An Evaluation of PCs for High Energy Physics under Windows NT and Linux [1]

Michael Athanas [a,b], Michael Ogg [c]

[a] *Department of Physics, University of Florida, Gainesville, FL 32611*

[b] *Department of Physics, Cornell University, Ithaca, NY 14853*

[c] *Department of Electrical & Computer Engineering, University of Texas, Austin, TX 78712*

The high availability and commodity prices of Intel-based PCs have made them serious contenders to workstations. We address the question whether PCs are appropriate for running real physics codes. We evaluate the performance of Intel-based systems with two popular 32-bit operating systems, Linux and Windows NT. We report on a suite of benchmark tests, both generic and HEP-specific, where we compare the performance of these two systems to each other, and to other popular workstation class machines.

Key words: Windows NT; Linux; PC; Performance; Benchmark

1 Introduction

The performance of the top-end of the PC market is not only comparable to mid-range workstations, but is actually superior to many workstations on the market today. As well as processor performance, the peripherals and PCI bus speed now match the CPU. In fact, the tremendous growth of 100 Mbps Fast-Ethernet was driven mostly by the PC market. In parallel to the advances in computer hardware, many important software technologies have been developed specifically for the PC. In particular, the availability of robust 32-bit operating systems (OS) allow PCs to compete with the workstation market.

In this paper, we evaluate the performance of Intel-based PCs, using both Linux and Windows NT, and compare them to other workstations. We have used both our own and some readily-available benchmarks written in C, C++,

[1] work supported by the NSF under grant ASC-9318151

and Fortran. We have made measurements of network and I/O performance. Finally, we compare the performance of a real physics analysis program processing data from the CLEO experiment [2].

2 Description of Systems

To minimize machine specific effects, we performed all measurements with the same hardware for both Linux and Windows NT. We used machines with 180 MHz Intel Pentium Pro CPU, Intel motherboard and PCI chipset, and 64 Mbytes of RAM. The data disks were Micropolis Tomahawk 3391 9.1 GB Ultra-Wide SCSI, with Adaptec 2940UW SCSI adapters. The network interfaces were SMC 9332BDT 10/100. Two IDE drives were used for system disks. On one IDE drive, we installed Microsoft Windows NT 4.0 Workstation using default drivers. On the other IDE drive, we installed Linux v2.0.18, using the RedHat 4.0 distribution. The only non-standard component was a more recent driver for the Ethernet NIC. We used two systems, each connected to a port of an SMC TigerSwitch 100 Mbps Fast-Ethernet switch.

For Windows NT, we used the Microsoft Fortran PowerStation 4.0 and Microsoft Visual C++ 4.0 compilers. Code was optimized with the `/G5 /Ox` flags. For Linux, we used the GNU g77 v0.5.18 Fortran compiler, and gcc v2.7.2 C/C++ compiler. Code was optimized with the `-O4` flag. The g77 compiler produced significantly higher performance code than translated f2c code.

As a reference platform, we performed the equivalent measurements on a Digital Alpha 250 4/266 running Digital Unix v3.2g. We compiled code with the Digital Fortran compiler f77 and the GNU gcc 2.7.2 compiler. Code was optimized with the `-O4` flag. For comparison with other workstations, we also did some of the measurements (all except those requiring a Fortran compiler) on a SUN UltraSparc with Solaris 2.5.

3 I/O Performance

As a first test, we measured the I/O performance of the systems. We found that this is almost purely a function of the hardware. We measured the speed by copying a 512 MB file to the `null` device to avoid any misleading measurements from caching a small file. An interesting feature of modern disks is that the bit density is essentially constant resulting in more sectors per cylinder at greater radius. Therefore, depending where the file was placed on the disk (which we controlled by creating several partitions), we could vary the I/O speed. We observed I/O rates ranging from 6.6 to 9.7 MB/sec, which

Processor	disk type	disk interface	I/O Speed
PPro, 180 MHz	Micropolis Tomahawk	Ultra-Wide SCSI	9.7 MB/sec
PPro, 180 MHz	Seagate Medalist	IDE	3.1 MB/sec
PPro, 200 MHz	Quantum	Wide SCSI	4.8 MB/sec
AXP, 266 MHz	Seagate Elite	Narrow SCSI	5.8 MB/sec

Fig. 1. I/O Speeds Measured using CLEO I/O Libraries

we compare to the manufacturer's internal data rate specifications of 76 to 125 Mb/sec [5].

We then used a program, *rio*, that is completely I/O bound and simply calls the CLEO low level I/O routines to read logical records from a local disk file. The results are shown in Figure 1. We show results for the highest performance drives (Micropolis Tomahawk), other SCSI drives on both PCs and Digital AXP, and for IDE drives. The latter are of interest because IDE drives are very cheap, but it is clear that their performance is not comparable to SCSI drives (in addition to being able to use fewer drives on a single machine). We note that the advantages of Ultra-Wide disks (with a SCSI bus speed of 40 MB/sec) become important only when there is more than one drive on the SCSI bus. However, even though the Digital AXP has a PCI bus, drivers for Ultra-Wide controllers such as the Adaptec 2940UW are not yet available.

4 Network Performance

Next we evaluate the network performance of NT and Linux on a 100 Mbps network. We created a simple sockets network benchmark test, which measures network throughput as a function of packet size for both TCP/IP and UDP/IP protocols. The program times the transmission of a series of buffers from a client machine to a server. These measurements are performed as a function of the buffer size. The results for the two protocols under both OS's are displayed in Figure 2.

For both protocols on NT, we observe a gentle rising slope for small buffer sizes which is attributed to the CPU's inability to construct the protocol stack fast enough to keep the network saturated. For NT, we observe nearly complete utilization of the network with buffer sizes of 1024 Bytes.

On Linux, we observe very poor TCP/IP performance, which we attribute to known problems with the NIC driver. Using a different NIC (3Com 595), we

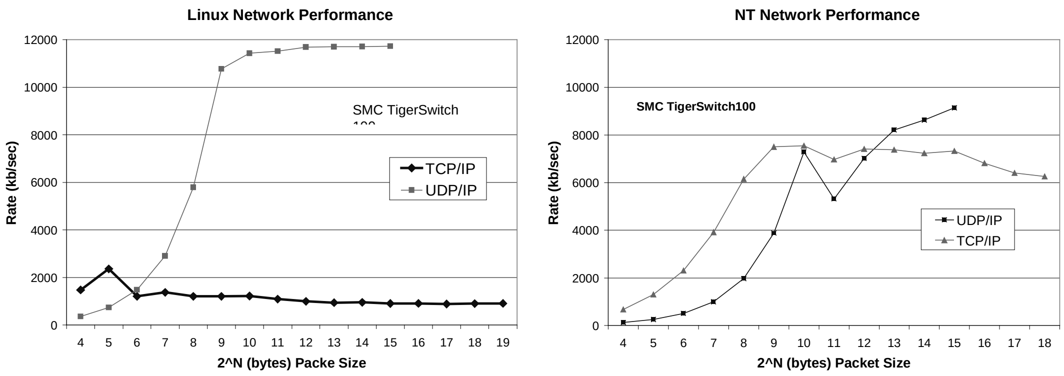

Fig. 2. Network performance for (a) Linux and (b) Windows NT on a SMC Tiger-Switch 100 Mbps ethernet switch

see more consistent results. However, UDP/IP manages to completely saturate the network for large packets.

5 CPU Benchmark Performance Comparison

We next examine the performance of compiled code across all three platforms to provide a comparison of CPU intensive non-IO scientific algorithms to obtain baseline measurements:

(i) *STL sort*, a simple program which populates STL containers, each with 10^6 random `int` and `float` variables, does mathematical operations on them, and then sorts the containers.
(ii) *Byte Benchmark*, a suite of programs that do extensive tests [3,7]. Here we just selected the arithmetic results of performing loops on `int` and `float` variables. This benchmark allows us to make comparisons with other workstations using C code.
(iii) *mdbnch*, a Molecular Dynamics Fortran program [8] which simulates the temporal evolution of multi-particle interacting systems.
(iv) *dft*, a Discrete Fourier Transform program [9] which is also in Fortran.

The results are shown in Figure 3. The performance varies considerably depending on the benchmark. Results for both Linux and Windows NT range from being relatively poor to being outstanding. Much of this can be attributed to floating point performance – the Intel x86 has excellent integer but weaker floating point performance. For applications that depend heavily on 64-bit floating point performance (*e.g. mdbnch*), the Digital AXP is a strong performer. Since we expect the basic performance of Windows NT and Linux to be similar, it is clear that there are also strong (and seemingly unpredictable) compiler differences.

Processor	OS	STL sort	Byte int/float	mdbench	dft `04/00`	CLEO `rchk`
AXP, 266 MHz	OSF/1	1.00	1.00/1.00	1.00	1.00/1.00	1.00
AXP, 500 MHz	OSF/1	1.66	1.93/1.52	1.68	1.73/2.12	1.57
PPro, 180 MHz	NT	0.97		0.83	0.78	0.25
PPro, 180 MHz	Linux	0.68	3.05/0.90	0.46	1.43/2.48	0.78
UltraSparc	Solaris	0.47	0.90/0.98			

Fig. 3. Benchmark Performance. Results are normalized to AXP 266 MHz. Higher numbers denote better performance.

6 Physics Analysis Performance

The fundamental goal and motivation of this work is to determine the relative (Linux vs. Windows NT) and absolute (Intel vs. Digital AXP) performance of a genuine physics analysis. We have ported the physics analysis libraries from the CLEO experiment to both Windows NTand Linux. For the Windows NT port, we used the Cygnus [4] NT/Posix utilities to adapt the existing `Makefile` system to Windows NT. We found Microsoft Fortran's inability to recognize C-preprocessor commands inconvenient but not too difficult to overcome. We rewrote the CLEO data I/O routines to take advantage of WIN32 specific I/O calls. After making the necessary code changes, we used this code base for the Linux port which progressed smoothly. On Linux, we used the GNU g77 compiler and observed extremely fast compilation speeds. Essentially all of the work of porting was because of poorly written code, in particular the use of non-standard Fortran features and sloppy use of pre-processor directives. Several bugs (in both Fortran and C code) were corrected. Both Microsoft Fortran and GNU g77 were extremely particular (as they should be) about adherence to ANSI Fortran 77 standards.

The CLEO software is dependent upon CERN libraries. The Linux port of the libraries was readily available. We obtained an unofficial pre-release of the CERN libraries for Windows NT – the first official release is scheduled for February 21.

We wrote a CPU bound sample physics analysis program, *rchk*. This program touches most of the data attributes, calculates masses, plots results, etc. The performance of *rchk* for Windows NT, Linux, and Digital Unix are also shown in Figure 3. The Linux results do not completely agree with the "correct" results (which may be due to porting issues or possibly program bugs that we have not yet fully corrected), and so might affect the measurements. Nevertheless, Linux appears to be very competitive with Digital Unix. The Windows NT result is about one third the speed of Linux. After examining the performance profile of *rchk* on NT, we noticed that a substantial time is spent in CERN library matrix manipulation routines. Evidently, the Windows NT version of these routines has not been optimized.

7 Comments on Linux

The Linux operating system was chosen because it is free and widely available, POSIX-compliant, open (allowing software and hardware customization where necessary), familiar (just another Unix), supports a range of development tools, and has a large world-wide support base. We have had substantial experience with many Linux systems for the last 3 years and found Linux to be stable and easy to install. Any user with Unix experience is able to use Linux instantly. As the porting work demonstrated, and we have seen from other applications, if a program runs on one Unix, there is little problem getting it to work under Linux. The GNU g77 Fortran compiler is still only a beta version, but it seemed adequate, although its optimizing could possibly be improved. We used the GNU gdb debugger with the `ddd` [10] interface. This is excellent for C++, but somewhat limited for Fortran. Linux makes a very large range of system information available through the `/proc` file system, although a user or system administrator might have to write specialized applications to view the information.

8 Comments on Windows NT

Windows NT is the most widely used true 32-bit operating system. The available quality software development tools are unsurpassed by any other contending OS. We found Windows NT easy and intuitive to use. Installation is quick, simple, and self-tuning. Once correctly configured, the system is secure, stable, and reliable. After a POSIX utility library is installed, the system may be configured satisfactorily for users accustomed to a Unix environment. The operating system is extensible through freely available device driver development kits. Windows NT has an impressive array of extensible built-in monitors which makes it very easy to analyze system performance. It is also host to several quality integrated development environments (MS Developer's Studio, Borland C++, etc). Windows NT's enforced cross application conformance to a common object model (DCOM/OLE/ActiveX) makes it a powerful platform for software development. Encapsulation and code reuse are only one aspect of the multitude of Rapid Application Design packages (Visual Basic, Delphi, Optima++, etc.).

9 Conclusions

We believe that the performance of Intel-based computers makes them impossible to ignore. On an absolute scale, they perform as well as, or even better, than other high performance workstations but for a fraction of the cost. Both Windows NT and Linux are quality operating systems. The choice between them will come down to preferences and style rather than performance and stability – each OS has its strengths and unique advantages. In fact, in many situations it will be desirable to use both OS together.

To really take advantage of a large number of cost effective machines, reliable systems' integration and management are required. This is the focus of the NILEproject [1,6], where we are building systems based on large numbers of commodity computers. Here we show that the performance of PCs (using either Windows NT or Linux) is comparable to conventional workstations at a fraction of the cost.

References

[1] D. G. Cassel and M. Ogg. Nile: Distributed Computing and Databases for High Energy Physics. In *NSF Grand and Naational Challenges Workshop*, March 1996.

[2] The CLEO Collaboration (Yuichi Kubota et al.). The CLEO II Detector. *Nuclear Instruments and Methods*, A320:66–113, 1992.

[3] `http://www.byte.com`.

[4] `http://www.cygnus.com`.

[5] `http://www.micropolis.com`.

[6] `http://www.nile.utexas.edu`. NSF, ASC-9318151.

[7] `http://www.silkroad.com/bass/linux/bm.html`.

[8] `http://www.sissa.it/furio/Mdbnch/info.html`.

[9] `http://www.tdb.uu.se/yggdrasil/f77test/html/f77-all.html`.

[10] Dorothea Lütkehaus and Andreas Zeller. Technische Universität Braunschweig.