CallBack 지연시간 감소를 통한 DMA 통신 기반의 클러스터 응용 시스템

김한호[0]* 박의수** 최현호*** 유원경**** 유관종*
*충남대학교 컴퓨터공학과 **대전보건대학 보건의료정보과
***대전보건대학 컴퓨터정보통신과 ****성신여자대학교 컴퓨터공학부
E-mail: {kastar1[0], kjyoo}*@miclab.cnu.ac.kr **uspark@hit.ac.kr
***hyuno@hit.ac.kr ****wyoo@sungshin.ac.kr

A Cluster Application System with CallBack Delay Time Decrease on DMA Communication

Han-Ho Kim[0]* Ui-Su Park** Hyun-Ho Choi***
Won-kyong Yoo**** Kwan-Jong Yoo*
*Dept. of Computer Engineering, ChungNam Univ.
**Dept. of Health & Medical Information, Daejeon health Sciences college
***Dept. of Computer Information & Communication, Daejeon health Sciences college
****Dept. of Computer Science, Sung-Shin Women's Univ.

요 약

최근 인터넷이 발전함에따라 사용자로 하여금 다양한 형태의 멀티미디어 서비스에 대한 요구를 증가시켰다. 하지만 욕구가 다양해짐에 따라 처리해야할 데이터는 단순 텍스트에서 멀티미디어 데이터로 전환되었다. 그로 인해 데이터의 용량은 기하급수적으로 늘어나고 이를 처리하기 위해서는 고성능 대용량 서버의 필요성이 대두되고 있다. 하지만 기존의 고성능 단일 서버의 모델은 멀티미디어 데이터를 처리하는데 있어서 사용자의 요구를 충분히 만족시키기에는 많은 문제점을 가지고 있다. 이러한 문제점을 해결하기 위해 리눅스 클러스터 시스템은 하나의 해결책으로 제시되고 있다. 본 논문은 Myrinet을 사용한 클러스터 응용에 있어서 DMA 통신을 통해 고성능 단일 서버의 문제점을 해결하고 성능을 향상 시키는데 있다.

1. 서 론

초창기 인터넷의 주요 서비스는 메인 프레임의 컴퓨팅 파워를 공유하기 위한 텔넷과 같은 터미널 기능이 주요 서비스였다. 하지만 최근 몇 년 동안 IT 산업을 육성하려는 정부의 의지와 초고속망의 보급으로 인해 인터넷은 단순한 컴퓨터 공유의 개념이 아닌 정보 전달의 의미가 더욱 중요하게 되었다. 또한 인터넷의 발전은 사용자로 하여금 다양한 서비스 욕구를 야기 시켰으며 현재의 인터넷은 생활의 모든 것을 융합하려는 움직임으로 나타나고 있다. 이러한 다양한 기능이 늘어남에 따라 처리되는 데이터는 기존의 단순한 텍스트에서 멀티미디어 데이터로 전환되는 추세이고, 그 용량도 기하급수적으로 늘어나게 되었다. 비록 초고속망이 빠르게 보급되고 대역폭도 늘어났지만 데이터의 양적인 증가 속도에 비해 현저히 느리다. 데이터의 증가는 사용자 입장에서는 응답 시간의 지연 및 처리 속도의 감소, 서비스를 제공해주는 사업자 입장에서는 네트워크의 과부하, 서비스 처리 속도의 저하 등 많은 문제점을 야기하고 있다. 따라서 고용량 멀티미디어 데이터를 효율적으로 처리하기 위해서 고성능 대용량 서버의 필요성이 대두되고 있다.

하지만 기존의 고성능 단일 서버 모델은 멀티미디어 데이터를 처리하는데 있어서 사용자의 요구를 충분히 만족시키기에는 많은 문제점을 가지고 있다. 수많은 사용자의 서비스 요구를 처리하기 위해서는 고성능 프로세서와 병렬화가 필요하게 되었고 늘어나는 데이터의 고속 처리를 위해서는 대용량의 저장 공간 확보가 필요하게 되었다. 이로 인해 단일 서버를 구축하기 위해서는 엄청난 비용이 발생하게 되었다. 그리고 고성능 단일 서버 자체에 장애 및 문제가 생길 경우에는 서비스 중단이라는 치명적인 단점을 안고 있다.

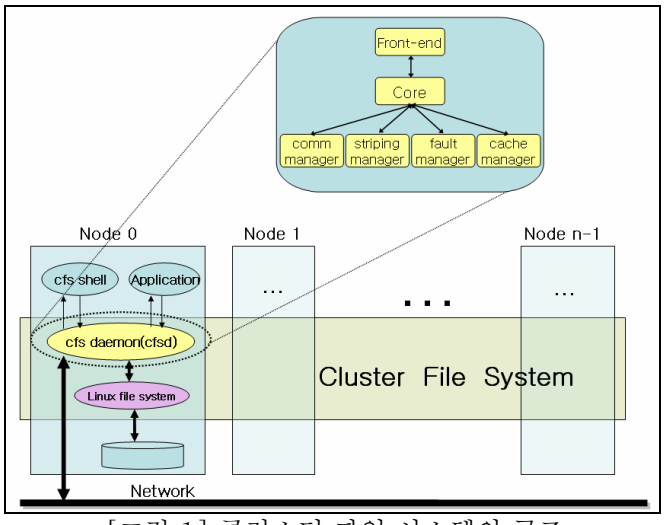
이러한 문제점을 해결하기 위해 현재 많은 방법들이 시도되고 있다. 그 중에 리눅스 클러스터 시스템은 단일 서버의 문제점을 해결하기 위한 하나의 해결책으로 제시되고 있다. 리눅스 클러스터 시스템은 단일 고성능 서버에 비하여 가격대 성능비가 우수할 뿐만 아니라 클러스터 시스템의 구조적 특징으로 인한 고성능, 고가용성, 고확장성을 지원할 수 있는 장점을 가지고 있다. 또한 리눅스는 공개 OS이기 때문에 이를 통해 다양한 응용 프로그램 개발이 저렴한 비용으로 가능하다. 이러한 특징을 가지고 있는 리눅스 클러스터 시스템은 기존의 고가의 서버 모델을 대치할 수 있는 가장 적합한 모델로서 자리 잡을 수 있게 되었다[1].

본 논문에서는 기존의 Myrinet GM(Glenn's Messages)을 사용한 클러스터 파일 시스템 통신 모듈의 문제점을 분석하고 제시한다. 그리하여 그 문제점을 해결하기 위한 새로운 방안을 제시하고 클러스터 파일 시스템의 성능을 올리고자 새로운 GM DMA(Direct Memory Access) 통신 모듈을 제안한다. 그리고 논문의 구성은 2장에서는 본 연구와 관련된 배경 연구들에 대해 알아보고 3장에서는 DMA 통신에 관한 설계 및 구현을 하였다. 4장에서는 제안 시스템의 실험 및 결과를 알아보고 5장에서 결론을 맺는다.

2. 관련 연구

2.1 클러스터 파일 시스템

최근 프로세서 및 메모리의 기술이 급속도로 발전함에 따라 컴퓨터의 동작 속도는 매우 빠르게 발전하였다. 하지만 이에 비해 디스크의 동작 속도는 프로세서와 메모리의 발전 속도를 따라가지 못해 입출력의 병목 현상이 날로 심해지고 있어 시스템 전체의 성능 저하로 나타난다. 이러한 문제점을 해결하기 위해 클러스터 파일 시스템이 대안으로 제시되고 있다. CFS(Cluster File System)는 대용량 멀티미디어 파일의 입출력 시간을 단축하고 결함 허용성(Fault Tolerance) 및 고가용성(High Availability)을 제공하는 기술로 주목 받고 있다. [그림 1]은 클러스터 파일 시스템의 구조를 나타낸 것이다.



[그림 1] 클러스터 파일 시스템의 구조

2.2 Myrinet GM

GM은 Glenn's Messages를 의미하며 Myrinet을 위한 low-level 통신 계층을 말한다. GM 시스템은 드라이버, Myrinet-NIC 콘트롤 프로그램, 네트워크 매핑 프로그램, GM API, 라이브러리와 헤더 파일을 포함한다. GM은 미리넷을 위한 메시지 기반 통신 시스템으로, 많은 메시징 기법처럼 GM은 CPU의 낮은 오버 헤드와 지연 시간, 높은 대역폭과 이식성을 제공하는데 설계목적이 있다. GM은 두 계층의 우선순위를 가진 " 포트(port)"로 불리는 통신상의 엔드-포인트(End-points)들 사이의 순서화되고 신뢰성 있는 전송을 제공한다. 이 모델은 클라이언트가 통신하기 위해 원격 포트와의 연결을 설정할 필요가 없다. 단지 클라이언트는 메시지를 만들어서 네트워크상에 있는 임의의 포트에게로 보낸다[2].

2.3 Direct Memory Access

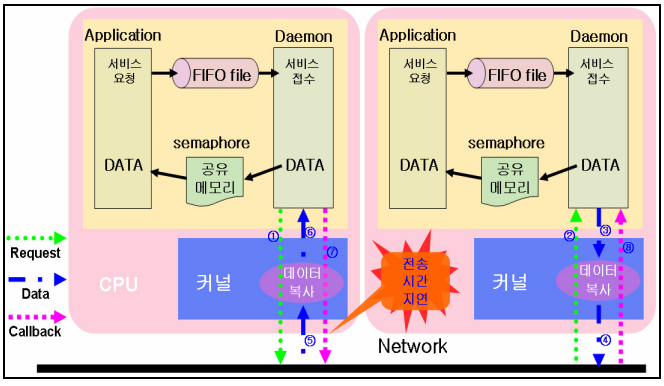
DMA(Direct Memory Access)란 입출력 장치에 관련된 개념으로 CPU의 개입 없이 주변장치와 주기억 장치와의 데이터 전송이 이루어지는 방법이다. 이는 프로그램이 실행되는 동안에 입출력을 위한 인터럽트의 발생 횟수를 최소화시켜 컴퓨터 시스템의 효율을 높이기 위한 방법으로, 주변 장치와 주기억 장치와의 입출력 데이터 교환을 CPU를 거치지 않고 고속으로 직접 행하게 한다. 이를 사용하는 것은 대개 디스크 등의 고속 입출력 장치로, 이러한 장치는 입출력되는 각 데이터 바이트를 CPU가 하나하나 처리하기에는 너무 빠르고, 또 일단 입출력이 시작된 다음에는 일정한 속도로 데이터가 전송된다는 특성이 있다. 따라서 이러한 장치의 입출력에는 CPU가 굳이 관여할 필요가 없다. DMA는 동작중에 시스템의 버스를 사용하게 되는데 이 때 CPU와 DMA가 동시에 버스를 요청했을 때 DMA에 우선권이 주어지는 것을 사이클 도용(cycle stealing)이라 하고 이와는 반대로 CPU의 처리에 따라 입출력이 이루어지는 것을 PIO(Programmed Input/Output)라고 한다.

3. 설계 및 구현

3.1 기존 시스템 소개 및 문제점

[그림 3]은 기존의 GM 전송 방식을 이용한 클러스터 파일 시스템을 나타낸 것이다. 클러스터 파일 시스템은 분산 저장을 기본으로 한다. 그래서 어플리케이션이 요청한 파일은 동일 노드에 모두 존재하지 않는다. 어플리케이션이 데몬에게 서비스를 요청하면 데몬은 그 서비스를 접수하고 해당하는 파일이 동일 노드에 있는 경우 파일을 디스크에서 바로 가져오지만 동일 노드 내에 없는 경우 다른 노드의 데몬에게 파일을 요청하게 된다.

하지만 기존 시스템은 파일 전송 메커니즘에 몇 가지 문제점을 가지고 있다. 첫 번째, 기존 시스템은 파일 전송하기 위해 CPU 자원을 많이 소모하는 문제점을 가지



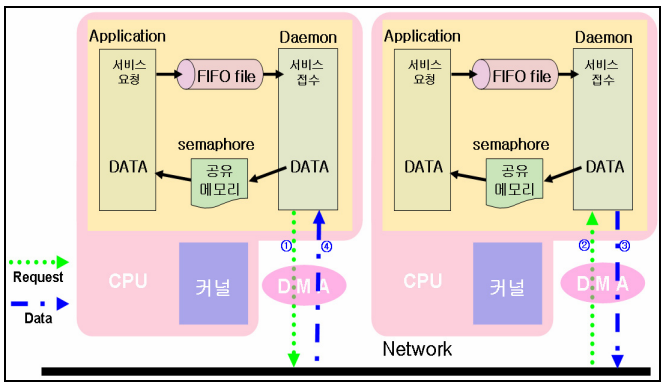
[그림 2] 기존 응용 시스템

어플리케이션과 데몬 간의 통신은 IPC를 사용한다.

고 있다. [그림 2]에서 보는 바와 같이 모든 어플리케이션 및 데몬 그리고 파일 전송이 이루어지는 데몬에서 네트워크 연결부까지 CPU가 파일 전송에 관여하게 된다. 그럼으로써 CPU가 다른 일을 처리할 수 있는 자원이 줄어들게 된다. 두 번째, 요청 받은 파일을 전송 후 다음 파일을 전송하기 위해서는 파일을 수신한 데몬이 파일을 정상적으로 받았다는 Callback 메시지를 송신 데몬에게 전송할 때가지 데몬은 그 다음 파일 전송을 할 수가 없다. 세 번째, 데이터를 전송하기 위해서는 데몬에서 커널, 커널에서 다시 네트워크 연결부까지 한 번의 Copy가 이루어진다. 이로 인해 데이터 전송 속도에 영향을 주게 된다. 이렇듯 세 가지 문제점으로 인해 데이터의 입출력 속도가 느려지게 된다.

3.2 제안 시스템

본 절에서는 제시된 기존 시스템의 문제점을 해결하고자 DMA 통신 모듈을 사용한 클러스터 파일 시스템을 제안한다. [그림 3]는 DMA 기법을 이용한 클러스터 파일 시스템으로 기존의 문제점을 해결한 시스템이다. DMA 통신 모듈을 이용한 클러스터 파일 시스템에서는 DMA 통신을 하기 때문에 데몬에서 네트워크 연결부까지 CPU의 관여 없이 파일을 전송할 수 있다. 그럼으로써 CPU는 다른 처리를 하기 위해 자원을 할당할 수 있다. 또한 DMA 통신 모듈은 수신 노드의 메모리 주소를 얻어 온 후 그 주소에 직접 데이터를 쓰기 때문에 수신 노드의 Callback이 없어도 파일 전송이 가능하게 된다. 이로써 Callback 지연 시간이 없어지게 되었다. 그리고 파일 전송에서 있어서 데몬과 네트워크 연결부까지 데이터가 커널을 거치지 않고 바로 복사되기 때문에 커널에서의 Copy 오버헤드를 줄일 수 있게 되었다[3].



[그림 3] 제안 응용 시스템

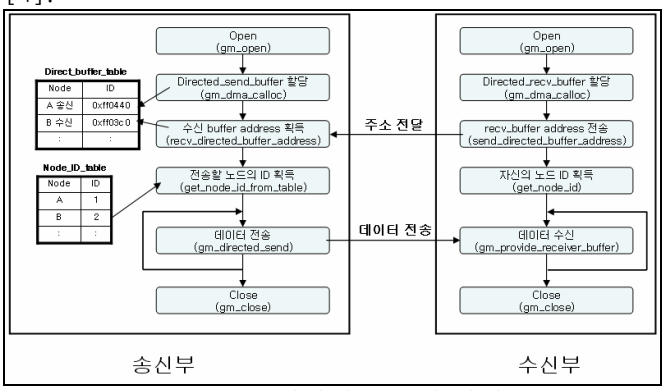
3.3 DMA 통신 모듈 설계 및 구현

클러스터 파일 시스템을 구성하는 노드는 GM DMA 통신 모듈을 이용하여 다른 노드와 통신을 수행한다. 정확히 말해 노드와 노드의 통신은 데몬과 데몬이 통신함으로써 데이터가 송수신 된다. [그림 4]은 본 논문에서 제안하고자하는 GM 통신 모듈의 구조를 나타낸 것이며,

노드와 노드간의 통신 연결의 흐름은 다음과 같다.

. 클러스터 파일 시스템의 데몬 실행시 GM 초기화
. 송수신할 포트를 Open
. 송수신에 쓸 DMA 가능한 directed_send_buffer와 directed_recv_buffer를 할당
. Node_id_table과 Direct_buffer_table를 이용하여 다른 노드에게 자신의 수신 버퍼(directed_recv_buffer) address를 전송
. 수신된 다른 노드의 directed_recv_buffer address를 direct_buffer_table에 기록 후 전송 대기

[그림 4]는 위의 통신 연결 흐름을 기본으로 하여 GM DMA 통신 모듈을 통해 노드 간 데이터가 어떻게 송수신 되는지 통신 모듈의 세부 동작 과정을 보여주고 있다[4].



[그림 4] 메세지 송수신 과정

4. 실험 및 결과

4.1 실험 환경

본 논문에서는 GM DMA 통신 모듈을 사용하여 각 패킷 사이즈별로 지원할 수 있는 대역폭과 지연 시간의 차이를 알아본다. 처음 실험 장비는 펜티엄4 시스템으로 진행하였으나 Myrinet NIC의 최대 성능을 이끌어내고자 64bit/133MHz PCI 슬롯을 지원하는 제온 서버로 교체하게 되었다. 결과값은 오차를 줄이기 위해 7번을 측정하여 최대값과 최소값을 제하고 평균값을 구하였다.

[표 1] 제 원

주변 장치	사 양
프로세서	Intel® Xeon™ 프로세서 3.0GHz (L1 cache 12KB) FSB 533MHz
캐 시	512KB L2 ECC
메 모 리	PC2100 ECC DDR SDRAM 1024MB (266MHz)
디 스 크	80GB Ultra ATA-100 buffer 2MB
확장 슬롯	I/O (총 1개 - 1×64bit/133MHz PCI)
Myrinet Switch Networks	Enclosures : M3-E16 (2U high enclosure for switches up to 16 ports

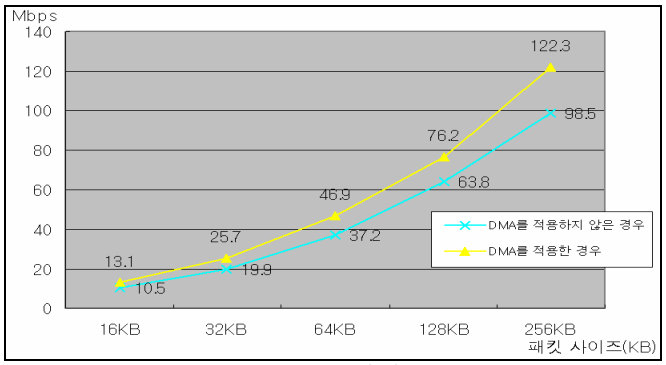
[표 2] 패킷별 성능 향상(대역폭/Mbps)

	16KB	32KB	64KB	128KB	256KB

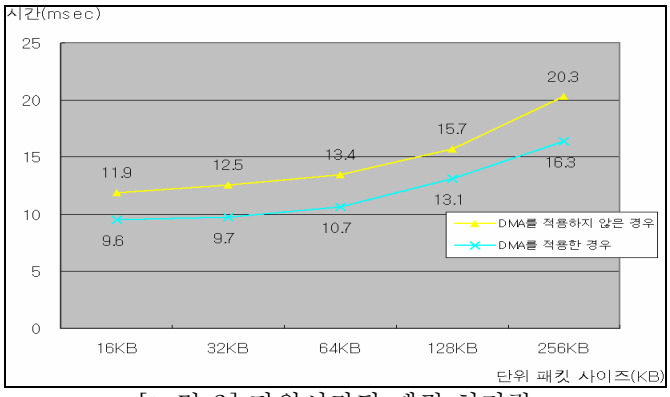
	Line Cards : M3-SW16-8F (Line-card switch 8 Fiber ports on the front panel and 8 SAN ports to the backplane
Myrinet NIC (M3-PCI64B-4)	PCI-bus Interface : 64/32bit, 66/33MHz Interface processor : LANai 9RISC (133MHz) Local Memory : 4MB (512K×8B) Myrinet-2000 Fiber port : 2.0+2.0Gb/s DMA controller
GM 드라이버	gm_1.5.1
운영 체제	커널 2.4.18

4.2 실험 및 결과

실험은 클러스터 파일 시스템에선 시스템 특성으로 인해 총 5 구간(16KB, 32KB, 64KB, 128KB, 256KB)의 패킷 사이즈로 대역폭 및 지연 시간을 측정하였다. 실험 방법은 가상 환경-GM API만을 이용하여 측정한 GM 성능, GM DMA API만을 이용하여 측정한 GM DMA 성능과 실제 환경-기존 GM CFS 성능, DMA기법을 사용한 CFS 성능에서 총 4가지 비교 대상별 테스트가 행해졌다.



[그림 5] 대역폭



[그림 6] 단위시간당 패킷 처리량

[그림 5]와 [그림 6]는 실제 환경(GM CFS와 GM DMA CFS)에서 대역폭과 지연 시간을 비교 분석한 것이다. 측정한 결과 값은 [표 2]에서 보는 바와 같이 대역폭과 지연 시간이 패킷 사이즈에 따라 약 20~30%까지 성능 향상이 있었다.

DMA를 사용하는 경우	13.1	25.7	46.9	76.2	122.3
DMA를 사용하지 않는 경우	10.5	19.9	37.2	63.8	98.5
향 상 폭	30%	30%	27%	19%	24%

그리고 송신측과 수신측의 CPU Workload를 측정하였을 때 [표 3]에서 보는것과 같이 DMA 통신 모듈을 사용하지 않았을 때 보다 각각 16%, 40% 감소하였다.

[표3] CPU Workload 감소폭

	송신측	수신측
CPU Workload	16% Down	40% Down

5 결 론

클러스터 파일 시스템의 가장 중요한 요소는 데이터의 입출력 속도 및 효율성이다. 이것이 얼마나 빨리 그리고 효율적으로 이루어지느냐에 따라 시스템 전체의 성능이 결정 된다. 단일 서버가 가지고 있는 대역폭의 한계로 인해 시스템 성능이 낮아지고 서비스의 질은 떨어지고 있다. 그래서 몇 년 전부터 이를 해결하기 위해 클러스터 파일 시스템이란 대안이 제시 되어 왔다.

클러스터 파일 시스템의 성능 높이기 위해서는 고성능의 네트워크 장비를 도입하여 성능을 올리는 하드웨어적인 방법과 네트워크의 프로토콜이나 어플리케이션의 최적화 등으로 성능을 올리는 소프트웨어적인 방법이 있다.

본 논문에서는 기존의 클러스터 파일 시스템의 성능을 높이기 위해 GM 기반의 CFS 문제점인 Callback 지연시간 및 커널에서 불필요한 Copy 문제 및 CPU 자원 문제를 해결하고자 GM DMA 기법을 도입하였다. GM DMA 기법은 Callback의 문제점이 없고 CPU를 통하지 않는 데이터 입출력으로 인해 CPU 자원 소모가 적은 장점이 있다. 또한 커널에서 Copy가 이루어지지 않아 보다 빠른 데이터 입출력이 가능하다. 이로써 보다 성능 좋은 클러스터 파일 시스템을 구현 할 수가 있었다.

[참고 문헌]

[1] 홍재연, 이재국, 이경희, 강미연, 김형식, "멀티미디어 서비스를 위한 사용자 수준 클러스터 파일 시스템의 최적화" 최종 보고서, Jan. 2003.

[2] Myricom, Myrinet FAQ (GM-1 Message Passing System),"http://www.myri.com/fom-serve/cache/11.html"

[3] Myricom, GM Reference Manual 1.6.3, Nov. 2002

[4] 박의수 " 리눅스 클러스터 파일 시스템을 위한 초고속 통신 모듈의 설계 및 구현" 석사학위 논문, 충남대학교 컴퓨터과학과 대학원, Feb. 2003