reason for disappearance: Kendall Square has terminated its business.

Machine: Kongsberg Informasjonskontroll SCALI.
Year of introduction: 1996.
Year of exit: 1997.
Type: Distributed memory RISC based system, max. 512 processors.
Theoretical Peak performance: 76.8 Gflop/s
Reason for disappearance: Kongsberg does not market the system anymore.

Machine: MasPar MP-1, MP-2.
Year of introduction: 1991 (MP-1).
Year of exit: 1996.
Type: Distributed memory processor array system, max. 16384 processors.
Theoretical Peak performance: 2.4 Gflop/s (64-bit, MP-2)
Reason for disappearance: Systems are not marketed anymore.

Machine: Matsushita ADENART.
Year of introduction: 1991.
Year of exit: 1997.
Type: Distributed memory RISC based system, 256 processors.
Theoretical Peak performance: 2.56 Gflop/s.
Reason for disappearance: Machine is obsolete and no new systems are developed in this line.

Machine: Meiko CS-1 series.
Year of introduction: 1989.
Year of exit: 1995.
Type: Distributed memory RISC based system.
Theoretical Peak performance: 80 Mflop/s per processor
Reason for disappearance: Meiko does not build complete systems anymore (but see below).

Machine: Meiko CS-2 series.
Year of introduction: 1994.
Year of exit: 1999.

Type: Distributed-memory RISC based system, max. 1034 processors.
Theoretical Peak performance: 200 Mflop/s per processor.
Theoretical Peak performance: 204.8 Gflop/s.
Reason for disappearance: Quadrics Supercomputers World Ltd. does not market the system anymore. The updated network technology is now offered for other systems like HP/Compaq's AlphaServer SC45.

Machine: nCUBE 2S.
Year of introduction: 1993. Year of exit: 1998.
Type: Distributed memory system, max. 8192 processors.
Theoretical Peak performance: 19.7 Gflop/s.
Reason for disappearance: NCUBE has withdrawn from the scientific and technical market. The nCUBE2S is now offered as a parallel multimedia server.

Machine: nCUBE 3.
Year of introduction: --- (see remarks at nCUBE 2S).
Year of exit: ---
Type: Distributed memory system, max. 10244 processors.
Theoretical Peak performance: 1 Tflop/s.
Reason for disappearance: Was announced several times but was never finished.

Machine: NEC Cenju-3.
Year of introduction: 1994.
Year of exit: 1996.
Type: Distributed-memory system, max. 256 processors.
Theoretical Peak performance: 12.8 Gflop/s.
Reason for disappearance: replaced by newer Cenju-4 series (see below).

Machine: NEC Cenju-4.
Year of introduction: 1998.
Year of exit: 2002.
Type: Distributed-memory system, max. 1024 processors.
Theoretical Peak performance: 410 Gflop/s.
Reason for disappearance: NEC has withdrawn this machine in favour of a possible successor. Specifics are not known, however.

Machine: NEC SX-3R.
Year of introduction: 1993.
Year of exit: 1996.
Type: Shared memory multi-processor vector processors, max. 4 processors.
Theoretical Peak performance: 25.6 Gflop/s
Reason for disappearance: replaced by newer SX-4 series (see below).

Machine: NEC SX-4.
Year of introduction: 1995.
Year of exit: 1998.
Type: Shared memory multi-processor vector processors, max. 256 processors.
Theoretical Peak performance: 1 Tflop/s.
Reason for disappearance: replaced by newer SX-5 series (see below).

Machine: NEC SX-5.
Year of introduction: 1998.
Year of exit: 2002.
Type: Shared memory multi-processor vector processors, max. 512 processors.
Theoretical Peak performance: 5.12 Tflop/s.
Reason for disappearance: replaced by newer SX-6 series.

Machine: Parsys SN9000 series.
Year of introduction: 1993.
Year of exit: 1995.
Type: Distributed memory RISC based system, max. 2048 processors.
Theoretical Peak performance: 51.2 Gflop/s
Reason for disappearance: Replaced by the newer TA9000 (but see below).

Machine: Parsys TA9000 series.
Year of introduction: 1995.
Year of exit: 1996.
Type: Distributed memory RISC based system, max. 512 processors.
Theoretical Peak performance: 119.3 Gflop/s
Reason for disappearance: Parsys does not offer complete system anymore. Instead it sells node cards based on the TA9000 for embedded systems.

Machine: Parsys GC/Power Plus.
Year of introduction: 1993.
Year of exit: 1996.
Type: Distributed memory RISC based system.
Theoretical Peak performance: 266.6 Mflop/s per processor.
Reason for disappearance: System has been replaced by the Parsytec CC systems (see below).

Machine: Parsytec CC series.
Year of introduction: 1996.
Year of exit: 1998.
Type: Distributed memory RISC based system.
Theoretical Peak performance: unspecified.
Reason for disappearance: Vendor has withdrawn from the High-Performance computing market.

Machine: Siemens-Nixdorf VP2600 series.
Year of introduction: ---
Year of exit: 1995.
Type: Single-processor vectorprocessors.
Theoretical Peak performance: 5 Gflop/s
Reason for disappearance: eventually replaced by the PRIMEPOWER series.

Machine: Silicon Graphics PowerChallenge.
Year of introduction: 1994.
Year of exit: 1996.
Type: Shared memory multi-processor, max. 36 processors.
Theoretical Peak performance: 14.4 Gflop/s
Reason for disappearance: replaced by the SGI Origin 2000 (see below).

Machine: SGI Origin2000.
Year of introduction: 1996.
Year of exit: 2000.
Type: Shared-memory multi-processor, max. 128 processors.
Theoretical Peak performance: 102.4 Gflop/s.
Reason for disappearance: replaced by the SGI Origin 3900.

Machine: Stern Computing Systems SSP.
Year of introduction: 1994.
Year of exit: 1996.
Type: Shared memory multi-processor, max. 6 processors.
Theoretical Peak performance: 2 Gflop/s
Reason for disappearance: Vendor terminated its business just before delivering first systems.

Machine: SUN E10000 Starfire.
Year of introduction: 1997.
Year of exit: 2001.
Type: Shared-memory multi-processor, max. 64 processors.
Theoretical Peak performance: 51.2 Gflop/s.
Reason for disappearance: replaced by the SUN Fire 3800-15K (see below).

Machine: SUN Fire 3800-15K.
Year of introduction: 2001.
Year of exit: 2004.
Type: Shared-memory multi-processor, max. 106 processors.
Theoretical Peak performance: 254 Gflop/s.
Reason for disappearance: replaced by the newer SUN Fire E25K.

Machine: Thinking Machine Corporation CM-2(00).
Year of introduction: 1987.
Year of exit: 1991.
Type: SIMD parallel machine with hypercube structure, max. 64K processors.
Theoretical Peak performance: 31 Gflop/s
Reason for disappearance: was replaced by the newer CM-5 (but see below).

Machine: Thinking Machine Corporation CM-5.
Year of introduction: 1991.
Year of exit: 1996.
Type: Distributed-memory RISC based system, max. 16K processors.
Theoretical Peak performance: 2 Tflop/s.
Reason for disappearance: Thinking Machine Corporation has stopped manufacturing hardware and hopes to keep alive as a software vendor.

5. Systems under development

실험적이고 특별한 목적을 가지는 시스템이나 아직 불확실한 시스템이 아닌 현실적이고 시장에 나올 수 있는 시스템에 대해 알아보고자 하고 있지만, 아직 개발 중이며 그래서 시장에 나올 수 있는 기회를 가지고 있는 시스템들을 이곳에서 다루고자 한다. 앞에서 이미 발표되고 6개월 내에 시장에 나올 수 있는 시스템들에 대해 다룰 것이라는 규칙을 정한바 있다. 여기서 설명하는 시스템들은 이 보고서가 완성되는 시점으로 1년 이내에 시장에 나타날 것이다. 그러나 일부 업체에서는 그들의 새로운 시스템이 판매되어 설치되기 전에 그 시스템의 자세한 데이터가 알려지는 것을 원하지 않을 수 있다. 그들은 해당 정보를 밝히지 않을 것이고 우리는 그런 업체들의 뜻을 충분히 알고 있다(단, 잠재적인 고객들에게 지나치게 긴 시간 기대하게 하는 것이 현명한 일은 아니다.).

지금부터 6개월에서 일년 이내에 시장에 나오게 될 상업적인 시스템을 선도할 시스템에 대해 아래에서 알아볼 것이다. 상업적인 시스템들은 대상이 되는 사용자 그룹이나 개발되는 방식(미국과 일본의 접근 방식이 현저히 다르다.)에 따라 원래의 연구 모델들과는 많이 다를 수 있다.

최근의 주목할 만한 사건은 Intel의 IA-64 Itanium 프로세서 계열이 소개된 것이다. 이미 4개의 업체에서 Itanium 2 기반의 시스템을 제공하고 있다. HP에서는 Itanium 프로세서 계열로 인해 Alpha와 PA-RISC 기반 시스템의 영업을 종료할 것으로 알려져 있다. 동시에 SGI도 MIPS 기반의 시스템 개발을 더 이상 하지 않을 것이다. 이것은 HPC 시스템을 위한 RISC 프로세서를 계속 생산하게 되는 업체는 앞으로 수년 내에 AMD와 IBM, Intel과 SUN만이 남게 될 것이라는 것을 의미한다. 한편으로, 일본의 Earth Simulator 시스템의 등장으로 인한 충격으로 미국이 또 다른 프로세서와 컴퓨터 아키텍처 연구를 위해 투자를 아끼지 않을 것으로 보여진다. 실제로 일부에서는 이미 그러한 계획이 시작되어 진행 중이기는 하지만, IBM의 BlueGene을 제외하고는 1, 2년 내에 가시적인 결과를 보기는 어려울 것이다.

5.1. Cray Inc.

2002년 말에 Cray사의 차세대 벡터 프로세서, X1이 시스템에 장착되기 시작하였다. X1은 Cray SV-1s에서 발견할 수 있는 기술들을 기반으로 하였다. Cray는 약 2010년까지 앞으로의 시스템들에 대한 로드맵을 널리 광고하고 있다. 그들 중 얼마나 많은 시스템들이 실현될지는 두고 봐야 하겠지만, 적어도 2종의 시스템은 2005년에 시장에 나올 것이 확실하다. 그 중 하나는 Cray X1e로 현재의 X1 시스템과 비교해 프로세서 클럭이 800 MHz에서 1.2 GHz로 향상된 것 이외에는 다른 것이 아무것도 없다. 다른 하나는 Cray Strider로 Cray가 Sandia Laboratories에 설치한 AMD Opteron(11,648개 프로세서) 기반의 Red Storm 시스템의 상업용 버전이다. 이 시스템은 값싼 프로세서와 AMD의 HyperTransport, Cray의 SeaStar router ASIC를 기반으로 하는 고속 네트워크로 인해 많은 관심을 끌고 있다.

X1e를 뒤이어 Black Widow가 2006년경에 시장에 나올 것으로 계획 중이다. 최근에는 코드명 “Rainier” 불리는 새로운 타입의 시스템 개발 계획이 알려졌다. 이 시스템에서는 프로세서 사이의 네트워크가 중요 부분이고 이 네트워크에서는 벡터 타입(Black Widow), 스칼라 타입(AMD or other) 또는 특별한 함수를 위한 FPGAs/DSPs 등과 같은 서로 다른 타입의 노드가 혼합되어 사용될 수 있다. Red Storm 네트워크를 더욱 업그레이드시킨 형태가 이와 같은 시스템을 위해 매우 적합해 보이기는 하지만 실현되기에는 아직 시간이 더 필요해 보이고 기술적인 문제에 부닥치게 될 것은 자명해 보인다.

5.2. Hewlett-Packard/Intel

HP와 Intel은 Itanium 프로세서 패밀리를 통해 다가올 수년 동안 아주 큰 영향력을 가지게 될 것이다. Itanium을 기반으로 하는 듀얼 코어 프로세서가 이미 drawing board상에 있고 1, 2년 이내에 시장을 강타할 것이다. 일반적으로 듀얼 코어 프로세서는 동일한 싱글 코어 프로세서에 비해 상대적으로 좋지 않은 성능을 가지므로 듀얼 코어 프로세서에 의한 성능 향상이 그다지 크지는 않을 것이다. 이보다는 시스템 아키텍처가 더 중요할 것이다. 또한 프로세서 자체의 변화가 성능을 향상시킬 것으로 보인다. HP/Intel이 가지고 있는 VLIW 프로세서에 대한 경험으로 인해, 사람들은 보다 긴 명령어 word를 가지는 프로세서 개발 쪽으로 연구 방향이 맞춰질 것을 기대한다. 그리고 FFT나 sparse Matrix-Vector 곱셈과 같은 고수준 연산을 위한 특별한 장치를 포함할 가능성에 대한 기대도 있다. 그렇지만 이와 같은 개선이 미래의 시스템에서 언제 어떻게 실현될지는 불확실하다. 분명한 것은 앞으로 2년 내에는 어렵다는 것이다. HP의 drawing board상에 올라온 시스템 중에 급진적으로 다른 형태의 아키텍처를 가지는 것은 아직 없다. 대신 HP는 이미 Itanium 기반의 대형 시스템들을 설치해 오고 있는 클러스터 분야에 더 많은 관심을 기울이고 있다.

5.3. IBM

IBM은 이미 몇 년 동안 Blue Gene 시스템에 힘을 쏟고 있다. 첫 시스템으로 Blue Gene/L이 2004년에 나올 것으로 보인다. 이 시스템은 700 MHz 클럭과 두 개의 부동소수 유닛을 가지는 프로세서 65,536개로 이루어질 것이다. 각 부동소수 유닛은 지금의 POWER4와 마찬가지로 fused multiply-add가 가능해 이론 최대 성능이 183.5 Tflop/s에 이른다. 또 다른 후속 Blue Gene 시스템들로 최대 성능이 1Pflop/s인 Blue Gene/P와 최대 성능이 3 Pflop/s인 Blue Gene/Q가 계획 중이다. 이 모든 시스템들은 HPC 사용자들을 주 대상으로 해서 개발 되고 있지만, 일반 시장에서도 사용 가능하도록 적절한 아키텍처 특성을 마련하려고 노력할 것이다.

물론 POWER4 프로세서의 개발도 계속될 것이다. POWER5 프로세서는 POWER4를 능가하는 기술적인 장점을 가지게 될 것이다. POWER5에서 고려되고 있는 것은 프로세서 8개를

묶어 60 ~ 80 Gflop/s에 이르는 하나의 가상 벡터 프로세서를 구성하는 방법이다. 이것을 ViVA(Virtual Vector Architecture)라고 부른다. 이 같은 방법은 Hitachi의 SR8000 프로세서나 Cray X1에서 사용된 MSP 프로세서를 연상 시킨다. ViVA의 구현은 2004/2005년에 POWER5가 나오고도 몇 년은 더 걸려야 가능할 것이다.

5.4. SGI

SGI가 MIPS기반 시스템 생산을 중단할 것이라는 사실이 최근에 알려지고 있다. 따라서 SGI는 Itanium 기반의 시스템을 제공하는 다른 업체들과의 차별성을 SGI 시스템의 macro-architecture에 두고자 한다. 시스템 아키텍처의 주요 개선 방향은 네트워크(Numalink3를 Numalink4 및 그 이상으로 더욱 개선)의 속도와 하나의 시스템 이미지 내에 대량의 프로세서를 가지는 시스템을 구성하는 것 등이다. 그 점에 있어서 SGI는 이미 미래의 Altix x000 시스템으로 확장됐을지도 모르는 MIPS 기반 Origin 3000 시스템에 대한 경험이 있다.

5.5. SRC

SRC사는 당대의 최대 속도를 끌어 올리는 경향을 보인다. SRC는 Field Programmable Gate Arrays(FPGAs)를 통해 일반 프로세서들을 보완하는 기술을 사용하고, 이를 통해 사용자가 정의하는 특별한 작업들을 위한 시스템 구성을 가능하게 한다. 이 기술을 사용하면 동일한 작업에 대해 일반 프로세서를 사용하는 것보다 현저하게 빠른 결과를 얻을 수 있다. SRC가 제공하는 SRC-6 시스템은 표준 IA-32 프로세서를 가지는 256개의 듀얼 프로세서 보드로 구성되며 각각의 보드는 하나의 Multi-Adaptive Processor(MAP) 유닛으로 연결된다. MAP는 FPGA, 개별 메모리, MAP 컨트롤러와 필요할 때 유닛을 재설정하는 logic으로 구성된다. MAP들은 링 네트워크로 서로 연결되고 로컬 메모리를 가지는 표준 프로세서 보드는 MAPs를 거쳐 읽기와 쓰기 크로스바를 통해 글로벌 메모리로 연결된다.

6. Glossary of terms

자주 사용되지만 본문에서 설명되지 않아 이해가 어렵거나 지나치게 광범위하게 설명되어 짧은 설명이 이해에 더 도움을 줄 수 있는 용어들이 있다. 이런 용어들에 대한 설명을 이곳에 실어 둔다.

Architecture: 컴퓨터 시스템이나 칩의 내부 구조. 아키텍처에 따라 해당 시스템이나 칩의 연산 기능과 성능이 결정된다.

Architectural class: 아키텍처에 따른 컴퓨터 시스템의 분류. 분산 메모리 MIMD, SMP 등의 구분이 이에 해당하며, 여러 가지 클래스들에 대한 설명은 이 용어집과 'architecture' 섹션에 나와 있다.

ASCI: Accelerated Strategic Computer Initiative의 약자로 고성능 시스템의 생산과 연구를 위한 미국의 대규모 펀딩 프로젝트. 이 프로젝트의 시작 동기는 미국이 핵실험을 실제로 수행하지 않고 전산 모의실험으로 대체하기 위한 것으로 알려져 있다. ASCI는 고성능 시스템이 SMP 클러스터 형태 한 방향으로 발전해 가도록 큰 영향을 주었다.

Bank cycle time: (캐시)메모리 뱅크에 요청된 데이터 접근을 처리하고 다음 요청을 받을 때까지 걸리는 최소 시간. 이 시간 동안 해당 뱅크에 다른 요청은 허용되지 않는다.

Beowulf cluster: PC나 워크스테이션을 사설 네트워크로 연결한 클러스터. 처음에는 통합 병렬 시스템의 값싼 대안으로 PC들을 Linux와 Ethernet을 이용해 연결한 DIY 클러스터를 지칭하는 용어였다. 현재는 그 정의가 광범위해져 O/S로 Linux나 Windows를 사용하면서 고속 스위치 네트워크와 RISC 기반 프로세서를 사용하고, 업체에서 미리 제작 완성한 rack-mounted 시스템에도 이 용어를 사용하고 있다.

Bit-serial: 바이트 또는 4/8 바이트 데이터에 대한 연산이 아닌 비트 기반의 데이터 연산. 한 번에 한 비트씩 처리된다. 프로세서 배열 시스템에서 신호처리나 영상처리를 위해 bit-serial 연산이 유용하게 쓰인다.

Cache --- data, instruction: CPU에 인접해 있는 작고 빠른 메모리로 처리될 데이터나 명령어들의 일부를 저장해 둘 수 있다. 주 캐시(레벨 1 이라고도 함)는 명령어 캐시와 데이터 캐시로 나뉘어지고 대부분의 경우 CPU와 동일한 칩에 위치한다. 레벨 2 캐시는 대부분 칩 밖에 위치하며 데이터와 명령어를 모두 저장할 수 있다. 캐시는 데이터를 메모리에서 꺼내올 때 발생하는 지연시간을 감추기 위해 사용한다, 필요할 것 같은 데이터나 명령어를 캐시

에 미리 올려 둠으로써 메모리 접근에 소비되는 지연시간을 대폭 감소 시킬 수 있다.

Capability computing: 대형 컴퓨팅의 일종으로 매우 크고 시간이 많이 소비되는 대형 작업들 자체의 실행 성능에 초점을 맞춘 컴퓨팅. 실행 작업의 효율성에 초점을 맞추므로 해당 시스템의 전체 컴퓨팅 자원 효율성은 최고로 유지되지 않을 수 있다.

Capacity computing: 대형 컴퓨팅의 일종으로 시스템 자원을 가능한 효과적으로 사용해 시스템 전체의 처리 용량이 최고가 되도록 하는 컴퓨팅. 시스템의 전체 사용을 최적화해서 개별 컴퓨팅 작업의 성능에는 오히려 좋지 않은 결과를 낼 수도 있다.

ccNUMA: Cache Coherent Non-Uniform Memory Access. 이와 같은 형태의 메모리 접근을 지원하는 시스템의 메모리는 물리적으로 분산되었지만 논리적으로는 공유된다. 데이터의 위치가 물리적으로 차이가 있기 때문에 데이터 요청에 걸리는 시간은 해당 데이터의 위치에 따라 달라지게 된다. 이 시스템에서는 메모리와 캐시가 모두 분산되어 있어서 시스템 전체에 걸쳐 데이터 일관성을 유지할 필요가 있으며, 이를 위해 디렉터리 메모리, 스누피 버스 프로토콜 등의 다양한 기법들이 존재한다. 이런 기법들 중 하나가 구현되어 캐시 메모리 일관성을 유지하는 시스템을 cache coherent 라고 한다.

Clock cycle: 컴퓨터의 기본 시간 단위. 컴퓨터에 의해 수행되는 모든 연산은 적어도 1에서 수 사이클이 걸리게 된다. 대체로 클럭 사이클은 1에서 수 나노초 범위이다.

Clock frequency: 클럭 사이클의 역수. 단위시간당 사이클 수가 Hz로 표현된다. 요즘의 클럭 주파수는 대체적으로 400 MHz ~ 1 GHz 이다.

Clos network: 로그함수적인 네트워크로 노드는 스위치에 연결된다. 이 네트워크에서 스위치는 모든 노드를 연결하는 이른바, 척추가 된다.

Communication latency: 메시지가 통신 네트워크를 거쳐 한 프로세서에서 다른 프로세서로 전송될 때 발생하는 시간 부하. 대체적으로 Infiniband나 Myrinet 같은 특별하게 설계된 네트워크에서의 통신지연은 수 μs 범위이고, (Gigabit) Ethernet의 경우는 약 100 μs 정도 이다.

Control processor: 프로세서 배열 시스템에서 모든 프로세서에 의해 실행될 명령어를 issue 하는 프로세서. 한편으로는 I/O 연산이나 순차 연산과 같이 프로세서 배열에 있는 프로세서들이 수행하지 못하는 작업을 수행할 수도 있다.

CRC: 데이터를 큰 이진수로 처리하는 것에 기반해 오류를 검색하고 수정하는 방법의 하나. 데이터 이진수를 다른 고정 이진수로 나눈 나머지를 정확성과 오류 회복을 위한 검사합(checksum)으로 이용한다. SCI 네트워크에서 CRC 오류 검사를 이용한다.

Crossbar (multistage): 다른 포트로부터의 메시지 간섭이 없도록 모든 입력 포트가 모든 출력 포트에 직접적으로 연결된 네트워크. 1단 크로스바에서 이런 직접연결은 컴퓨터 시스템의 모든 메모리 모듈이 모든 CPU와 직접 연결되는 효과를 가진다. 이것은 다중 CPU 벡터 시스템에서 흔히 볼 수 있다. 다단 크로스바 네트워크에서는 한 크로스바 모듈의 출력 포트들이 다른 크로스바 모듈들의 입력 포트들과 연결된다. 이런 방법으로 거대 네트워크 구축에 드는 비용을 감소시키며, 복잡성이 로그함수적으로 증가하는 네트워크를 구축할 수 있다.

Distributed Memory (DM): 시스템 메모리가 시스템 내의 여러 노드에 걸쳐 분산되어 있는 시스템 아키텍처 클래스. 시스템에 있는 데이터에 대한 접근은 노드들을 연결하는 네트워크를 거치게 된다. 이때 메시지 패싱을 통한 명시적인 접근이나 암시적인 접근이 가능하다. 암시적인 데이터 접근은 HPF를 이용하거나 ccNUMA 시스템에서는 자동으로 수행된다.

Dual core chip: 두 개의 CPU와 캐시(공유될 수도 있음)를 가지는 하나의 칩. 집적 수준의 발달로 인해 더 많은 장치들이 하나의 칩에 올라갈 수 있다. 실제로 IBM에서 듀얼 코어 칩(POWER4)을 생산하고 있으며, 다른 업체들도 곧 따라 갈 것으로 보인다.

EPIC: Explicitly Parallel Instruction Computing. Intel의 IA-64 칩과 이를 위해 정의된 명령어 집합을 지칭한다. EPIC은 향상된 VLIW(Very Large Instruction Word) 컴퓨팅으로 볼 수 있다. 주요 골자는 RISC 프로세서와 같이 명령어에 대해 동적 스케줄링을 하지 않고 코드의 추정 실행과 명령어 스케줄링이 프로그램의 컴파일 단계에서 미리 결정되는 것이다. 이로 인해 칩 설계는 간단해 지고, 한편으로는 많은 명령어들이 병렬로 실행될 수 있는 가능성을 가진다.

Fat tree: 이분 (사분) 트리 구조를 가지는 네트워크. 루트 근처의 가용 대역폭이 트리의 잎 근처보다 더 크도록 수정되었다. 트리 구조에서 루트 프로세서로부터 다른 모든 프로세서로 데이터를 gather 하거나 broadcast 하는 경우, 이와 같은 수정이 없다면 루트 근처에서 경합(contention) 또는 병목이 심해질 것이다.

FPGA: Field Programmable Gate Array. 사용자가 지정한 작업을 수행하도록 하드웨어적으로 프로그램될 수 있는 논리 게이트의 배열이다. 이를 통해 사용자는 어떤 제한적인 작업에 매우 효과적일 수 있는 특별한 목적의 functional 유닛을 고안할 수 있다. FPGA들은 동적으로 재설정될 수 있어서(초당 100 ~ 1000번 재설정 가능), 더욱 복잡하고 특별한 작업에

대해 일반적인 프로세서들로 얻을 수 있는 것보다 더 높은 속도를 얻도록 최적화하는 것이 이론적으로 가능하다.

Functional unit: 미리 정의된 함수의 실행을 전담하는 CPU 내의 유닛. 주 캐시에서 데이터를 로딩하는 유닛, 부동소수 덧셈을 수행하는 유닛 등이 있다.

Grid --- 2-D, 3-D: 노드들이 2-D나 3-D 그리드 레이아웃으로 연결된 네트워크 구조. 대부분의 경우에 있어 그리드의 끝점들은 다시 시작점으로 연결되어 2-D나 3-D 토러스 구조를 형성한다.

HBA: Host Bus Adapter. 외부 네트워크의 한 부분으로 네트워크 자체와 계산 노드의 PCI 버스 사이 인터페이스를 구성한다. HBA는 통신의 시작, 버퍼링, 정확성 검사 등에 대해 상당한 양의 프로세싱 정보를 제공한다. HBA는 Infiniband에서는 HCA 또는 TCA, Myrinet에서는 LANai, QsNet에서는 ELAN 등으로 네트워크에 따라 달리 호칭된다.

HPF: High Performance Fortran. Fortran 프로그램을 분산 메모리 시스템 상에서 공유 메모리 시스템에서와 같이 실행시킬 수 있는 컴파일러와 런타임 시스템. 데이터 분배, 프로세서 레이아웃 등이 주석 지시어(comment directives)로 기술되어 순차적으로 실행하는 것이 가능하다. 현재 상업적으로 가용한 HPF는 간단한 분배 체계와 모든 프로세서들의 동시에 동일한 코드를 실행하는 SPMD 스타일만을 허용하고 있다.

Hypercube: 일반화된 큐브 구조를 가지는 네트워크. 로그함수적인 복잡성을 가진다. 현재 구조의 경계선을 두 배로 하고 꼭지점을 오리지널 구조와 연결해 다음 차원의 하이퍼큐브를 얻는다.

Instruction Set Architecture: CPU가 실행하도록 설계된 명령어들의 집합. ISA는 설계자가 해당 CPU를 위해 적합하다고 결정한 명령어들의 목록을 나타낸다. 다르게 만들어진 CPU들도 동일한 ISA를 가질 수 있다. 예를 들어, AMD 프로세서는 다른 구조를 가지면서 Intel IA-32 ISA를 실행하고 있다.

Memory bank: 전체 메모리 뱅크들의 집합에서 연속적으로 어드레스되는 (캐시) 메모리의 부분. 데이터 a(n)이 뱅크 b에 저장될 때 a(n+1)은 뱅크 b+1에 저장된다. (캐시) 메모리는 여러 뱅크들로 나눠져 뱅크 사이클 시간에 의한 효과를 피하고 있다. 즉, 데이터가 연속적으로 저장되고 꺼내어 질 때 각 뱅크는 해당 뱅크에 다음 요청이 도착하기 전에 충분한 회복시간을 가지게 되는 것이다.

Message passing: 분산 메모리 시스템을 위한 병렬 프로그래밍 형태. 명시적으로 요구되는 논로컬 데이터가 적절한 송신과 수신 메시지에 의해 필요한 프로세서로 전송되어야 한다.

MPI: Message Passing Interface. 메시지 패싱 라이브러리의 하나로 메시지 패싱 스타일의 프로그래밍을 구현한다. 현재 MPI는 메시지 패싱 스타일의 프로그래밍을 위한 사실상의 표준이다.

OpenMP: 공유 메모리 시스템과 SMP에서 병렬로 실행될 수 있는 공유 메모리 병렬 프로그래밍 모델. 병렬화를 주석 지시어(Fortran) 또는 pragma(C/C++)로 제어 해서 동일한 프로그램을 별다른 수정 없이 순차 시스템에서도 실행할 수 있다.

PCI bus: PC 노드의 버스. 대체적으로 I/O를 위해 사용되며, 통신 네트워크와 노드를 연결하기 위해서도 사용된다. 형태에 따라 110 ~ 480 MB/s의 대역폭을 가진다. 새롭게 업그레이드된 PCI-X와 PCI Express가 곧 사용 가능해 진다.

Pipelining: 명령어를 전체 실행하는 데 많은 사이클이 걸리더라도, 매 사이클마다 새로운 연산수를 받을 수 있도록 Functional 유닛을 구분하는 것. 파이프라인 구조는 컨베이어 벨트가 파이프라인이 채워질 때까지 유닛을 받아서 매 사이클마다 결과를 내놓는 것과 같이 동작한다.

Processor array: 간단한 단일 프로세서들의 배열(대부분 2-D 그리드)이 프로그램 명령어를 동시 동작(lock-step)으로 실행하는 시스템. 명령어 실행을 제어하는 제어 프로세서가 있다.

PVM: 널리 사용되어 온 또 다른 메시지 패싱 라이브러리. 원래는 워크스테이션들을 연결한 환경에서 실행하도록 개발되었고 작업을 실행하면서 프로세스를 동적으로 생성하고 제거하는 것이 가능하다. 현재 PVM은 대부분 MPI로 대체되고 있다.

Register file: CPU내의 프로세서 레지스터 집합. 0/1과 같은 상수 저장을 위한 레지스터, 중간 결과를 재명명하기 위한 레지스터 그리고, 경우에 따라서는 함수 인수와 루틴의 리턴 주소들을 저장하기 위한 분리된 레지스터 스택 등이 보완된다.

RISC: Reduced Instruction Set Computer. 이전의 Complex Instruction Set Computers(CISCs)와 비교해 보다 단순해진 명령어 집합을 가지는 CPU. 명령어 집합이 이상적으로는 1 사이클에 실행되는 간단한 명령어들로 축소되었다.

Shared Memory (SM): 모든 프로세서들이 시스템내의 모든 메모리에 직접 연결되는 컴퓨터의 메모리 구성. 공유되는 대역폭에 대한 기술적 한계로 인해 공통의 메모리를 공유하는 프로세서는 일반적으로 16개 보다 많지 않다.

SMP: Symmetric Multi-Processing. 보다 큰 시스템의 일부가 되는 공유 메모리 계산 노드를 지칭하는 용어로 흔히 사용되며 이런 노드들을 연결해 전체 시스템을 구성한다. 이 노드들이 ccNUMA 시스템으로 구성되거나 또는 분산 메모리 시스템으로 구성될 수 있다. 이때 분산 메모리 시스템의 SMP 노드에서는 OpenMP로 프로그램 될 수 있지만, 노드와 노드 사이의 통신은 메시지 패싱으로 구현되어야 한다.

TLB: Translation Look-aside Buffer. 프로그램 코드에서 사용되는 가상주소에 대응하는 물리적 주소의 테이블을 저장하는 특수 캐시.

Torus: 그리드의 끝점과 시작점을 연결한 구조. 병렬 시스템의 연결 네트워크로 2-D나 3-D 그리드 구성을 자주 사용한다.

Vector unit (pipe): 벡터 레지스터에서 연산수를 받는 파이프라인된 functional 유닛. 파이프라인이 채워진 후 매 사이클마다 벡터 레지스터의 완전한 항목들에 대한 결과를 내놓을 수 있다.

VLIW processing: Very Large Instruction Word processing. 많은 functional 유닛들이 병렬로 항상 실행상태에 있도록 하기 위해 큰 명령어 워드를 사용하는 것. 명령어 스케줄링은 컴파일러가 확률적으로 처리해서 컴파일러에 의한 고품질의 코드 생성이 필요하다. VLIW 프로세싱은 IA-64 아키텍처에서 되살아나서 EPIC으로 불린다.

참고자료

[1] C. Amza, A.L. Cox, S. Dwarkadas, P. Keleher, H. Lu, R. Rajamony, W. Yu, W. Zwaenepoel, Tread-Marks: Shared Memory Computing on Networks of Workstations, to appear in IEEE Computer (also:www.cs.rice.edu/~willy/TreadMarks/papers.htm)

[2] W.Anderson, D.W. Hess, P. Briggs, A. Khoklov, R. Rosenberg, C.S. Hellberg, M. Lanzagorta, Early Experience with Scienti_c Programs on the Cray MTA-2, Proc. SC2003, November 2003, Phoenix, AZ, USA.

[3] The ASCI program: http://www.llnl.gov/asci/.

[4] S.H. Bokhari, J.R. Sauer, Sequence Alignment on the Cray MTA-2, Proc. HICOMB2002, April 2002, Fort Lauderdale, USA.

[5] L. Carter, J. Feo, A. Snavely, Performance and Programming Experience on the Tera MTA, Proc. SIAM Conf. on Parallel Processing, March 1999.

[6] K. Cassirer, B. Steckel, Block-Structured Multigrid on the Cenju, 2nd Cenju Workshop, October 1998, Sankt Augustin, Germany.

[7] R. Chandra, L. Dagum, D. Kohr, D. Maydan, J. McDonald, R. Menon, Parallel Programming in OpenMP, Morgan Kaufmann Publishers Inc., January 2001.

[8] D.E. Culler, J.P. Singh, A. Gupta, Parallel Computer Architecture: A Hardware/Software Approach, Morgan Kaufmann Publishers Inc., August 1998.

[9] J.J. Dongarra, Performance of various computers using standard linear equations software, Computer Science Technical Report CS-89-85, Univ. of Tennessee, July 7, 2002 (early version).

[10] T.H. Dunigan, M.R. Fahey, J.B. White, P.H. Worley, Early Evaluation of the Cray X1, Proc. SC2003, November 2003, Phoenix, AZ, USA.

[11] Overview page of Distributed Shared Memory systems: http://www.ics.uci.edu/~javid/dsm.html.

[12] Directory with EuroBen results: www.euroben.nl/results.

[13] M.J. Flynn, Some computer organisations and their e_ectiveness, IEEE Trans. on Computers, Vol. C-21, 9, (1972) 948{960.

[14] A. Geist, A. Beguelin, J. Dongarra, R. Manchek, W. Jaing, and V. Sunderam, PVM: A Users' Guide and Tutorial for Networked Parallel Computing, MIT Press, Boston, 1994.

[15] D.B. Gustavson, Q. Li, Local-Area MultiProcessor: the Scalable Coherent Interface, SCIzzL Report, Santa Clara University, Dept. of Computer Engineering, 1995. Available through: www.scizzl.com.

[16] J.M.D. Hill, W. McColl, D.C. Stefanescu, M.W. Goudreau, K. Lang, S.B. Rao, T. Suel, T. Tsantilas, R. Bisseling, BSPlib: The BSP Programming Library, Technical report

PRG-TR-29-9, Oxford University Computing Laboratory, May 1997. (Compressed Postscript with ANSI C examples, 142K; Compressed Postscript with Fortran 77 examples, 141K) REFERENCES 67
[17] R. W. Hockney, C. R. Jesshope, Parallel Computers II, Bristol: Adam Hilger, 1987.
[18] T. Horie, H. Ishihata, T. Shimizu, S. Kato, S. Inano, M. Ikesaka, AP1000 architecture and performance of LU decomposition, Proc. Internat. Symp. on Supercomputing, Fukuoka, Nov. 1991, 46-55.
[19] High Performance Fortran Forum, High Performance Fortran Language Specification, Scientific Programming, 2, 13, (1993) 1-170.
[20] D.V. James, A.T. Laundrie, S. Gjessing, G.S. Sohi, Scalable Coherent Interface, IEEE Computer, 23, 6, (1990),74-77. See also: Scalable Coherent Interface: http://sunrise.scu.edu/
[21] D.J.Kerbyson, A. Hoisie, S. Pakin, F. Petrini, H.J. Wassermann, A performance evaluation on an Alpha EV7 processing node, Int. J. of High Perf. Comp. Appl., 18(2), 199-209, Summer 2004.
[22] M. Snir, S. Otto, S. Huss-Lederman, D. Walker, J. Dongarra, MPI: The Complete Reference Vol. 1, The MPI Core, MIT Press, Boston, 1998.
[23] W. Gropp, S. Huss-Ledermann, A. Lumsdaine, E. Lusk, B. Nitzberg, W. Saphir, M. Snir MPI: The Complete Reference, Vol. 2, The MPI Extensions, MIT Press, Boston, 1998.
[24] http://www.myrinet.com
[25] N.J. Boden, D. Cohen, R.E. Felderman, A.E. Kulawik, C.L. Seitz, J.N. Seizovic,Wen-King Su, Myrinet { A Gigabit-per-second Local Area Network, IEEE Micro, 15, No. 1, Jan. 1995, 29-36.
[26] W.E. Nagel, Applications on the Cenju: First Experience with Effective Performance, 2nd Cenju Workshop, October 1998, Sankt Augustin, Germany.
[27] Web page for the NAS Parallel benchmarks NPB2: http://science.nas.nasa.gov/Software/NPB/
[28] OpenMP Forum, Fortran Language Specification, version 1.0, Web page: www.openmp.org/, October 1997.
[29] F. Petrini, D.J. Kerbyson, S. Fakin, The Case of the Missing Supercomputer Performance: Achieving Optimal Performance on the 8,192 Processors of ASCI Q, Proc. SC2003, November 2003, Phoenix, AZ, USA.
[30] F. Petrini, W. Feng, A. Hoisie, S. Coll, E. Frachtenberg, The Quadrics Network: High-performance clustering technology, IEEE Micro Jan/Febr 2002, 46-57.
[31] T. Sato, The Earth Simulator, Proc. SC2002, Nov. 2002, IEEE Press. (Proceedings

only on CD-ROM.)
[32] T. Shanley, In_niband Network Architecture, Addison-Wesley, Nov. 2002.
[33] D.H.M. Spector, Building Unix Clusters, O'Reilly, Sebastopol, CA, USA, July 2000.
[34] A.J. van der Steen, Exploring VLIW: Benchmark tests on a Multiow TRACE 14/300, Academic Computing Centre Utrecht, Technical Report TR-31, April 1990.
[35] A.J. van der Steen, The benchmark of the EuroBen Group, Parallel Computing 17 (1991) 1211-1221.
[36] A.J. van der Steen, Benchmark results for the Hitachi S-3800, Supercomputer, 10, 4/5, (1993) 32-45.
[37] A.J. van der Steen, ed. Aspects of computational science, NCF, The Hague, 1995.
[38] A.J. van der Steen, Benchmarking the Silicon Graphics Origin2000 System, Technical Report WFI-98-2, Dept. of Computational Physics, Utrecht University, The Netherlands, May 1998. The report can be downloaded from: www.euroben.nl/reports/REFERENCES 68
[39] A.J. van der Steen, An evaluation of some Beowulf clusters, Technical Report WFI-00-07, Utrecht University, Dept. of Computational Physics, December 2000. (Also available through www.euroben.nl, directory reports/.)
[40] A.J. van der Steen, An evaluation of Itanium 2-based high-end servers, Technical Report HPCG-2004-04, Utrecht University, High Performance Computing Group, July 2004. (Also available through www.euroben.nl, directory reports/.)
[41] T.L. Sterling, J. Salmon, D.J. Becker, D.F. Savaresse, How to Build a Beowulf, The MIT Press, Boston, 1999.
[42] H.W. Meuer, E. Strohmaier, J.J. Dongarra, H.D. Simon, Top500 Supercomputer Sites, 23rd Edition, June 2004, The report can be downloaded from: www.top500.org/.
[43] Task Force on Cluster Computing home page: www.clustercomputing.org.

찾아보기

A

ALAT, 36
ASCI, 4, 5, 98

B

Beowulf 클러스터, 4, 21, 22, 37
BHT, 27, 30, 32
bit-serial, 13, 98
branch predictor, 27
Butterfly-network, 15

C

Cache Coherent, 20, 99
capability computing, 21
capacity computing, 21
chassis, 57
Chip Multithreading, 42
CMT, 43, 79
COMPAS, 60
cut-through wormhole, 45

D

directory memory, 21, 62
DSP, 23, 96

E

Earth Simulator, 4, 5, 95
EM64T, 38
EPIC, 35, 100, 103
execution trace cache, 38
Explicit Parallel Instruction Computing, 35

F

fat tree, 18, 45, 47, 49, 57, 66
Flynn, 7, 8
FMA, 30, 32
FMAC, 30, 36
FPB, 74
FPGA, 23, 57, 96, 97, 100
FPU, 14, 24, 42, 69

G

GPR, 39

H

HCA, 44, 45, 101
High Performance Fortran, 9, 68, 101
hypercube, 17, 18, 19, 76
Hyper-threading, 38
HyperTransport, 23, 25, 57, 95

I

IEU, 41
IIU, 41
IPC, 37
IRB, 29

L

Lanai, 46, 47
Linpack, 5, 52, 70

M

MCU, 42
MPI, 9, 44, 45, 46, 47, 49, 51, 52, 56, 57, 58, 61, 65, 66, 67, 68, 71, 73, 76, 77, 102

MSP, 55, 56, 97
multi-threading, 34

N

NUMA factor, 21, 62, 65
Numalink3, 75, 76, 97

O

OpenMP, 9, 56, 61, 62, 64, 65, 66, 68, 71, 73, 76, 79, 102, 103
out-of-order, 23, 24, 26, 29, 32, 39
out-of-phase, 29

P

PCI-X, 44, 46, 47, 102
Processor-array, 12
PVM, 9, 61, 66, 68, 77, 102

R

ROP, 24

S

Scalable Coherent Interface, 15
SCI, 15, 22, 48, 49, 100
S-COMA, 20, 21
SIU, 42
SMT, 34
snoopy bus protocol, 20
speculative branch, 30
speculative processing, 26
SSE, 25, 38, 39
switchless SMP processing, 27

T

TCA, 44, 45, 101
TLB, 30, 41, 103

V

Very Large Instruction Word, 35, 100
VLIW, 9, 35, 96, 100, 103
VPU, 10, 11, 14

W

wormhole routing, 18

Ω

Ω-network, 15

공

공유 메모리 시스템, 8, 9, 14, 16, 17, 47, 53, 57, 66, 72, 101, 102

다

다단 크로스바, 19
다중 스레딩, 34, 43, 53, 79

벡

벡터 프로세서 시스템, 7

분

분기 예측기, 27
분산 메모리 시스템, 8, 101, 102, 103

사

사이클당 처리 명령어 회수, 37

섀

섀시, 57

선

선인출, 41, 60

스

스누피 버스 프로토콜, 20, 48, 99
스눕 컨트롤러, 42

실

실행 추적 캐시, 38

전

전체 요구 대역폭, 11

중

중앙 버스, 15

추

추정 분기, 29
추정 실행, 26, 36, 100

크

크로스바, 15, 19, 21, 31, 46, 59, 60, 62, 63, 66, 71, 72, 78, 80, 97, 100

통

통합 병렬 시스템, 4, 22, 24, 43, 45, 47, 98

프

프로세서 배열, 12, 13, 16, 98, 99