The ACSS

Newsletter

University of Minnesota
Twin Cities
May 1986

VMS 4.2 Service Begins

Marisa Riviere

On April 16 the VAX 8600, ACSS's new VX system, became available to our users. This larger, faster VAX permits us to offer new VMS services that were difficult or impossible to offer on the VA, a smaller VMS machine, and will cost users 20 to 30 percent less than the VA.

The ACSS additions to the VMS operating system have been transferred to the VAX 8600. There are, however, some important differences between VMS 4.2 (the operating system on the VX) and VMS 3.6 (the operating system on the VA). Some changes are discussed later in this article. For information on changes in control language commands, see the on-line writeup VMS42 on the VX; type the command

```
$ type sys$writeup:vms42
```

For those VMS users who did not log on to the VX during the test period (March 1 to April 15), you will find that copies of your VA files were transferred to your VX account on March 29. Your passwords on the VAX 8600 are the same as those you were using on the VAX 11/780 at the time of the transfer.

As previously announced in our March *Newsletter*, we discontinued our VMS 3.6 service on the VA a few days after the VX became available.

Not all the software available on the VAX 11/780 was transferred to the VAX 8600. See the March *Newsletter* or the VMS42 writeup on the VX for details.

Logging On

To log on to the VX, you simply type the **vx** code (and press **RETURN**) in response to the ACSS network prompt. The log-in message from the VX system will then be displayed, along with the same prompts for user names and passwords that VA used.

If you are using a telephone and a modem to access the VX, dial 376-9070. Then respond to the network prompt as described above.

You may also have to change your terminal parity; in VMS 4.2 the parity default is even.

Training Software

We plan to offer several on-line training packages on the VX. As we go to press, two packages are available: the Introduction to VMS and the Introduction to EDT. (EDT is the VMS line and full-screen editor). To use either package, first type in the `set term/vt100` command. Then, to use the Introduction to VMS, type

```
$ run vms2cai
```

To use the Introduction to EDT, type

```
$ run edt2cai
```

After each package begins to run, you simply follow the instructions that appear on your screen.

Continued on page 66

Continued from page 65

DCL Compatibility

Most Digital Command Language (DCL) of VMS 3.6 is upwardly compatible with the new version, VMS 4.2. What follows are brief descriptions of some of the more important new and changed features in version 4.2. The changes and additions are not described in detail, but are simply mentioned to point users to potential problems and to improvements. If any commands appear especially troublesome (or helpful), you can read more about them in the VMS 4.2 HELP library or in the appropriate VMS reference manual. Copies of VMS manuals are available in the ACSS Reference Room in 128A Lind Hall. To purchase VMS manuals, order them at the Electronics Desk in the Minnesota Book Center in Williamson Hall.

Our new VX system permits us to offer new VMS services at a cost 20 to 30 percent less than the VA.

The **output** format of *many* commands has changed. In most cases this was done to make output more readable. An unfortunate side effect is that old programs that processed output in VMS 3.6 may need to be modified.

A change to new versions of **compilers** (FORTRAN 4.3, Pascal 3.0, and COBOL 3.2) may require some programs to be compiled and linked again.

Passwords in VMS 4.2 must have at least 8 characters. If your password is now less than 8 characters it will remain as such until the first time you attempt to change it. At that time the system will require you to provide a longer password.

In VMS 4.2 the **parity** default is even parity; the default for VMS 3.6 was no parity. You may need to change the parity setting on your terminal. Most users should not notice any difference; however, Kermit users and people who use micros to up/down load files may have to change a software setting in the file transfer program.

UIC's (**User Identification Codes**, e.g. [011,212]) can now be specified in alpha-numeric form in addition to numeric form. The most obvious change users will see is in the output from a `directory/owner` command.

File names and file types may now contain up to 39 characters each. Programs that access data files may need larger text buffers for holding the data file name.

Use of **logical names** usage has been enhanced to include: user definable logical name tables; a job logical name table in addition to the process, group, and system logical name tables; logical name directory tables; and search lists to equate multiple equivalence strings with a logical name. For information on ACSS logical names see the section in the VMS42 writeup on "New Logical Names."

A **file archiving** system will be implemented on the 8600. At the end of each month files that have not been accessed for three months will be archived onto tape and removed from the packs.

The **software** on UCCLIB on the VAX 780 will be reviewed and divided on different directories on the VAX 8600. ACSSLib will contain ACSS-supported software; utilities will contain software that ACSS does not officially support but which users may find useful; VMSInfo will contain informational notes. These directories will be accessed with the logical names ACSS$Lib, ACSS$Util and ACSS$Info respectively. See the section entitled "New Logical Names" in the VMS42 writeup for more information.

The output format of many commands has changed in VMS 4.2.

The **writeup directory** will remain much the same; however, the logical name pointing to it will eventually change from `sys$writeup` to `acss$writeup`.

For More Information

Space does not permit us to give a full description of our new VMS services in this article; watch future issues of this *Newsletter* for more information. Meanwhile, see the VX writeup VMS42 for information on these topics:

- Changes to system security
- Significantly changed VMS commands
- Slightly changed commands
- New commands
- New logical names
- Hardware

You can also get more information with the VX on-line documentation command MOREHELP. (Type **`morehelp`**.) MOREHELP now contains information on VX logical names, new and planned software, training packages, and hardware.

Seymour Cray's Machines (Part 2)

Lawrence Liddiard

In last month's column I began an historical examination of the seven computers designed by Seymour Cray: the 1604, 160, 6600, and 7600 systems of Control Data Corporation (CDC) and the CRAY-1, -2, and -3 of Cray Research Incorporated (CRI). These machines have several things in common: They were all produced by Minnesota and Wisconsin firms, and all (except the CDC 7600) have been part of the computer configurations at the University of Minnesota (or will be in the near future).

Floating Point Arithmetic: Special Values

One of the most difficult tasks in floating point design is allowing the user to recover from arithmetic errors. Cray's CDC 6600 machine introduced infinite (N/0) and indefinite (0/0) values that correctly propagated and produced an error indication when they were used as operands in a floating point functional unit. These values have proven very useful in presetting program values to give error indications when variables are not initialized. Since the error indication is not given until the abnormal value is used, an infinite operand produced by squaring 10 to the 200th on the 6600 would not be detected until sometimes much later in the program. The 7600 was changed to provide an error indication when the infinite (indefinite) result was produced, but, as in the 6600, the mantissa of the result was set to zero, making meaningful recovery difficult.

The IEEE standard requires that the floating point unit be able to initiate an immediate trap for an error such as overflow, so that the user can recover. Note how difficult such an action would be (i.e., how much hardware would be required) on a vector pipeline arithmetic unit that is producing one result per clock cycle. The CRAY-1 floating point exponent was designed with two leading sign bits, thus allowing a squared 10 to the 2000 to have a correct mantissa and exponent representation as 10 to 4000, with an error indication after the vector unit is finished. Any overflow, underflow, or illegal operand can easily be found after the vector operation by testing which result values have the double leading exponent sign bits changed to a single bit.[1]

Floating Point Arithmetic: Speed vs. Accuracy

In the implementation of fast vector pipeline functional units, it is desirable to produce a result every clock cycle. The CDC 7600 adder unit could accept a new pair of operands every cycle, the multiplier unit every other cycle, and the divide unit every 18 cycles. In the CRAY-1 the adder unit was changed to discard all bits shifted out of the upper 48 bits in the aligning process, to add or subtract the operands, and then normalize the result. (Numerical analysts would rather these single guard and shifted bits be saved when normalization will bring them back into the upper 48.) To achieve a new result every clock period, the multiplier unit was changed to use a truncated multiply pyramid with a round bit entered at 2**(-49) that may round up the correct answer by 2**(-48). The original algorithm sometimes caused unsymmetric rounding so that A*B was not equal to B*A. This no-no has since been corrected in all CRAY-1 computers.

To achieve better than 29 and 18 clock cycle divide times on Seymour Cray's 6600 and 7600 machines, the exact division was replaced by a reciprocal approximation method on the CRAY. Thus to compute C=A/B on a CRAY: an initial 1/B reciprocal approximation is formed, followed by a reciprocal iteration and numerator, A, times the initial approximation done in parallel, and completed by multiplying the results at of the two parallel calculations to arrive at C. This Newton's method iteration can be done in 3 clock cycles in the vector units, which is 6 times faster then the exact divide on the 7600. Note that this method approaches the answer from below, so, for example, the integer 9 divided by 3 may produce the result 2.9999 99999 99999. This is a good floating point approximation, but a number of programs expect integer quotient results for exact integers.

The specialization of the vector units has hurt the generality of the arithmetic in the evolution of Cray's systems; thus DOUBLE PRECISION operations on the CRAY are 10 times slower than REAL compared with 4 times slower on the 7600 and 6600. In addition the numeric qualities of the resultants are hard to explain and numerical analysts are not able to prove nice properties for such arithmetic.

Cooling and Design

Besides the logical design, the

Continued on page 68

Continued from page 67

CRAY machines are the "right thing" at their time in history. The 1604 box was small compared to others of its generation; so a wide three-section console with octal numeric read-outs in the center was accompanied by the system typewriter to the left and the paper tape reader on the right. This made you feel you had received your money's worth in those days of "ironware." The 6600, in the shape of a large cross, used stacked ("cord wood") modules that were cooled with freon.

The 7600 was Cray's first system that had the rough shape of a **C** (from a bird's eye view), matching the designer's last initial. Bench seating over the power supplies was provided around the exterior of the C-shaped CRAY-1, and the modules were attached to heavy copper substrates that transferred the chip heat to the columnar freon cooling. The CRAY-1 design allowed the system to be turned on and off without component failure or transients, an action that was not prudent on the 6600 or 7600. As the cycle time gets smaller, the system must have shorter connections, and the waist-high CRAY-2, cooled with inert fluid, has 4 processors and 256 million 64-bit words in the shape of a compact and familiar **C**. Cray's computer designs are thus recognizable by their small "foot prints" in required floor space.

In a speech last year, Seymour Cray said the CRAY-3 processors would occupy *at most* a cubic foot of space to achieve minimal interconnection delay and the cooling inert fluid would have to pass the modules at ten times the CRAY-2 rate. The actual shape and design of the CRAY-3 should be interesting when it is delivered in 1987.

Assessment

The CDC 1604 was Fifties-traditional with a single accumulator, buffered I/O, and about 2/3 the speed and 4/3 the word length of its IBM 7090 rival. The CDC 160, rumored to have been designed over a weekend by Cray, was CDC's first $60,000 *desk* (not *desktop*) computer that became the prototype I/O processor for the peripheral processors surrounding the CDC 6600 and 7600. The CDC 6600 was the first RISC machine for scientific computation and its final FORTRAN compilers made it about 25 times the speed of the 1604. Our original benchmarks gave a factor of 13 over the 1604, which shows that software can improve with age. The 7600 was the improved 6600 by a factor of 4 or 5 that became the standard weather and atomic energy computation machine for a decade with its segmented parallel arithmetic units. Although many of the early 7600s had a mean time to failure of four hours, the computations you could do in just two hours equalled a full shift of 6600 time. The 7600 returned to buffered I/O by the central processor and required fewer of the small I/O processors.

The processor on the CRAY-1 did all of its I/O without the help of small processors and had long mean time to failure rates after single error correction, double error detection (SECDED) was added to its memory modules. The reported average improvement of the CRAY-1 over the 7600 for the latest CRAY FORTRAN compiler is about 2.5 for scalar and 5 for vectorizable problems. (These were originally 2 and 4 respectively.)

Seymour Cray has been striving for improvement by a factor of 10 from the CRAY-1 to the CRAY-2 and from the CRAY-2 to the CRAY-3. To achieve a factor of 10 for the CRAY-2 the cycle time was reduced from 12.5 nanoseconds in the CRAY-1 to 4.1 in the CRAY-2 giving a factor of about 3 for vectorizable problems. This rate, combined with 4 processors addressing 256 times the memory of the CRAY-1, provides an expected 10 speedup for vector problems that can do parallel processing. The scalar side of computation is estimated to achieve about 1.5 times the CRAY-1 speed due to the relative slowness of the new memory and the difficulty of using local memory. Many problems do not currently achieve these factors, but will be able to when software improves and better use is made of the 16,384 words of local memory.

For the CRAY-3 the estimated speedup factors over the CRAY-2 will be: 4 due to 16 processors rather than 4, 2 to 3 due to cycle time improvement, and 1.5 due to scalar improvements for a total of 12 to 18 in scalar and 8 to 12 in vector. Thus from Seymour Cray's 1604 to the CRAY-2 the lower order of execution speed improvement is about 25*4*2.5*6=1500. The memory word capacity ratio for those systems is 2**28/2**15=2**13=8192. In 1987 the CRAY-3 should multiply both of these factors by 10, with the proviso that parallel processing by 16 processors is easily done by the FORTRAN compiler.

Thus our first director, Dr. Martin Stein, recognized the value of Seymour Cray's products by installing the CDC 1604 and CDC 6600 as our first ventures in reliable computing hardware. I will allow a future writer to report on the next 25 years.

The topic for my column next month will be parallel processing: If 16 processors do it nicely, why not 4096 to really get some speed?

Note

1 L. A. Liddiard, "Required Scientific Floating Point Arithmetic," *4th Symposium on Computer Arithmetic Proceedings*, IEEE Computer Society, No. 78 CH1412-6C (October 1978).

An Introduction to CDCNET

Paul Tranby and Roger Gulbranson

The Control Data Distributed Communications Network (CDCNET) is a family of new products that ACSS is using to replace our current PDP/11 line of front-end communications processors. CDCNET connects terminals and other computer-related resources to Control Data Corporation (CDC) mainframes. ACSS is now in the process of installing CDCNET to connect campus terminals to our three CYBER computers.

Our previous CYBER front-end hardware, the PDP/11s, which have served faithfully for many years, are now outdated and do not provide the line speed and other capabilities that are required for doing such things as full-screen editing and graphics. (The PDP/11s have problems supporting line speeds any higher than 1200 baud, giving a much lower effective rate at higher settings.)

CDCNET will support line speeds from 50 to 38400 baud for normal interactive terminal traffic. With the help of our Tellabs network and the new phone system, we plan to connect all hardwired CRT-type terminals at 9600 baud and hardcopy devices at their maximum speeds. The time you spend logged into NOS should be more productive, and you will be able to make full use of new software like full-screen editors and procedure files. CDCNET also supports *type-ahead.* (Type-ahead allows you to type in many messages (NOS commands, input lines, etc.) in succession without needing to wait for a response from each one before you continue with the next.)

Only terminals in the public labs that access the ME and MD CYBER mainframes are connected with CDCNET at the present time. CA will be connected sometime this summer. When the new phone system installation is complete, all NOS access (including dial-up) will be through CDCNET.

This is the first in a series of *Newsletter* articles that will describe this new communications front-end to our CYBERs. This series of articles will explain in some detail the new configuration, the differences in terminal characteristics, setting up your terminal, and some examples of using the features that CDCNET now gives you.

CDCNET at ACSS

Here at the University we currently have CDCNET access to the ME and MD CDC mainframes. To connect to MD or ME via CDCNET, enter the MD or ME code in response to the ACSS-net prompt:

```
    ACSS-net (NOS,ME,MD,VX) ?
```

In this example, when you type in MD, a message similar to the following will be displayed on your terminal:

```
Copyright Control Data Corporation, 1985.                }
                                                         }
DI System Name is 080025300162, TDI1                     }      Note 1
Terminal Name is 000000, $CONSOLE_300162_000000          }
You may enter CDCNET commands.                           }

Connection TO_MD created.                                }      Note 2
Procedure SETUPMD completed.                             }
```

Continued on page 70

Continued from page 69

```
WELCOME TO THE NOS SOFTWARE SYSTEM.                   }
COPYRIGHT CONTROL DATA  1978, 1985.                   }
                                                      }      Note 3
86/02/18.  12.19.22.  T070601                         }
MERITSS/MD  (02/12-AN)        NOS 2.4.3 - 647/642     }
FAMILY:                                               }
```

Notes:

1) This is the CDCNET header that identifies the logical terminal name and logical Device Interface (DI) system name that you are connected with. If you are having problems, it will be helpful to the support staff if you have this information; otherwise you can ignore it.

2) This message shows that CDCNET has executed a procedure of CDCNET commands that configures your connection with our site-defined defaults and then creates a connection for you between CDCNET and NOS. These messages are sometimes misplaced after the FAMILY: prompt. This has been fixed in the next release of CDCNET from CDC. In fact, these messages will disappear in that release.

3) This is the normal NOS log-in message. Enter your user name and password as you normally do. If you are prompted with the following prompt:

```
    APPLICATION:
```

type **IAF** in response. IAF stands for Interactive Facility, the software subsystem you use when logged on to the CYBERs.

To disconnect from the CYBERs you still type the **BYE** command. You will see the ACSS-net prompt about four seconds after the last IAF messages are displayed.

Getting to Know Your Terminal Attributes

Every terminal connected to a computer has many characteristics that define how the terminal is going to work, such as CRT versus hard copy; how many characters can be displayed on a line; and how many lines can be displayed on a screen. Other characteristics refer to the data on the communication line between your terminal and the computer, such as line speed, number of bits of data in each character, and parity. Characteristics like echoplex, line folding, and page waiting are features that the host can provide to help customize your connection. All of these characteristics are referred to as "attributes" by CDCNET.

CDCNET, like any computer or communications processor, must know about or at least make certain assumptions about these terminal attributes in order to communicate properly with your terminal.

Besides being able to provide you with attributes like character echo and page waiting, CDCNET also allows you to do such things as the following:

- Define which key forwards the current input line
- Define which key will delete the last character entered
- Define which key will delete the current input line
- Define which key is the ATTENTION key
- Define what you want the BREAK key to do—interrupt or abort
- Define what you want the ATTENTION key to do—interrupt or abort
- Define the number of columns and lines displayed on your terminal

The following list describes how the defaults for these and other attributes are currently set:

advance the current input line	: `(CR)`
delete the last character	: `(control-h)` or `(BACKSPACE)`
delete the current input line	: `(control-u) (CR)` or `(BREAK)`
discard input within CDCNET	: `%0 (CR)`
interrupt the current job	: `%1 (CR)` or `(BREAK)`
terminate the current job–ATTENTION signal	: `%2 (CR)` or `(control-y)`
stop output temporarily	: `(control-s)`
restart output	: `(control-q)`
ENQUIRE about the current job	: `%e (CR)`
show status of the current job	: `%s (CR)`
abort transparent input mode	: `(BREAK) (control-a) (control-x)`
force logout	: `(BREAK) (control-a) (control-t)`
force ACSS-net disconnect	: `(BREAK)` QQQ

The control commands require that you keep the CONTROL key depressed while you strike a letter key. The `(CR)` is used to indicate whatever key on your keyboard sends the carriage return character (0D hex). This may be labeled RETURN, NEXT, or ENTER and also can be sent by entering `control-m`.

When you hit the `BREAK` key or enter `%1 (CR)` to interrupt the current job, the system displays the message `*INTERRUPTED*` at your terminal and stops. In response to this message, a carriage return will cause the job to continue. If the job was generating output, some of the output will have been discarded. A response of any character other than the carriage return will abort the job and `*TERMINATED*` will be displayed at your terminal.

The `BREAK` key will also discard the current input line but will not cause the `Input Canceled` message to be displayed as it is for the `control-u (CR)` sequence. Note that you do need the carriage return after the `control-u` to delete the current input line.

Changing Your Terminal Attributes

With CDCNET there is a lot more flexibility in what you can do with your terminals. Along with this flexibility comes a good deal more complexity, however. All of the terminal attributes mentioned previously and many others can be changed at any time by the user or by software running in the host.

The user can change these attributes from either the host side with the NOS TRMDEF command or from the terminal side with commands sent directly to CDCNET itself. The CDCNET commands to do this will be described in one of the subsequent articles on CDCNET. These commands are fully explained in the *CDCNET Access Guide* and the *CDCNET Terminal Interface Usage* manual, both of which are on reserve in the ACSS Reference Room in 128A Lind Hall.

TRMDEF is described in the on-line document EXPLAIN, M=COMMAND and in the *NOS V2 Reference Set,* Volume 3 (System Commands). One thing to be aware of when using TRMDEF is that the current version of TRMDEF does not directly support all of the CDCNET attributes. It was written to support the CDCNET predecessor, the CDC 2550, and its attributes. At present, the TRMDEF parameters are mapped into CDCNET attributes by CDCNET. The next release of NOS will contain a TRMDEF which is fully compatible with CDCNET. This release will be out this summer. Also note that you cannot enter the TIP commands as described in the *NOS Reference Set.* TIP commands are to the CDC 2550 what CDCNET commands are to CDCNET.

Changing terminal attributes is largely dependent on what type of terminal you are using. There are literally hundreds of terminals manufactured today, all with their own peculiarities. Some manufacturers, however, try to follow accepted industry standards that make terminals easier to set up and use. For example, many terminal manufacturers build a VT100-compatible terminal. (The VT100 is a terminal manufactured by Digital Equipment Corporation [DEC], and has become a standard throughout the

Continued on page 72

Continued from page 71

world.) Many of the terminals on campus are either VT100s (or VT100 compatible) or Zenith Z19/Z29s.

Control Data has taken many of the terminals available and combined them into groups with similar characteristics. CDC calls these groups "terminal types" or "terminal classes." For example, all VT100 and VT100-compatible terminals belong to terminal class seven (tc=7), most hardcopy terminals belong to terminal class one (tc=1), most "dumb" CRT-type terminals are in terminal class 2 (tc=2), and CDC 721 terminals belong to terminal class three (tc=3). These classifications are discussed more completely in the NOS manuals. If, for example, you want to define your terminal as a VT100 with echoplex on, page width of 80, page length of 24, and page waiting on, you can enter the following command:

```
TRMDEF,TC=7,EP=Y,PW=80,PL=24,PG=Y.
```

A procedure has been made available on user name LIBRARY, called TERMTYP, which can help you configure your terminal to the appropriate terminal class. To use it, just acquire the file from LIBRARY (type `GET,TERMTYP,UN=LIBRARY`) and enter `BEGIN,,TERMTYP`. The procedure will then prompt you for the type of terminal you are using.

You could also set up a log-in procedure, designated by the UPROC command, to configure your terminal. For example, type `UPROC,MYFILE` and create and save a file called MYFILE that contains the following:

```
.PROC,MYFILE.
GET,TERMTYP/UN=LIBRARY.
BEGIN,,TERMTYP.
REVERT(NOLIST)
```

If you usually use the same type of terminal, it might be to your advantage to have ***myfile*** look like the following:

```
.PROC,MYFILE.
GET,TERMTYP/UN=LIBRARY.
BEGIN,,TERMTYP,VT100.
SCREEN,VT100.
REVERT(NOLIST)
```

This procedure will not prompt you for input because TERMTYP has its required parameter. It also places you in screen mode, which is handy if you use the Full Screen Editor (FSE), EXPLAIN, or interactive procedures.

Things to Try

Although this article does not contain the CDCNET commands, you can use many of its capabilities without them. For example, if you have a VT100-compatible terminal, enter the following commands, if necessary replacing *VT100* with whatever terminal you have (see the SCREEN command in the *NOS Reference Set* for details).

```
GET,TERMTYP/UN=LIBRARY.
BEGIN,,TERMTYP,VT100.
SCREEN,VT100.
GET,myfile.
FSE,myfile.
```

(See WRITEUP,FSE, the FSE *Brief*, and the CDC NOS manuals for more information on FSE.) Now create the following file with any editor.

```
.PROC,TEST*I,P1,P2,P3,P4.
NOTE./  /  P1    P2    P3    P4   /
REVERT(NOLIST)
```

Now enter the following and notice the differences.

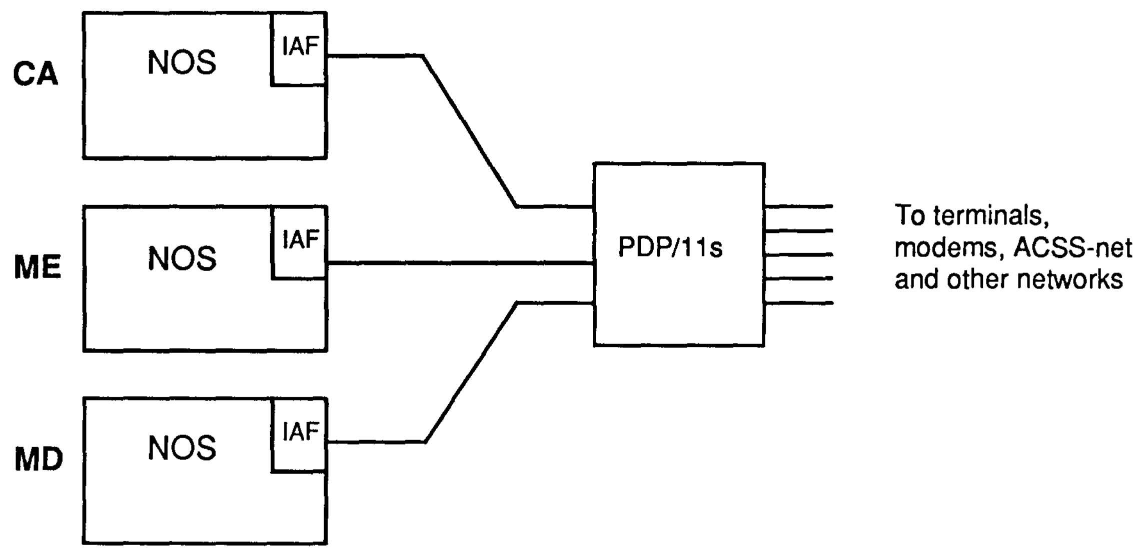
```
LINE.                   {NOS LINE command}
BEGIN,,TEST.            {calls procedure}
aa                      {enter procedure parameter values}
bb                          .
cc                          .
dd                          .
SCREEN.                 {NOS SCREEN command}
BEGIN,,TEST.
```

Also try EXPLAIN in SCREEN mode and experiment with the `TRMDEF,PG=Y` and `TRMDEF,PG=N` commands, which control page waiting.

The Old and the New

ACSS has for many years used DEC PDP/11s as communication front-ends to our CDC CYBER mainframes. The software used in the PDP/11 was developed here several years ago to provide access for the large number of users that the University supported. Figure 1 shows the configuration of the PDP/11s.

Continued on page 74



IAF is the software subsystem that you are conversing with when you are logged in to NOS. It accepts your commands (CATLIST, SAVE, etc.) and submits them to NOS for execution.

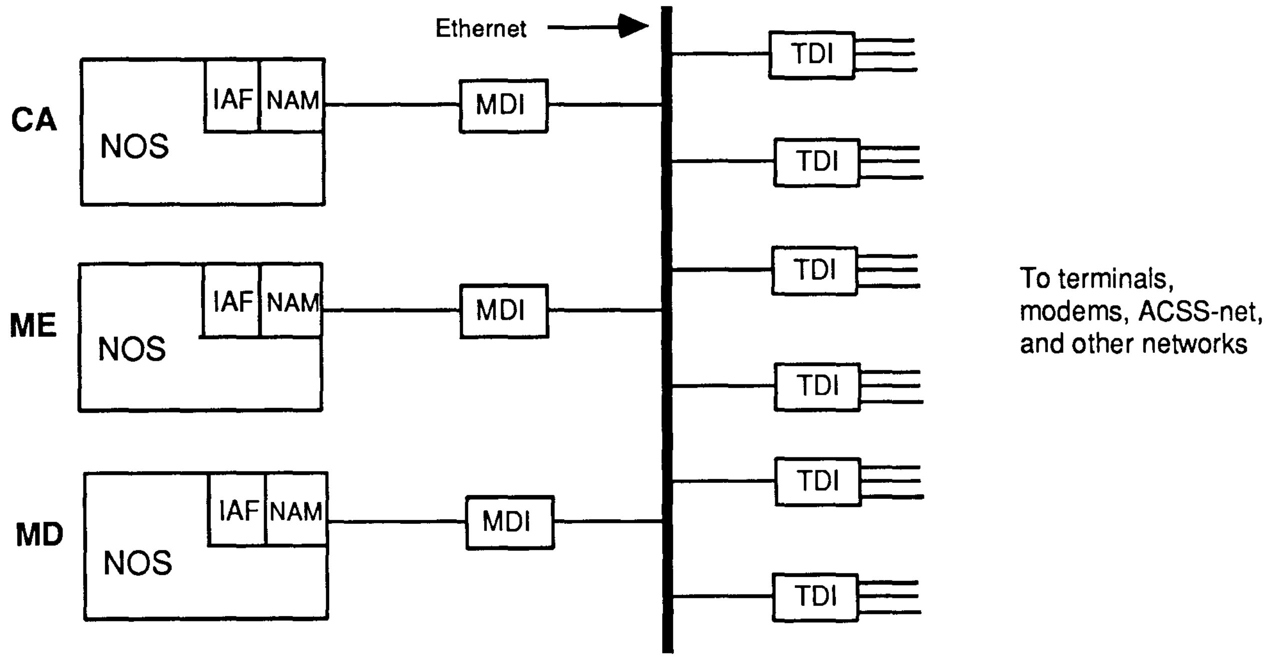
Figure 1. PDP/11 Configuration.

Continued from page 73

The new CDCNET was designed under a distributed communications philosophy that allows us to "plug together" a number of small, powerful boxes in any configuration that we think will best fulfill our needs. It also allows us to switch to the new system in smaller increments and to expand easily.

Figure 2 shows CDCNET access to all three CYBER mainframes. However, we currently only have access to MD and ME.

In CDCNET terminology, each small box in Figure 2 is called a DI—for Device Interface. What each DI actually does depends on the hardware boards that are plugged into it, and this can be changed by simply plugging in different boards.



MDI refers to a CDCNET box configured to be a Mainframe Device Interface—it connects a mainframe to the rest of the network. TDI refers to a CDCNET box configured to be a Terminal Device Interface—it connects terminals to the rest of the network. NAM is a software subsystem that passes terminal data between CDCNET and the NAM application subsystems such as IAF and RBF. NAM stands for Network Access Method.

Figure 2. CDCNET Configuration.

Known Problems and Restrictions

Often when you log in, the `Procedure SETUPME completed` message (or the `SETUPMD completed` message) appears after the `FAMILY:` prompt. This problem has been fixed in the next release of the CDCNET software, which we will receive this summer.

Auto mode (invoked by the NOS AUTO command) works differently through CDCNET. You cannot backspace over the line number and change it, which makes it hard to enter FORTRAN comments or continuations because they require a "+" or "*" immediately after the line number. To enter a character immediately after the line number, type `control-u (CR)` after the line number prompt to delete it, then retype the line number followed by the "+" or "*". You can backspace over the line number if you invoke the store_backspace_characters connection attribute with the CDCNET command **`%chaca sbc=on`**. This sends backspace characters to the host where the application program then figures out what to do with them. You must then enter the **`%chaca sbc=off`** CDCNET command when you exit auto mode, however. To exit auto mode you can enter `BREAK` or `control-y`, or `control-u (CR)` followed by another carriage return. Because of these difficulties, we recommend that you do not use auto mode.

The BREAK key on some of the Raster Technology terminals in the public labs appears to be incompatible with CDCNET and our ACSS-net in general. This is a problem with the firmware in the Raster Technology terminals, which we hope to replace soon.

All terminals should be configured with the NEWLINE setup option disabled.

All terminals should be set up to run in full-duplex mode (i.e., the terminal should not display your input itself). Operating in half-duplex interferes with security and will also cause problems if you attempt to do screen operations such as using screen-oriented editors.

There is a conflict between the HOME key and the attention character `(control-y)` when using FSE from a CDC 721 terminal. To get around this, you can redefine your attention character by entering the following:

```
  %chata ac='x'
```

where the *x* is something meaningful to you like `control-t` for terminate or `control-a` for attention.

<u>*The Classifieds*</u>

For Sale: CPT 8525 word processor and CPT Rotary VIII Printer, like new. Best offer. Call Janelle at 373-0825.

<u>*Documentation News*</u>

New *Guide to Subprogram Libraries*

The latest edition of ACSS's *Guide to Subprogram Libraries* is now available. You can purchase copies through the University bookstores or refer to copies in the ACSS Reference Room, 128A Lind Hall.

FSE on the CYBERs

Space does not permit us to publish the promised article on using the CYBER Full Screen Editor (FSE) this month; the article will appear in our June issue. Meanwhile, you can refer to an on-line document, WRITEUP,FSE, on the CYBERs. There is also a new FSE *Brief* available free in our Reference Room, 128A Lind Hall. Also in the Reference Room is a reference copy of Control Data's *NOS Full Screen Editor User's Guide.*

ACSS Computer Grants for 1986-1987

Lawrence Liddiard

The grant program announced last August for 1985-86 that provided centrally administered computing services for graduate thesis work and for faculty research not supported by granting agencies will be continued during the 1986-1987 fiscal year.

In August 1985 this *Newsletter* reprinted a letter from Acting Vice President for Academic Affairs and Provost V. Rama Murthy that described the grant program. As the start of the new fiscal year for the grant program rapidly approaches, the details from that letter bear repeating.

- The program exists for all central computer systems in Information Systems. These systems include ACSS's CYBER/NOS, VAX/VMS, and UNIX services; the St. Paul Computing Center's IBM service; the Health Sciences CYBER NOS and NOS/VE services; and the Duluth campus's Computing Services.

- An initial payment of $50 (not refundable) per user will establish an account of $1000 for both computing costs (03) and on-line computing-related charges (05). This schedule will apply to computing for graduate student theses as well as to faculty research computing.

- All off-line computing-related charges and computing costs beyond $1000 will be fully charged to the user. Therefore, all users must keep track of their current, accumulated charges by using the ACCSTAT program on ACSS systems or comparable methods at other service centers. The funds for these grants are "real" dollars and thus the grantee is required to be alert to total computer usage.

- Grant additions will be available under the program presently managed by the Computer Grants Committee. Lawrence Liddiard (ACSS, 5 Wulling Hall, 86 Pleasant Street SE, Minneapolis, MN 55455) currently is chairman of the committee. In the past, problems have occurred for individuals who have overrun their grant amount: **an overrun cannot be handled retroactively** by the Computer Grants Committee.

Vice President V. Rama Murthy suggested that departmental budget funds might be used for the initial charges of faculty who have unfunded research needs. One purpose of the grant program is to reach new faculty members and provide computing resources for their research and background requirements. As Information Systems' centers continue lowering their rates, these grants will provide a considerable amount of computing at low cost and go far in meeting the computing needs of many faculty and graduate students.

At the University of Minnesota, grants and contracts for research projects are expected to include funds for research computing and to pay for the full costs of such computing whenever possible. This practice should be followed whether the computing is done by faculty members or by graduate students. If this practice is not followed, available University funds may not permit continuation of this policy for unfunded research. The Computer Grants Committee will not favorably evaluate requests for unfunded research when sincere efforts to acquire funding have not been pursued through granting agencies. Your cooperation is needed in implementing this policy which, if not followed, will lead to inequities for those who do obtain funds for research computing.

Logic Programming and PROLOG

Ron Zacharski

PROLOG is becoming an increasingly popular language in artificial intelligence programming. PROLOG is radically different in its structure from traditional programming languages. In FORTRAN, Pascal, C, COBOL and most programming languages, a programmer must precisely describe **how** to compute a result and not **what** it is that must be computed. The programmer writes a sequence of steps that the computer follows to accomplish a task. However, most of us first precisely specify what a problem is before we can determine the correct sequence of steps that will lead to its solution. It is (at least) a two-part process:

1. Problem Specification
2. Actual Programming

In logic programming, the problem specification is the program. The programmer only needs to describe **what** the problem is and the programming language itself takes care of **how** to reach the solution. The advantages are obvious. It takes less time to write programs and, more importantly, we can mathematically prove that programs meet their specifications.* If we use the formal language of predicate logic to express our specifications, we could directly execute them with a powerful mechanical theorem prover. Many researchers are currently working on the development of such a theorem prover. Meanwhile we can use a sublanguage (for example, Horn clauses) for which a theorem prover does exist. PROLOG is perhaps the most well-known of these sublanguage theorem provers.

Logic programming is based upon the interpretation of rules of the form

A if B and C and ...

where A, B, and C are relations.

For example,

Rule 1 *x* is a bachelor if *x* is an unmarried man.

This is interpreted as

to show *x* is a bachelor
  show *x* is an unmarried man

Consider the rule

Rule 2 Ross is an unmarried man.

This is interpreted as

to show Ross is an unmarried man
  do nothing.

To find a bachelor we pose the query

Find *x* where *x* is a bachelor.

The program reasons backwards and by use of Rule 1 reduces the problem to the subproblem of showing

Continued on page 78

Continued from page 77

Find *x* where *x* is an unmarried man.

This is solved directly by Rule 2 and the solution

x = Ross

is given.

We use the declarative statements of predicate logic to create our specification and it is the system's interpretation of these statements that leads to a program. The preceding example is admittedly a simple one. However, PROLOG can be used for a variety of applications of symbolic computation including data bases, circuit design, natural language processing, expert systems, symbolic equation solving, and abstract problem solving. The most intriguing thing about PROLOG is that the programmer is attempting to build a logical description of certain phenomena. However, these logical descriptions are not just theoretical devices but also computer programs that can be used in the analysis of the phenomena.

If you would like more information on logic programming, we highly recommend that you review the following:

BYTE, August 1985. Contains several articles on PROLOG.

Clark, K.L. and F.G. McCabe. *micro-PROLOG: Programming in Logic.* Englewood Cliffs, New Jersey: Prentice-Hall, 1984. 401 pages. A lucid introduction.

Clocksin, William and Christopher Mellish. *Programming in PROLOG.* Berlin: Springer-Verlag, 1981. 297 pages. $17.95. Considered the definitive introduction to PROLOG. Not quite as enjoyable to read as Clark and McCabe.

Hoare, C.A.R., and J.C. Shepherdson, eds. *Mathematical Logic and Programming Languages.* Englewood Cliffs, New Jersey: Prentice-Hall, 1984. 184 pages. $29.95. A collection of papers describing current research in logic programming.

We have a public-domain version of PROLOG for the IBM-PC. If you are interested in obtaining a copy, please contact us. We also have LPA PROLOG for the Macintosh. This is an outstanding product that includes an interpreter and an incremental compiler. This PROLOG is almost as good as one you would find on a LISP machine. In a future issue of the ACSS *Newsletter,* we will provide an in-depth look at PROLOG versions on mainframe computers.

If you would like to see a demonstration of LPA PROLOG, or if you would like further information about PROLOG, logic programming, or any AI topic, please feel free to contact us.

Ron Zacharski
Tom Rindflesch
Special Projects
124 Shepherd Laboratories
(612) 376-2944

*Pure logic programs can be verified without taking their behavior into account. This would make it possible to use increasingly sophisticated proof procedures to execute logic programs without complicating correctness proofs. Unfortunately, the execution of pure logic programs in PROLOG may lead to infinite loops and other less extreme forms of inefficiency. To avoid such loops, the programmer may need to control the ordering of the rules. Now we must take behavior into account to verify the program and this verification may be more complicated than verification of conventional programs.

The IBM Convertible

Mark McCahill

IBM announced its IBM PC Convertible on April 2. A brief description of the Convertible and other product announcements follows. If you are interested in more detail see the April *Microcomputer Newsletter.* You can also stop by the Micro HelpLine and read the official announcement or talk to a consultant.

Hardware

The PC Convertible is a true portable; it weighs only 12.2 pounds and can run off its battery pack for 6 to 10 hours between recharges. The Convertible comes with 256K memory standard, and you can expand the memory in 128K increments to a maximum of 512K. An optional internal 300/1200-baud modem is available. An optional serial/parallel interface is also available, as well as a 40 character-per-second dot matrix printer. Both the serial/parallel interface and the printer are compact units that fit onto the back of the computer. The internal modem fits *inside* the Convertible's case. The Convertible's screen is a 25-line liquid crystal display (LCD).

The Convertible comes with two 3.5" disk drives (similar to the Macintosh 3.5" drives). The 3.5" disks are double sided and hold 720K; this is quite respectable compared to the 360K capacity of the IBM PC's 5.25" disks. The short-term drawback to the 5.5" disks is that nearly all IBM software is only available on 5.25" disks. If a large number of Convertibles are sold, we expect software vendors to make their programs available on 3.5" disks. At the moment, the supply of software on 3.5" disks is a bit limited.

The Convertible comes with two 3.5" disk drives (similar to the Macintosh 3.5" drives). The 3.5" disks are dou

Software

The Convertible comes with PC-DOS 3.2. This new version of PC-DOS supports the 3.5" disk drive. If you plan to use the 3.5" external disk drive with a PC, XT, or AT, you also will need to purchase PC-DOS 3.2.

In addition to PC-DOS, the Convertible also comes with a shell (the Application Selector) and several programs collectively called SystemApps.

Other IBM Announcements

IBM-XT: At the same time IBM announced the PC-Convertible, they also announced a drop in the IBM-XT retail price. As of press time we do not yet know what the new price for the XT will be at the Book Center.

The XT is now available with a 20 MByte hard disk drive (formerly, the only hard disk drive available with the XT was a 10 MByte drive). The 5.25" floppy disk drives are now *half-height* drives, so it is possible to fit two floppy drives into the space that was formerly occupied by one drive. An internal half-height 3.5" diskette drive is available.

IBM-AT: As in the case of the IBM-XT, IBM announced retail price reductions for the IBM-AT and changed the machine a little. In the case of the AT, the changes include increased speed and a new keyboard.

More Microcomputer HelpLine Hours

The Microcomputer HelpLine hours have been extended. Effective *now* you can call 376-4276 or visit 125 Shepherd Labs during the following hours:

Wednesday & Thursday	9:00 a.m. to 4:00 p.m.
Monday, Tuesday, Friday	9:00 a.m. to noon and 1:30 to 4:00 p.m.

These extended hours should reduce the waiting time when you call or stop by.

MEWS: A Tool for Mac Programmers

Mark McCahill

The Microcomputer Systems Group has developed a set of software tools for TML Pascal programmers who want to develop professional-looking Macintosh applications. *MEWS* (for Menu, Event, and Window System) is designed to speed the development of Macintosh software and eliminate the need of rewriting code that is required of nearly all Macintosh programs. Included on the MEWS disk are a manual (as a MacWrite document) and several source code example programs.

An early version of MEWS, beta-site test Version 1.0, was distributed at the Mac Users Group meeting in February. If you have the early copy, we suggest that you replace it with the updated Version 2.1. The bugs that existed in the early version have been fixed and the manual has been considerably imroved.

The Microsystems Group highly recommends TML Pascal and MEWS to anyone who plans to develop Mac programs. (We have moved all development off the Lisa Development System and onto TML Pascal.) MEWS will be in the public domain and is available to anyone who visits the Micro HelpLine with a blank, initialized disk and makes a copy of it.

NOSNEWS for CYBER Users

CYBER users should note that the MAIL topic NOSNEWS is now open to all MAIL users who wish to read information on user-visible changes to the NOS system or to any of its products or utilities, including CALLPRG products and/or WRITEUPs. All MAIL users may **read** NOSNEWS. Only selected ACSS staff may write to the topic. See WRITEUP,MAIL for details on MAIL topics.

Project Assist Workshops

Kit Eastman, Project Assist

Project Assist, a campus-based group that was created to assist University of Minnesota faculty members who want to use computers for instructional purposes, is sponsoring two workshops during the month of May:

Design Considerations for Computer-Based Instruction
Thursday, May 22; 9:00 a.m. to 12:00 noon

An Instructional Design Model
Thursday, May 29; 9:00 a.m. to 12:00 noon

The workshops were designed to be taken consecutively, and Project Assist recommends that you enroll in both workshops in order to get a thorough background on the principles of computer-based instruction.

Workshops are available and free of charge to faculty, but space is limited. If you wish more information, call the Project Assist office at 373-2660. If you wish to register for a workshop, please send a memo listing the workshops in which you wish to enroll and your name, campus address, and phone number to the Project Assist office at 148 Peik Hall. Project Assist staff will contact you concerning workshop meeting places.

Computing for the Future: The 1986 Users' Meeting

Steven Brehe

On March 20, administrators and staff from Academic Computing Services and Systems met with about 45 computer users at the ACSS Users' Meeting in Coffman Union. For two hours, ACSS group managers presented their groups' plans for new and expanded services, and University students, faculty, and staff, and off-campus users spoke with ACSS staff about their computing projects.

Peter Oberg, manager of User Services, opened the meeting, welcoming users, explaining the agenda, and introducing the acting director of ACSS, Michael Skow.

Mr. Skow established the context for the presentations to follow. First, he explained how ACSS fits into the **administrative structure** of computing at the University: ACSS is a department of Information Systems, which in turn is part of Academic Affairs. The Institute of Technology, also part of Academic Affairs, administers the Supercomputer Institute. The Institute provides University researchers with supercomputing services obtained from Research Equipment Incorporated, a private corporation affiliated with the University, that operates supercomputers for research and business use.

Mr. Skow then outlined the two-part organization of ACSS—Services and Systems. Services contains these three groups: Distributed Services, Central Services, and Engineering Services; Systems, these three: Central Systems, Network Systems, and Operations. (For descriptions of each group's responsibilities, see the January *Newsletter*.)

The **mission** of ACSS, Mr. Skow said, is to provide computing support for academic programs of the Twin Cities campus, consistent with the goals and resources of the University. In pursuit of this mission, ACSS is introducing new services and expanding others, offering, for example research grants on the new VAX 8600, VAX instructional computing, support for artificial intelligence and expert systems programming, electronic mail, and the national BITNET network. To perform its mission better, ACSS is also evaluating its services in several ways, including questionnaires to users, interviews with randomly chosen users, and visits with departments and user groups. We have also established an Academic Computing Advisory Committee, which is preparing an evaluation of ACSS.

Deputy Director Lawrence Liddiard described planned improvements in **computing systems**: This year we expect to reduce charges for computation and storage on the CYBERs by 15 percent, reduce VMS computing charges with the advent of the 8600, and, with the completion of the new phone system and computing network, offer campus-wide communications that are eight times faster than present. These 9600-baud communications speeds, at the cost of previous 1200-baud service, will also enable us to provide FSE, the Control Data full screen text editor, on all CYBER systems.

In the **future**, Mr. Liddiard said, we plan to provide instructional computing services on a UNIX 4.2 system and make advanced campus workstations available. When supercomputers and their peripheral devices are moved from our Lauderdale site to the University facility in the High Technology Corridor, we will also offer the service of housing and operating departmental computers.

Michael Frisch, manager of software applications and tools, described his group's plans for text processing and analysis, math and engineering packages, graphics, and artificial intelligence support.

In **graphics**, we will emphasize the VAX 8600 as our primary graphics machine rather than the CYBERs. ACSS will also emphasize Precision Visuals (DI-3000) graphics software (including their Grafmaker, PicSure, TextPro, and Contouring packages), eventually de-emphasizing ISSCO software (DISSPLA and TELL-A-GRAF, new versions of which we will soon install), and dropping support of our own MNCORE software. We will help users convert their existing jobs to Precision Visuals software. ACSS has ordered plotting hardware for the new 8700 laser printer; this will permit users to print 8 by 11-inch graphs at a resolution of 300 dots per inch on the 8700. We are also adding the DI-3000 graphics package to the MERITSS MD machine, will support the Raster Tech ONE/10 graphics terminal, and will soon inaugurate a new slide generation service through an outside vendor. ACSS is also considering acquiring a new electrostatic plotter, a new

Continued on page 82

Continued from page 81

color hardcopy device, and its own slide generation equipment.

In **artificial intelligence**, ACSS will provide consulting and documentation for AI research and instruction on micros and mainframes, make available demonstration copies of AI software for the IBM-PC and Macintosh, and will support the OPS5 and PROLOG programming languages on the VAX 8600.

ACSS has hired a support specialist for SCRIBE and other **text processing**, will install the TEX mathematical text processing package on the VAX 8600, and the EQN/TROFF package on the future UNIX system. New **text analysis** packages will be available in the future for the CYBERs, the VAX, and micros.

Math and engineering packages formerly available only on the CRAY will be available on the CYBERs. Most packages on the CYBERs will be available on the VAX 8600, and others will be added.

Janet Eberhart, a **programming languages** supervisor, then discussed plans for the VAX 8600, which will have Pascal, FORTRAN, COBOL, LISP, OPS5, and Ada. (See the April *Newsletter* for details about these VAX compilers.) The VAX will provide a collection of new programming and data management tools, including the Language Sensitive Editor. All the VAX languages will be able to share the Common Run-time Library. On the CYBERs, updated versions of Pascal and M77 will be installed.

Bruce Center, supervisor of **statistical software**, discussed new and expanded services in his area. On the CYBERs, we have added SPSSX, the new, improved version of SPSS, two time series and regression packages (SHAZAM and RATS), and have made SPSSX, BDMP, and LISREL available on the ME instructional system.

On the VAX 8600, we will install SPSSX, SAS, and MINITAB. The VAX version of SPSSX will include SPSS/GRAPHICS, a full-color graphics system, and TABLES, for sending camera-ready tables to the Xerox 8700 laser printer. SAS, a popular statistics package not available for CYBER systems, will also include a color graphics package, SAS/GRAPH. MINITAB Version 5, also not available for CYBERs, will be the version on the 8600.

Dr. Center will also evaluate microcomputer statistics software to recommend to users, including micro versions of SPSS, SAS, and MINITAB for the IBM/XT and its compatibles. He will evaluate other micro packages as they become available; he has found no statistical packages for the Macintosh that he can recommend.

Mark McCahill, a manager in the Microcomputer Group, described plans to expand **microcomputer** support. The group has expanded the number of Micro short courses it offers and provides courses on new topics as the demand for these becomes evident. The hours of the Micro HelpLine are being expanded as new staff can be hired and trained. In the future the Micro Group will provide consulting support for the Sun and Apollo workstations and also for local area networks (LANs) for micros, permitting the interchange of information and the sharing of peripheral devices among clusters of microcomputers. This includes taking advantage of the high-speed transmission capabilities of the new University phone system. The Micro Group has also developed some micro programming tools now available to Macintosh and IBM programmers.

Gene Kath, manager of field engineering, discussed the expanding **maintenance programs** offered by Engineering Services. The group offers a variety of maintenance programs for lab and departmental terminals and all microcomputers and peripherals now offered in the University's Microcomputer Discount Program. The group will expand its micro maintenance services as other hardware becomes available in the discount program, and recently began providing security kit hardware and installation for micros.

Concluding this portion of the meeting, Peter Oberg returned to discuss our plans for **data bases**: On the CYBERs, there will be major updates to System 2000 and SIR Fall Quarter. The Statistical Analysis Systems (SAS) Institute has taken over the maintenance of System 2000 and plans to add statistical routines to the package. SIR plans to add a graphics module and a PC version of SIR is currently available from our Micro Group. For the 8600, we will make SIR, Datatrieve, and Ingres available. Ingres is a fourth-generation language that is non-procedural —i.e., one-line commands retrieve data, produce reports, or update your entries.

After the staff presentations, there was an opportunity for **questions**. A user asked when ACSS will offer instructional services on the VAX 8600. (Answer: We are not prepared to offer this service yet, but we hope to provide it soon.) Another user asked if it will be possible in the future to access supercomputers from ACSS systems. (Answer: We are willing to provide that service when requested by the Supercomputer Institute; user demand will be an important factor in determining whether we make such a service available.)

Regression Analysis of Time Series Statistics Package

Bruce Center

Regression Analysis of Time Series (RATS) is a batch-oriented statistics package for the analysis and forecasting of time series data. RATS is particularly useful for dealing with econometric analysis, forecasting, and ARIMA (autoregressive integrated moving averages) models. It includes standard techniques such as ordinary least squares, weighted least squares, least squares with omitted observations, two- and three-stage least squares, seemingly unrelated regressions, univariate ARIMA estimation, and single equation non-linear least squares. It also includes instructions for analyzing multivariate time series using vector autoregressions, spectral analysis, and Fast Fourier Transforms. RATS only handles linear models.

RATS can be run interactively, but we do not recommend it. Normally you will run RATS as a SUBMIT job with the following set of commands:

```
/JOB
/NOSEQ
JOBNAME.
/USER
RATS.
/EOR
   .
   .         {RATS control cards}
   .
/EOR
```

SUBMIT,*lfn*,TO will then return the output and dayfile to the WAIT queue.

QGET,*jsn*, will transfer it from the WAIT queue to the local file ***jsn***.

RATS data can be read by variable or by observation in formatted, format-free or binary form from an external data file or from the input file. Data can be printed by the simple instruction PRINT or by COPY, which is a flexible instruction with options similar to those for DATA.

The general data transformation instruction called SET is designed to create polynomial trends or take quarterly averages of monthly series, in a straightforward way, by explicitly using the time subscript (T). For instance, in the example below, GNP is set to the sum of CONSMPTN, INVESTMT, and GOVT; TREND is set to a time trend; and TBILLQ is set equal to the quarterly averages of monthly data for TBILL.

```
SET GNP 46,1 80,2 = CONSMPTN(T)  + INVESTMT(T) + GOVT(T)
SET TREND 47,1 80,2 = T
SET TBILLQ 47,1 80,2 = (TBILL(3*T-2) + TBILL(3*T))/3
```

Matrices can be manipulated by using algebraic formulas, such as:

INV(TR(X1)*X1)*TR(X1)*Y for $(X1'X1)^{-1}X1'Y$

Individual elements of arrays can be accessed as subscripted variables:

X(3,I)

for instance.

Continued on page 84

Continued from page 83

Calculations using scalars can be done using FORTAN-like expressions, such as:

```
SET F = (SSR - SSU)*DF2 / (SSU*DF1)
```

The usual FORTRAN functions such as SQRT, ABS and LOG are also provided.

RATS provides a set of control instructions for structured programming, including IF-ELSE, WHILE, UNTIL, DO, and subroutine-like procedures. These may be nested.

RATS includes special instructions for creating seasonal dummies (SEASONAL) and for filtering series (FILTER). An EQUATION option on FILTER can be used to determine the filter empirically.

RATS contains commands for testing hypotheses: TEST, EXCLUDE, RESTRICT, and MRESTRICT test linear restrictions on the coefficient vectors of any of the regression commands (including the multivariate three-stage and seemingly unrelated regression systems). RATIO computes the likelihood ratio statistics from two sets of series of residuals.

Forecasting instructions operate on vector autoregressions, linear simultaneous systems, ARIMA models, and on any combination of these. These instructions can be used for conditional and unconditional forecasting, simulation with random shocks, and generation of statistics on out-of-sample forecast errors. The instructions IMPULSE, ERRORS, and HISTORY are used for studies of the properties of multivariate linear systems. SIMULATE can be used for generating data for Monte Carlo tests.

RATS includes a data bank facility in which RATS-readable time series and matrices can be kept, modified, and deleted.

More information on RATS can be obtained from WRITEUP,RATS and from the *RATS 4.1 User's Manual.* We strongly recommend that you read the reference manual before running RATS. You may examine the manual in the ACSS Reference Room in 128A Lind Hall. Contact the Statistics HELP-Line (376-1761) from 1:00 to 2:00 p.m. daily for help on RATS.

MINOS 5 Installed

Mike Frisch

MINOS (Modular In-Core Nonlinear Optimization System) Version 5 package has been installed on ACSS's CYBERs as an M77 user library named MIN5LIB. The library can be accessed with the FETCH command as
`FETCH,MIN5LIB/V=M77.`

MIN5LIB can be used to solve problems in linear programming, unconstrained optimization, linearly constrained optimization, and nonlinearly constrained optimization. The optimization problems must have smooth, nonlinear parts; that is, they must have known gradients. Gradients needed for optimization can be provided by the user, be approximated by MIN5LIB, or be provided through a combination of the two. A gradient-checking option is available to help users test their code. Problems can be large-scale but must be able to fit in memory, although sparse matrix methods are used to minimize the amount of memory required. Users can pick a starting guess for the solution or let MIN5LIB choose one.

To handle optimization problems, MIN5LIB must be provided with a subroutine to evaluate the objective function (FUNOBJ) and a subroutine to evaluate the constraints (FUNCON), if there are any. As an option, a subroutine can be provided for problem modification (MATMOD). The only routine in MIN5LIB called by the user is MINOS1, which runs complete problems defined for the linear part by data files and for the nonlinear part by subroutines

Continued on page 85

Continued from page 84

FUNOBJ, FUNCON, and MATMOD.

MIN5LIB requires users to provide a workspace in blank COMMON and to call MINOS1. A data file (called the SPECS file in the user's manual) must be provided for specifying information to MIN5LIB about the number of unknowns, number of constraints, number of nonlinear variables, tolerances, etc. Within this file is the location of the file containing the coefficients of the linear part of the problem in a special format called MPS (Mathematical Programming System). This format is used by commercial linear programming packages, such as APEX, as a standard way to input data, particularly for large problems. Also included in the SPECS file are locations of optional files for saving solutions to be used elsewhere, such as start-up values for other runs.

MINOS Version 5 was written by Bruce Murtagh and Michael Saunders of the Systems Optimization Laboratory, Department of Operations Research, Stanford University, Stanford, CA 94305. Together with the ACSS document WRITEUP,MIN5LIB, their *MINOS 5.0 User's Guide* provides the information needed to use MIN5LIB on the CYBER. A copy of the *User's Guide* is on reserve in the ACSS Reference Room, 128A Lind Hall, and is available from the Systems Optimization Laboratory at Stanford. Also on reserve in the Reference Room are reports giving algorithm details.

ACSS has tested MIN5LIB on programs provided with the package, and all test programs have run satisfactorily (although some of them use a lot of memory). ACSS regards MIN5LIB as a high-quality, full-capability optimization package. It has a good user's manual and provides a large number of user options. It is, however, somewhat more complicated to use than its main competitor, the GRG2 (Generalized Reduced Gradient) optimization library. Questions about MIN5LIB should be directed to Mike Frisch, (612) 376-1636. Sample job runs, with problem descriptions, input, and MIN5LIB output, can be seen on WRITEUP,MIN5LIB.

Note: ACSS's contract with Stanford limits use of MINOS to noncommercial research purposes and to University students and staff.

Expanded Services and Hours

Elaine Collins

Expanded text-processing service and phone-consulting hours are now available to all ACSS system users. Staff can be reached at 376-5262, a new phone number, from 9 a.m. to 11:00 a.m., Monday and Tuesday, and from 1 p.m. to 3:00 p.m., Wednesday and Friday.

Staff from the text-processing area can answer specific questions and assist with problems, as well as respond to general inquiries about text processing. In addition, document-coding service is available on a contract basis.

Currently, the focus of the text-processing area is on text-formatting and printing options that are available on the VAX 8600 (VX). SCRIBE formatting for the Xerox 8700 laser printer and the Spinwriter is now supported on ACSS's VX operating system.

SCRIBE Document Production Software is a powerful, widely distributed system, which has been used on campus to produce theses and dissertations, manuals, reports, and camera-ready copy for publications. SCRIBE uses the font-handling capabilities of the Xerox 8700 laser printer to produce true proportionally-spaced text in a variety of type styles, sizes, and character sets.

SCRIBE document design service is available to all ACSS system users from staff in the text-processing area.

Reference copies of SCRIBE documentation are located in the ACSS Reference Room in 128A Lind Hall. This documentation includes *SCRIBE Document Production Software User Manual,* published by Unilogic, Ltd., Pittsburgh, PA; *Scribe at UCC*; *Theses in Scribe*; and two ACSS *Briefs*, *NEC Spinwriter Typewheels* and *Xerox 9700 Fonts.*

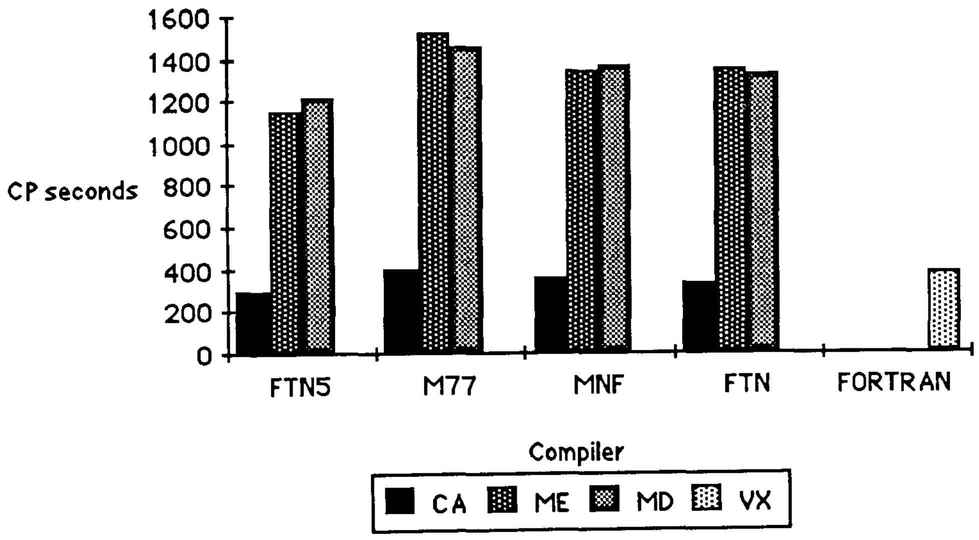
Contents *continued from page 88*

Correction

The statement made in the April *Newsletter* article "Choosing a FORTRAN Compiler for your Job" that the "VAX no longer maintains a 66-standard compiler" is incorrect. The /NOF77 qualifier causes the compiler to select the FORTRAN-66 interpretation in cases of incompatibility.

In the same *Newsletter* article, the figure that appeared on page 55 was incorrect as printed. The correct figure appears as follows:

Figure 2: Average Execution Time



PHONE NUMBERS

Access:	
CYBER(CA)—10, 30 cps	376-5730
—120 cps	376-5706
MERITSS(ME)—10, 30 cps	376-7730
—120 cps	376-7120
VAX/VMS(VX)—(autobaud)	376-9070
Accounts:	
MERITSS	373-7745
User Names	373-4548
Computer-Aided Instruction	376-2975
Computer Hours (recorded message)	373-4927
Consulting:	
HELP-Line 8 a.m.–5 p.m., Monday-Friday	376-5592
Statistics Packages 1–2 p.m., Monday-Friday	376-1761
Data Bases 10–11 a.m., Monday-Friday	376-1761
Microcomputers 9:30 a.m.–noon and 1:30–4 p.m., Monday, Tuesday, Friday 9:30 a.m.–4 p.m., Wednesday, Thursday	376-4276
Text Analysis, Artificial Intelligence 3-4 p.m., Monday-Friday	376-2944
Text Processing 9–11 a.m., Monday, Tuesday 1–3 p.m., Wednesday, Friday	376-5262

Contract Programming	376-1764
Data Base Applications	376-1764
Engineering Services	376-1023, 376-8153
Equipment Maintenance/Information	376-8153
Lind Hall I/O	373-4596
Graphics Software	376-5592
HELP-Line 8 a.m.–5 p.m., Monday-Friday	376-5592
HOURS-line (recorded message)	373-4927
Information, Wulling Hall	373-4360
Information, Lauderdale	373-4912
Instructional Labs	376-2703
Instructional Services	373-7745
Lauderdale Computer Room	373-4940
Lauderdale Services	638-0523
Newsletter Subscription	376-1491
Permanent File Restoration	376-5605
Professional Services Division	376-1764
Project Assistance	376-1764
Reference Room	373-7744
Remote Batch (RJE) Services	376-2703
Short Courses	376-8806
Shuttle Bus Service	376-3068
System Status (recorded message)	373-4927
Tape Librarian: see Lauderdale Services	

OPERATING HOURS

	CYBER (CA)	Low Rate	MERITSS (ME)	MERITSS (MD)	VAX (VX)
M-F	7 a.m. - 4 a.m.	8 p.m - 4 a.m.	7:45 a.m. - 3:30 a.m.	8:00 a.m. - 1:30 a.m.	7 a.m. - 4 a.m.
Sat	4 a.m. - 5:15 p.m	4 a.m. - 5:15 p.m.	7:45 a.m. - 3:30 a.m.	8:00 a.m. - 1:30 a.m.	4 a.m. - 5:15 p.m.
Sun	4 p.m. - 1 a.m.	4 p.m. - 1 a.m.	4 p.m. - 3:30 a.m.	4:00 p.m. - midnight	4 p.m. - 1 a.m.

PUBLIC LABS-TWIN CITIES CAMPUS

Location	Batch	Interactive	Micro
East Bank			
Arch 160			X
CentH		X	
ComH		X	
DiehlH 207		X	
EltH 121, 124		X	
EltH N640	X		
FolH 14, 14a	X[1]	X*	X
FronH		X	
LindH 26		X	
LindH 128B	X	*	
LindH 306B			X
MechE 308		X	
Physics 69		*	
PlH		X	
SafH		X	
TerrH		X	
VincH 4		X	
Walib 9		X	X
West Bank			
AndH 170			X
BlegH 25		*	
BlegH 90	X		
BlegH 140		X	
MdbH		X	
OMWL 2	X[1]	X	
St. Paul			
BaH		X	
CentLib B50			X
ClaOff 125	X	X	

* Research cluster; access to CYBER CA and VAX/VMS
X in interactive column indicates access to MERITSS
1 Printer only.
For more information see WRITEUP(LABS)

Contents

Continued on page 75

***The* ACSS Newsletter**
May 1986
Volume 20, Number 5

Acting Director: *Michael M. Skow*
Editors: *Steven Brehe, Paula Goblirsch*

The *ACSS Newsletter* is published monthly by Academic Computing Services and Systems (formerly the University Computing Center) of the University of Minnesota, Twin Cities. Deadline for articles is the 10th of the month preceding publication; deadline for short announcements is the 15th. The *Newsletter* is produced with an Apple Macintosh running Microsoft Word, MacPaint, MacDraw, and Aldus Pagemaker software, with camera-ready copy produced on the Apple LaserWriter.

Direct comments, suggestions, articles, announcements, and subscriptions to the editors at the address below, or call (612) 376-1491. On-campus address changes *must* include your department's name and your *departmental* address. Subscriptions are free.

The ACSS
Newsletter

Academic Computing Services and Systems

Technical Publications
5 Wulling Hall
University of Minnesota
86 Pleasant Street SE
Minneapolis, Minnesota 55455

Nonprofit Org.
U.S. Postage
PAID
Minneapolis, Mn.
Permit No. 155

KAREN KLINKENBERG
10 WALTER LIBRARY
UNIVERSITY ARCHIVES.

Deliver to current occupant.