리눅스 상의 VIA 구현 비교

김강호° 김진수 김해진
한국전자통신연구원 리눅스연구팀
{khk, jinsookimm haejinkim}@etri.re.kr

A Comparison of VIA Implementations on Linux

Kangho Kim° Jin-Soo Kim Haejin Kim
Linux Research Team, ETRI

요 약

최근에 클러스터 시스템 사용이 보편화되어 가고 있지만 클러스터를 구성하고 있는 노드 사이의 통신이 여전히 전체 성능 향상의 병목요인으로 지적되고 있다. 현재 클러스터 시스템의 노드간 통신은 TCP 프로토콜을 이용하고 있는데, 동질적이고 전송에러를 무시할 수 있는 클러스터 통신망에는 적합하지 않다. TCP의 단점을 극복하기 위하여 클러스터를 위한 다양한 사용자 수준 인터페이스가 제안되고 구현되었다. 이 중 Intel, Compaq, Microsoft가 주축이 되어 정의한 VIA는 SAN 환경에 적합하도록 기존의 소프트웨어 오버헤드를 줄인 사용자 수준의 통신 프로토콜이다. 본 논문에서는 현재 리눅스 클러스터 상에서 사용가능한 VIA 구현들에 대해 특징을 살펴보고, 동일한 환경에서 성능을 비교해 보았다.

1. 서론

근래 들어 리눅스를 이용한 클러스터 시스템의 구성이 보편화되어 가고 있지만, 클러스터를 구성하고 있는 노드 사이의 통신이 여전히 전체 성능 향상의 병목요인으로 지적되고 있다. 현재 대부분의 리눅스 클러스터 시스템은 노드간 통신을 위하여, WAN(Wide-Area Network) 환경에서 양극단간의 신뢰성있는 패킷 송수신을 위해 고안된 TCP/IP 프로토콜을 이용하고 있다. 그러나 클러스터 내부의 통신망은 동질적이고, 전송 에러를 무시할 수 있을 만큼 신뢰성이 높기 때문에, 기존의 TCP/IP 프로토콜 내부에서 수행되는 여러 불필요한 기능을 제거한 경량의 프로토콜을 사용하는 것이 바람직하다. 또한, 기존의 TCP/IP 프로토콜은 통신 프로토콜이 커널 내부에 존재하기 때문에 매 패킷 전송시마다 사용자 모드에서 커널 모드로의 문맥 교환이 수반되며, 사용자 주소 영역에 있는 데이터를 커널 내부로 복사한 후 전송해야 하는 단점이 있다. 이러한 TCP/IP 프로토콜의 단점을 극복하기 위하여 AM, BIP, FM, PM, U-Net, VMMC, VIA 등 클러스터를 위한 다양한 종류의 사용자 수준 인터페이스가 제안되고 구현되었다[1].

이 중 Intel, Compaq, Microsoft가 주축이 되어 정의한 VIA(Virtual Interface Architecture)[2]는 SAN(System Area Network) 환경에 적합하도록 기존의 소프트웨어 오버헤드를 줄인 사용자 수준의 통신 프로토콜로써, 새로운 산업계의 표준으로 대두되고 있다.

본 논문에서는 현재 리눅스 클러스터 상에서 사용가능한 VIA 구현들에 대해 특징을 살펴보고, 동일한 환경하에서 성능과 특징을 비교해 보고자 한다.

2. VIA (Virtual Interface Architecture)

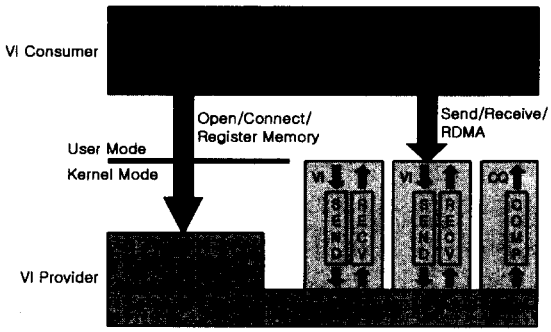
VIA는 그림 1에 보이는 바와 같이 네 개의 기본 요소, 즉 가상 인터페이스 (VI: Virtual Interface), 완료 큐, VI-제공자, VI-수요자로 구성된다. VI-제공자는 물리적인 네트웍 어댑터와 커널 에이전트 및 라이브러리를 포함하며, VI-수요자는 VIA를 이용하는 응용 프로그램을 말한다.

일반적으로 노드상의 여러 프로세스는 하나의 네트웍 어댑터를 공유하지만, VIA에서는 각각의 VI-수요자에게 가상으로 하나의 전용 어댑터가 할당되어 있는 것과 같이 보이도록 지원한다. 따라서 두 프로세스가 통신을 하기 위해서는 우선 각 프로세스가 VI를 하나씩 생성하고, 이들 VI 사이에 가상 회선을 설정하여야 한다. VI는 송신 큐와 수신 큐의 쌍으로 이루어지며, VI-수요자는 전달할 또는 전달받을 메시지에 대한 정보를 담고 있는 디스크립터(descriptor)를 큐에 추가함으로써 데이터를 교환한다. 한편, 여러 개의 송수신 큐를 하나의 완료 큐(CQ: Completion Queue)와 연결시킴으로써, 매번 모든 큐를 조사할 필요가 없이 완료 큐를 통해 완료된 디스크립터들을 처리할 수 있다.

VIA는 기존의 NIC(Network Interface Card) 위에 소프트웨어적으로 구현될 수 있으나, NIC 수준에서 하드웨어 혹은 펌웨어를 통해 구현될 수도 있다. 본 논문에서는 현재 리눅스 상에서 사용가능한 세가지 VIA 구현을 대상으로 하였으며, 구현상의 특징을 표 1에 비교하였다.

M-VIA(Modular VIA)[3]는 NERSC(National Energy Research Scientific Computing) 센터에서 개발된 것으로 소프트웨어적으로 VIA의 기능을 에뮬레이션한다. 반면, UC

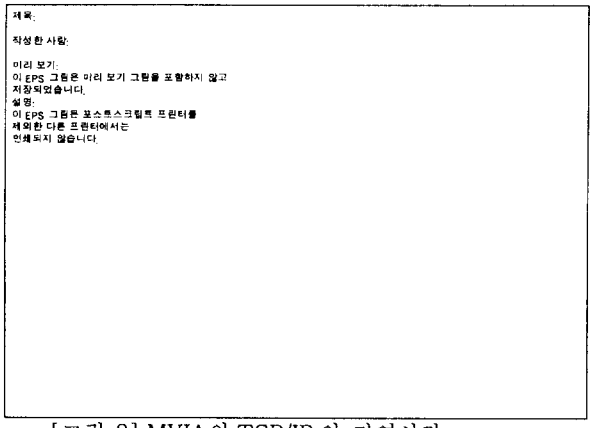
Berkeley 대학에서는 Myrinet 상에서 펌웨어를 수정함으로써 VIA의 기능을 구현하였으며[4], Giganet의 cLAN[5]은 하드웨어적으로 VIA 규격을 지원한다.



[그림 1] VI의 구조

3. 성능 비교

[표1]에 보이는 VIA 구현들의 성능을 비교하고자 Intel L440GX+ 보드, Pentium III-500MHz CPU 및 리눅스 커널 2.2.16(UP)이 탑재되어 있는 플랫폼 상에서 지연시간(latency) 및 대역폭(bandwidth)을 측정하였다. 지연시간은 메시지 왕복 시간(round trip time)의 반으로 측정하였으며, 대역폭은 단위 시간동안 송신측에서 수신측으로 전송된 데이터 양으로 측정하였다.


제목:

작성한 사람:

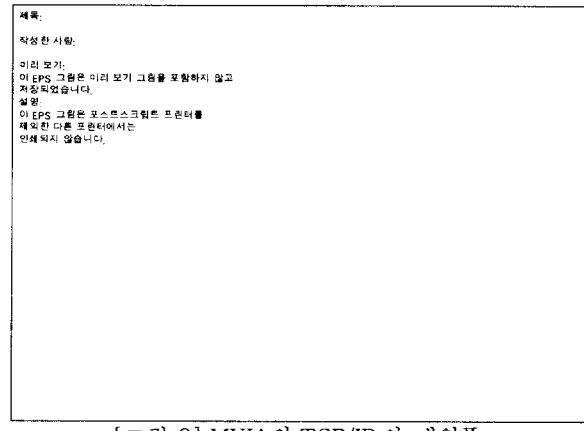
미리 보기:
이 EPS 그림은 미리 보기 그림을 포함하지 않고
저장되었습니다.
설명:
이 EPS 그림은 포스트스크립트 프린터를
제외한 다른 프린터에서는
인쇄되지 않습니다.

[그림 2] MVIA와 TCP/IP 의 지연시간

3.1 M-VIA와 TCP/IP

동일한 NIC (Intel eepro/100)에서 M-VIA와 TCP/IP를 비교하여 그림 2와 같은 결과를 얻었다. 모든 메시지 크기에서 M-VIA가 TCP/IP보다 작은 지연시간을 보였다. 메시지 크기가 4바이트(bytes)인 경우 M-VIA는 약 42us, TCP/IP는 약 58us의 지연시간을 보인다. M-VIA의 경우 소프트웨어적으로 VIA를 에뮬레이션하기 때에 완전한 zero-copy가 되지 않음에도 불구하고 지연시간이 작다는 것은 VIA 프로토콜이 상대적으로 경량화되어 있다는 것을 의미한다. 동일한 실험을 SMP 커널을 사용하여 수행하면 지연시간이 전체적으로 커지는 것을 관찰할 수 있다.

대역폭의 경우 TCP는 Nagle 알고리즘[6]의 영향으로 작은 메시지의 경우 불규칙한 성능을 보이나 TCP_NODELAY 소켓 옵션을 사용하여 Nagle 알고리즘을 사용하지 않으면 불규칙성이 사라진다. 지연시간의 감소로 인해 512바이트 이하의 메시지에 대해 M-VIA의 대역폭이 우수하고, 큰 메시지의 경우에도 다소 우수하다.


제목:

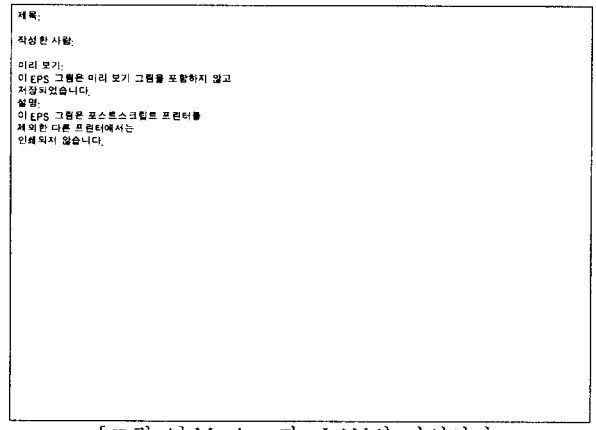
작성한 사람:

미리 보기:
이 EPS 그림은 미리 보기 그림을 포함하지 않고
저장되었습니다.
설명:
이 EPS 그림은 포스트스크립트 프린터를
제외한 다른 프린터에서는
인쇄되지 않습니다.

[그림 3] MVIA와 TCP/IP 의 대역폭

3.2 Berkeley VIA와 cLAN

그림4은 B-VIA과 cLAN의 지연시간을 측정한 그래프이다. 모든 구간에서 cLAN의 지연시간이 B-VIA의 지연시간보다 작게 관찰되었다. cLAN의 경우는 4바이트를 전송할 때 약 10us의 지연시간이 있고, 그 외 전구간에서도 아주 작은 지연시간을 보인다. 그러나 B-VIA의 경우는 cLAN보다 월등히 긴 지연시간을 보일 뿐만 아니라 작은 메시지에 대해서는 M-VIA와 비슷한 지연시간을 보인다. 데이터를 송수신할 때 송수신 완료시점까지 기다리지 않고 계속 확인하는 방법(polling)을 사용하면 지연시간을 줄일 수 있다. 그러나 이 방법은 폴링하는 동안 CPU를 사용하기 때문에 많은 시스템에 부하가 많은 경우 적합하지 않다.


제목:

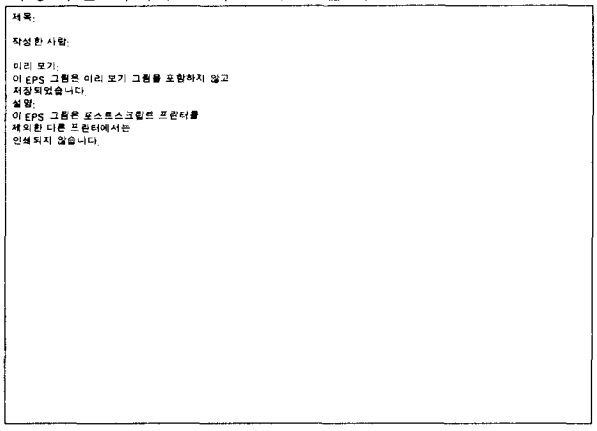
작성한 사람:

미리 보기:
이 EPS 그림은 미리 보기 그림을 포함하지 않고
저장되었습니다.
설명:
이 EPS 그림은 포스트스크립트 프린터를
제외한 다른 프린터에서는
인쇄되지 않습니다.

[그림 4] Myrinet과 cLAN의 지연시간

그림5는 B-VIA와 cLAN의 대역폭을 보여주는 그래프이다. VIA는 긴 메시지를 MTU단위로 잘라서 보내고, 결합하는 기능이 없으므로 MTU까지만 메시지 크기를

변화시키면서 대역폭을 측정하였다. 메시지 크기가 증가함에 따라 대역폭도 함께 증가하였으나 cLAN의 대역폭이 B-VIA의 대역폭보다 월등히 우수하다. cLAN의 경우는 giganet에서 제공하는 cLAN전용 VIA 드라이버를 사용하여 하드웨어가 제공하는 최대 성능을 잘 이용하였으나, B-VIA의 경우는 Myrinet의 하드웨어 성능을 발휘하지 못하여 대역폭 성능이 크게 차이가 나는 것으로 분석된다. 대역폭 측정에서도 지연시간 측정과 같이 폴링 방법을 사용하면 대역폭 증가 효과를 볼 수 있다.



[그림 5] Myrinet과 cLAN의 대역폭

VIA가 기본 구조가 간단하고 사용자 영역 통신(user level networking)의 산업계 표준으로 정의하려는 장점이 있지만 개선할 여지가 많이 있다. VIA는 작은 메시지를 처리할 때 효율이 떨어지는 것이 가장 큰 단점이다. 실제 데이터 전송 이전에 메모리에서 NIC으로 두번의 메모리 전송(도어벨, 디스크립터)이 필요하다. 대부분이 작은 메시지인 환경에서는 이것이 큰 부담으로 작용한다. 두번째로 멀티캐스트(multicast)를 지원하지 않는 점이다. 현재는 여러 개의 노드에 동일한 메시지를 보내려면 각 노드와 VI로 연결하여 동일한 메시지를 각각 전송해야 한다.

프로그래머 관점에서 기존의 네트웍 프로그램 인터페이스(예: 소켓)보다 프로그램하기 쉽지 않다. 성능 향상을 위해서 메시지 전송에 관련된 기본 기능만 제공하고 있으므로 그 외의 기능은 프로그래머가 부담해야 한다. 예를 들어, 흐름제어, 수신측의 디스크립터 준비와 같은 문제를 프로그래머가 신중히 고려하여야 한다.

위에서 언급한 단점이 있지만 VIA는 급속히 발달하는 고속 네트워크 장비를 표준화된 인터페이스로 사용할 수 있고, 참조 구현에서 검증이 되었듯이 클러스트 노드 간의 통신 규격으로 사용하기에 적합하고 잠재적 발전 가능성 또한 높다고 판단된다.

[표 1] 특성 비교

	M-VIA	Berkeley VIA	cLAN
지원되는 NIC	Fast Ethernet (DEC Tulip, Intel), Packet Engines GNIC-I/II	Myrinet	Giganet cLAN1000
지원되는 OS	Linux 2.2.x	Solaris 2.6, Windows NT, Linux 2.2.x	Linux 2.2.x, Windows NT
RDMA: Write	×	×	○
Read	×	×	×
신뢰성: 비신뢰적	○	○	○
신뢰적 전송	×	×	○
신뢰적 수신	×	×	×
보호 태그	○	×	×
Notify 함수	○	×	×
완료 큐	○	×	○
원격 노드 주소	Ethernet MAC 주소 (6바이트)	노드 ID (4바이트)	어댑터 주소 (6바이트)

4. 결론

본 논문에서는 현재 리눅스 플랫폼 상에서 사용할 수 있는 VIA 구현들을 비교하였다. TCP/IP에 비해 지연시간이 감소, 대역폭이 증가하여 근거리 고속통신에 적합한 것으로 판단된다. VIA는 일종의 인터페이스 규격으로 앞으로 하드웨어적으로 지원하는 VI NIC가 다수 출현할 것으로 전망된다. (예: Tandem Servernet II, Giganet cLAN 등) 이에 따라 클러스터 시스템을 위한 경량 고속 통신에 많이 활용될 전망이다. 그러나 Giganet cLAN을 제외한 구현들이 아직 다소 안정적이지 않다. VIA가 활성화되려면 좀 더 많은 NIC이 VIA를 지원하고, 편리하고 안정적인 프로그램 환경이 갖추어져야 한다.

참고문헌

[1] Rajkumar buyya, "High performance cluster computing," Prentice Hall, vol 1, 1999.

[2] Virtual Interface Architecture Specification, draft revision 1.0, 1997.

[3] mvia, "http://www.nersc.gov/research/FTG/via/

[4] Philip Buonadonna, Andrew Geweke, and David E. Culler, "An Implementation and Analysis of the Virtual Interface Architecture,"

[5] "VI Architecture Software Developer's Guide," Giganet, 1999.

[6] Richard Stevens, "TCP/IP Illustrated," vol. 1, 1994.