

SAS® 6.09 on a CONVEX® C3840: First Impressions

Kenneth A. Hardy

Sally M. Muller

University of North Carolina at Chapel Hill

Abstract

The architecture of the CONVEX C3 series of processors should offer memory address space, I/O and computational speed advantages for running the SAS System. The University of North Carolina at Chapel Hill has been an experimental and beta test site for the development of SAS 6.09 under ConvexOS® and has been running the production version of SAS 6.09 since January, 1994. This paper provides our initial impressions of running version 6.09 on a CONVEX C3840 and compares it to SAS running on several other platforms on our campus.

Introduction

One of the major advantages of the SAS System is the consistency of its user interface, syntax and powerful data access, reporting and analytical procedures across a wide variety of computing platforms. It has been far and away the most heavily used applications software at the University of North Carolina at Chapel Hill. It is used by students faculty and staff on platforms that include Intel based machines running DOS, Windows and OS/2®, UNIX workstations from DEC, Hewlett Packard, IBM® and Sun, DEC VAX machines running VMS and IBM mainframes running under both MVS/TSO® and VM/CMS®. Until recently, the only publicly available processor on campus that did not run SAS was our CONVEX system. Users of this machine needed the many data management and statistical features of the SAS System but had to move their data to other systems to get them. Thus, we were intrigued when we heard that SAS Institute in cooperation with CONVEX Computer Corp. was developing a version of SAS 6.09 for CONVEX systems and offered to become a development and beta test site. This paper will try to summarize our first impressions of using the production release of SAS 6.09 on our CONVEX C3840. We will begin with a brief description of the architecture of a CONVEX C series machine, then provide some performance benchmarks and finally review some problem areas.

What is the CONVEX C3 series?

The CONVEX Computer Corp. has been shipping low cost "supercomputer" class machines since 1985. The C3 series of 64 bit, tightly coupled multiprocessors was introduced in 1991 and consists of the C3200, C3400 and C3800 product lines. Each processor has both a scalar and integrated, high speed vector processor. According to Ihnen and Crawford (1993) the features of the C3 series that have implications for running the SAS System include :

1) Very large physical and virtual memory space.

The C3200 line supports up to 2 gigabytes (GB) and both the C3400 and 3800 lines support up to 4 GB of physical memory. Virtual memory, of course, can be in excess of physical memory but is limited to 4 GB. Practically, it is also limited by disk space available for swapping. While the maximum virtual address spaces of many open systems platforms are currently 2 GB, their physical memories are generally substantially less.

2) High speed computation

By providing a crossbar feature that allows both CPU and I/O functions to access memory simultaneously, the C3 series makes efficient use of multiple processors. The C3200 and C3400 series can be expanded to four processors while the C3800 is expandable to 8. In addition numerically intensive tasks that can be vectorized can benefit substantially from the fact that each CPU has an integrated vector processor.

3) Large file support

ConvexOS supports single file sizes up to 100 GB which is in excess of the usual 2 GB limit of most other open systems. Thus researchers desiring to process very large data files (e.g. 1990 Census 5% PUMS) may do so without having to segment those files.

4) Very fast sequential I/O

The C3 series I/O architecture is optimized for very fast sequential I/O. Disk striping and very large dynamic buffer caching (up to 512 megabytes) can yield very high data transfer rates for single file tasks, multiple file tasks and sorts. Maximum transfer rates vary from 65 MB/Second on C3200 series machines to 100 MB/Second on the 3800 series.

5) Enhanced UNIX operating system

While somewhat enhanced to provide greater I/O capacity and large systems administration features, the basic operating system commands are virtually identical to Berkeley 4.2/4.3 UNIX allowing users familiar with open systems to begin working productively on the system quickly.

SAS 6.09 on UNC CONVEX Platforms

UNC has implemented four versions of SAS 6.09 under ConvexOS. In July 1992 we began running the pre-release version of SAS

6.09 under ConvexOS on a model C240. We installed the experimental SAS 6.09 in March, 1993. The beta release was installed in August 1993 on the C240 and in September we moved the beta release to our newly acquired C3840. We received the production version of SAS 6.09 in November 1993. This was a complete version of the SAS System and included all SAS System products available on other UNIX platforms. Although there were some initial problems with the production release, the SAS Institute developers and technical support were very cooperative and solved these problems in a timely fashion.

CONVEX C3 Advantages for SAS

What are the implications of the above features for users of the SAS System? There are several. First, the large physical and virtual memory support allows analytical PROC's which can create very large matrices a greater chance to keep either all or a very significant portion of these matrices in memory. Memory intensive Proc's like those in SAS/OR, the more complex linear modelling PROC's like VARCOMP, MIXED, GLM, REG and CATMOD and iterative estimation PROC's like LOGISTIC benefit from this feature. Some PROCs require that the design matrix or matrices be kept resident in memory (GLM, VARCOMP, MIXED, REG, CATMOD and LOGISTIC). On the C3840 at UNC the physical memory is 1 GB and the SAS system MEMSIZE option is set at 512 MB. This can greatly enhance the speed of calculations for large problems and can allow users to solve very large SAS problems without receiving SAS memory error messages.

A second implication is that many of the same PROC's that use large matrices can employ vector processor support which further improves execution speed. Users of some UNIX workstations and PC's at UNC can take advantage of the large memory and vector processing support on the CONVEX C3840 to undertake problems that would either take too many hours or not run at all on these smaller platforms.

Third, many SAS programs are characterized by multiple DATA and PROC steps which involve one or more PROC SORT steps and can easily become I/O bound on many systems. The fast sequential I/O of the CONVEX can speed up multistep SAS programs working with very large files.

Some Cross Platform Comparisons

The University's central computing organization will be moving to a distributed UNIX based environment over the next several years and is considering the current CONVEX machine as an important transition machine for providing central SAS services for the campus. In order to see where the CONVEX fits in the campus computing picture we wanted to get a preliminary assessment of the efficiency of the CONVEX C3840 relative to the current IBM MVS system, a DEC 5000 workstation and an INTEL platform for handling SAS tasks. The current UNC CONVEX system is a quad processor C3840 with 1 GB of RAM and 20 GB of disk space available to users. Our IBM MVS system is a 3090 170J with 256 MB RAM (128 MB central plus 128 MB expanded), and approximately 100 GB of older 3380 disk drives and controllers. The DEC platform is a DEC 5000 Model 240 with 32 MB of memory and 2 GB of disk running Ultrix. The INTEL machine is a 66 MHz. 486DX2 IBM ValuePoint with 24 MB of RAM and a 528 MB IDE hard drive running under OS/2. See Appendix A for further details on these systems.

Our initial benchmarks consisted of a series of SAS DATA and PROC steps designed to represent a range of SAS computing tasks found on our campus. We tried to use reasonably challenging tasks that we felt stretched the capacity of smaller platforms and would give us an indication of the relative performance of the two larger platforms. As indicated in Table 1 we divided these tasks into those that were meant to be primarily I/O intensive (multistep PROC FREQ and PROC MEANS and two PROC SORT's), primarily CPU intensive (PROC MIXED), a combination of I/O and CPU intensive without use of vector processing (PROC CATMOD) and I/O and CPU intensive with vector processing (PROC LOGISTIC). Each benchmark was run on each platform at least three times. All benchmarks were run in "batch" mode without the SAS Display Manager. We also tried to run each benchmark under low, medium and high system loads on the CONVEX and IBM MVS systems by running them either early morning or late night, the middle of the morning and during the middle of the afternoon. All code was run with the FULLTIMER system option. The MEMSIZE= option, however was different on each platform. For the CONVEX this was 512 MB, and for IBM/MVS 32 MB. For the DEC it was 32 MB and for the OS/2 system there is no MEMSIZE= option rather the system simply uses as much memory as is available. No special "tuning" was done with any system options other than MEMSIZE. The settings of other performance options, are contained in Appendix B.

Because task CPU times on all platforms were remarkably stable, all the CPU time results we present are average figures. Of course, elapsed times on the two shared machines varied with system load so we report ranges for elapsed times. SAS under OS/2 does not report elapsed time. However, elapsed times under OS/2 were very close to CPU times.

Table 1: Benchmarks for Comparing SAS on a CONVEX 3840 to Other Platforms

<u>Benchmark Characteristics</u>	<u>SAS PROC</u>
Test 1: I/O Intensive, Multistep	DATA step generates 100,000 observations. Then 2 PROC FREQ's and PROC MEANS construct multiple tables using 100,000 observation file.
Test 2: I/O Intensive, Sort	PROC SORT on 128 MB file of short (128 byte) records and 128 MB file of long (2048 byte) records
Test 3: CPU intensive	PROC MIXED for about 16,000 observations
Test 4: I/O & CPU No Vector Processing	PROC CATMOD, large model, 102,000+ observations
Test 5: I/O & CPU, Vector Processing	PROC LOGISTIC, large model, 102,000+ observations

Test 1: I/O Intensive Multistep

Many researchers at UNC use large United States Census or national health survey files to construct crosstabulations and tables of means. Our first benchmark attempted to simulate this activity by running a multistep SAS job that generated 100,000 observations in a DATA step and then used that data set for two PROC FREQ's and a PROC MEANS. The code used to construct this benchmark is essentially the same as that presented in Sapp and Tschudi (1992) but with the number of cases increased to 100,000. The code used for this benchmark was :

```
DATA a;
  LENGTH DEFAULT=4;
  DO n=1 TO 100000;
    y=INT(UNIFORM(77777)*5);
    x=INT(UNIFORM(77777)*7);
    w=UNIFORM(77777)*10;
    z=INT(UNIFORM(77777)*24);
    c=' ';
    IF w>2 THEN c='A';
    IF w>7 THEN c='B';
  OUTPUT;
  END;
run;

PROC FREQ;
  TABLES (y c) * (x z) w;
  TABLES x*z / NOROW NOCOL NOPERCENT;
  FORMAT w 4.1;
run;

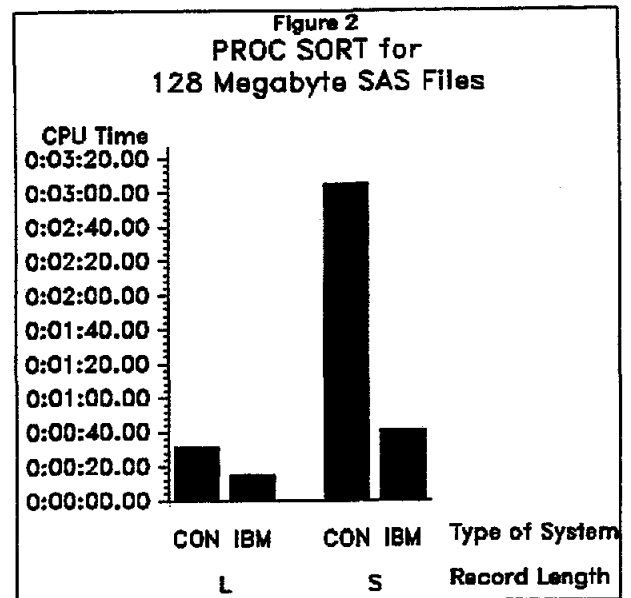
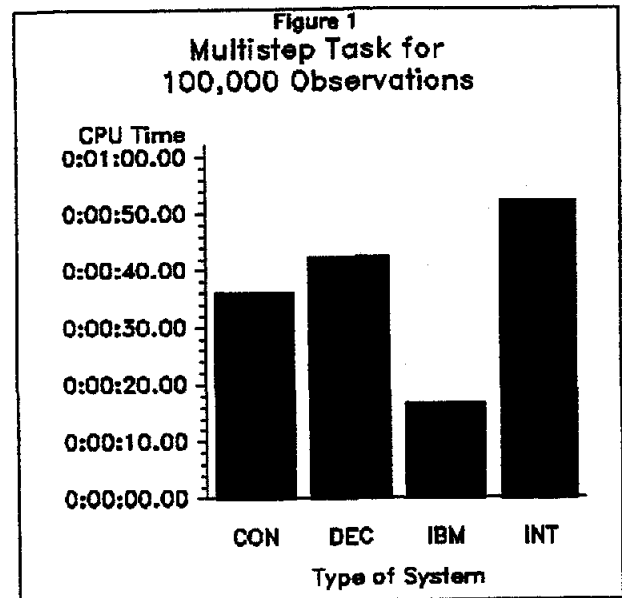
PROC FREQ;
  TABLES x*y / NOPRINT OUT=b;
run;

PROC MEANS DATA=a;
run;
```

The comparison of the mean CPU time used by the four platforms is presented in Figure 1. Note that SAS under IBM/MVS is about twice as quick as the CONVEX and about 3 times faster than the DEC and INTEL-OS/2 system. IBM/MVS was also quicker with respect to elapsed time. It always completed this benchmark in between 50 and 70 seconds. The DEC and INTEL/OS/2 platforms completed the task in one to two minutes. Elapsed times on the CONVEX were slowest and ranged from about 2 minutes 30 seconds to about 52 minutes.

Test 2: I/O Intensive Sort

Our second I/O benchmark sorted two approximately 128 megabyte SAS data sets on two variables with randomly assigned values. The first was composed of 65,536 records with a length of 2048 bytes while the second contained 1,048,576 short records with a length of 128 bytes. Our DEC and INTEL-OS/2 systems could not perform these sort tasks due to the lack of adequate free disk space. The mean CPU times for the CONVEX and IBM systems are shown in Figure 2. For long records the IBM platform is about twice as fast as the CONVEX. However for short records it is over four times faster. Elapsed times were another story, however. For both short and long records the CONVEX System's elapsed times were generally shorter. For short records the CONVEX's elapsed times ranged from about 1 minute to 4



minutes while the IBM/MVS system's times varied from 7 to 10.5 minutes. For long records, the CONVEX had times of 54.89 seconds, 1 minute 5 seconds and 2 hours 3 minutes. The IBM/MVS machine's times were 7 minutes 2 seconds, 10 minutes 7 seconds and 10 minutes 25 seconds. Thus, with one notable exception, the CONVEX's elapsed times were quicker than the IBM's.

Test 3: CPU Intensive

The CPU intensive task involved a PROC MIXED step that we borrowed from our Department of Biostatistics Collaborative Studies Coordinating Center. It involved a 1/2 MB SAS data set with clinical data from a repeated measures study involving approximately 16,000 observations. The PROC MIXED code was

```
PROC MIXED DATA=nomiss
  METHOD=ml ABSOLUTE CONV=0.0000001 NOCLPRINT;
  CLASS id site ;
  MODEL y = site cage1 site*cage1 / PREDICTED S;
  REPEATED site / SUBJECT=id TYPE=un;
  ID id site;
  MAKE 'Predicted' OUT=pred(KEEP= id site obser pred);
  MAKE 'Covparms' OUT=var ;
```

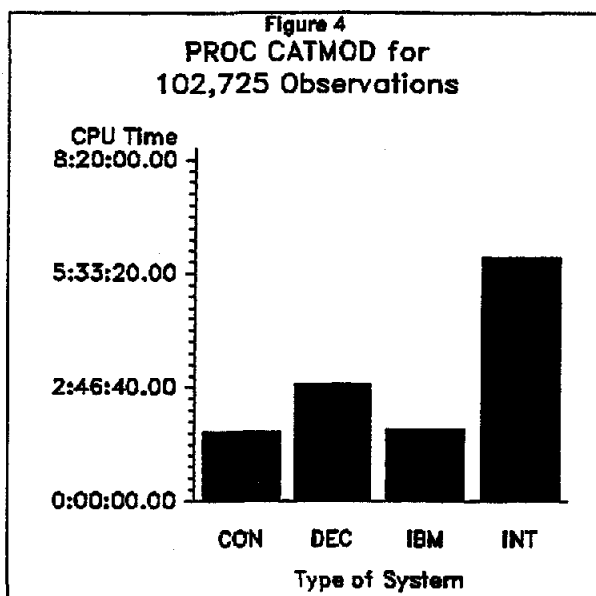
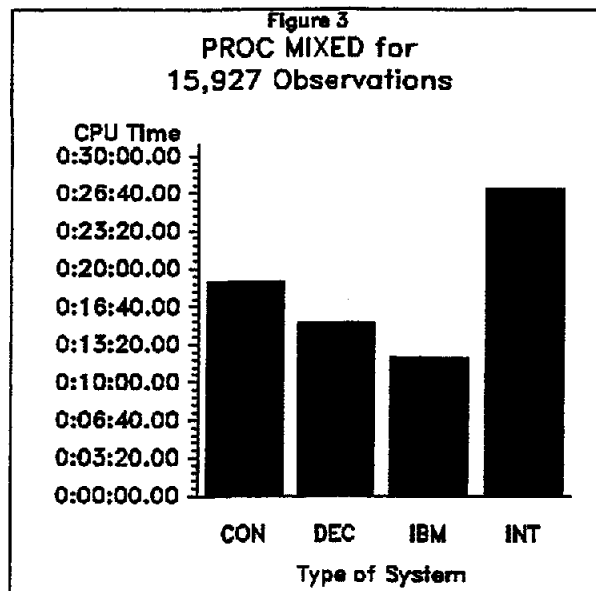
From Figure 3 we can see that the CONVEX mean CPU time was about 1.5 times slower than IBM/MVS and just slightly slower than the DEC 5000's. Not surprisingly, the INTEL-OS/2's mean time was the slowest -- about 1.5 times that of the CONVEX. Elapsed times, however, proved to be a very different story. The INTEL-OS/2 platform's CPU times were very close to its elapsed time so it consistently completed the benchmark in less than 30 minutes. The DEC machine's elapsed times ranged from about 23 minutes to 43 minutes. IBM/MVS's times were from about 38 minutes to 1 hour 42 minutes and the CONVEX's lowest time was about 1 hour 38 minutes and the highest was about 11.5 hours.

Test 4: I/O & CPU

Our I/O and CPU intensive task without a vector processing component was borrowed from a sociologist. It involved using PROC CATMOD to estimate a multinomial, logistic regression model with 25 predictor variables for over 102,000 observations using maximum likelihood. It proved to be the most CPU intensive benchmark we ran. The PROC CATMOD code was:

```
PROC CATMOD DATA=in.biglogis ;
  RESPONSE LOGIT;
  DIRECT age agesq vertical division Insicb
    denp denpsq birthden Incmpete agesqdr affil2
    rep war1 nira1 strk1 unfl1 unall1
    totmer11 tottdh1 found11 lgnpl1
    cmerge ctransf cspIn peakassc;
  MODEL ends=
    age agesq vertical division Insicb
    denp denpsq birthden Incmpete agesqdr affil2
    rep war1 nira1 strk1 unfl1 unall1
    totmer11 tottdh1 found11 lgnpl1
    cmerge ctransf cspIn peakassc
  /ML NOGLS NOPROFILE NODESIGN ;
```

The mean CPU times presented in Figure 4 show that the IBM and CONVEX platforms achieve CPU time parity for this task. Each devoted roughly one hour and forty minutes of CPU time to this problem. The DEC machine took just over an hour longer on average while the INTEL/OS/2 platform needed close to six hours. Both the best and worst elapsed times were turned in by the larger machines. They each completed a CATMOD run in about 3 hours. However, the CONVEX machine had one run that took 41.5 hours during a period of very heavy system use and the IBM/MVS system had one run that took 9 hours. The DEC 5000 had elapsed times between 4 and 5 hours while the INTEL/OS/2 platform consistently took just under 6 hours.

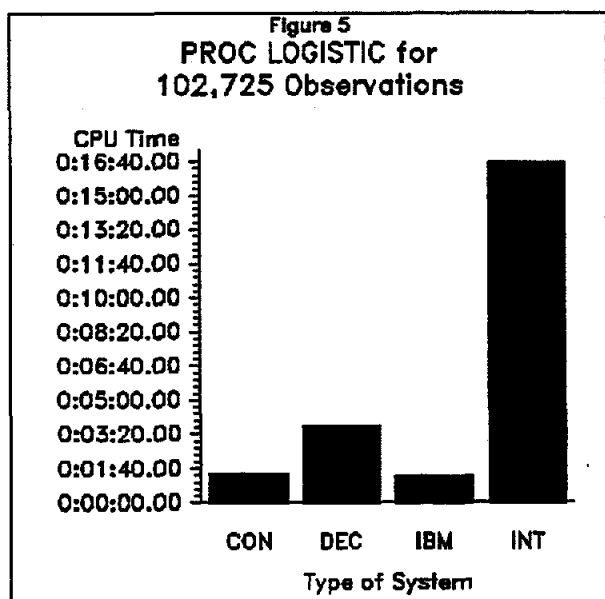


Test 5: I/O & CPU Vector Processing

We used the same sociologist's data set for our I/O and CPU intensive task with vector processing. We simply recoded the dependent variable down to two categories instead of three and ran the same 25 variable model with PROC LOGISTIC which does contain vectorized code for both the CONVEX and MVS versions of SAS. The MODEL statement code was similar to that shown above for PROC CATMOD.

```
PROC LOGISTIC DATA= in.biglogis ;
  MODEL ends2=
    age agesq vertical division Insicb
    denp denpsq birthden Incmpete agesqdr affil2
    rep war1 nira1 strk1 unfl1 unall1
    totmer11 tottdh1 found11 lgnpl1
    cmerge ctransf cspIn peakassc;
```

The relative CPU performance of the platforms is shown in Figure 5. As in the case of our PROC CATMOD benchmark, the MVS and CONVEX systems are quite close in terms of CPU time with the CONVEX system being only marginally slower. The DEC platform was about 3 times slower than the large machines. OS/2 devoted about twelve times more CPU time to this task than either the CONVEX or IBM platforms. The CONVEX system's elapsed times ranged from under two minutes to five minutes to just over 21 minutes, while MVS's elapsed times ranged from about 4 to 6 minutes. The INTEL-OS/2 system consistently spent about 17 minutes on this problem. The DEC machine turned in times ranging from 27 minutes to 40 minutes.



DATA Steps and Missing Values

According to Ihnen and Crawford's 1993 SAS Institute white paper on using the CONVEX C series with SAS mentioned above, the C series was designed with scientific computing for data without missing values in mind. They indicated in that paper that DATA steps used to calculate new variables from existing variables containing missing data may experience "severe performance degradation". Because many SAS applications in the health and social sciences on our campus can involve data with significant amounts of missing values, we were somewhat concerned to read this and thought we'd best try to investigate the matter further. We constructed a modest size SAS data set with five variables and 5,000 observations. We used separate RANUNI functions to set the values of all the variables to independent, uniform random numbers in the range zero to 1. The code to construct this data set was

```
DATA complete ;
  n = 5000 ;                /* number of cases */
  ARRAY xs x1 - x5;        /* number of vars */
  DO i = 1 to n ;          /* generation loop */
    DO OVER xs ;
      xs = RANUNI(7943287) ; /* uniform random values */
    END;
  OUTPUT ;
END;
```

We then took this basic data set and used the values of the five variables to set approximately 2.5, 5, 10 and 15 percent of the values to SAS missing values. The resulting data sets were then run through a DATA step that calculated 4 additional variables and the datasets with the computed values were then run through PROC MEANS and PROC REG. For example, the code for 2.5 percent missing data was

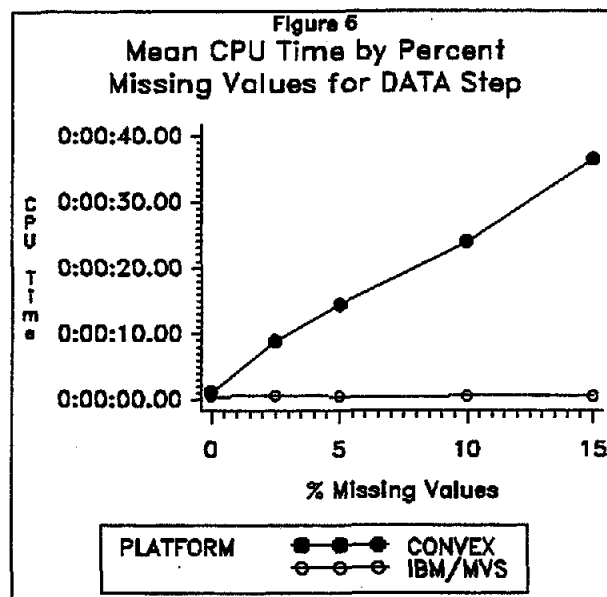
```
DATA pct2_5 ;
  ARRAY xs x1 - x5 ;        /* original variables */
  SET complete ;            /* complete data set */
  DO OVER xs ;
    IF xs <= .025 THEN xs = . ; /* approx. % missing */
  END;
OUTPUT;
```

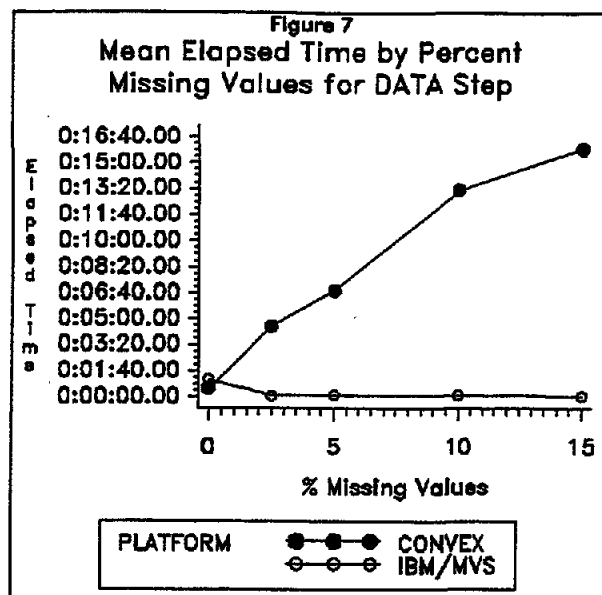
```
DATA step2_5 ;              /* calculate 4 more */
  SET pct2_5 ;
  y1 = (x1 + x2 + x3) + RANNOR(3209845) * .2;
  y2 = x1 + x2 + x3 + x4 + x5 + RANNOR(217691) ;
  y3 = SUM(OF x1,x2,x3,x4,x5) + RANNOR(674391) * .5 ;
  y4 = LOG(x4/x5) ;
```

```
PROC MEANS ;                /* use each variable */
```

```
PROC REG ;                  /* use all together */
  MODEL y1 = x1 x2 x3 x4 x5 y2 y3 y4 / COLLIN ;
```

Figure 6 plots the mean total CPU used for the DATA steps against the percentage of missing data. The top curve gives the results for the CONVEX while the bottom curve gives results for MVS for comparison. It is apparent that there is a significant problem with handling missing values in DATA steps on the CONVEX. While performance on the IBM platform remains about constant, on the CONVEX CPU time devoted to the DATA step increases at an alarming rate as the percentage of missing values increases. The mean CPU time expended on the DATA step for the SAS file with fifteen percent missing values is about thirty times more than that used for no missing values. The problem appears more severe when elapsed time is examined. Figure 7 presents mean elapsed





time by percentage of missing data. The average time spent waiting for the DATA step to run increases from about 30 seconds to 16 minutes for a relatively small file! Even for a file with a modest 2.5 percent missing values, the time increases by a factor close to 9.

What can be done to deal with this problem? The obvious solution is for SAS Institute and CONVEX to get together, figure out a fix for the problem and implement it in either a maintenance release or the Version 6.10 release. Our understanding is that due to the way ConvexOS handles interrupts, implementing a clean fix may be quite challenging. We are not aware of any plans for providing a fix for this problem. Until more definitive information is available from SAS Institute on this point, users of Version 6.09 should be aware of the problem and handle it by explicitly checking for missing values in the DATA step code before assignment statements involving variables with missing values. To illustrate the major performance improvement that handling missing values explicitly can make, we recoded the DATA step above using IF statements and the NMISS function so that it appeared as follows:

```
DATA if2_5 ;      /* check mvs before doing calculations */
SET pct2_5 ;

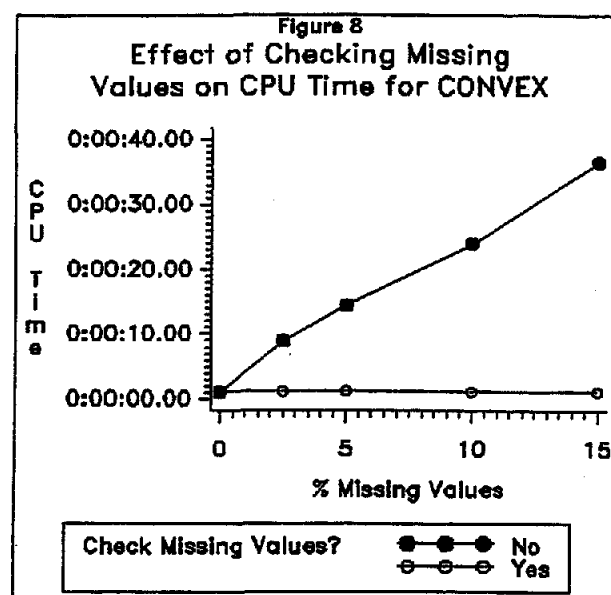
IF NMISS(OF x1 - x3) = 0 THEN
  y1 = (x1 + x2 + x3) + RANNOR(986735) * .2;

IF NMISS(OF x1 - x5) = 0 THEN DO ;
  y2 = x1 + x2 + x3 + x4 + x5 + RANNOR(217691) ;
END;
ELSE DO;
  y2 = . ;
END;

y3 = SUM(OF x1,x2,x3,x4,x5) + RANNOR(674391)*.5 ;

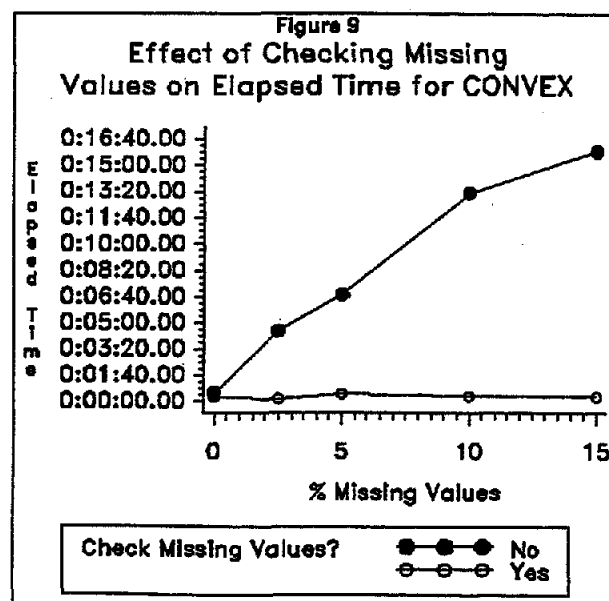
IF NMISS(OF x4, x5) = 0 THEN y4 = LOG(x4/x5) ;
```

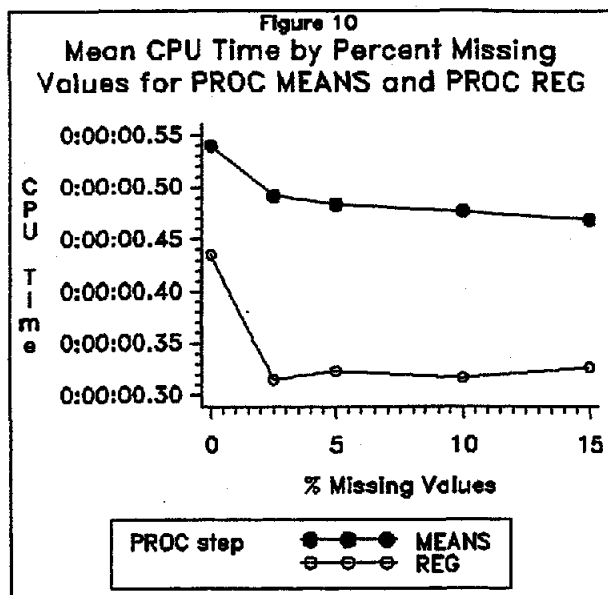
Figures 8 and 9 show the differences this can make in CPU time and elapsed time respectively. The top curve in each graph



corresponds to the ones in Figures 6 and 7 while the bottom curves are the mean CPU and elapsed times for the recoded DATA steps. Clearly, the extra coding effort is rewarded with much more efficient use of CPU resources and speedier execution time.

Do missing values also cause performance degradation in PROC steps? We certainly didn't have time to check every PROC, but we did use the same strategy of using data sets with increasing percentages of missing values with PROC MEANS(SUMMARY) and PROC REG. PROC MEANS treats each variable independently when dealing with missing values while, by default, PROC REG looks at the set of variables comprising a single MODEL statement. Figure 10 shows that there seems to be no impact of missing value on these PROC's and our guess is that this would be true for most PROC's that screen data for missing





values prior to computing statistics. It may very well be true of most PROC's, but further investigation is needed.

Summary

So how does running SAS 6.09 on the CONVEX stack up against running SAS on other platforms? It would appear that the general answer is that "It depends on what you're doing and don't trust benchmarks based on CPU times." The CONVEX's CPU performance was never superior to that of the IBM/MVS system, although for the two I/O and CPU intensive tasks it was very close. It was usually better than the DEC 5000 and always better than the INTEL system. However, its performance with respect to elapsed time, which is most important to faculty and students needing to get work done, depended on the task being performed and the concurrent load placed on it by other users. Under light to moderate loads, its elapsed times on the CATMOD and LOGISTIC benchmarks were equal to or better than those of any platform. However, its elapsed time performance on the PROC MIXED task were always eclipsed by the two smaller platforms. The CONVEX's most impressive elapsed times were on the two PROC SORT benchmarks where its quickest times were 1.5 to 10 times faster than the next best system.

While the CONVEX could be better at times on some tasks than the other platforms, the elapsed times for any given task often fluctuated wildly. For example, one of the multistep I/O intensive benchmarks ran in under 3 minutes while another took over 50 minutes. The compute intensive PROC MIXED ran to completion once in 1 hour 38 minutes and another time in 11 hours and 38 minutes. Moreover, for DATA steps, the problem with handling missing values in assignment statements can lend another source of unpredictability to forecasting task completion time. These rather radical fluctuations in elapsed times makes it difficult to recommend this platform for routine interactive or batch use to students and researchers on our campus who need a working environment where they can predict with reasonable certainty how long a given SAS task is likely to take. We are aware that our CONVEX C3840 may be a bit overburdened at the present time and that other CONVEX installations could experience much less variation in

elapsed times and find SAS 6.09 a valuable addition to their applications software library.

References

Ihnen, Leigh and Isom Crawford, *The SAS System under ConvexOS Release 6.09*, SAS Institute, white paper, 1993.

Sapp, Armistead W. III and Robert Tschudi, *Workstation Benchmarking: Choosing the most Appropriate Platform for Your Application*. Observations, Vol. 2 #1 (Fourth Quarter, 1992), pp. 12-17.

Acknowledgements

Thanks to Leigh Ihnen of SAS Institute for answering our many questions and furnishing us his code for the sort benchmarks. Bob Jones of SAS Institute also provided much needed information which we greatly appreciate. Thanks also to UNC-CH CONVEX keepers Mike Padrick and Larry Mason for their assistance with installing SAS 6.09 beta and production so quickly and helping us with using SAS on that platform. Also, thanks to Tom Rutledge who installed SAS 6.09 on the DEC 5000 system very quickly and helped us with SAS on that platform. Finally, we express our appreciation to Arturo Barrios and Jennifer O'Lear who edited and made this manuscript camera ready.

Disclaimer: The ideas and opinions contained in this article are those of the authors and do not reflect any official position of the University of North Carolina at Chapel Hill or any of its departments.

This paper was presented previously at the 19th annual SAS® Users Group International Conference in Dallas, Texas on Tuesday April 12, 1994.

Appendix A: Hardware Configuration Details

CONVEX C3840

Software:

- SAS 6.09
- OS is ConvexOS Release 10.2

System:

- four processors, clock speeds 66 megahertz
- one GB of real physical memory
- two GB of virtual memory
- 25 GB disk

DecStation 5000 Model 240

Software:

- SAS 6.09 TS027
- Ultrix version 4.3A (Rev. 146)

System:

- 40 megahertz CPU (RISC)

- 32 megabytes (4-Mbit) DRAM
- 64K cache
- 2 GB of disk space

trademarks or trademarks of CONVEX Computer Corp. ® indicates USA registration. Other brand and product names are registered trademarks or trademarks of their respective companies.

IBM 3090 170J

Software:

- SAS 6.08 TS404
- MVS/ESA operating system with JES2 under VM/XA

System:

- 1 processor (est 17 mips) with a vector facility
- 128 megabytes of central memory
- 128 megabytes expanded memory
- 24 parallel channels - 4.5 megabytes each
- 64K cache
- 102.5 GB disk

IBM Value Point 486DX2

Software:

- SAS 6.08 TS405
- System OS/2 2.1

System:

- CPU Intel 486DX2 66 mhz.
- Ram memory 24 MB
- 64 kb cache
- hard Drive 528 MB IDE

Appendix B: Host SAS Options on Benchmark Machines

Benchmarks involving Memory and/or I/O (all benchmarks):

	CONVEX	DEC	IBM	Intel - OS/2
MEMSIZE	512MB	32MB	32MB	is not reported
BUFNO	1	1	5	1
BUFSIZE	0	0	23040	0
SORTSIZE	536MB	16MB	MAX	1MB
SORTPGM	SAS	SAS	BEST	SAS
SYSLEAVE	64K	64K	(0,102400)	15K

SAS is a registered trademark or trademark of SAS Institute, Inc. in the USA and other countries. IBM, MVS/TSO, OS/2 and VM/CMS are registered trademarks or trademarks of International Business Machines. CONVEX and ConvexOS are registered