NEC 시스템에서 크로스 컴파일러 자동화 구현

이영주, 성진우, 장지훈, 이혁로
한국과학기술정보연구원
e-mail:yjlee@kisti..re.kr

The Implementation of Automatic with Cross Compiler in NEC System

Young-Joo Lee, Jin-Woo Sung, Ji-Hoon Jang, Hyeak-Ro Lee
Korea Institute of Science Technology Information

요 약

컴퓨터 시스템의 프로세스 종류는 크게 벡터 시스템과 스칼라 시스템의 두 가지 방식으로 나눌 수 있다. 이는 명령어 처리 방식을 구분한 것으로서 벡터 시스템은 명령어의 처리를 벡터 파이프라인 방식으로 수행한다. 이러한 벡터 시스템은 일반 명령어를 처리할 때는 명령어 실행을 벡터화하여 처리하기 때문에 속도가 향상되지만 반면에 컴파일을 수행할 때는 일반 명령 처리장치에서 스칼라 형태로 처리되기 때문에 그 처리 속도가 저하된다. 이러한 벡터 시스템에서의 컴파일 속도를 개선하기 위하여 별도의 크로스 컴파일러를 사용하는데, 이 때 별도의 크로스 컴파일 서버의 사용을 자동화하여 편리하게 사용할 수 있도록 구성하였다. 벡터 방식은 주로 Cray, NEC, Fujitsu 등의 컴퓨터에 채용하고 있으며 현재 KISTI에서 보유하고 있는 NEC 컴퓨터는 벡터 시스템이다. 본 연구는 크로스 컴파일러 서버를 구축하고 이 서버를 이용하여 자동적으로 크로스 컴파일을 함으로서 사용자는 기존의 일반 컴파일 방식과 동일하게 이용할 수 있기 때문에 사용하기 편리하고 NEC 메인 시스템에 부하를 주지 않으면서 컴파일 속도의 성능 향상을 가져오는 구성 방법을 제시한다.

1. 서론

요즈음은 IT의 발달로 인한 빠른 인터넷 시대이다. 대부분의 업무처리는 고성능의 컴퓨터와 방대한 데이터의 저장 공간을 필요로 한다. 컴퓨터 시스템에 있어서도 신속한 업무처리를 위한 대용량의 컴퓨터와 과학계산을 위한 고성능의 컴퓨터가 계속 발전하고 있다. 이러한 컴퓨터 시스템은 명령어 처리방식에 따라 벡터와 스칼라 컴퓨터로 구분할 수가 있다. 현재 KISTI에 설치된 NEC 시스템은 벡터 컴퓨터로서 명령어의 처리를 벡터 파이프 라인을 통하여 처리한다. 컴퓨터를 프로그램 처리 방법으로 구분할 때는 병렬처리와 벡터 처리로 구분 할 수가 있다. 벡터 컴퓨터는 프로그램이 구조상 병렬처리가 어려운 프로그램에서 처리 속도를 향상시킬 수 있다.

벡터 시스템은 프로그램을 처리할 때는 벡터 처리장치를 사용하고 일반 명령어는 스칼라 처리장치를 사용하기 때문에 벡터 컴퓨터에서 컴파일을 수행할 때는 상대적으로 처리속도가 저하된다. 그래서 컴파일을 수행할 때는 별도의 크로스 컴파일러 서버를 사용하여 컴파일을 한 실행파일을 벡터 컴퓨터에서 수행하는데 이러한 과정이 불편하기 때문에 사용자는 속도가 느리더라도 사용하기 편리한 벡터 컴퓨터에서 그대로 컴파일 하려는 경향이 있다.

따라서 크로스 컴파일 과정을 자동화 하여 벡터 컴퓨터에서 자체 컴파일을 하는 것과 동일한 방법으로 크로스 컴파일을 편리하게 할 수 있으므로 컴파일 속도를 향상시키고 벡터 컴퓨터의 부하가 거의 발생하지 않도록 하였다.

2. 관련 연구

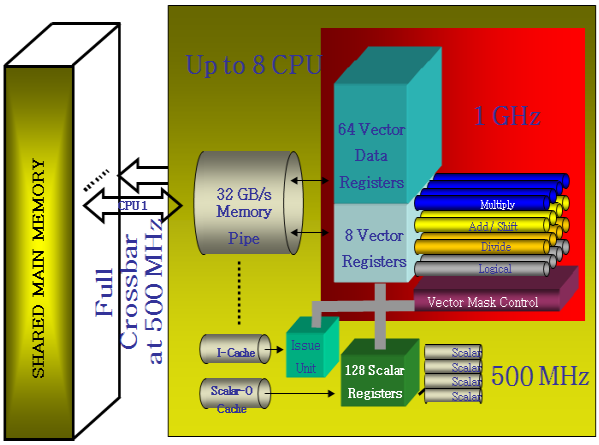
2.1 벡터시스템 특성

NEC 벡터 시스템은 내부에 벡터처리 장치와 스칼라처리 장치를 각각 가지고 있으며 처리하고자 하는 프로그램의 코드에 따라서 각각의 장치에서 처리된다. 스칼라 처리를 할 경우에는 그 속도가 상대적으로 느리게 되는데 그것은 벡터 시스템은 벡터 코드에 더 적합하게 설계되었기 때문이다. (그림 1)에서 NEC 벡터 시스템의 처리장치들을 보여주고 있는 것과 같이 각각 16개의 Add-Shift 파이프라인과 Multiply-Shift 파이프라인이 있다. 각각의 파이프 장치가 모두 동작 중일 때 1Machine cycle(625MHz) 당 1회의 실수연산이 가능하고 모든 파이프 장치가 동시에 동작할 때 1Machine cycle 당 32회의 실수 연산이 가능하게 된다. 스칼라 장치는 2개의 스칼라 로더와 2개의 Floating point/Integer 연산장치가 있다.

10Gflops = (Peak Performance=32/3.2ns) = 10 x 10^9/sec

2.2 파이프 라인

실수 연산을 위하여 4단계의 연산장치를 동시에 작동하여 N+3 Cycle 동안에 N개의 실수 연산이 완료된다.

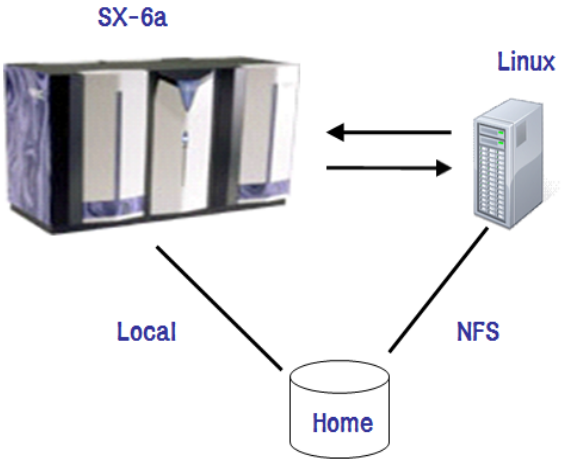


(그림 1) 벡터 CPU 구조

3. 시스템 환경

3.1 시스템 구성

(그림 2)는 벡터 컴퓨터(SX-6)와 크로스 컴파일러 서버의 Linux 컴퓨터의 구성을 나타낸 것이다. 벡터 컴퓨터에서 크로스 컴파일 명령을 수행하면 자동적으로 Linux 서버에서 크로스 컴파일을 수행하여 실행파일이 생성된다. 사용자의 홈디렉토리는 벡터 컴퓨터와는 직접 연결되어 있고 Linux 서버는 NFS를 통하여 연결되어 있다.



(그림 2) 시스템 구성도

3.2 크로스 컴파일러 서버 자동화 구현

벡터 컴퓨터에 로그인하여 Remote shell 등의 명령어를 이용하여 크로스 컴파일을 수행할 경우 각각 서버들 간의 매개변수를 서로 전달하기가 어렵기 때문에 정확한 크로스 컴파일을 할 수 있었던 것을 보완하기 위하여 크로스 컴파일을 하기 위한 별도의 자동화 프로그램을 C로 각각 구현하였다. 홈디렉토리에서 구현한 C 프로그램을 이용하여 매개변수를 전달하여 크로스 컴파일을 하면 소스 프로그램이 크로스 컴파일 서버에서 컴파일되어 현재 위치의 홈디렉토리에 실행파일이 생성된다.

3.3 시스템 사양

<표 1>은 각각의 시스템 사양을 나타낸 것이다. 크로스 컴파일러 서버 성능은 벡터 시스템에 비하면 성능이 매우 낮다.

<표 1> 시스템 스펙

구분	내 용	
	주컴퓨터	크로스 컴파일러
모델명	SX-6 두대	인텔 PC
O.S	SUPER-UX R13	Linux8.0
CPU 수	16	1
Clock(MHz)	625	800
성능	160GFolps	2.7GFolps
주메모리	128GB	1GB

4. 실행결과

<표 3>은 벡터 컴퓨터와 Linux 서버의 크로스 컴파일러와 컴파일 속도를 비교한 것이다. 평균적으로 크로스 컴파일 서버가 8배정도의 성능 향상이 있음을 알 수 있다.

<표 3> 컴파일 성능 비교

	Lang	Compile Time(sec)		Ration
		SX-6	Cross	
Couple	Fortran	40	7	5.7
YONU AGCM	Fortran	906	116	7.8
QCD_vector	Fortran	42	7	6.0
FEM_LSA	C++	884	60	14.7

4. 결론

NEC 벡터 컴퓨터에서 크로스 컴파일러 서버를 구축하고 자동적으로 크로스 컴파일을 할 수 있도록 구성하여 실행한 결과 약 8배의 성능 향상이 있었다. 이러한 크로스 컴파일 과정을 프로그램으로 자동화하여 사용자들이 기존의 벡터 컴퓨터에서 실행하는 것과 같은 방식처럼 쉽고 편리하게 사용할 수 있게 하였다.

따라서 이러한 구성은 컴파일의 성능 향상과 편리하게 이용할 수 있고 벡터 시스템에는 부하를 주지 않는다.

참고문헌

[1] Advanced Computer Architecture by Kai Hwang
[2] NEC SUPER-UX System Design Guide
[3] NEC FORTRAN90/SX Programmer's Guide
[4] NEC C++/SX Programmer's Guide