DESY Computing Newsletter No. 3 May 1992

A Publication of the DESY User Support Group

Contents

- Editorial 1
- RACF Registration on IBM 2
- Fortran News 3
- Usage of CERNLIB on the IBM 5
- Electronic Mail at DESY 6
- Network Behaviour 9
- The UCO Diskette Copying Facility 11
- The DESY Staging System 13
- From 0 to X in five minutes 15
- INTERLINKing VAX and IBM 17
- How to Teach Yourself VAX/VMS 21
- XFIND/WWW on the DESY IBM and Central VAX 22
- GEP Version 5.0 24
- Questions and Answers from the UCO 27

DESY Computing Newsletter No. 3

• Editorial

by Michael Behrens

In the first issue of this newsletter we announced that we would produce one issue per quarter. You may have noticed that we have missed this target now for the second time. The reason for this in both cases was not lack of themes, but simply lack of time to write, prepare for publication, and finally publish what we had prepared. Like many of you, we have several duties, and some of these duties seemed more urgent in this phase just before the start of HERA operations, than the timely production of the next issue.

We will nevertheless continue with the newsletter, and we will try to improve on its timeliness. We have even been asked quite often for the publication date of this issue and we take this as an expression of your interest in our product.

In this issue you will see less articles covering the central IBM. This reflects the increasing diversity of our computing environment and the large amount of activity in the less well established areas.

You will nevertheless find an article describing a new development on the IBM, the staging system that was recently made available. Staging works on the IBM, but not only for the IBM. It will be used by applications running on other computing platforms at DESY as well. This co-operation between the various systems shows up in other articles too, and actually only few articles this time deal with single-machine issues.

This new paradigm of distributed and cooperative work starts to become regular practice at DESY. As a consequence, life becomes more complicated for both sides, the users as well as the providers of these services.

Our role as newsletter editors is twofold. On the one hand we try to educate you – the users – on what services are being offered to you, on the other hand we try to help you – the providers of services – to distribute the results of your effort to the user community.

We had more contributions from other authors this time, in some cases without even pushing them, and we are very happy about this positive development. We have to thank these authors for the effort they put into their articles. We also hope for more contributions in the future, and we will happily assist every prospective author in the preparation of his/her text.

Editors of the DCN are:		
Katherine Wipf	RØ1WIP	ext. 3222
Michael Behrens	RØ1BEH	ext. 2556
Jan Hendrik Peters	RØ1NET	ext. 2583

• RACF Registration on IBM

by Jan Hendrik Peters

RACF, the IBM data protection system, was introduced at DESY years ago and most users are defined within RACF already. Now everybody not yet registered under RACF will have to do so by June 1992. After this time, only users covered by RACF will be able to use the computer. This action is required in order to maintain our level of data protection in a distributed computing environment. In addition, an increasing number of services is available only if you are covered by RACF (e.g. the IP-services ftp, smtp).
If you are one of those users not yet covered,you will now have to register as a RACF user. This is no big thing, and should have no impact on your usage of the computer.
The technicalities needed are handled by "Group Administrators" for the different groups at DESY. If your userid does not belong to one of the larger groups, the User Support Group will act as administrator for you. The main task of the administrators is to take care of your "Admission Form", to certify your identity and to be contacted by you in case of "revoked password" or other changes.

What is RACF?

If you do not know what RACF is all about, you will find details of RACF functions with `Help RACF` and the names of the administrators and the structures of the groups with `Help RACGRPS` under NEWLIB. For you the main effect of the RACF introduction will be the following:

- You will be able to change your password whenever you like it. (You will be required to change it at certain intervals. Currently a password expires after 6 month.)
- You may have "personal" files readable only by yourself (see `Help RACF`)
- You may have "group" or "public" files that can be modified freely by any member of your "group" or by anybody respectively (see `Help RACF`)
- Your password will be made invalid (revoked) after too may attempts to LOGON with an invalid password. In this case you have to ask your group administrator to reestablish your userid.

You will be registered under RACF as soon as we received your Application Form. You will then see an additional field
`NEW PASSWORD ===>`
on your LOGON-panel, indicating that you are now registered. You will then have to change your password on the first LOGON following your registration. From then on you may change it whenever you like, and in addition you will be reminded to change it at regular intervals. To improve security, short passwords are not allowed. You need a 7- or 8-character password which may contain characters, numbers and any of the symbols #, @, $.
If you are not yet under RACF, please consult your groups RACF administrator. See the following table or `Help RACGRPS` on the IBM to find who is responsible for you and register as soon as possible.

See table of RACF groups on next page

Group	Administrator	Userid
Argus/F15	U. Djuanda	F15DET
	R. Mankel	F15MAN
	H.D. Schulz	F15HDS
Cello/FCE	J.H. Peters	F14JHP
F58	V. Heynen	F58HEY
Hasylab	O. Beimgraben	F41BEI
	T. Kracht	F41TKR
	H. Schultz	F41SCH
HERA-p	I. Borchardt	RØ1BOR
	R. Pforte	F51PFR
H1	A. De Roeck	H1KADR
	H.-U. Martyn	F34MAR
	J. Meyer	F14JME
IfH Zeuthen	I. Meier	IFHIME
	W. Niepraschk	IFHNIE
KRYO	M. Clausen	KRYMCL
	U. Knopf	BØ1KNO
	H. Lierl	F35LIE
L	H. Preissner	LØØHTP
M	C. Kluth	F35KLU
	S. Wipf	MPYSZW
R1	D. Wiemann	RØ1WIE
R2	B.-W. Hellwig	RØ2HEL
	P.-K. Schilling	RØ2SCH
Theory	A. Ali	TØØALI
	I. Montvay	TØØMON
Theor. Nucl. Physics	H.V. von Geramb	I04GER
ZEUS	E. Lohrmann	F35LOH
	D. Notz	F1BNOT
1^{st} Inst. for Exp. Physics	H. Krause	IØ4KRA
	H. Kreiser	IØ4HKR
	W. Scobel	IØ4SCO
2^{nd} Inst. for Exp. Physics	T. Voss	IØ1VOS
2^{nd} Inst. for Theor. Physics	J. Bartels	IØ2BAR
	G. Kramer	IØ2KRA
All other Groups	M. Behrens	RØ1BEH
	J.H. Peters	RØ1NET
	K. Wipf	RØ1WIP

• Fortran News

by Michael Behrens

New FORTRAN Release on IBM

A new release of the IBM VSFORTRAN compiler has been installed on the central IBM. The new release is named Version 2.5. It is currently under test and it will replace the current Version 2.4 after previous announcement. If you specify `CVER=9` in your job, you will get the new version.
Please test the new release now. It will become the production version soon.

The new compiler has several new features, but some of them will not be of general interest. The important new features are:

Padding for variable length records possible

With previous versions of the compiler, an I/O error was produced when one tried to read more characters with a formatted read statement than were actually present. This usually happened when reading 80 characters, including trailing spaces, from a dataset in V–format. The new compiler option `RECPAD` will supply an unlimited number of logically blank characters, thus imitating the case of a fixed format dataset.
(logically blank means: blank for character data, 0 for integer, 0.0 for real ...)

The compiler was installed with the default `RECPAD(all)` which means padding for all types of records by default. You may overwrite this default upon compilation, e.g. with the compile time option `NORECPAD`.

Compiler runs in upper region

Not only the generated code – as in previous releases – but also the compiler itself is now XA-capable and runs in the extended region. Especially packages like KUIP which call the compiler from inside the package will benefit from this improvement.

Improved compiler options for DC (dynamic COMMON)

Up 'til now users of dynamic COMMON had to specify all COMMON names separately. Now it is possible to declare all COMMONs dynamic by specifying `DC(*)`.

There is a corresponding new option `SC` for static COMMON, which is the usual COMMON. This way you can say "everything but ...". See the following examples.

DC(*)	All named COMMON blocks become dynamic COMMON
DC(*),SC(C1,C2)	C1 and C2 are static COMMON, all others dynamic
SC(C1,C2),DC(*)	All COMMONs are dynamic. SC was overwritten because DC(*) was last
SC(C1),DC(*),SC(C2)	Only C2 is static

You have to remember that *blank COMMON* cannot be used as dynamic COMMON in principle, `DC(*)` excludes blank COMMON.

Parallel Fortran support

The new release supports Parallel Fortran, which means it supports automatic parallelization (like vectorization) and multitasking. This way a user program can consist of several tasks that could run on several processors concurrently. In principle this is a very large and important step, but will and should not be of interest to most users. The computer center strongly discourages use of the parallel features. Using parallel programs is complicated and imposes additional restrictions on programs as well as additional overhead. The main benefits would come from reducing elapsed time by using several processors of our IBM. This behaviour would only be acceptable on a machine dedicated to very few users or programs, because all other programs will be slowed down for the benefit of a single user program. There are normally easier ways to get really important jobs faster through the machine than by using the parallel feature. If you feel that you nevertheless have a reason for going parallel, you should discuss it with me beforehand.

More new features

There are several more features especially for very large programs (programs which use several 1000's of Megabyte memory). This is definitely not the type of program used at DESY. A detailed description of these features is omitted here for this reason.

Fortran 90 Compiler on Apollo

A compiler for Fortran 90 is available on the Apollo. This compiler – currently the only compiler available – was developed by NAG (Numerical Algorithms Group) and is available for several Unix machines. We have purchased this compiler to make the new language available. I do not expect that it will be used for production work on such a small machine, but is intended as a training ground to gain experience in using the new language and to try out its possibilities. The compiler implements the full language, and tests, e.g. at CERN, have shown that it works for real physics code.

It is unlikely that this compiler will also be available under MVS or VMS in the near future. I know that IBM is working on a compiler for MVS and there will be a compiler on VMS as well, but it will take some time before it becomes available.

Currently Release 1.0 of the compiler is installed. It will soon be replaced by Release 1.1.
In the DESY library you will find several books describing Fortran 90, e.g.

- M. Metcalf, J.Reid : Fortran 90 explained
- W. Brainerd, C.Goldberg, J. Adams : Programmers Guide to Fortran 90

The first book concentrates more on the new things in Fortran 90, the second starts almost from scratch and does not refer to previous FORTRAN versions.

• Usage of CERNLIB on the IBM

by Harald Keller

Current State of CERNLIBs

New numbering scheme of CERNLIB releases

CERN has separated the release of the CERN program libraries from the publication of the CERN Computer Newsletters. From this year on, the version stamping has the format 'Vyyl' with yy = year and l = upgrade level, e.g.

'V92A' first release in 1992
'V92B' 2nd release in 1992

Change of format of CERNLIB source files

The old PAM files are packed into binary FORTRAN records for rapid handling both by the program PATCHY and by the auxiliaries. Nowadays, however, the speed problem is no longer relevant and the binary format has the inconvenience that the files cannot be inspected with an editor without a preliminary translation. In addition there are often problems in transporting binary sources between different systems.

This is why, in 1992, CERN has started to distribute the PAM files in card format. These CARD PAM files are called CAR files for short.

PATCHY itself is not affected by this change. It continues behaving as expected for both PAM or CAR files. The only difference is that PATCHY attaches a CAR file with:

```
+PAM, 11, T=CARDS.
```

CMZ users are not affected, because CMZ already uses CAR files.

Current state of CERNLIBs on the IBM

NOTE: You should use the command

```
Help CERNLIB
```

under NEWLIB to verify the current state of the CERNLIBs on the IBM, because by the time this information is printed, it might already be obsolete.

Version Line	OLD-PRO-NEW Line
'CNLxxx' versions: software upgrade associated to a CNL issue number. **'Vyyl'** versions: software upgrades are no longer associated with a CNL issue number.	From the Version Line libraries 3 generations are maintained.
r∅1utl.cern.cnl201.lib.*suffix* r∅1utl.cern.v92a.lib.*suffix* r∅1utl.cern.v92a.lib.*suffix*	r∅1utl.cern.OLD.*suffix* r∅1utl.cern.PRO.*suffix* r∅1utl.cern.NEW.*suffix*

Dynamic COMMON

For all versions beginning with 'CNL200' you have to use the compile option

DC(PAWC,CMDATA,CMSEQ0).

The CERNLIBs on the IBM are compiled with this option as well. If you omit this option, your program will not run as expected and produce strange results (or abend) without any error message during compile or link.

• Electronic Mail at DESY

by Jan Hendrik Peters

Mail Addresses

Mail addresses have a fixed format which depends on the network the computer sits on. A user on this computer is defined by his/her username and the host or node name of the computer on the network.

Typical addresses on the various networks appear in one of the following forms:

Internet	user@host.subdomain.domain
DECnet	NODE::USER
Bitnet	USER at NODE
uucp	host3!host2!host1!user
X.400	S=surname, OU=org.unit, U=unit, PRMD=priv.mgmt.domain, ADMD=admin.domain, C=country

Since this first form is understood by most mail systems, one should try to use it for **all** mail addresses, regardless of the system used.
Sometimes the user part in the Internet or **domain style** address is replaced by another full address. This address then contains a %-sign instead of the @-sign and the host-domain part is a gateway system: user%host.net@gateway.domain

Address Convention at DESY

At DESY we have adopted this domain style address convention for all mail hosts. This requires additional programs on the machines which don't run TCP/IP natively. On the central VAX cluster the handling of this address convention is done by a program called PMDF which interfaces to VMS mail, on the IBM this is done by the MAIL utility itself. Fully qualified addresses of our systems at DESY are e.g.:

r01net@dsyibm.desy.de
— for the IBM on the Internet side,
f14jhp@dhhdesy3.bitnet
— for the IBM on the BITNET side,
postmaster@vxdesy.desy.de
— for the central VAX, and so on.

Usually the domain part can be omitted when mail is sent within the same domain, the hostname may in addition be omitted if mail is sent on the same host (so called short form names or addresses).
The important message is that one should **always** use a fully qualified domain style address. These addresses are understood by almost all mailing systems around the world, while e.g. BITNET or DECnet style addresses are only understood by systems on these networks. Since some of our computers are connected to more than one network, they carry more than one name. If you are in doubt which address to use, use the Internet address for the computer you want to reach, since all other addresses might disappear in future. The addresses for DESY computers can be found in the following table:

computer	**Internet address (preferred)**
IBM	**dsyibm.desy.de**
central VAX	**vxdesy.desy.de**
other VAX	node**.desy.de**
Apollo	**apollo.desy.de**

computer	Bitnet	DECnet
IBM	dhhdesy3.bitnet	—
central VAX	desyvax.bitnet	VXDESY
other VAX	—	NODE
Apollo	—	—

A warning: The Internet address for the IBM may only be used for users that are registered under RACF. Mail to non-RACF users will be rejected. (See article about *RACF Registration on IBM* in this Newsletter)

How to Convert to Domain Style Addresses

Most people still use addresses in their native format and tell you that you can reach them at NODE::USER or host3!host2!host1!user and you have no idea how to reach them from your computer. The first action

Converting to Domain Style Addresses		
Given Address	Node at DESY	Node not at DESY
node::user	user@node.desy.de	user%node@vxdesy.desy.de
hostn! ... host1!user	—	user@host1.uucp
user@node	user@node.bitnet	user@node.bitnet
user at node	user@node.bitnet	user@node.bitnet
S=surname,OU=ou,U=u,PRMD=prmd, ADMD=admd,C=c	—	surname@ou.u.prdm.admd.c

is to translate this address to a fully qualified domain style address.

The above table will give you some hints how this could be done. Unfortunately there are no strict rules for the conversion of DECnet addresses (the ones with the ::) and uucp addresses (the ones with the !) for other networks. The assumption is that there is a gateway on the network that knows more about the host you want to reach.

How to Send Mail from the IBM

Call the NEWLIB command MAIL and select the action SM, or call SENDMAIL directly. In the full screen panel give the full address of the person you want to reach, and fill out the subject line to make it easier for the receiver to sort and select the message. Then press the ENTER key to edit your message with the NEWLIB editor. Typing RETURN in the command line or pressing the corresponding key displayed in the =MSG=> line, will bring you back to the full screen panel, where the next steps to send, store and print the message are displayed.

If you would rather receive all your mail on a different computer you can set an automatic forward for your mail. Select the action F in the MAIL panel and give the full forwarding address. Press the ENTER key to see whether the address is syntactically correct and then PF2 to activate the forwarding. Currently only mail coming from Bitnet will be forwarded, the forwarding mechanism for Internet mail will be implemented soon.

How to Send Mail from the Central VAX

Execute the command MAIL and type SEND after the MAIL> prompt. Give the fully qualified address in the form IN%"address" if you are on a VAX where PMDF is already installed, otherwise type VXDESY::IN%"address" and press the RETURN key. Fill out the subject line to make it easier for the receiver to sort and select the message. You then enter a line mode editor where you can type the message you want to send line by line. Press CTRL-Z to leave the editor and to send the message. If you want to use your favourite VAX editor instead, you have to set the editor:

MAIL> SET EDITOR editor-name

and type

MAIL> send/edit

instead of the usual send command.

If you would rather receive all your mail on a different computer you can set an automatic forward for your mail. There are two possibilities to do so, one is using VMS mail and one PMDF. The preferred way is with PMDF:

$ PMDF FORWARD "address"

but you could also use the VMS mail forwarding mechanism:

MAIL> SET FORWARD IN%"""address"""

Please notice that you have to use 3(!) sets of quotation marks. There are of course some trade-offs in each of the methods. The PMDF forward does not catch pure DECnet mail. The VMS mail forward produces very long message headers and is slower since mail is usually channeled through PMDF be-

fore it is passed on to VMS mail.

Here is a little example session:

```
$ mail
MAIL> send
To:         in%"petersjh@sscvx1.ssc.gov"
Subj:       here are the latest results
Enter your message below.  Press CTRL/Z when
complete, or CTRL/C to quit:
Dear friend,

I finally succeeded to ....

Yours ...
Exit
MAIL> exit
```

How to Send Mail from a Station in the APOLLO Domain Ring

Execute the command *mailx address* to send a mail. The system will then prompt you for a subject which you should fill out in order to make it easier for the receiver to sort and select the message. You then enter a line editor where you can type the message you want to send line by line. If you have finished the message text press the RETURN key once more and then CTRL-D to finish editing and to send the message. Some systems will prompt you for additional carbon copies (cc:) of your message. If you want to send this message to someone else give another address, otherwise just press the RETURN key. If you want to use your favourite editor instead of the line mode editor *ed*, you have to set the corresponding environmental variables EDITOR or VISUAL, e.g.:
ips02> EDITOR=vi
ips02> export EDITOR
and then invoke the editor by typing ~v for the VISUAL editor or ~e for the EDITOR editor in the input mode after having filled out the subject. Once you have finished editing, you still have to press CTRL-D to send the message.

Here is a little example session:

```
ips02> mailx petersjh@vxdesy.desy.de
Subject: what I need to say
Here comes the message you want to write.

Greetings John
EOT
ips02>
```

• Network Behaviour

by Jan Hendrik Peters

With the start of the HERA experiments, it will be quite important that everybody sticks to a few rules when he/she is using the networks at DESY or to remote sites anywhere else. One of the crucial items in this game is file transfer via ftp. It uses not only a considerable amount of the network but also of the cpu time of the hosts involved. There is something you should clearly keep in mind: *the higher the transfer rate, the higher the load on the cpu.* Another important aspect is the length of time you are using the network for your file transfer and how much you hinder the interactive sessions of other users using the same data path.

CPU Time needed for File Transfer

I will give some examples of cpu time usage on the IBM for transfering large amounts of data (POTs, DSTs, mini DSTs, n-tupels, etc.) and of the network load created by such transfers.

First some basic numbers: transfering data with a rate of 100 kB/s via the ethernet (that's the rate you can achieve during the day) will create a 10 % load on the IBM cpu. This number scales linearly with the data rate (i.e. 400 kB/s is equivalent to a 40 % load). A DST file of 200 MB (one cartridge) will then need 3 minutes and 20 second of cpu time on the IBM. The transfer time is 2000 seconds (about half an hour !). If 200 people transfer 1 cartridge per day to their workstations this will use 670 min of cpu time on the IBM. **Compare this with your daily cpu account in your collaboration !** By the way, this amounts to a total of 40 GB disk space attached to individual workstations alone for the ZEUS collaboration, and thus is not an unrealistic amount of data to be transfered. Clearly, transfering data in this way is something to be avoided by all means!

There is a better way of transfering massive amounts of data from the IBM to your workstation. By using Ultranet and the Silicon Graphics cluster as an intermediate buffer, data can be transfered at a much higher rate (1 MB/s is realistic) with a much lower load on the IBM (about 5 % at 1 MB/s transfer rate – a factor 20 lower than via Ethernet !). Part of the TCP protocol is realized in the Ultranet hardware and no longer needs to be handled by the IBM host. Another advantage is the increased packet size which can be used with Ultranet. To transfer data in this way, the group R2 has prepared a script called **ibmget**, which is already available on some unix workstations and will be available on others soon (see *man ibmget* or ask your system manager). Even with this high rate **out** of the IBM you still have the bottle neck in the network to get the data **into** your machine. This means that transfering DST or mini DST data over the network requires some organisation within your analysis group **before** you actually start the transfer.

File Transfer Times

Here are some other numbers which you should keep in mind (transfer rate and transfer time for a 1 MB file):

line to	transfer rate	time
cernvm	max. 50 kB/s	20 s
vxcern (decnet)	10 kB/s	100 s
slacvm	10 kB/s	100 s
sscvx1	5 kB/s	200 s
German HEP site	(night) 4 kB/s	4 min
German HEP site	(day) 1 kB/s	17 min
Univ Bonn (pib)	(day) 0.5 kB/s	34 min
MICOM printer	max 1 kB/s	17 min
printer R02PS2	6 kB/s	3 min

This clearly **rules out** sending **any** DST cartridge or the equivalent amount of mini DST or n-tuples via the network to collaborating institutes! The last number in the table is also of some importance. Think twice before you send screen dumps or complicated graphics to any of the MICOM printers (R02PS1, H01PS1, ...). A screen dump easily has 2 MB of data and so do detailed correlation plots!

Printer Speed and Printing Costs

Talking about printers: Someone recently printed simple linemode output on the relatively fast IBM printer LI1. These 70000 lines of output, however, took 1 hour 40 min to print! Please, think about the speed of the printer before you choose where to print and what to print. As a reminder here are some numbers for the speed of the printers available at DESY:

destination	print "speed"
LOCAL	100 pages/min
EXT	30 pages/min
LI1	20 pages/min
L1-L6	12 pages/min
Kyocera-P2000 (...PS.)	8 pages/min
HP Laserjet IIISi (R02PS2)	16 pages/min

Another item which you should have in mind before you start printing is the cost of a single page of output. These numbers vary widely depending on the printer, the method of printing (thermo transfer, ink, ...), the quality of the paper, and whether you choose paper or slides. Test printouts should certainly be printed on the cheapest medium and long outputs (e.g. program listings) clearly on the LOCAL IBM printer. Here are some prices we calculated for a single page (paper, toner, spare parts, maintenance):

printer	color	medium	size	cost[DM]
r02pc1	color	slide	A3	6.00
	b/w	slide	A3	4.60
	color	slide	A4	4.20
	b/w	slide	A4	3.20
	color	paper	A3	2.30
	b/w	paper	A3	0.84
	color	paper	A4	1.60
	b/w	paper	A4	0.56
Kyocera	b/w	paper	A4	0.03
L1-L6, LI1	b/w	paper	A4	0.03
EXT	b/w	paper	endless	0.03
LOCAL	b/w	paper	endless	0.03

As a general rule, remember that the distributed printers are self service printers: YOU should check whether there is enough paper for your printout and add paper if neccessary (note: the printable side of the paper is marked with an arrow on the wrapper), and YOU should remove the printouts from the tray and SORT them into the output bins located near the printer. Do not expect that these tasks will be done by your colleagues. In case of trouble with the printers, read the instructions posted on the printer or on the wall next to the printers. Never switch the printer off. If you cannot resolve the problem yourself, please inform Zbigniew Jakubowski from R2 by email r02jak@dhhdesy3.bitnet or jakubowski@vxdesy.desy.de or by phone (2345).
For problems with printers EXT, LOCAL or LI1 inform the operators of the computer center (ext. 3818).

• The UCO Diskette Copying Facility

by Michael Behrens and Katherine Wipf

The User Consulting Office now provides a facility for copying files from PC diskettes to any computer connected to Internet and vice versa. Two diskette drives (one for $3\frac{1}{2}$ and one for $5\frac{1}{4}$ inch diskettes) are available for this task, so that any IBM compatible diskette can be used. The drives are located just outside the UCO, and are connected to a PC inside the office which runs an FTP server. This server runs continuously, but is unattended except during the UCO office hours. If you have any problems you can either report them via mail to `RØ1UCO@DHHDESY3.BITNET` or ask the UCO consultants for assistance during office hours.

File copying is done via FTP (File Transfer Protocol). Help for `FTP` on the computer you wish to transfer files to or from will explain how to use FTP on this machine. Once you have entered FTP, you may ask for help with `HELP` and on some computers also with the command `?`.

How to Transfer Files to/from Diskettes

The name of the PC you have to connect to via FTP is `UCOPC` or `UCOPC.DESY.DE` from outside DESY. You can transfer files to or from any $3\frac{1}{2}$ inch (device name is `D:`) or $5\frac{1}{4}$ inch (device name is `E:`) IBM compatible diskettes. The procedure described below is valid for any host on the DESY site:

1. Insert your diskette into the appropriate drive
2. Choose a suitable terminal and logon to the computer you wish to transfer files from or to
   Note: The Falco terminal next to the drives is reserved for this purpose, but other terminals can also be used.
3. Give the command `FTP UCOPC`
   Note: This is the correct syntax for the DESY IBM, the central VAX, and the Apollo. The command may differ slightly on other computers.
4. You will be prompted for a user name. On the VAX or Apollo you can simply press `<RETURN>` because your userid is the default value; on the IBM you should type in your userid.
5. Now you can transfer your files using the FTP `GET` (from diskette) or `PUT` (to diskette) commands
   Note: Do not forget to specify the correct device name (`D:` or `E:`) for your PC files
6. Close the FTP connection with the `QUIT` command on the IBM or Apollo or the `EXIT` command on the VAX
7. Don't forget to remove your diskettes when finished!

Some Remarks

- The FTP implementations on various different computers are not identical. Please refer to the local help for usage details.
- Once you put your diskette into one of the PC's drives, it is accessible to anyone who connects to the PC via FTP (and several users can be connected simultaneously). For this reason we recommend that you write-protect all your source diskettes to prevent another user from accidentally overwriting or deleting your files.
- We have disabled write access to the hard disk in order to preserve the integrity of the PC.
- A last remark: Copying licensed software is generally prohibited.

A Few Examples

These examples demonstrate a few basic commands you will need in order to transfer files with FTP. In the examples, user commands are written in the standard font, FTP prompts in a type-writer font, and our comments and explanations in italics. The rather verbose FTP output is partially omitted.

FTP on the IBM

ftp ucopc	*connect to UCOPC via FTP*
USER (identify yourself to the host):	
userid	*give your IBM userid, no password required*
Command:	
put seq.file e:seq.txt	*copies an IBM sequential file onto the diskette in drive E:*
dir d:	*list the files on the diskette in drive D:*
get d:myfile myfile.pc (replace	*overwrites an IBM sequential file with a file from the diskette in drive D:*
cd d:	*changes the default device name to D:*
mget *.for	*copies all FORTRAN files on the diskette onto the IBM*
mput *.tex	*copies all TEX files onto diskette*
lcd my.lib	*local directory change on the IBM to my.lib, a library containing members*
mput *	*copies all members in the library my.lib onto the diskette*
	Note: This will NOT work for partitioned data sets created in the internal NEWLIB format.
binary	*changes the file transfer mode to binary (no ASCII/EBCDIC conversion)*
get e:binary.dat binary.dat	*transfers a binary file to the IBM*
quit	*close the connection and leave FTP*

FTP on the Central VAX

ftp ucopc	*connect to UCOPC via FTP*
Name (ucopc:userid):	
	simply press <RETURN>
FTP> put my.txt e:my.txt	*copies a VAX file onto the diskette in drive E:*
FTP> dir d:	*list the files on the diskette in drive D:*
FTP> get d:myfile.dat myfile.pc	*copies a file from drive D: onto the VAX*
FTP> set default d:	*changes the default device name to D:*
	Note: This command only works with a diskette in the drive
FTP> get *.txt	*copies all diskette files with a .txt extension onto the VAX*
	Note: The PUT command cannot be used with wildcards
FTP> set type image	*changes the file transfer mode to binary*
FTP> put binary.dat e:binary.dat	*transfers a binary file to the diskette*
FTP> exit	*close the connection and leave FTP*

Please always leave FTP on the VAX with the EXIT command. DO NOT use the DISCONNECT command, NEVER leave FTP with <CTRL> Y, and DO NOT allow the server to time out (this will happen if you don't enter any commands for 5 minutes). All these actions will cause the server to hang indefinitely.

FTP on the Apollo

The FTP version on the Apollo is very similar to the IBM version. The FTP prompts differ somewhat and the Apollo version also offers a wider range of commands. However, all the commands in the IBM example above are also valid on the Apollo (with the exception of the (replace parameter which isn't necessary for UNIX systems).

• The DESY Staging System

by Dieter Wiemann
and Michael Behrens

The following text is a modified and shortened version of Help STAGE on the IBM.

Following requests from the users, a staging system was developed and implemented on the DESY IBM. It became generally available in April after previous testing by selected users.

The principal idea of the staging concept is to copy tape/cartridge read-only data sets to disk before using them and then to use this copy of the data set on the disk. This is done automatically as far as possible. The disk versions reside on a staging disk pool for several hours or days according to their last use date/hour. Inactive data sets are deleted automatically in order to get free space for new stage requests. The tape data sets remain unchanged all the time.
Some of the advantages of the staging approach are:

- Parallel access to the same tape data set from independent jobs is possible after the data set has been staged.
- Jobs reading tape data sets can be active for many hours. Using staging, the tape/cartridge drive is only allocated during the copy operation, then it is freed for other jobs.

The DESY staging system consists of two parts: a staging subroutine and a permanently active staging task.

- The staging subroutine has to be called by the user in his program as described below ("The STAGE subroutine"). If the data set is already staged, the staging routine simply allocates the staged version of the data set, otherwise it stages (i.e. copies) the tape data set to a staging disk and then allocates the staged version.
- The permanently active staging task maintains all the staging volumes. It deletes staged data sets at certain intervals according to their age (last use) etc., but does no staging by itself.

In the foreground, already staged data sets are usable but no "real staging" (i.e. copying from tape to disk) is allowed.

Staging can be requested for tape or cartridge data sets. The term 'tape' can always be replaced by 'cartridge' and vice versa.

A Newlib interface is available.

The STAGE subroutine

Staging is requested by the call shown below. Programs calling STAGE may be written in any high-level programming language (FORTRAN, PL/1, PASCAL, C) or assembler. The parameters are similar to those for DYNAM.

```
Call STAGE (function, returncode,
                      reasoncode,
                   parmstring_1,
                            ...
                   parmstring_20)
```

The following functions are available:

'INPUT '	stage from tape to disk for reading
'DELETE '	delete staged data set
'QUERY '	give staging information

`Parmstring_n` contains staging parameters in the form:

`'keyword1=value1 keyword2=value2 ...;'`

Most parameters are optional, a detailed list of parameters can be found in **Help STAGE**.

Exceptional Conditions

STAGE passes back a return code and a reason code to indicate the success or failures of the stage request. Callers can request the level of message printing (parameter PRINTLev). If any printing of messages is requested, please supply a print file with the ddname STGPRT, otherwise messages will appear in the joblog (file JESMSGLG for batch) or directly on the screen for TSO.

Most of the reason codes have corresponding message-ids, i.e. reason code 0161 corresponds to

message-id STG0161c, where "c" marks the severity (**A**(ction), **E**(rror), **W**(arning), **I**(nfo)).
The return codes, reason codes, the messages and their meaning can be found under **Help STAGE**

An Example

Call from VS FORTRAN.
More examples in **Help STAGE**.

```
* STAGE a dataset from tape to disk,
* and allocate it as file FT20F001.
* It should stay for at least 5 hours
* on disk before being deleted.
*
      INTEGER*4 RETCOD,REACOD
      CHARACTER DSN*80
*
      DSN = 'DSN=tape.data.set.name ;'
*
      CALL STAGE('INPUT ',RETCOD,REACOD,
     +     DSN,'DDNAME=FT20F001 MINT=5;')
      IF(RETCOD .NE. 0) THEN
        WRITE(6,*)'Unsuccessful STAGE.',
     +            ' Retcod=',RETCOD,
     +            ' Reacod=',REACOD
        STOP
      END IF
*
* the data set is ready for processing:
*
      READ(20) ...
*
      END
```

Usage Considerations

Hints and Tips:

- Use CALL STAGE(...) for all allocations of staged data sets. This insures that last use date/time is always updated and that your data will not be deleted from the staging pool while being used.
- If you know that the staged version is only used for a short time please specify the MINTIME keyword. The staging system is then able to remove this data set as early as possible.
- If you know that a staged version is no longer needed by anyone please delete this version by

  ```
  call STAGE('DELETE ',...)
  ```

  or NEWLIB command

  ```
  STAGED DSN=...
  ```

  This ensures that the occupied space is freed and reusable by other stage requests.

Restrictions:

- only standard label tapes are supported
- only physical sequential data sets are supported
- only cataloged data sets are supported
- only single-volume data sets are supported

NEWLIB / STAGE interface

NEWLIB supports DESY-staging with general- and line-commands. For each stage function there is one command of the form "STAGEf", where f specifies the function (D=Delete, I=Input, Q=Query):

STAGED	delete a staged data set
STAGEI	allocate a staged data set for input if already staged. Otherwise submit a job that stages the data set.
STAGEQ	list staging information for a data set or a group of data sets with the same data set name level.

Parameters for the general commands are the keywords as explained above.
Parameters for the linecommands are taken from the line in which the command is entered. For the STAGEI linecommand the dsname field may be overwritten by a ddname, this is equivalent to specifying keyword DDNAME=ddname .
All staging linecommands can be given in LC and STAGEQ screens.

A detailed description of all parameters, messages etc. is available under **Help STAGE** *on IBM.*

• From 0 to X in five minutes

by Thomas Finnern

X-terminals will become the standard general purpose graphics terminal of the future. In principle X-terminals are small computers which, above all, know the TCP/IP network protocol and the X-window graphics protocol. The three major reasons for installing X-terminals are:

- Reasonable costs
- Every workstation "speaks" TCP/IP and X
- No fan (They really do work SILENTLY !!)

The current state of the installations made by the group R2 is:

30	NCD X-terminals
48	TEKTRONIX X-terminals
42	APOLLO Workstation Displays
7	HP Workstation Displays
127	Terminals for X Graphics

The workstation displays can be used in the same way as the X-terminals. Additionally, the workstations act as support servers for the X-terminals. The following should outline how X-terminals are booting and which prerequisites are needed before a user has a running session on the screen.

To serve or not to serve

The heart of an X-terminal is a program called XSERVER which runs locally in the terminal and listens on the network for graphic requests to be drawn onto the local screen. XSERVER also sends keyboard and mouse activities back to the graphic programs.

Before the XSERVER can start to work, it needs a number of other servers to perform various functions:

- a boot and configuration server
  This server tells the terminal its hostname and its ip number, which name server and which gateway it has to use, and which programs and which setup files it has to load from which file server. The protocols used here are BOOTP (Boot Protocol) for infos and TFTP (Trivial File Transfer Protocol) for loading data files.
- one or two font servers
  These font servers send image files with character fonts on request to an XSERVER. Currently each manufacturer supplies about 40 Mbytes of font data specially prepared for it's own terminals. TFTP is used to load the font data.
  All X-terminals and APOLLO displays support a set of font aliases to provide the user with uniform font names independent of the X-terminal or workstation display used. There are fixed, variable, and vt fonts in sizes tiny, small, medium, large, and huge, matched to the actual screen resolution.
  With the upcoming release 5 of X there will be the possibility of installing special font servers which have a set of outlined font families which can be converted dynamically to every weight, slant, and point size independent of the manufacturer.
- a login server
  X release 3 login servers have a list of X-terminals for which they should start a login session. X release 4 login servers are contacted by the X-terminal itself: The X-terminal is either configured to use a fixed login server or it broadcasts into the network to find one of the available servers. In future, the X-terminals should get the name of its login server indirectly by asking a special server for the actual name. On all these terminals a login panel will appear, displaying the name of the login server and requesting the user's login name and password.

The final step

Currently the login server acts as a user session server too. Additionally this server is the default CPU server during the actual user session.

The future might bring a modified algorithm where a user session is either started on the workstation with the user's home directory or on a worksta-

tion with more disposable CPU power. This would minimize network traffic and/or provide a better load-sharing for the CPU servers.

The following describes the login procedure for X displays connected to the APOLLO cluster.

After the user has successfully typed his login name and his password, a program called XSESSION is started in order to prepare the X display for the user:

- The individual UNIX user environment is set up so that all programs started at a later time know about the user's preferences, such as the default printer ...
- The display is loaded with network wide default values for standard X clients. These values are matched to the current size and resolution of the screen.
- The keyboard mapping is modified to provide the user with some kind of virtual terminal because the supported keyboards differ in their mapping, composition and layout of special keys.
- The session looks for network wide user default values and merges these into the systems defaults.
- The Motif window manager *mwm* will be started. Although mwm is mainly designed for managing the screen, it is configured to do some session manager tasks like starting terminal emulators (e.g. 3270 to the IBM).
- Some icons are placed at the bottom of the screen to help the user in starting standard clients like new terminal windows or a clock or any other X clients the user requested when setting up his UNIX environment. These icons do not need any resources from the server because they are only activated on a user's mouse click.
- A control program named *xctrl* is started. This program is a sort of session manager. It lets the user log out, may lock the screen for lunch, starts terminal emulators, or provides manual pages for help.
- The first active terminal program is started. This "login" window makes a notification to the system that the user has successfully logged in. Here the user receives notices for new mail, system news and other messages.

Give me an X

As shown in the above, an X-terminal is not a standalone terminal. Because the networks, the terminals, and the servers are occasionally unavailable or slow, some tasks like booting and logging in may take a little bit more time. Proper configuration should, however, minimize the interdependencies of the various support servers.

A well designed user session provides the user with a more or less virtual X-terminal independent of the manufacturer and the connected workstation. As the support for the X-terminals becomes more and more perfect, this type of terminal will stay as a general purpose, desktop, *plug-in-and-go* display untouched by the fast-changing workstation market.

• INTERLINKing VAX and IBM

by Katherine Wipf

INTERLINK is a hard- and software package, installed on DESY's IBM mainframe, which serves as a gateway between the DEC and IBM "worlds." Seen from a DECnet node, the DESY IBM appears as the node DSYIBM with the DECnet address 13.49. INTERLINK makes this possible by receiving all file access requests coming to the IBM from DECnet and translating them into a form recognizable to the IBM file system. INTERLINK can thus be used for performing VMS file operations (such as `$ COPY`, `$ DELETE`, `$ SORT`, ...) on IBM files via DECnet.

Prerequisites for using INTERLINK

For security reasons, user access to INTERLINK is restricted. If you wish to use INTERLINK, you must fulfill the following requirements:

1. You must have userids on both VXDESY and the IBM. In addition, your IBM userid must be registered under RACF (Resource Access Control Facility - a standard IBM security package).
2. You need a proxy definition linking your VAX and IBM userids. This definition gives you the same IBM file access privileges when manipulating files via INTERLINK as you would have if you were actually logged on to the IBM.

If you have a field marked `NEW PASSWORD ===>` in your IBM logon panel (where you enter your password), then your userid is registered under RACF. If this is not the case, you need to talk to your RACF administrator, who can also handle your request for a proxy definition. If you need an account on the VXDESY, your VAX group administrator can give you one. The article *Introducing the Central VAXcluster* in **DESY Computing Newsletter No.2** gives you a list of the VAX group administrators and the article *RACF Registration on IBM* in this volume contains a table of RACF administrators. You can always find an up-to-date list in `HELP RACGRPS` on the IBM.

INTERLINK Syntax

You don't need to call up a special facility in order to use INTERLINK. All changes to remote IBM files are made with the standard VMS file manipulation commands. In these commands, IBM files are specified according to the VMS rules, with the node name DSYIBM. For example, the specification:

```
DSYIBM::[F99ABC]FILE.NAME
```

refers to the IBM file

```
F99ABC.FILE.NAME
```

If the IBM file belongs to you, you can leave out the [userid]. If the IBM file name would not be valid on the VAX (this is the case with members of a partitioned dataset, with generation data groups (GDG's), and with files that have more than two qualification levels) it must be enclosed in quotation marks. Here are some examples:

```
DSYIBM::"[F99ABC]FILE.NAME(MEMBER)"
DSYIBM::"[F99ABC]FILENAME(+1)"
DSYIBM::"[F99ABC]LONG.FILE.NAME"
```

Quotation marks are also necessary if you wish to use any of the INTERLINK options. Note that INTERLINK options must be placed INSIDE the quotation marks, whereas options for the command you are executing (i.e. - options for a `$ COPY` or `$ BACKUP`) appear outside the quotation marks. The INTERLINK options in the following example will be explained later:

```
$ COPY MY.TXT
_To:  DSYIBM::"MY.TXT/CARD/FF"/LOG
```

Copying Files between VXDESY and the IBM

When you copy a VAX file to the IBM, you can only copy one file at a time. The target file on the IBM must be sequential (it cannot be a member of a partitioned dataset). Since the IBM does not support multiple versions of a file, you must always use the copy command with replace option (`$ COPY/REPLACE`) in order to overwrite IBM files.

When copying IBM files to the VAX you have a little more freedom. You can copy several IBM files at once into a single VAX file. You can also

copy members of a partitioned dataset, provided the dataset was *not* created by NEWLIB (see the section *Copying NEWLIB Members with the COSY Command*). Here are a few examples:

```
$ COPY MY.FILE DSYIBM::[F99ABC]PUB.FILE

$ COPY DSYIBM::PART.ONE,DSYIBM::PART.TWO
_To:  TWOPART.TXT

$ COPY DSYIBM::"CARD.S(MEMB)" VAX.TXT
```

Interlink offers a large number of options which can be used with the COPY command. Some of the more useful ones are listed below and a complete list can be found in `$ HELP INTERLINK` on the VAX. The last example of the section *INTERLINK Syntax* above shows you how to incorporate the INTERLINK options into your `$ COPY` command.

`/ALLOC:n` Indicates the number of 512-byte blocks to be allocated on the IBM for the file being copied. The maximum possible value for n is 99999999.

`/BINARY` Indicates that the file consists of binary data (no ASCII/EBCDIC translation) and that no implied record delimiters (carriage return/line feed) are present.

`/CARD` Forces the record format of the IBM target file to be card image (this option has the same effect as specifying `/FIXED:80` and `/EXPAND` for a file in character format).

`/EXPAND` Expands VAX tabs into blanks, assuming the DEC standard tab stops of 9,7,25,....

`/FF` Converts form feed (<FF> and <VT>) characters into carriage control top-of-page information. This option can only be applied to a VAX to IBM transfer and produces an ASA file (must be printed from the IBM with the `OVFL ONA` option).

`/FIXED:n` Forces the records of the new file to be of fixed length n, regardless of the source file's attributes. Records are truncated or padded to the specified length. If the length n is not specified, it defaults to the maximum record length of the source file.

`/IMAGE` Indicates that the data should be transfered as a binary image with no translation performed. When transfering from the IBM to the VAX, the carriage-return-carriage-control record attribute will be assigned.

`/STRIP` Removes trailing blanks or nulls from each record in the new file. If the file is fixed, it is padded to the maximum record length after `/STRIP` processing. If the file is variable, `/STRIP` will set each record length equal to the length of the record with trailing blanks and nulls removed. The new file's logical record length (LRECL) will be the maximum record length of the source file.

`/VARIABLE:n` Forces the new file into a variable length record format, regardless of the source file's attributes. The value n defines the maximum record length for the file. If n is not specified, it defaults to the maximum record length of the source file.

Copying NEWLIB Members with the COSY Command

Libraries created by NEWLIB have a special compressed (COSY) format which saves space. Unfortunately it also makes NEWLIB members very difficult to read for non-NEWLIB applications. In order to solve this problem for INTERLINK, Mr. M. Landon of H1 created the `COSY` command. This command copies a NEWLIB library member to the VAX using a similar syntax to the `COPY` command. The

chief difference in the syntax is that COSY command options are placed after rather than within the quotation marks enclosing the IBM file name. A COSY command with options would look something like this:

```
$ COSY DSYIBM::"PROGS(MAIN)"/FIX/NOHEAD
```

It is not necessary give an output file name, although it can be specified if required. If the output file name is missing, COSY will construct a name from the member name (stripped of any invalid characters) and the language flag in the NEWLIB header line. Thus the NEWLIB member MYPROG in the language FVS would become MYPROG.FOR on the VAX. The default name COSY is used if the member name cannot be used and if the language flag cannot be determined, the default type is TXT. The options used in the example above have the following effect:

/FIXED	By default the output VAX file will have variable length records with trailing blanks removed. If you absolutely must have a card image output file, specify /FIXED.
/NOHEADER	Controls whether the NEWLIB header line (if present) is copied to the output file. The default is to keep the header line. Use /NOHEADER to prevent it from being copied.

The COSY command has a few other options which are described in $ HELP COSY on the VAX.

Making VMS Backups on the IBM

INTERLINK has a very useful feature which few people take advantage of. It offers the possibility of making VMS backups (either of the entire directory structure or of selected files) and storing these backup files on the IBM. This eliminates the necessity of handling and storing your own VAX backup tapes.

When making backups on the IBM, you must use the INTERLINK /IMAGE option so that the backup files are stored without any ASCII to EBCDIC translation. Use the /SAVE_SET qualifier so that the remote file will be treated as a BACKUP save set. It is generally also a good idea to use the /BLOCK_SIZE qualifier to change the default block size of 32,528 bytes. The following example creates a backup of a user's directories and files on the VAX:

```
$ BACKUP/LOG [...]  -
_$ DSYIBM::"VAX.BCK/IMAGE" -
_$/SAVE/BLOCK=8192
```

Once you have created a BACKUP save set, it is possible to obtain a list of the files it contains with the following command:

```
$ BACKUP/LIST -
_$ DSYIBM::"VAX.BCK/IMAGE"/SAVE
```

The files can also be restored to a subdirectory with the BACKUP command as follows:

```
$ BACKUP DSYIBM::"VAX.BCK/IMAGE"/SAVE
_To:  [.RESTORED]
```

Using VAX editors on IBM Files

For people who prefer one of the VAX editors to NEWLIB or ISPF, INTERLINK offers an interesting alternative. You can edit any sequential text file cataloged on the IBM with your favorite VMS editor (EDT, EVE, LSE, ...). The only restriction is that since MVS doesn't support multiple file versions, you must specify a new file name when you want to save your editing changes. You can start an editing session for an IBM file with a command similar to this one:

```
$ EDIT/EDT DSYIBM::MY.FILE
```

When leaving the EDT editor, you must specify a new file name like this:

```
*EXIT DSYIBM::MYNEW.FILE
```

If you are using the EVE or LSE editors, you should give the command:

```
WRITE DSYIBM::MYNEW.FILE
```

before leaving the editor.

Other Useful Commands

Quite a number of the VMS file manipulation commands can be applied to IBM files with the help of INTERLINK. A detailed description of these commands can be found in the *INTERLINK User's Guide* which is stored in the IBM file `RØ1UTL.INTERDOC`. You can print it with the following command:

```
$ LPRINT DSYIBM::"[RØ1UTL]INTERDOC"
```

A number of the more useful commands are briefly explained below:

- `DELETE` - You can delete any IBM file for which you have delete permission with a VMS `DELETE` command from the VAX. In order to conform with the VMS `DELETE` syntax (with explicit version numbers) you must add a ';' after all DSY-IBM files which are not enclosed in quotation marks. Thus `$ DELETE DSYIBM::MY.FILE;`
  NOTE: If the IBM file has been migrated, the `DELETE` command will take a VERY long time.
- `DIFFERENCE` - This command is very useful for comparing two files when one of them is stored on the IBM. It produces an output file listing any differences.
- `DIRECTORY` - The `DIRECTORY` command gives you a list of datasets cataloged on the IBM. You cannot obtain a list of the members in a partitioned dataset with this command, just of the datasets you would also see with the NEWLIB `LC` command. `DIRECTORY/DATE` can be useful for determining if a file has been migrated - a MIGRAT file will return the information `<None specified>`.
- `SEARCH` - Use the `SEARCH` command to search one or more IBM files for a specified string or strings. This command will only work on sequential files and members of partitioned datasets which were *not* created by NEWLIB.
- `SORT` - The `SORT` command will invoke the VAX Sort Utility which reorders records in the input file as directed and creates a new output file containing the reordered records. Again this command will not work on members of partitioned datasets which were created by NEWLIB.
- `SUBMIT/REMOTE` - This command can be used to submit batch jobs to the IBM. The files you submit must contain a complete JCL job including a jobname and account number. If you include the following JES3 statement:
  `//*MAIN ORG=UILØ1`
  you will be able to inspect your job output with the JES utility on VXDESY (see chapter 7 of the *INTERLINK User's Guide* for details).

You can use wildcards with many of the commands above, but they may not work quite the way you expect them to. You should read the explanation of the MVS catalog structure in section 4.2.8 of the *INTERLINK User's Guide* before relying on wildcards. INTERLINK also offers the possibility of accessing VSAM datasets, of using lexical functions to gain information about IBM files, and of reading IBM files within DCL command procedures. Since these operations are a little more complex, they are not covered in this article. Details can be found in *INTERLINK User's Guide*.

• How to Teach Yourself VAX/VMS

by Katherine Wipf

A couple of Computer Based Instruction (CBI) courses are now available on the central VAX cluster. These are programs which teach users at the terminal by providing instruction and practice exercises. At the moment there are two courses available:

- Introduction to VMS
- Using the VAX Language-Sensitive Editor

How to use CBI Courses

You can start a CBI course at any time and work through as much or as little of it as you like. To start a course for the first time, give one of the following commands:

```
$ RUN VMSENG   to start "Introduction to VMS"
$ RUN LSECBI   to start "Using the VAX LSE"
```

After you have read the "Welcome" panel you will be prompted to press <RETURN> if you are a first time user. In the next panel you are prompted to type in your first and last name and then asked for a code name with 1 - 6 letters. Once you have completed these steps, you are registered for the course. From now on, whenever you want to continue in the course or start a different CBI course, you only need to type in your chosen code word.

Once you have finished the registration/identification process you are asked if you are taking the course for the first time. Answering Y will lead to a course introduction and overview which sometimes includes a pre-test. If you answer N you will come directly to the course Main Menu from which you can begin with the course by choosing a topic.

If you have any trouble with or questions about one of the courses, please ask Katherine Wipf (Email VXDESY::WIPF, Tel. 3222) or come to the UCO.

New Courses

We have a German version of the *Introduction to VMS* course which we hope to be able to install soon. In addition we have 3 courses which consist of student instruction- and workbooks (they are not computer based) on the following topics:

- VMS Utilities and Commands
- Advanced VMS Command Procedures
- Utilizing VMS Features from VAX C

Anyone who is interested in one of these courses should contact Katherine Wipf (Email VXDESY::WIPF, Tel. 3222). DEC offers quite a variety of courses on various topics and we can order a few more if there is enough interest. You can come to the UCO to look at the course catalogue and tell us which courses interest you.

• XFIND/WWW on the DESY IBM and Central VAX

by Peter Dobberstein
and Michael Behrens

XFIND

XFIND is a keyword-based approach to accessing Help- and other information. It was created and is being developed at CERN. The XFIND service at DESY runs as a client on the IBM or VAX and uses the normal XFIND server at CERN. The DESY specific data - mainly from NEWLIB Help - are prepared at DESY and copied to the CERN VM system at certain intervals. Afterwards XFIND uses these data over the network from CERN. XFIND is callable on the IBM and the VAXes with the command

```
XFIND keyword1 ...keywordN
```

where keyword1, ..., keywordN are the topics which are simultaneously searched for (logical AND). On the VAXes, the number of keywords is limited to 8.

```
XFIND TAPE
```

generates a list of all information belonging to the keyword TAPE. If you are only interested in how to copy a tape, you should try the call

```
XFIND TAPE COPY
```

to get a (shorter) list of the information belonging to the keywords TAPE and COPY simultaneously.

The command XFIND mentioned above is, technically speaking, a short-cut saving you from going through the initial panel of WWW each time.

WWW

WWW (World Wide Web or w3) is a vehicle for making data residing on one or more servers accessible to clients on various computer systems connected to these servers via INTERNET. WWW uses the HYPERTEXT approach explained later in this article and is being developed at CERN.
At DESY we installed line-mode WWW on the IBM and on the central VAXes (VXDSYA, VXDSYB, VXDSYC) to give an uniform access to XFIND- and other data (a NEWLIB full-screen version and an X/MOTIF version are planned for the future). Additionally, we filled most of the help information stored on our IBM into the XFIND database to make it usable on IBM and VAX.

WWW is callable on the IBM and the VAX with the command

```
WWW
```

and then should present information on how to proceed. As part of the available data you will find the CERN phonebook, the XFIND data of CERN and DESY, some NEWS, the hierarchical VAX help (currently this is the Help for VXCRNA, not VXDESY) and useful information about remote procedure calls (RPCs). In contrast to XFIND, WWW displays information from many sources. Therefore the maintenance level may be different for different information sources.

What is Hypertext ?

Hypertext is a method to maneuver trough a set of documents with less restriction than in the traditional tree-like structure. The documents of a hypertext system are connected at a number of anchor points defined by the author of each document. By "pointing" on an anchor point you enter the document "behind" that anchor. You may proceed this way as often as you like and as long as you find new anchor points, and you can go back as many levels as you like exploring the full system.

This HYPERTEXT approach is realized in the XFIND/WWW system in line-mode by showing the anchors an numbers included in brackets ([n] or <n>) at all anchor points to another document. By entering the desired number one can display the new document and so "navigate" through all the documents available. For a document/listing longer than one screen page, the next pages are shown by a repeated pressing of `RETURN` or `ENTER` in VT-terminal style. The most important commands available in the actual XFIND/WWW-version are the following (valid in upper and lower case as well as mixed):

B(ack)	Return to the previous document
H(elp)	Explain the commands available at the present level,
HOME	Return to the first document shown
K(eyword)	Search document(s) belonging to the following keyword(s){only applicable if indicated in the prompt line}
Q(uit)	Leave XFIND/WWW
T(op)	Display the first page of the present document/listing (node)

More XFIND details

XFIND details

Most users of CERN's IBM surely know the XFIND system, which gives an easy access to a lot of information. Contained are helps belonging to CERN's VM system, CERN's CRAY, computer newsletter articles, writeups from different sources and many other data. In 1991 an additional data group, DESY, was added, which contains most help information available at DESY's IBM mainframe.

XFIND is a keyword-based system. It does not require the user to select the proper name of a search item directly, which is sometimes difficult to guess. It is also not a tree-like system as on the VAX where you have to correctly guess the path that leads to your destination. Instead, many keywords are provided for each item, so that hopefully many users with different ideas of the "name" or their search item can be served properly. In addition, a keyword based approach is in principle open, so that more keywords can be added at any time if needed.

Nevertheless, there is an internal structure of the data base, which is useful to know in order to get an overview of the different kinds of data stored. The data are spread over several groups, of which

DESY
PUBlic = CERN PUBLIC
USER = CERN USER

are the most interesting for DESY. By default, these disks are searched through for each request. The group DESY comprises most helps accessible on the DESY-IBM under NEWLIB, PUB covers the information about CERN's central computers and connected areas, whereas USER contains some useful, but not officially supported topics concerning the CERN computer centre.

For each group of data substructures exist,they are called categories. For group DESY the following structures exist currently:

FUTURE	Future versions of helps
HELP	IBM commands and clists
INFO	Miscellaneous INFOrmation
NEWLIB	Helps belonging to NEWLIB
OBSOLETE	Items remembered but obsolete

WWW services

Since WWW is intended as a general tool for accessing information servers, the number of available servers is expected to increase in the future.

Help	Gives information about WWW (usage as well as in depth details and installation instructions).
Phone Book	Contains actual extracts of existing CERN databases which are used for the administration of the CERN phone book, the accounts on CERN computers and the CERN Yellow Pages (the last in English and French).
XFIND index DESY docs	Via these anchors one can access the data directly available with the XFIND command. 'XFIND index' covers the XFIND groups PUBLIC and USER, 'DESY documents' covers the XFIND group DESY. In contrast to the direct usage of XFIND, where all three groups DESY, PUBLIC and USER are searched through (in this order), here one can search through the group DESY separately.

News	News information available from the CERN XNEWS system, organized into various groups.
VMS help	A complete tree of the VMS help is available in a hierarchical organization. No keywords available.
CERN RPC	User guide for the RPC system developed by CERN-CN (not Sun/RPC). This documentation is (partially) converted into hypertext.

Both on the IBM and on the VAX detailed examples of XFIND and WWW sessions are provided with `Help XFIND`.

Abnormal Situations

WWW is a network application involving different computers and networks. Therefore, its availability depends on the availability of all participating components. If XFIND/WWW doesn't answer, in most cases the server machine is in maintenance or the network line is down: try again later. If a special error (e.g., a breakdown of the line after a certain operation) appears repeatedly, you should inform the DESY User Consulting Office (`RO1UC0@DHHDESY3`). The same applies if the program reports an error.

• GEP Version 5.0

by Erich Bassler

After many years a new version of GEP has been completed and is available for tasting.

Analysis of Requirements

Since the beginning of the 80's we have recorded the size and the number of distributions of each data base generated by a GEP batch jobs. From this data we observe some shortcomings: a sharp cut at 500 distributions, which was a limit defined by the library structure of the old GEP, indicating a request for more distributions in the data base. In addition we observed databases with more than 50 Mbytes, much larger than the default values, which had been overwritten by the user. Comparing HERA physics with PETRA physics, we expect larger data bases with more complexity.

There was a further shortcoming of the old GEP architecture: the final storing of the data base after filling. Since the final data base was only stored after completion (of the GO-step), it occasionally happened that a batch jobs ran several hours and then found no disk space for the final data base. To avoid this problem, it is reasonable to require the possibility of writing out during filling and of updating the data base from a check point set by the user. Furthermore it is nice to have the option of an early look into the data bases during the filling.

Consequences for the Data Management

Because we expect larger data bases, we studied the data management and the I/O (especially what happens when a new data base is opened). One design criteria of the old and new GEP was and is the assumption that the data base can be larger than the available memory space. If only binned distributions are stored in the data base, the estimation of its actual size is easy. The difficulties in estimating the actual size of the data base arise if single entry recording (e.g. unbinned histograms for dynamic

binning, scatter plots, physical polylines (field lines), and ntupels) is allowed.

If FORTRAN direct access is selected for the data base (ANSI FORTRAN without tricks in order to have good portability), we measure results on the IBM main frame which we could not accept if we want to offer reasonable default values for the data base size. The reason is that FORTRAN direct access does not allow any extents on the data set. We have found that index-sequential I/O (keyed access) is a better solution. It is extendable on the IBM main frame, supported on DEC VMS and the OS/2 operating system, and is part of the X-OPEN Guidelines since 1987. The I/O type allows direct access to each record of a data base with variable length records. For multiple write out during filling, it allows the user to extend the distributions (for entry recording) and/or overwrite distributions (for binned data). Making these I/O changes was a major task. It required much more than the just substitution of one WRITE statement for another.

Changing the Architecture of GEP

We have restricted our task of changing the GEP architecture to the IBM main frame and VAX VMS computers, keeping portability in mind. This was done with the intention of using these computer types as a model. We believe that with this model we have met all possible problems: hidden ASCII-EBCDIC dependencies, different floating point presentations, high-endian/low-endian architecture of the processors, the allocation problems, different handling of key-stroke interrupt (with respect to the full screen I/O), and different language extentions. For GEP Version 5 the entire architecture has been changed. The layout of the internal banks and record formats has been modified. Each record now has a format descriptor which allows programs running on the VAX to read from and write to a GEP data base on the IBM using the INTERLINK software.

As in the past, the user only has to specify the data base name. The allocations and the I/O routines are hidden from the user. There is also no need for the user to calculate the amount of memory needed for workspace.

What is New ?

For GEP Version 5 no special GEP-procedure is necessary (see example below).

Example of a GEP-batch-job on the IBM:

```
//f99abc   JOB '12345678',CLASS=K,TIME=2
//A EXEC VFORTCLG,GPRM='f99abc.RNDMV',
//     LLB1='R02BAS.GP.L',
//     LLB2='R01UTL.CERN.PRO.KERNLIB'
//C.SYSIN DD *
       CALL RDMIN(1)
       X1 = 1.
       X2 = 1.
       CALL GEPIC
       DO 1 I = 1,10000000
       X=RNDM(DUMMY)
       IF(X.LE.0.001) CALL CORR(2,X1,X2)
       X2 = X1
       X1 = X
 1     CONTINUE
       CALL GEPW
       STOP
       END
//G.SYSIN DD *
MPRINT
DST 2 'Random numbers ?
ENDQ
```

The new version is almost fully backwards compatible. (Exception: Old data bases cannot be used because of missing information for updating some of the distributions. These data bases can be partly converted using the EXPORT-IMPORT-facility of GEP.) The number of distributions is now extended to < 100000. There are three classes for the size of the data base: small < 4 Mbytes (default), medium < 40 Mbytes and large ≥ 40 Mbytes. The user can write out to the data base and then continue filling, and can have a first look into the data base while the batch job is still running. Furthermore two new types of distributions have been introduced: the efficiency

histogram EHIST and the VTUPEL, which is similar to an NTUPEL with entries of variable length.

Consequences for the Interactive Part

The interactive part of GEP has also been revised. Colour has been introduced, as well as dynamic binning of unbinned histograms and the analysis of VTUPELs. On the IBM there are 2 versions available (GDDM-version load module size: 1900 kbytes + 1400 kbytes workspace,[1] GKS-version load module size: 3900 kbytes + 1400 kbytes), which differ in the handling of the full screen I/O. The generation and maintenance of these 2 versions is not a major task. It was the revision of the architecture which took the time.

The GDDM version is implemented for the IBM-3270G-terminal and its emulations. This version allows graphical input e.g. for positioning of text strings, but hard copy is only possible on the IBM 3820 laser plotter (LI1).

The GKS version is implemented for all graphic-terminals supported by the old GKS version. This version does not allow graphical input on the IBM-3270G-terminals or Falco-terminals (because of the fact that it is not possible to clear alphanumeric and graphical I/O separately). Hard copy, however, is possible on all the plotters supported by the old GEP.

Language Considerations and Portability

One important question was: is it reasonable to continue to require that a fit must be correct not only in the graphical sense but also in the mathematical sense? (This requires good exception handling, but on the other hands languages with good exception handling are not available on all hardware platforms.) It is a question of feeling responsible for the fit results. A single user on the the IBM can, on his own responsibility, switch off exception recording in his program (in FORTRAN by CALL ERRSET). It is different in a generally used application program, where the user cannot decide this.

[1] The workspace can be extended in steps of 1 Mbytes.

We continued in using PL/I for the GEP code. As in the past, this will not be noticeable for the user. PL/I has good exception handling and we have made good experiences with this language. We know that there are some restrictions in the portability of PL/I, but PL/I is an ANSI language which has a new standard since 1987. PL/I is also an SAA language since 1989, and is available under DEC VMS and on several UNIX-platforms for e.g. i386, i486, SPARC, RS6000, M88000, some M860X0 etc.. Among the customers using PL/I under UNIX are such well known companies as AT&T, General Electric, Ford, EDS, Grumman, American Airlines, Stanford University, Great Western Life Insurance, and the U.S. Naval Observatory.

There are many language independent portability restrictions. For example, it is unreasonable to study a scatter plot with 100000 dots via a slow network which cannot be controlled by the application programs. We believe portability is more a question of the task itself than of the language used for the program which should perform the task. Using standards (like GKS or PHIGS) does not solve the portability problem. We observe that the interpretation of a standard is unfortunately not standardized. Workstation-identifiers, line-width, and text font identifiers have no standard, they are dependent on the selected software package. A similar situation can be observed by analyzing the different Tektronix escape sequences.

Future of GEP

We believe that the redesign of GEP has improved its functionality and portability and is a good basis for future developments. The interactive part of GEP on the VAX and the access via INTERLINK to the IBM will be available soon. The IBM-version is available through the NEWLIB-CLIST GEP.

• Questions and Answers from the UCO

by Katherine Wipf

The User Consulting Office answers quite a variety of questions, but usually only a few people hear the answers. This column tries to bring the answers to more frequently asked questions to a larger audience.

. . . about NEWLIB

◦ *Question:* I have repeatedly changed my telephone number with USERLIST after being told by my colleagues that my number is wrong under INQUIRY. When I call USERLIST I see my correct number, but INQUIRY still gives my old number. How can I change it?

◦ *Answer:* INQUIRY now displays the combined data from the USERLIST and the DESY telephone book. Each night a job runs which merges the two lists. This is why changes made with USERLIST do not appear under INQUIRY until the next day.

◦ *Question:* I'd like to know how it is possible to create a member from an output file (with HBOOK histograms) that has 132 columns?

◦ *Answer:* You can copy most sequential files (such as HBOOK FZ-files) into members, but if the original file has lines longer than 80 columns, they will be folded and a funny character (¢) will be placed at the end of each incomplete line. This is a concatenation character which allows the lines to be stuck back together when you list or print the member. Unfortunately when you edit the member the lines remain folded (which makes editing rather tedious).

If you create a special partitioned dataset, however, you can edit members with up to 133 columns. The following command will create such a partitioned dataset with the name W133.S:

```
ALLOC DA(W133.S) DIR(10) LRECL(133) -
  BLKSIZE(11438) RECFM(F B) SP(10) TR
```

You can start a normal NEWLIB session with this dataset and copy in any sequential file (with up to 133 columns) with the command:

```
GET seq.file INTO membname SEQ
```

. . . about IBM Batch Jobs

◦ *Question:* Sometime I have the following problem: I submit a job which reads from or writes to a datasets. After it starts running I keep getting the message:

```
JOB xxx NEEDS aaa.bb
```

where xxx is the name of my job and aaa.bb is the dataset which the job reads or writes. What does this message mean? What am I supposed to do?

◦ *Answer:* The message means that your job cannot use the dataset mentioned because you already have it allocated in your interactive session. You need to give the command ALC, which will show you a list of all the datasets which you have allocated. Find the dataset in question and write FREE in the field to the left of the name. Another possibility is to simply give the command `FREE DATASET(aaa.bb)`. Occasionally it can happen that another user has allocated the dataset your job needs (possibly by listing the dataset). You will notice this by your job taking *much* longer to run than usual. The other user will be getting the `JOB xxx NEEDS aaa.bb` messages, but if he doesn't respond, your job will simply wait for the dataset. You can give the command:

```
ENQDSN aaa.bb
```

in order to find out who has allocated your dataset and then ask him to give the FREE command.

◦ *Question:* Sometimes I would like to have both my compiler listing and my job-output printed together, so that just releasing the 'FT06F001'-part of the job is not sufficient. When I use The 'release-all' command however, there are always several pages of 'obscure' code between the 'CL' and the job-output. How can I avoid printing it?

◦ *Answer:* You should use the LOCate line command for the job you want printed. You will

see a listing of all the files produced by your job. Now you can give the RELease line command for all the files you are interested in (in your case SYSPRINT STEP C and FT06F001 STEP G). You should remember to PURge the job output afterwards, because it will not be purged automatically as it would be after a RELease of all the files.

◦ *Question:* I have a job which reads one of my colleague's datasets. It used to work perfectly, but now my colleague is under RACF and my job doesn't run anymore. What can I do?

◦ *Answer:* RACF (Resource Access Control Facility) provides a much better protection of our user's data than we used to have. In particular, it prevents other user from writing on your datasets unless you explicitly give permission (by making your dataset into a group or public dataset). This protection makes it a little more difficult to read other peoples datasets in batch, because both the JCL DD statement with DISP=SHR and the Fortran OPEN statement allocate a dataset for reading and writing (which RACF does not allow). This means that you must always specify your intention to read the dataset. In Fortran this is done with the ACTION='READ' parameter (which is not standard Fortran). Here is an example:

```
      OPEN(UNIT=3,FILE='/userid.his.file',
     * ACTION='READ')
```

If you are allocating your dataset with a JCL DD statement, you need to add a LABEL=(,,,IN) parameter (the three commas are not a misprint - you really need them). Your DD statement might look something like this:

```
//G.FT03F001  DD  DSN=userid.his.file,
//              DISP=SHR,LABEL=(,,,IN)
```

If you must use standard Fortran, you will have to provide a DD statement like the one above for all foreign datasets.

. . . about other Topics

◦ *Question:* Very often when I want to log on to the IBM through MICOM, I am placed in the waiting queue at a position around 10. That implies a wait of about 20 minutes until I can connect to the IBM. Can MICOM be enlarged so that these waiting times can be avoided? Is there a better way of connecting to the IBM?

◦ *Answer:* The MICOM switch is already at the hardware limit. There is no way of increasing the number of sessions MICOM can handle. There is, however, a way of connecting to the IBM via the MICOM option LAT instead of IBM. This option connects you to a terminal server which can handle up to to 256 sessions.

Select LAT, press <RETURN> twice, and you will get the prompt

```
MIClocal>
```

answer with

```
MIClocal> c ibm
```

and follow the instructions for connecting to the IBM via TCP/IP.

◦ *Question:* I need to find out the E-mail address of someone working at CERN. Is there any way of getting this information at DESY?

Answer: There are several methods of finding out CERN userids. One possibility is to use the World Wide Web (WWW), a server-client-structure which makes data residing on various server machines (including one at CERN) available to its clients. Both the central VAX and the IBM are WWW clients. To find out a CERN userid with WWW you must do the following:

1. Give the command `WWW` on the central VAX or the IBM.
2. Type in the number 3 to select the CERN phone book.
3. Type in the letter K followed by a blank and the surname, first name, or phone number of the person whose userid you need.
4. You will now see a list of all the names which match the string you gave. Each name is followed by a number so that you can select the person you are interested in.

5. The address given behind the word `Email=` is the person's prefered mail address.

If you are working on the IBM you can also find CERN userids by giving the following command:
`TELL EMDIR@CERNVM QUERY surname`
The answer comes with a certain delay depending on network activity.