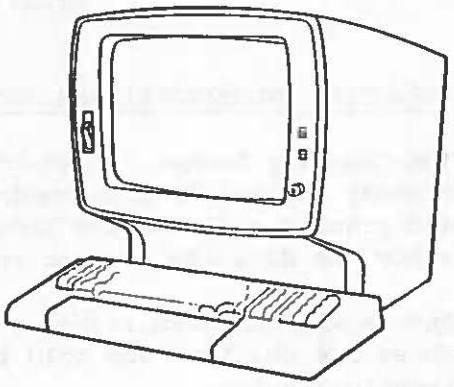
1982-05-28

Computer Center News Letter / May 26, 1982

Computer Center News Letter



NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA

May 26, 1982
Volume 14, No. 4

TABLE OF CONTENTS

* * *

SOME PUBLIC TERMINALS MOVED

Substantial progress has been made with the cabling of the other academic buildings. Public Works Department has almost completed work in Spanagel and Root Halls. We have moved some terminals from Ingersoll Hall, Rooms 148 and 158, to the following work stations:

		Terminals	
Building	Room	IBM 3278	IBM 3277 Graphics
Spanagel	311	9	1
Root	222	6	1

These are public terminals available to any users around the clock. We are following right on the heels of Public Works. As soon as they run the cables we will install terminals in the remote sites.

DEPARTING STUDENTS PLEASE NOTE

The Computer Center is not on all official checkout sheets, but we would greatly appreciate your coming by the User Registration Office, In-147, and signing out. Please have your userid number ready and be prepared to state the date you want to relinquish your disk space.

Anyone who has been issued a users' manual is asked to return it also. These are the blue and gold binders containing a selection of the Center's technical notes.

All students who have been extended beyond their original graduation dates to finish theses, teach at NPS, etc., are asked to come by and extend the termination date of their private disk allocation. This will prevent problems arising from reassignment of space before the original owner is through with it. Normally, all private disk space is reissued after the graduation date of the previous registered owner. If you have a special need, please tell us and we will be happy to make adjustments.

* * *

STAFF CHANGES

The Center has had two recent retirements. Mr. Milton (Mick) Sholley, who has been a Computer Operator at the Center since the days of the CDC 1604, recently retired from Civil Service. (He had already retired from 20 years service in the U.S. Army). We wish him very many years of happy leisure.

Our second retiree is Thelma (Duffy) Tegtmeier of the Operations staff. She has left us after many years of effective and loyal service to the School. Duffy's many friends are sorry to see her go, but glad she will now have time for her many projects.

Mr. Hans Doelman who has been a senior programmer for the Center since 1967, has transferred to the Naval Environmental Prediction Research Facility. He will be sorely missed here by both users and staff who could always depend on him to find the most recalcitrant bugs.

* * *

MOVING BATCH DATA SETS TO MVS

A new CMS command, GETMVS, is available to help you copy data sets from the transfer disk MVS004 to a CMS disk (your A-disk or a temporary disk you have defined before invoking GETMVS).

GETMVS can

-copy a sequential data set

-copy all the members of a partioned data set, or

-copy selected members of a partitioned data set.

In addition it will display information on the size and format of the data set to be copied, and the space available on your CMS disk(s).

To invoke GETMVS, just type

```
GETMVS
```

NOTE: GETMVS doesn't help you to get your data set to MVS004. That must be done by submitting the necessary MVS job(s) before invoking GETMVS. (See Section 1.4 of Technical Note MVS-01.)

* * *

WATFIV COMMAND ON CMS

Some changes have been made to the WATFIV Command:

1) You can get on-line help on the use of the WATFIV command by typing

```
WATFIV ?
```

2) By typing

```
WATFIV ? CMS
```

you can get a printed copy of a write-up on how to use WATFIV from CMS (the same information as in the CMS file WATFIV LISTING Y).

By typing

```
WATFIV ? IO
```

you can get a printed copy of a write-up on some of the stickier points of doing terminal I/O with WATFIV.

3) The WATFIV command is now an exec. If you intend to invoke it in another exec, issue the command in the form

```
EXEC WATFIV . . . . .
```

Questions, suggestions or complaints about the WATFIV command may be directed to Joanne Bogart, In-105, X2651.

NEW VERSION OF SAS

SAS 79.5 has been installed as the default SAS package. This version replaces the older SAS 79.3A. To invoke SAS 79.5, via the correct cataloged procedure use

```
//   EXEC SAS
```

Anyone who has been accessing SAS 79.5 with procedure SAS795 should switch to the above name.

The basic documentation for SAS 79.5 remains the SAS User's Guide, 1979 Edition. The supplemental procedures are also available. These are described in the publication SAS Supplemental Library User's Guide, 1980 Edition.

SAS 79.5 contains one major bug. PROC CONVERT will not read an SPSS system file. To access an SPSS system file from SAS 79.5 requires a two-step process. First, use SAS 79.3A to read the SPSS system file using PROC CONVERT. In that same job, output that file in the form of an SAS system file. Then, the SAS system file created by SAS 79.3A can be read by a SAS 79.5 job. Until this difficulty with PROC CONVERT is cleared up, SAS 79.3A will be kept on the system. The convert cataloged procedure can be invoked by using

```
//   EXEC SAS793
```

Questions about SAS system files should be directed to Dennis Mar, In-102A, ext. 2672.

* * *

CHOOSING THE RIGHT BLOCK SIZE FOR A PARTITIONED DATA SET

(This article was condensed from one appearing in the Clemson University Newsletter of December 1981.)

For partitioned data sets on MVS full track blocking is not optimal. In a partitioned data set, many blocks are shorter than the block size specified when the data set was allocated. A block which follows one of these short blocks must be written on the next track if there is insufficient space remaining for it on the current track. This is usually the case when full track blocking is used. Full track blocking also introduces complications in moving load libraries across devices with different track lengths.

In general, for partitioned data sets, a good block size is one which fits two or three blocks on a track. A block size of 6160 is a good choice as it is compatible across several different device types: two blocks per track on a 3330, three blocks per track on a 3350, seven blocks per track on a 3380.

FORTRAN PROGRAMMING PITFALLS DURING EXECUTION

(The following article is adapted from one in the Mar/Apr issue of the MIT Academic & Research CS Newsletter by David Burleigh.)

By far the most common and potentially the most dangerous "bugs" are those which slip past the compiler's scrutiny and nestle snugly in the "object code"-- the executable form of a program. These run the gamut from trivial mistakes which are easily found to errors of such subtlety that years may pass before they are detected. In the March Newsletter we discussed errors that can occur in editing and compiling a Fortran program. Once you have corrected these you can run your program. Now you have to worry about execution errors, which may cause your program to fail completely (perhaps because of system limitations), or produce incorrect results. If you're lucky, any mistakes are obvious from the output. It's tough, though, when the output "looks good" but isn't, or when the one case you hadn't considered occurs after you put the program into production!

Execution errors may be broadly divided into two groups: those which provoke error messages from the system, and those which do not. Be glad when the system detects your errors! The error messages may seem obscure, but they can help you pinpoint problems rather quickly, if you use the method discussed below. Finding execution errors which are not detected by the system may require detailed tracing of the flow of execution. We plan to return to that subject in a later Newsletter.

Finding Errors which Cause the "Program Interrupt Condition"

A "program interrupt" occurs when the execution of an instruction results in a violation of a system specification. For example, if an arithmetic operation produces a number too large to be represented in the storage allocated for it, a program interrupt occurs. In that case, you receive an error message similar to this:

```
IH0207I IBCOM--PROGRAM INTERRUPT (P)--OVERFLOW  OLD PSW is FFE4000C8202025C
```

Here "207" is the FORTRAN message number, and "OVERFLOW" is a terse description of the problem. For a full description, consult the section "Program Interrupt Messages" in IBM OS FORTRAN IV (H Extended) Compiler and Library (Mod II) Messages (SC28-6865). The string of characters beginning with "FFE4..." is the hexadecimal representation of the "Program Status Word" (PSW) which will be discussed below. Additional information which may be used to pinpoint the offending statement, follows the error message. For the program which produced the message in the example above, the additional information might look like this for CMS execution:

```
TRACEBACK  ROUTINE CALLED  FROM  ISN  REG. 14   REG. 15   REG. 0    REG. 1
           MULT                  0002  4202010A  00020140  00000000  00020098
           MAIN                        0001419A  00020000  00000000  00000868
ENTRY POINT=  00020000
STANDARD FIXUP TAKEN , EXECUTION CONTINUING
```

Listed under the "ROUTINE" heading are names of routines in reverse order of calling, beginning with the routine in which the error was detected. Of course, the error, though detected within one routine, may have resulted from a problem in another. If the first routine is one which you did not write--a system library routine or a "canned" routine--then you should probably begin looking for the error in the first routine listed that is your own; the system-supplied routines are rarely at fault. Next to the routine name, under the heading "CALLED FROM ISN", is the "Internal Statement Number" (ISN) from which that routine was called. ISNs are given in decimal form. In the example above, the interpretation is: Subprogram MULT was called from ISN 0002 in routine MAIN (the main program). If a subprogram is called from many places in the program, this information can help you determine at which call the error occurred.

Often the reason for the problem becomes obvious when you review the routine pointed to by the traceback information. For the overflow condition, look for an arithmetic operation which could produce a floating-point number greater than about 7.2E75, or for an arithmetic operation on a variable containing character information, or for an expression referring to an uninitialized variable.

If the error was detected in one of your own routines, and if you compiled that routine using FORTGI with the "MAP" compiler option (always a good idea while debugging), then you can use the compiler listing and the traceback information to determine exactly which statement caused the error.

First, write down the rightmost six digits of the PSW from the error message. This hexadecimal number is the address of the instruction which follows the one at which the error occurred. Below that number, write the last six digits of the first number under the heading "REG. 15". This is the address of the beginning of the routine in which the error occurred.

Now subtract (using hexadecimal arithmetic) the second number from the first. The result is the relative address of the instruction pointed to by the PSW. For the above traceback:

02025C -- the last six digits of the PSW

020140 -- the last six digits of the address in Reg. 15

00011C -- the relative address of the instruction following the one which caused the error

Next compare this relative address to the addresses printed in the "STATEMENT LABEL MAP" of the compiler listing for the routine in which the error occurred. For our example, this looks like:

STATEMENT LABEL MAP

LOCATION	STA NUM	LABEL	LOCATION	STA NUM	LABEL	LOCATION	STA NUM	LABEL
00110	1		00011A	3		00011F	4	

Locate the largest address that is less than the relative address you computed. Immediately to the right of that address is the ISN (not the label) of the

offending statement. Now if you look at the compiler listing of the routine itself, you will see a sequence of numbers to the left of the program statements-- these are the ISNs generated by the compiler. Find the offending FORTRAN statement by matching the statement number you located from the statement label map with one of the numbers in this sequence. (Do not confuse ISN's with either the sequence numbers of the lines in the source file or with the labels of FORTRAN statements; they are usually unrelated.)

In the example above we conclude that the error occurred at ISN 3, which happens to correspond to the FORTRAN statement "X = X * 1.25E30". We can see that if X is larger than about 7.E45, an "overflow" results. In the next issue of this Newsletter we shall discuss "standard fixups" and fatal execution errors.

* * *

RECENT ADDITIONS TO THE COMPUTER CENTER LIBRARY

Books

Author	Title
Jensen, Randall W.	1979 Software Engineering
Morris, Michael F.	1982 Computer Performance Evaluation
Thomas, Charles R.	1981 Administrative Information Systems: The 1980 Profile

Reports

No.	Author	Title
1392	Schneidewind, Norman F.	1982 Software Maintenance: Improvement Through Better Development Standards and Documentation
1393	Schneidewind, Norman F.	1981 Stockpoint Logistics Integrated Communications Environment (Splice) Networking Study
1394	Schneidewind, Norman F.	1982 Evaluation of SecNavinst 3560.1 Tactical Digital Systems Documentation Standard for Software Maintenance
1395	Schneidewind, Norman F.	1982 Evaluation of Maintainability Enhancement for TCP/TSP Revision 6.0 Update .20

Proceedings

Author	Title
IFIP	1980 Information Processing 80
ACM	1981 APL 81 Conference Proceedings
ACM	1982 Symposium on Architectural Support for Programming Languages and Operating Systems

EXTERNAL GRAPHICS INTERFACE ADDED TO IODE

Users of the Interactive Ordinary Differential Equations Solver (IODE) are now able to save plotting points so that they can be sent to MVS for Versatec plotting or used with the 3277-based Graphics Attachment software. Points are saved on the user's A-disk and he must supply the program(s) to use them elsewhere. One invokes this feature by issuing

```
iode eg
```

when beginning the solution of a problem. The points for up to three graphs (any number of ordinates) may be saved.

This feature is documented in the IODE Writeup. To obtain a copy issue:

```
iode d
```

* * *

The Newsletter appears semi-quarterly and is written by members of the staff, W. R. Church Computer Center (Code 0141), Naval Postgraduate School, Monterey, California 93940. Requests for further information or suggestions for articles for the Newsletter may be addressed to the User Services Manager, Code 0141 (In-133), ext. 2752 (or ext. 2573 for messages).

The Center provides batch-processing service under OS/VS2 MVS with JES3 and timesharing service under IBM VM/SP CMS. These services are based on an IBM 3033 Attached Processor System with 16 megabytes of storage.

DISTRIBUTION

List 3, plus 1-A5, 300-B2, 3-B3, 1-B13, 3-F3, 1-F4, 1-F5, 1-F6, 1-Code 49, 60-Student Mail Center (Lobby)