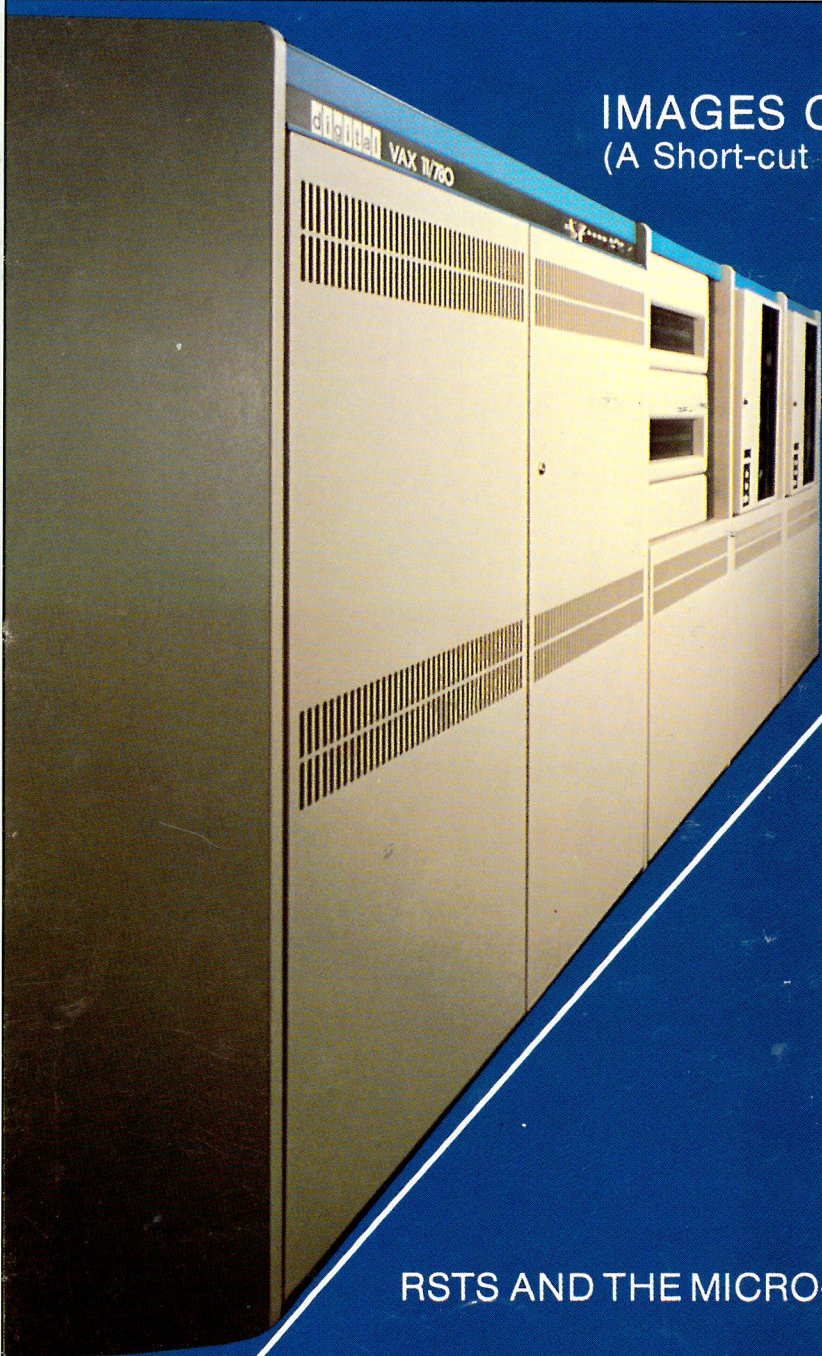


VAX RSTS PROFESSIONAL

Volume 5, Number 4

August 1983

\$10⁰⁰/issue, \$35⁰⁰/year

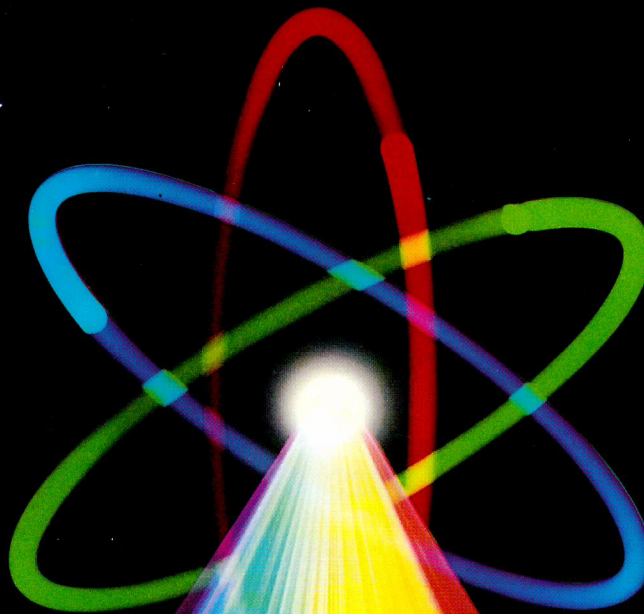


IMAGES COMMA SHAREABLE
(A Short-cut Through VMS Manuals)



RSTS AND THE MICRO-11

NOW AVAILABLE FOR VAX



USER-11: POWERFULLY PRODUCTIVE.

People productivity. It's more important than ever. And a good database system can mean *real* productivity.

USER-11 is a high-performance database system.

It is a fact: Software designed with USER-11 is built more quickly, operates more reliably, and performs better than other software techniques.

USER-11 is unique. It's easy to install. Easy to learn. And easy to apply. Adaptive tools and a standard approach ensure that maintenance is easier than ever.

A key to USER-11's success is its powerful, dictionary-based modules. Software developers simply describe and assemble these modules to create custom business packages—at an unprecedented rate.

Naturally USER-11 is supported with excellent documentation and a variety of training options for beginner to expert. Our commitment is to your complete satisfaction.

Whether you are a software provider or a software user, we guarantee you will be delighted.

Ask us about USER-11 and our family of business software products, or better yet, ask a *productive* USER!



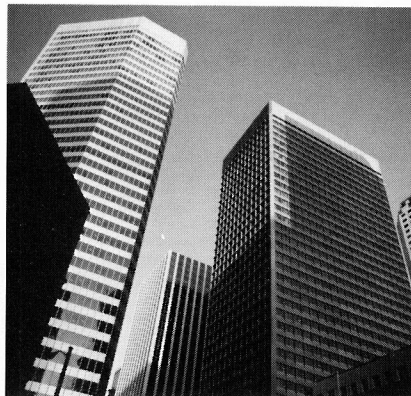
North County
Computer Services, Inc.
2235 Meyers Ave.,
Escondido, California 92025
(619) 745-6006, Telex: 182773

*USER-11 is currently available for DEC computers using the RSTS and VMS operating systems.

©NCCS 1983

CIRCLE 30 ON READER CARD

Introducing
the Most



POWERFUL

General Ledger Software Solution for the VAX

What is MAPS/GL?

MAPS/GL is a comprehensive general ledger/financial management system designed for DEC's VAX series of minicomputers. General ledger accounting, financial control, budgeting and advanced reporting functions are incorporated in MAPS/GL, and integrated with decision support to form a state-of-the-art financial management software system.

Standard general ledger functions are only the beginning. MAPS/GL includes many advanced features: a powerful, user-friendly report writer; extensive audit trails; three levels of security; capabilities for custom allocations and complex consolidations; and more.

All the Flexibility You'll Ever Need

You can tailor MAPS/GL to your individual needs, controlling accounting structures, organizational roll-ups and

reporting specifications. As your company grows and your needs change, you can change your system to adapt.

Its powerful but easy-to-use report writer brings flexibility to your daily needs as well. Ad hoc reporting from MAPS/GL's financial database can solve your special information needs.

Truly Interactive Processing

MAPS/GL was designed for today's powerful, interactive minicomputers—not adapted from an out-of-date mainframe-oriented batch system. Inquiry, updating and reporting functions can be performed online, right now. So your accounting staff becomes more productive. And your information much more accessible to management.

Integrated with Decision Support

Best of all, MAPS/GL can be linked with Ross' easy-to-use packages for financial modeling and business graphics, MAPS/MODEL and MAPS/GRAPH. Together these packages form an integrated financial

management system, including micro-computer-based distributed computing power with our MAPS/PRO package.

More MAPS from Ross Systems

The MAPS™ (Management, Accounting, and Planning Software) family includes MAPS/AP for accounts payable, MAPS/DB for database management, and MAPS/ISO for employee stock option tracking. All Ross products are available for license or via ROSS/NET™, our VAX-based remote computing service.

And our commitment includes customer and product support that reflects 11 years of consulting experience.

MAPS from Ross Systems. The single source for integrated, interactive financial management software.

DEC and VAX are trademarks of Digital Equipment Corporation. MAPS and ROSS/NET are trademarks of Ross Systems, Inc.

MAPS/GL was developed by Price Waterhouse under the name "FM80" and adapted for the VAX by Ross Systems, Inc. Our license includes the support necessary to keep MAPS/GL current with new FM80 releases. Price Waterhouse markets and supports FM80 on computers other than VAX.

MAPSGL™

from Ross Systems

Financial Software Solutions

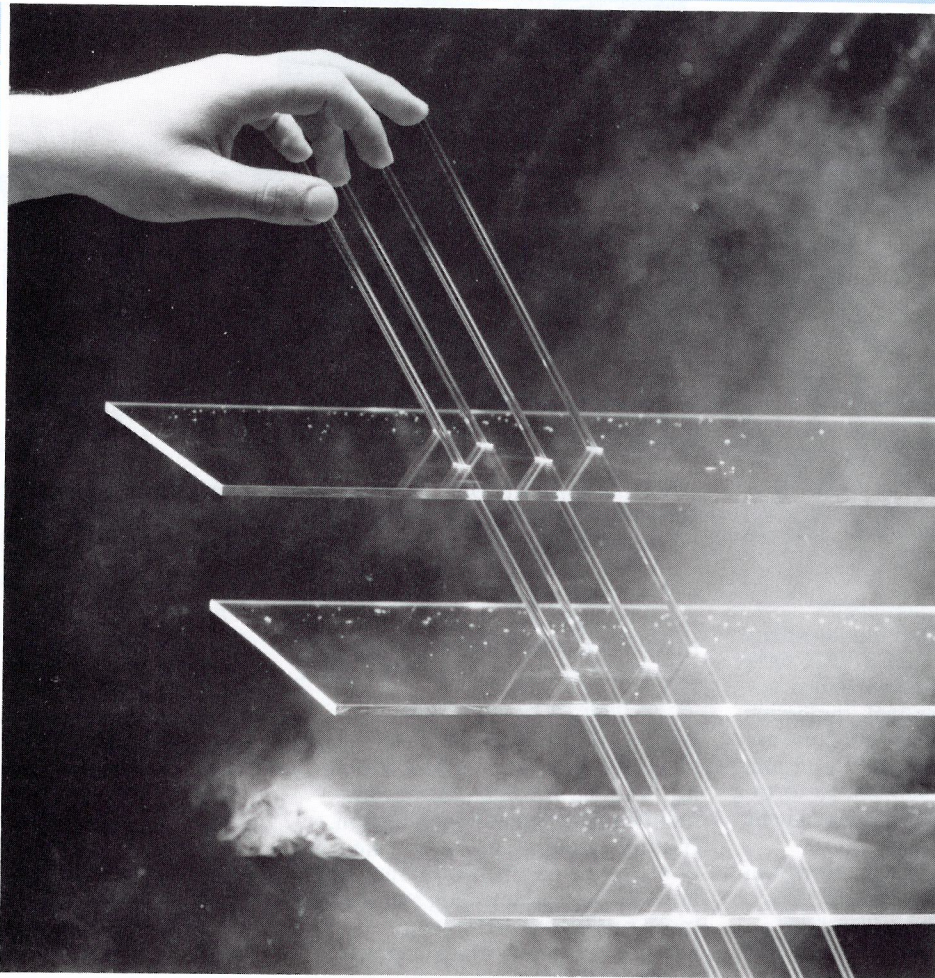


DISTRICT OFFICES:
Palo Alto, CA
San Francisco, CA
Los Angeles, CA
Dallas, TX
New York, NY

CIRCLE 1 ON READER CARD

ROSS SYSTEMS
CORPORATE HEADQUARTERS
1860 Embarcadero Road
Palo Alto, CA 94303
(415) 856-1100

Summoning Solutions DIGICALC v1.2^{T.M.}



- GOAL SEEKING
- ARCHITECTURE ALLOWS
DIVISION CONSOLIDATION
- TEXT OR NUMBER SORTING
- MULTIPLE USER CAPABILITY
- ADVANCED CLASSROOM
TRAINING

DIGICALC 1.2 brings new dimensions to the DEC system user. This truly advanced version has immense capacity for spreadsheets that generate budgeting, financial modeling, project management and many types of analyses. This most flexible software is the perfect combination of procedural and non-procedural language. Once learned, the scientific concept of pattern recognition that has been designed into the keypad aids recall for professionals and executives with little computer experience. No other program for DEC equipment has such extensive HELP built in. The user is in command and can define functions, consolidate multiple spreadsheets . . . even integrate with existing systems on a multi-user level.

With over 400 copies installed and 8000 users world wide, a simple call to WHY Systems will give you finger tip access to the more perfect power of DIGICALC[™] 1.2. Runs on VAX/VM5*, PDP-11* and RSTS/E*.



Seeking Solutions . . . Mankind Asks WHY

16902 Redmond Way, Redmond, WA 98052 U.S.A. (206) 881-2331

*Registered trademarks of Digital Equipment Corporation

- BUILT IN TRAINING
- EXTERNAL FILE INTERFACE
- UP TO 17 DIGITS OF
NUMERICAL PRECISION
- BOARDROOM QUALITY
REPORTS
- THREE-DIMENSIONAL
CONSOLIDATION

Contents

IMAGES COMMA SHAREABLE — A Short-Cut Through the VMS Manuals	8
Al Cini VAX introduced many new concepts to the DEC world, the idea of sharing images is one of them. All of this is in the manuals of course, but this article might save you some time.	
SUMMER HEAT AND WINTER COLD	20
Carl B. Marbach Building a computer room? Ever hear the story of the one that was built with a door that was too small for the computer? Avoid that and other problems by planning your next room with care.	
RSTS/E VERSION 8.0 — A First Report	22
Mark E. Derrick V8.0 of RSTS was an unpleasant surprise for many of us.	
WRITING AST ROUTINES IN VAX-11 BASIC	26
Kevin Paul Herbert Asynchronous System Trap routines are helpful; here is a technique for using them.	
TAPE COPY	28
W. Franklin Mitchell, Jr. You have only one tape drive and you need to make a copy of a tape? Try this for the answer.	
BASIC PLUS TECHNIQUES FOR TABLE ORGANIZATION AND LOOKUP	34
Dennis Thibeault You can't index everything in a file, but you can organize the data so that you can find what you need when you need it. Tables are one way to do it.	
4TH GENERATION SYSTEM DEVELOPMENT	48
D.G. Burnley System development methodologies are an important part of any manager's portfolio. Will automatic/smart programming systems reduce our need for programmers, or will increased productivity make computers better and help proliferate systems?	
RSTS AND THE MICRO-11	50
Carl B. Marbach Is this the perfect machine/software match? A marriage made in Maynard.	
SMARTDCL — A DCL Alternative	52
Bob Stanley The users do it better. Just look at SMARTDCL.	
TIPS & TECHNIQUES — Understanding Terminal Interface Performance	58
Kevin Paul Herbert Which communication multiplexer or what single line interface should be configured with a computer can be a mystery. There are significant differences which can impact system performance.	
VAX SECURITY	60
Terry C. Shannon A subject of some importance which needs to be explored fully is STARTED here. We would like to hear more about this timely subject.	
ISCAN.BAS	62
David S. Williams This scanning program for RSTS could be changed to look for more than just ISAM files.	
CASTAT — An Optimization Tool	66
Joe Bigley When caching was introduced to RSTS it was hard to tell if it was doing any good. Sometimes, it is still hard to tell. Help is here.	
TRAPPED BY CONTROL-C	72
Philip G. Anthony Catch-22 or ERR=28%? Error trapping can be a trap itself.	
TICKETS PLEASE: Setup/Reserve	74
Don Buhman How about a seat reservation system in BASIC. Take a seat!	
EFFECTIVE USE OF THE VT-100 PRINTER PORT	78
Lawrence F. Koolkin The printer ports are not only on the VT-100 but also on the VT-102 and the DEC personal computers. Using this feature can be tricky.	
ON2OF1.B2S	79
Dave Goodman An improvement on an old program first published in the RSTS Professional , March, 1981.	
A PROPER BALANCE BETWEEN MEMORY AND DISK	80
M. Christopher Getting and Philip G. Anthony I know memory is getting cheaper, but is it that cheap?	

Coming

- New VAX Hardware
- Using the RSTS/E Data Space Feature
- DYNPRI.MAC
- Transferring Files Between RSTS and UNIX
- VMS Directories: Some Navigation Aids
- Patching PIP for Protection
- "Civilized" Image Activation
- VAX Clusters: Then, Now, and in the Future
- Make Your PDP/34 Work Harder
- Halloween Special, RSTS Under Wraps
- A User Control Run-Time System
- What's Wrong with VMS
- KIM. Program to Transfer Files to and from TU-58 Cartridge Tapes
- Constructing An Optimized Disk for RSTS/E
- More . . .

Departments

From the Publishers	4
"Pro" Talk (letters to the editor)	6
Dear VAX/RSTS Man (questions & answers) ..	36
News Releases	82
Classifieds	87
List of Advertisers	88

The VAX/RSTS Professional Magazine, August 1, 1983, Vol. 5, No. 4. ISSN 0745-2888 is published bi-monthly by M Systems, Inc., 161 E. Hunting Park Avenue, Philadelphia, PA 19124. Subscriptions: single copy price \$10⁰⁰, \$35⁰⁰ per year; \$50 Canada and 1st class. All other countries, air mail, \$60 US. Second Class postage paid at Philadelphia, PA. POSTMASTER: Send all correspondence and address changes to: VAX/RSTS Professional, P.O. Box 361, Ft. Washington, Pa. 19034-0361, telephone (215) 542-7008. COPYRIGHT © 1983 by M Systems, Inc. All rights reserved. No part of this publication may be reproduced in any form without written permission from the publisher.

From the publishers . . .

HIT OR MIPS

Dave Mallery

Well, where is it??

Where is the 'big VAX'? Where is the response to the DG 10000? I heard that they are talking about a machine in the CRAY-1 performance range by the early 1990s. My wrist watch will perform in that range by then.

In this week's ELECTRONICS, there was a technical article announcing a 1.5 mips 32-bit CHIP from Zilog. It features six level pipelining and an on-chip cache. Its I/O bandwidth can reach 13.3 megabytes per second. It's called the Z80,000 and will start samples next spring. Motorola, Intel and National Semi are all very serious players. Western Electric has a working chip.

No one really knows just what these chips are going to do when they grow up. Judging from the current rate of growth of personal computer utilization, they will be just in time.

Just how does a half-million dollar 1 mips 780 stack up in the face of a 1.5 mips desktop running UNIX? Sure, they are not the same. A desk-top personal is not a main-frame. Or is it?

VMS may be great, but will it sell iron in our brave new world?

WHAT'S GOING ON THERE?

Carl B. Marbach

DEC has recently undergone a major internal reorganization. From all indications it was overdue.

The "State of the Corporation," from the outside world is not good. Four areas of concern have been:

1. The lack of a suitable replacement for the VAX 11/780

2. The early failures and delays in delivering the UDA-50

3. The complete collapse of the "Jupiter" project which was to be the next 36-bit machine

4. The late entry into the personal computer field

The VAX 11/780, which was introduced more than five years ago, is still the largest and most powerful VAX available (the 11/782 is just two 11/780s, not a newer machine). In those five years we have developed the PDP-11/70 on a chip, 64K memory chips, much improved packaging, performance and reliability. Although the 64K chips were forced into the VAX because of economic considerations, we have not benefited from other technological improvements. Other manufacturers are bringing multiple-MIP machines to market;

. . . continued on page 63

From the VAX editor . . .

SHARE IT WITH FRIENDS

Al Cini

The RSTS PROFESSIONAL has grown from humble beginnings just a few years ago to a widely read and respected technical journal because its readership PARTICIPATES. The little unsolicited "jewel" articles — coding tips, system management tricks, product tutorials — have inspired many a creative solution to thorny RSTS problems the world over.

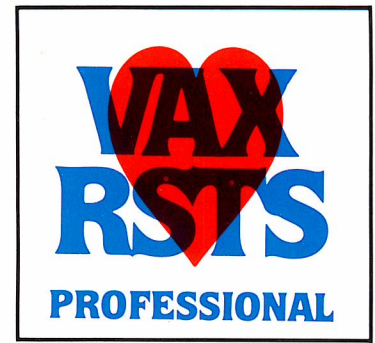
Now that we're the VAX/RSTS PROFESSIONAL, we intend to earn the same level of respect from our VAX readers — but we'd like your help! You, VAX user, are a prospective author!

Your audience? System managers, novice and advanced programmers, project leaders, anybody within "bitshot" of a VAX.

Your topic? VMS performance advice, programming language notes, system programs, product reviews, anything you know and would like to tell others about.

Your motivation? A little ego, a little greed (the magazine pays you an honorarium when it publishes your article), or maybe even an honest desire to help your colleagues.

So, what's stopping you? Write that idea up and send it in! And grow a little bit along with us!



Publishers: R.D. Mallery, Carl B. Marbach

Managing Editor: James L. Trichon

VAX Editor: Al Cini

Director of Advertising: Helen B. Marbach

Business Manager: Peg Leiby

Assistant to the Editor: Hope Makransky

Editorial Assistant: Linda DiBiasio

Production Manager: Georgia F. Conrad

Subscription Fulfillment:

Kathi B. Campione, Claire Hollister

Steven Barsh, Margie Pitrone

Connie Mahon, Terry McMahon

United Kingdom Representative:

Pauline Noakes

RTZ Computer Services Ltd., P.O. Box 19

1 Redcliff Street, Bristol, BS99-7JS

Phone: Bristol 24181

Mid-Atlantic Representative:

Ed Shaud

P.O. Box 187, Radnor, PA 19087

Phone: 215/688-7233

N.Y. & New England Representative:

Jack Garland

P.O. Box 314 S.H.S., Duxbury, MA 02332

Phone: 617/934-6464

Contributors:

Philip G. Anthony, Joe Bigley, Don Buhman,

D.G. Burnley, Mark E. Derrick,

M. Christopher Getting, Dave Goodman,

Kevin Paul Herbert, Lawrence F. Kookin,

W. Franklin Mitchell, Jr.,

Terry C. Shannon, Bob Stanley,

Dennis Thiebaud, David S. Williams,

Cartoons: Frank Baginski, Douglas Benoit,

Emily Dahlstrom, Brian Hansen

Design, Typesetting & Layout: Grossman Graphics

Printing & Binding: Schneider Litho Co., Inc.

ALL PROGRAMS PUBLISHED IN THE VAX/RSTS PROFESSIONAL ARE WARRANTED TO PERFORM NO USEFUL FUNCTION. THEY ARE GUARANTEED TO CONTAIN BUGS. THEY ARE DESIGNED TO GET YOU THINKING. THEY ARE INTENDED TO EDUCATE AND ENTERTAIN. THEY ARE PUBLISHED ON THE PREMISE THAT IT IS BETTER TO SPREAD PEOPLES' BEST EFFORTS AROUND EVEN IF THERE IS AN OCCASIONAL PROBLEM. IF YOU USE THEM, MAKE THEM YOUR OWN, AND YOU WILL NOT GO WRONG.

Editorial Information: We will consider for publication all submitted manuscripts and photographs, and welcome your articles, photographs and suggestions. All material will be treated with care, although we cannot be responsible for loss or damage. (Any payment for use of material will be made only upon publication.)

*This magazine is not sponsored or approved by or connected in any way with Digital Equipment Corporation. "VAX", "RSTS" and "DEC" are registered trademarks of Digital Equipment Corporation. Digital Equipment Corporation is the owner of the trademarks "VAX", "RSTS" and "DEC" and is the source of all "DEC" products--.

Material presented in this publication in no way reflects the specifications or policies of Digital Equipment Corporation. All materials presented are believed accurate, but we cannot assume responsibility for their accuracy or application.

DOPTER

The Time & Money Saver

1.

PERFORMANCE

DOPTER is a RSTS/E disk optimization program which provides dramatic improvements in response time and up to 50% more throughput. DOPTER produces a better organized disk than any other product!

2.

EASE OF USE

The Disk Analyzer and other automatic functions require almost no intervention by the user. Yet features not offered by anyone else are standard with DOPTER.

3.

PRICE

At \$750 for the first CPU, DOPTER is priced so you can't afford to have fragmented disks and a slow system. And with each license we include a 30 day unconditional money-back guarantee.

If you would like more information on how you can increase the performance of your RSTS/E system with DOPTER, phone or write today.

RSTS/E is a registered trademark of Digital Equipment Corporation.



**System
Performance
House, Inc.**

51 North High Street • Columbus, Ohio 43215 • 614-464-4411

CIRCLE 108 ON READER CARD

PRO TALK



Send letters to:
PRO TALK
P.O. Box 361
Ft. Washington, PA
19034-0361

I have recently seen the cover of Vol. 5, No. 1 of *RSTS Professional* [Feb. 1983]. I would appreciate any information you can supply which would help me to acquire some of those buttons (especially the Wombat Wizard).

Concerning text editors! I use VTEDIT frequently but it sets the terminal to ANSI mode and I need it in VT52 mode. I realize that I probably need to change VTEDIT.TEC, but I don't know TECO well enough to even consider this. I have the same problem with EDT when used in the CHANGE (full screen) mode. Is there a patch for EDT that will remedy this?

One of the features of VTEDIT which I appreciate is the storing of the text's file name in the file TECFjj.TMP (jj is the user's job number). I have written a program to simulate this for the EDT text editor, and have

enclosed a copy of the source. It involves the creation of a CCL (in this case I used "ED2") which retrieves a text file name if one exists or stores the text file name in the file EDT2jj.TMP (much the same as VTEDIT uses TECFjj.TMP). It then uses SYS(CHR\$(14%)+ "EDT...") to execute EDT. This allows you to use the EDT editor without having to repeatedly give the text file name. It also saves the name of the EDT initializing file if given.

James Miner, Computer Services
Church of the Brethren
Elgin, Ill.

* * *

I would like to take this opportunity of saying that I think it is a fantastic magazine and keep up the good work.

Also, is there anyone who could tell

me how to kill a high priority CPU Bound job, which is holding the system hostage on a PDP 11/44.

R.B. Lautenbach, Director
Cape Town, Kaapstad

* * *

Many owners of 11/44 procesors running RSTS have tried the built-in TU-58 tape unit once or twice and given up under a deluge of "Device hung or write-locked" errors. The problem is that 9600 baud is just too fast for RSTS. DEC's solution is evident: they are discontinuing support for it when (and if) V8.0 is replaced. This is tolerable for most users, after all, not even Field Service will use it if there is any alternative device.

We, however, are developing code for TU-58 based stand-alone systems, and the necessity of copying software to floppy, and putting it on tape via RT11 (whose driver does work) got on my nerves. Luckily, there is a solution for some configurations. The 11/44 TU-58 port may be set for 9600 baud, 38.4 K baud (heaven forbid), or the CONSOLE BAUD RATE! If your console is an LA120, you may reset the

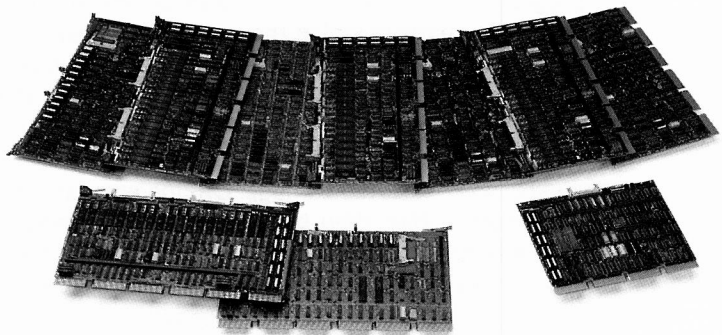
... continued on page 70

```

1-----
1      ED2.B2S          BY JAMES E MINER
1
1      INITIALIZER FOR EDT.
1      SETS UP FILE "EDT2JJ.TMP", WHICH IS USED BY THE CCL
1      ED2 IN THE SAME WAY "TECFJJ.TMP" IS USED BY TE/VT
1
1      TO SET UP CCL:
1          UT CCL ED2=ED2.TSK;30000
1
1      FORMAT:
1      ED2 [PPN1]OUTPUT.TXT=[PPN2]INPUT.TXT,[PPN3]EDTINI.TXT
1      MINIMUM:
1      ED2 OUTPUT.TXT
1
1      IF "ED2" ONLY IS TYPED, THEN THE PROGRAM LOOKS FOR
1      "EDT2JJ.TMP" FOR EDITING INFORMATION.
1-----
1
1      EXTEND
1      ON ERROR GOTO 19000
1
1      OPEN "KB:" AS FILE #1$
1      \ PRINT #1$, "EDT>";
1      \ INPUT #1$, C$
1      \ CLOSE #1$
1
100     GOSUB 1000          I SET-UP INIT$ I
1      \ GOSUB 1010 IF C$=""
1      \ GOTO 200 IF C$=""
1      \ GOSUB 1100
1      \ DUMPT$=SYS(CHR$(14%)+ "EDT "+C$)
1      \ GOTO 32767
1
200     PRINT "EDT? NO FILENAME"
1      \ GOTO 10
1
1000    I-----
1      I      SET UP TEMPORARY FILE'S NAME (EDT2JJ.TMP),
1      I      USING JOB NUMBER FROM SYS CALL.
1-----
1      S$=MID(SYS(CHR$(6%)+CHR$(26%)+CHR$(0)+CHR$(0)),3%,1%)
1      \ JN$=ASCII(S$)/2
1      \ JN$="00"
1      \ RSET JN$="0"+NUM1$(JN$)
1      \ INIT$="EDT2"+JN$+".TMP"
1      \ RETURN
1
1010    I-----
1      I      GET NAME OF INPUT-FILE FROM EDT2JJ.TMP
1-----
1      OPEN INIT$ FOR INPUT AS FILE #1$
1020    \ INPUT #1$, C$
1      \ PRINT "EDIT: "+C$;
1      \ SLEEP 1
1      \ PRINT
1030    \ CLOSE #1$
1      \ RETURN
1
1100    I-----
1      I      SAVE FILE NAME IN EDT2JJ.TMP
1-----
1      OPEN INIT$ FOR OUTPUT AS FILE #1$, MODE 1536$
1      \ GOSUB 1110
1      \ PRINT #1$, D$
1      \ CLOSE #1$
1      \ RETURN
1
1110    I-----
1      I      BREAK DOWN INPUT STRING FROM: OUTPUT=INPUT,INIT
1      I      INTO: OUTPUT,INIT SO THAT IT CAN BE SAVED IN
1      I      EDT2JJ.TMP
1-----
1      D$=C$
1      \ EQ$=POS(D$,"=",1%)
1      \ RETURN IF EQ$=0%
1      \ EQ1$=POS(D$,"=",1%)
1      \ EQ2$=POS(D$,"=",1%) IF EQ1$=0%
1      \ CP$=EQ$-1%
1      \ CP$=POS(D$,"[",EQ1$+1%)+1% IF EQ1$<0%
1      \ CP$=POS(D$,"[",EQ2$+1%)+1% IF EQ2$<0%
1      \ C$=POS(D$,"[",CP$)
1      \ D$=LEFT$(D$,EQ$-1%) IF C$=0%
1      \ D$=LEFT$(D$,EQ$-1%)+RIGHT$(D$,C$) UNLESS C$=0%
1      \ RETURN
1
19000   I-----
1      I      ERROR TRAPPING ROUTINE
1-----
1      IF ERL=10 AND ERR=11
1      THEN PRINT "EXITING. .";
1      RESUME 32767
1      ELSE
1      IF ERL=1010
1      THEN C$=""
1      RESUME 1030
1      ELSE
1      PRINT "ERR =" ;ERR, "ERL =" ;ERL
1      ON ERROR GOTO 0
1      GOTO 32767
1
30000   I-----
1      I      *** CCL ENTRY
1-----
1      CCL$="ED2"
1      ON ERROR GOTO 19000
1      C$=CVT$(SYS(CHR$(7%)),1%+1%+8%+16%+32%+128%+256%)
1      \ FOR J$=LEN(CCL$) TO 1% STEP -1%
1      \ IF LEFT$(C$,J$)=LEFT$(CCL$,J$) THEN
1      \ C$=RIGHT$(C$,J$+1%)
1      \ C$=RIGHT$(C$,2%) IF LEFT$(C$,1%)=SP
1      GOTO 100
1
30010   NEXT J$
1      \ PRINT "?"+CCL$+" - UNRECOGNIZED CCL COMMAND - ";C$
1
32767   END

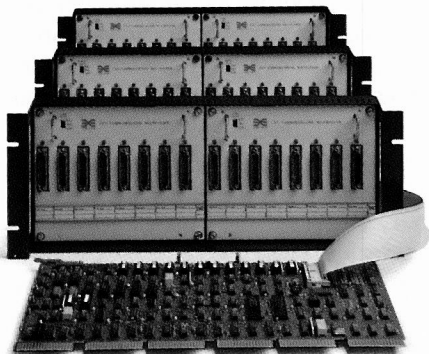
```


Five reasons why DEC users should buy Emulex communications controllers.



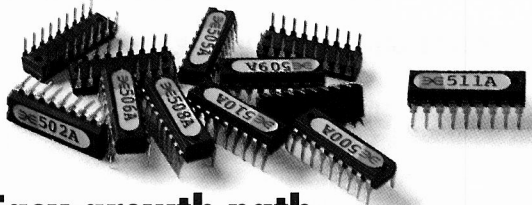
Broad product line featuring our new DMF-32 emulation.

Nobody covers LSI-11, PDP-11, and VAX-11 users' needs like Emulex. More than 15 software-transparent controllers emulating DH11, DZ11, DV11 and DMF-32. All deliver improved line-handling capabilities, in a smaller package, at lower costs.



More channels.

Emulex's new DMF-32 emulation is typical. One controller board handles up to 64 lines, vs. only eight per DEC module. And Emulex offers *all* lines with modem control, not just two. For even more lines, Emulex's Statcon Series is the answer. We simply add a low-cost port concentrator, so that with one controller board you can connect up to 256 remote *and* local terminals.



Easy growth path.

As your system grows, upgrading is simple with Emulex controllers. Just change PROM sets. Example: DH to DMF for \$350. In addition, Emulex's advanced microprocessor architecture is consistent throughout the product line. Think of the inventory savings.



Fewer backplane slots.

Emulex communications controllers pack so much capability onto each board that fewer boards are needed. Take a 64-line DH11 emulation. Emulex does on one board what it takes DEC to do on 36. Think of the savings in rack space, to say nothing of price.



Lower prices.

For instance, a DEC DH11 controller lists at \$8,950 per 16 lines, with expansion chassis costing \$3,000 or more. Compare that to Emulex's CS11/H at \$4,500 for the first 16 lines and \$3,000 for each additional 16 lines. At 64 lines, you suddenly have savings of about \$23,000 and a lot of extra slots to boot.

Don't speculate with your communications controller dollars. Invest in Emulex. Phone toll free: (800) 854-7112. In California: (714) 662-5600. Or write: Emulex Corporation, 3545 Harbor Blvd., P.O. Box 6725, Costa Mesa, CA 92626.



The genuine alternative.

IMAGES COMMA SHAREABLE

(A short-cut through the VMS manuals)

By Al Cini

Software engineers can use shareable images under VMS to set up more easily maintained subroutine libraries, to reduce system-wide paging overhead, to establish virtual memory data areas in which multiple processes (or processors, for you rich folk with a 782) can share data — in short, shareable images are worth knowing about. This article will present example shareable image construction for a wide number of representative cases, illustrated using simple VAX-11 BASIC programs (FORTRAN, PL/I, COBOL, or other compiled language coders should be able to generalize most of these examples with little difficulty; in some cases, key BASIC language features will be tied explicitly to their equivalents in other languages).

WHAT ARE SHAREABLE IMAGES?

Every VAX/VMS programmer is familiar with the traditional role of the LINKER utility in program development:

```
$LINK A,B,C
```

In this example, the files A.OBJ, B.OBJ, and C.OBJ are object modules, produced by a compiler or assembler from a source program file. Object modules contain generated machine instructions, virtual memory storage allocation requests, local and global symbols, and other things called for by your source program (see Appendix A of the Linker Manual for a detailed description of the VMS object language). Except in a few unusual cases known only to MACRO programmers, the addresses assigned to data and instructions in an object module are relocatable — which means that they're tentatively zero-based, to be assigned actual virtual addresses later.

Given our previous example DCL command, the LINKER utility takes the A,B, and C object modules and creates an A.EXE executable image file, converting the relocatable object module addresses into actual virtual memory addresses as it goes along. Later, a \$RUN A command tells the image activator to load the program in A.EXE into virtual memory and transfer control to it.

Alternatively, we could have pre-linked subroutines B and C into a shareable image offering the following advantages:

- If INSTALLED, shareable images accessed by many processes concurrently will share the same physical memory in the machine, thus reducing paging I/O. If these installed shareable images contain data program sections, the data in these sections can be accessed transparently by concurrently active processes.
- If transfer vectors are used (described later), the shareable image containing B and C can be modified

without re-linking the main program A (or any other program which calls B or C and is linked to the shareable image in which they are maintained). Also, because a shareable image is "pre-linked," linking calling programs to shareable images is faster than linking calling programs to object modules.

A shareable image is sort of half-way between an object module and an executable image. Linking programs to a shareable images rather than to individual object modules defers the assignment of some virtual memory addresses until image activation time, which accounts for the greatly increased flexibility shareable image subroutine libraries afford a programmer.

DO I NEED TO BE A GENIUS TO USE SHAREABLE IMAGES?

Did you need to be a genius to be a programmer?

No.

HOW DO SHAREABLE IMAGES WORK?

Both physical memory on your VAX and virtual memory in your program space are organized into 512-byte segments called pages, and your programming language (or you, if you code in MACRO) will assign characteristic attributes to each of the pages in your program. (To simplify image activation, the linker will gather pages with similar characteristics into image sections). When a program is executed, the image activator and the VMS operating system decide where in physical memory each of the pages in your virtual memory should reside when they are referenced, and, based on page attributes, the system software can decide which physical pages can be simultaneously "shared" among several working sets.

This little bit is already a lot more than you need to know to use shareable images. If you're still curious about the mechanics, refer to the LINKER manual for "Shareable Images" and the SYSTEM MANAGER'S GUIDE for the INSTALL utility. You can also get a discussion of the LIBRARIAN's use with shareable images in the UTILITIES manual. The rest of this article will "cook-book" the procedures for building and using shareable images.

HOW DO I CREATE SHAREABLE IMAGE SUBROUTINE LIBRARIES?

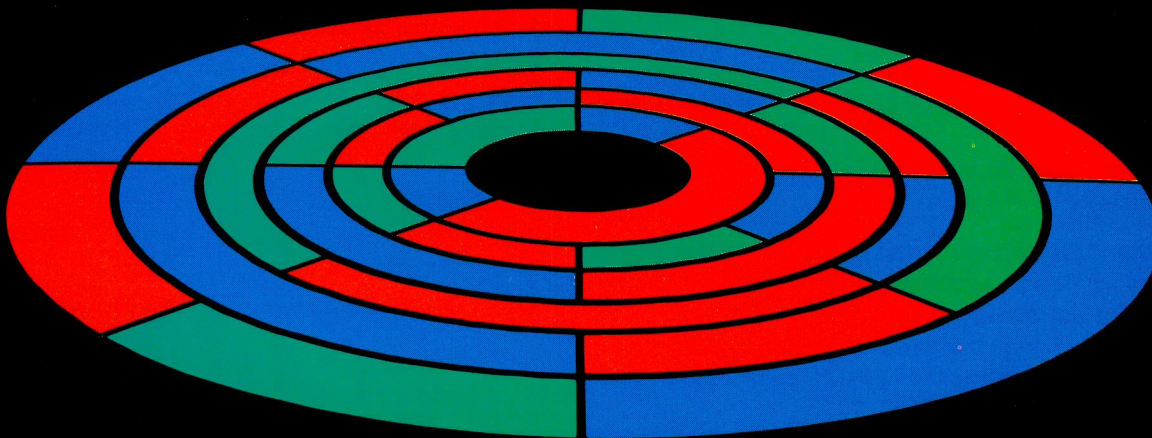
Some of the steps in creating shareable image subroutine libraries are optional. To start with, let's leave them out and see what happens.

To create shareable image subroutine libraries without transfer vectors:

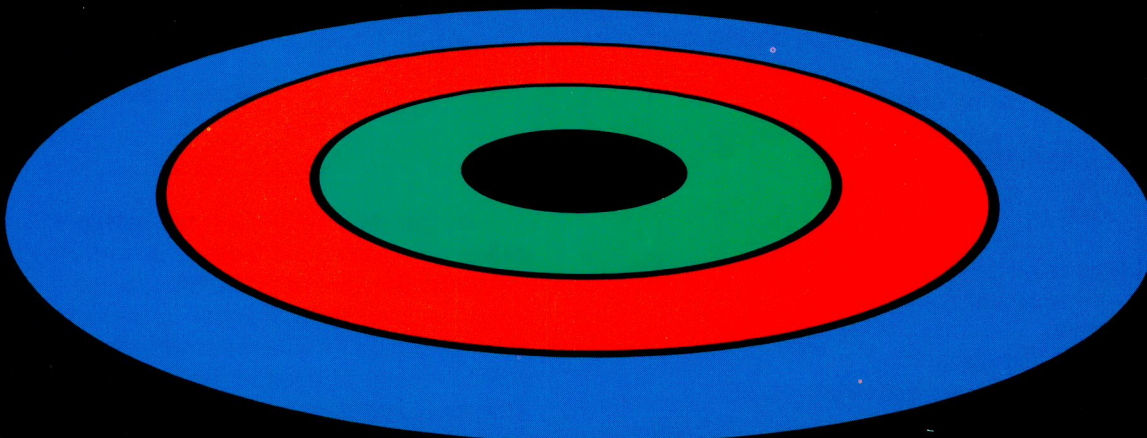
1. Compile your subprograms and external functions, for example:

NOW FOR V8.0

WHAT YOU DON'T KNOW ABOUT YOUR DISKS IS COSTING YOU MONEY



If your disk looks like this, you're wasting system performance.



If your disk looks like this, you're using DISKIT.

When the job you're running requires reading the "red" file, it naturally happens faster on a well-ordered disk. Disks become "fragmented" as you use your computer. The system slows down. And that costs you money.

Now, you can restructure your disks and get back that lost performance (up to 50%) without spending a dime on new hardware. DISKIT is the original software system that makes this possible.

But don't confuse DISKIT with other system utilities, DISKIT is a complete "software tool kit" that optimizes your RSTS/E system.

DISKIT is:

- DSU — The utility which restructures the information on your disk, making data fast and easy to access.
- DIR — The incredible directory tool that finds files at the rate of 400 per second.
- RDR — Reorders disk directories 30 times faster than ever before possible.
- OPEN — Displays complete job statistics and file activity so you can see what your system is doing.
- DUS — The set of CALLable subroutines which pre-extend file directories, reducing fragmentation.

In today's tight economy, it's more important than ever to get the most out of your hardware investment. Call or write today and start getting your money's worth from your computer.

Software
Techniques
Incorporated

5242 Katella Avenue
Los Alamitos, CA 90720
United States
Phone: [714] 995-0533

287 London Road
Newbury, Berkshire RG13 2QU
United Kingdom
Phone: 44 [0] 635-30840

CIRCLE 65 ON READER CARD


```

SUB1.BAS:
10 SUB SUB1
  PRINT "[1] SUB1, V1"
END SUB
SUB2.BAS:
10 SUB SUB2
  PRINT "[2] SUB2, V1"
END SUB

```

2. Make a shareable image library out of them using the linker:

```

$LINK/SHARE = SHARELIB SUB1.SUB2,SYS$INPUT/OPTIONS
GSMATCH = LEQUAL,1,0
UNIVERSAL = SUB1.SUB2

```

The GSMATCH option is used to enforce upward-only compatibility in routines which are linked to a shareable image. If this is of no concern to you, everything will be ok as long as you never change the arguments to this option when you link your images. Check the LINKER manual for a "how to use" discussion of GSMATCH.

The UNIVERSAL option tells the linker that the external symbols SUB1 and SUB2—the subroutines' entrypoints — are to be "known" to modules linked to the shareable image. Any other subroutines included in SHARELIB.EXE, perhaps referenced by SUB1 or SUB2, will NOT be callable from outside the shareable image.

3. Compile your calling program:

```

MAIN.BAS:
10 PRINT "SHARED LIBRARY TEST"
  CALL SUB1
  CALL SUB2
END

```

4. Link your main program to the shareable image:

```

$LINK MAIN,SYS$INPUT/OPTIONS
SHARELIB/SHARE

```

5. Invoke the main program:

```

$DEFINE/USER SHARELIB dev:[directory]SHARELIB.EXE
$RUN MAIN
SHARED LIBRARY TEST
[1] SUB1, V1
[2] SUB2, V1

```

The \$DEFINE command isn't necessary if SHARELIB.EXE is in SYS\$SYSROOT:[SYSLIB]. If the shareable image file is kept anywhere else, you need to set up a logical name associating it with its device and directory location. If you're interested, this little inconvenience is brought to you by the image header format, which lacks the space to carry a fully qualified VMS file specification for a shareable image name, hence defaults to SYS\$SHARE unless over-ridden as above.

Be careful NOT to use a logical name that conflicts with another logical or file name in your program!

WHAT IF I NEED TO MODIFY ONE OF THE SUBROUTINES IN THE SHAREABLE IMAGE FILE?

Let's try it:

1. Compile the edited subroutine:

```

10 SUB SUB1
  PRINT "[1] SUB1, V2"
  PRINT "THIS IS THE MODIFIED SUBROUTINE"
END SUB

```

2. Re-build the shareable image (as in step 2 above).

3. RUN THE MAIN PROGRAM:

```

$DEFINE/USER SHARELIB dev:[directory]SHARELIB.EXE
$RUN MAIN

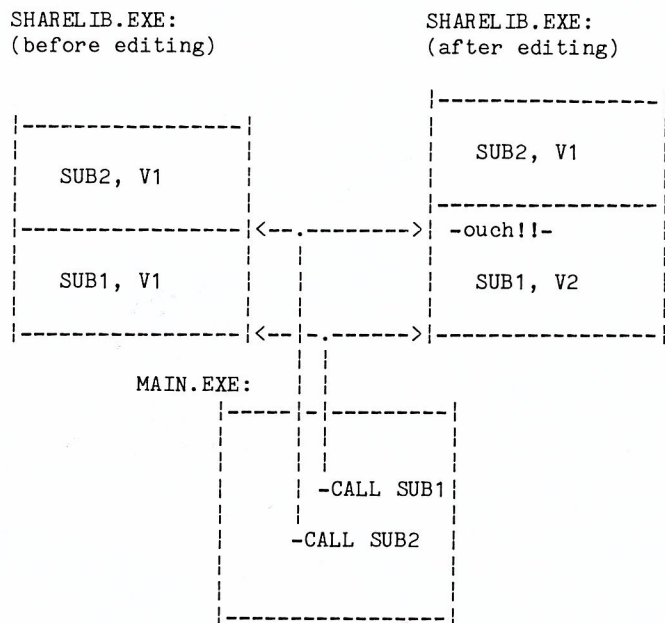
```

```

[1] SUB1, V2
THIS IS THE MODIFIED SUBROUTINE
%BAS-F-MEMMANVIO, Memory Management Violation
-BAS-I-USEPC-PSL ...

```

As you can see, we had very little luck trying to change SUB1 within SHARELIB.EXE without re-linking MAIN.EXE, because:



As this diagram shows, the entrypoints into shareable image subroutines are "bound" to a referencing program at link-time. Our edit to SUB1.BAS caused it to "grow" a few bytes, pushing SUB2 further up in virtual memory. Since MAIN.BAS didn't know about this, the CALL instruction into SUB2 actually jumped at random somewhere into SUB1. This confusing state of affairs is corrected by using a device known as "transfer vectors."

HOW DO I USE TRANSFER VECTORS?

All you need is a little MACRO routine (requiring no MACRO coding experience). To continue with our example:

1. Assemble a file with the transfer vector directives required by your situation:

```

XFER.MAR:
.TITLE      "TRANSFER VECTORS"
.PSECT      XFER.EXE,NOWRT,PIC,SHR,GBL
.TRANSFER   SUB1
.MASK       SUB1
JMP         L1SUB1+2
.TRANSFER   SUB2
.MASK       SUB2
JMP         L1SUB2+2
.END

```

(You assemble this source program with the DCL command \$MAC XFER)

2. Link the transfer vectors into your shareable image along with your subroutines:

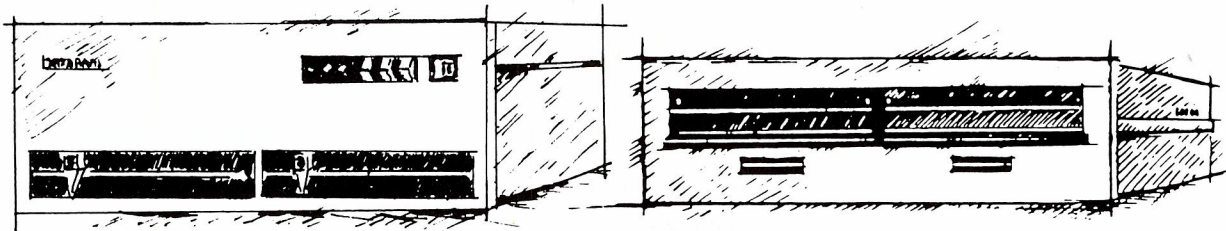
```

$LINK/SHARE = SHARELIB XFER.SUB1.SUB2,SYS$INPUT/OPTIONS
GSMATCH = LEQUAL,1,0
CLUSTER = UPFRONT
COLLECT = UPFRONT,XFER

```

Note that the UNIVERSAL=SUB1.SUB2 option is no longer required; the .TRANSFER directives provide the linker with this information.

DEC-COMPATIBLE DISKS FOR LESS



Dataram has acquired Charles River Data Systems' DEC-compatible product line. We will continue to offer their popular FD-311 dual floppy subsystems and have added an exciting new floppy-based system, Dataram's A21.

Q-bus and UNIBUS compatible versions of the FD-311 provide dual RX02-compatible 8" floppy drives for \$2,490. Our new 7" high A21 combines dual RX02-compatible 8" slimline floppies with an 8-quad slot Q-bus card cage for only \$3,600. Both products are supported by the industry's widest range of LSI-11 compatible products. Call or write for details.

DATARAM

Dataram Corporation □ Princeton Road □ Cranbury, New Jersey 08512 □ Tel: 609-799-0071 □ TWX: 510-685-2542

DEC and LSI-11 are trademarks of Digital Equipment Corporation.

CIRCLE 55 ON READER CARD

The CLUSTER and COLLECT options "force" the linker to place the transfer vectors in the beginning (i.e., at the lowest addresses) in the shareable image. The CLUSTER option simply assigns the name "UPFRONT" to the first "cluster" of memory in the shareable image; the COLLECT option then instructs the linker to place the program section XFER (remember the .PSECT directive in the transfer vector MACRO source program?) containing our transfer vectors into this first cluster. IT IS VITALLY IMPORTANT THAT THE TRANSFER VECTORS FOR A SHAREABLE IMAGE NEVER "MOVE" WITHIN THE SHAREABLE IMAGE! This CLUSTER/COLLECT device forces the issue by allocating the transfer vectors at the beginning of the image, where nothing ahead of them can "grow" and accidentally move them.

3. Re-link the main program to the shareable image:

```
$LINK MAIN,SYS$INPUT/OPTIONS
SHARELIB/SHARE
```

4. Whenever you edit anything inside the shareable image, or add any new subroutines to it, you can repeat step 2 above without re-linking your main program PROVIDED:

- YOU NEVER CHANGE THE ORDER OF THE TRANSFER VECTORS.

- YOU ALWAYS ADD NEW VECTORS AFTER EXISTING VECTORS.

Transfer vectors can be used in other interesting ways. Let's define a macro to set up transfer vectors a little more easily:

```
.MACRO      VECTOR VECTOR__NAME, ENTRY__NAME
.TRANSFER   VECTOR__NAME
.IF NB ENTRY__NAME
    .WEAK   ENTRY__NAME
    .MASK   ENTRY__NAME
    JMP     L1ENTRY__NAME + 2
.IFF
    .MASK   VECTOR__NAME
    JMP     L1VECTOR__NAME + 2
.ENDC
.ENDM
```

We can use this macro to define the transfer vectors in our previous example as follows:

XFER.MAR (using transfer vector set-up macro):

```
.TITLE      "New Vector Routine"
.MACRO      VECTOR VECTOR__NAME, ENTRY__NAME
.TRANSFER   VECTOR__NAME
.IF NB ENTRY__NAME
    .WEAK   ENTRY__NAME
    .MASK   ENTRY__NAME
    JMP     L1ENTRY__NAME + 2
.IFF
    .MASK   VECTOR__NAME
    JMP     L1VECTOR__NAME + 2
.ENDC
.ENDM
.PSECT      XFER,EXE,NOWRT,PIC,SHR,GBL
VECTOR SUB1
VECTOR SUB2
.END
```

— OR we can re-direct calls from SUB1 to SUB2 and vice-versa:

```
VECTOR SUB1,SUB2
VECTOR SUB2,SUB1
```

As you can see, transfer vectors in shareable images can be written to transparently re-direct a call intended for one routine into another. They can be used in this way to "front-end" the VMS run-time library:

```
VECTOR SYS$ASSIGN,ASSIGN__INTERCEPT
```

```
(later, in subroutine ASSIGN__INTERCEPT:)
... CALL SYS$ASSIGN
```

In this example, any program which is linked to this shareable image and calls the system service SYS\$ASSIGN will, without knowing it, actually call the user-written routine ASSIGN__INTERCEPT within the shareable image. This routine may pre-process input parameters, call the "real" SYS\$ASSIGN, and then post-process the output parameters — all unbeknownst (sic) to the referencing MAIN program!

Transfer vectors may also reference non-existent routines yet-to-be-implemented:

```
VECTOR FUTURESUB1
VECTOR FUTURESUB2
```

Or, instead, these unimplemented vector paths could be routed to a specially coded error routine:

```
VECTOR FUTURESUB1,NOTIMPYET
VECTOR FUTURESUB2,NOTIMPYET
```

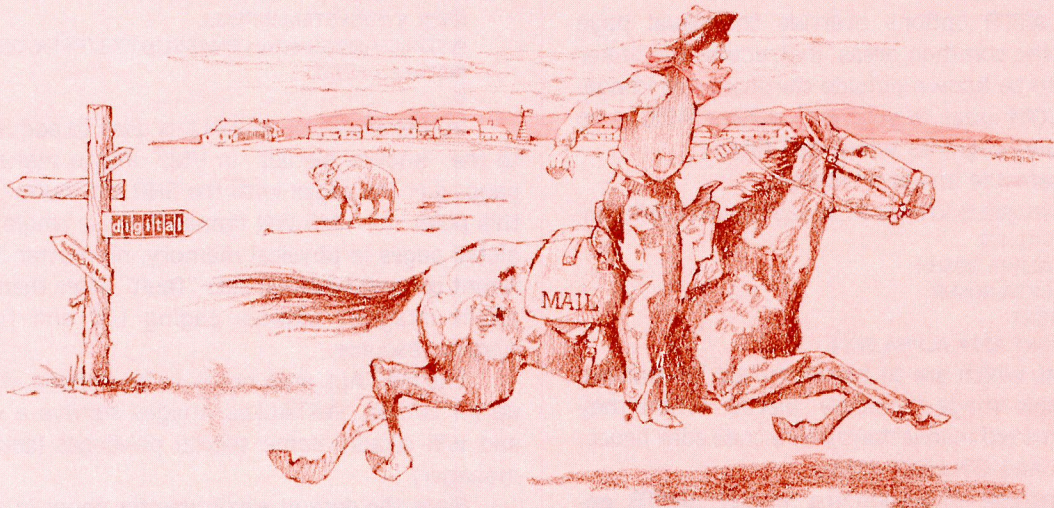
High-level language programmers should note that, when intercepting system service calls or when "stubbing" unimplemented routines with special error routines, the number of arguments passed from a caller may need to agree with the the number expected in the called routine (with VAX-11 BASIC, if the size of the arglist in the CALL statement has more or fewer arguments than the SUB statement, you'll get an "Arguments don't match" error at run-time).

WHAT IF MY SUBROUTINES HAVE MAP OR COM STATEMENTS IN THEM?

MAP and COM statements define blocks of virtual memory for data storage purposes, the actual allocation of which is performed by the LINK utility. (In FORTRAN these are COMMON areas, and in PL/I or COBOL these are EXTERNAL data items or structures.) If the subroutines you include in shareable images define such areas, you will need to take some special steps:

(Assume that both SUB1 and SUB2 in our running example have MAP (ALPHA) ... and MAP (BETA) ... statements in them.)

The most exciting MESSAGE DELIVERY NEWS



since the Pony Express...

**DIGITAL ANNOUNCES DECmail/RSTS VERSION 1.0 ELECTRONIC MAILING SYSTEM
AT AN AFFORDABLE PRICE TO SPUR YOU ON!**

Once, a letter had to travel tortuous routes to reach its destination. Via the swift Pony Express. Or more recently, by way of the office message box.

Now, the wide open spaces between your office and company headquarters, the 18th and 22nd floor, the New York office and the Toronto branch are bridged easily — electronically — with DECmail/RSTS.

DECmail/RSTS provides a low cost, reliable message handling system for RSTS/E sites. With this new system, you can create, edit and process messages — from simple memos to complicated specifications. You can forward your message for delivery almost instantly to a business associate down the hall, across town, around the country or overseas.

DECmail/RSTS also brings you up-to-date in the morning and holds messages for you while you're out. It will organize your material for you — storing, searching and retrieving messages as needed. If you have a problem, an on-line help facility comes to the rescue.

REWARD!

Request full information on your DECmail/RSTS message system now. If you order within 90 days, you will receive a RSTS western-style belt buckle FREE.



Mail the coupon today or call 1-800-DEC-INFO any working day between 8:30 a.m. and 5:00 p.m. E.S.T.

If this sounds as easy as falling off a horse, it is. DECmail/RSTS was designed by the original RSTS/E development team. They put together a system with easy to learn commands and flexible options that make DECmail/RSTS sophisticated enough for advanced users yet simple and uncomplicated for computer novices. DECmail runs on RSTS/E Versions 7.2 and 8.0. No update will be needed until after RSTS/E V8.0 is released. Thanks to DECmail update kits, you won't be saddled with an obsolete software product down the line.

What does all this cost? A lot less than you might think. And the increase in productivity in your working environment — not to mention the wonders of electronic communication — is worth every penny.

Find out about this new electronic message system for your office. Complete, clip out and return the coupon — pronto!

MAIL this coupon to: Digital Equipment Corporation
Direct Response Center, POB 279
Maynard, MA 01754

Please send me complete information on DECmail/RSTS and my eligibility for a Western belt buckle.

Name _____

Title _____ Telephone No. () _____

Company _____

Address _____

City _____ State _____ Zip _____

If the MAP/COM areas in your subroutines are NOT to be referenced from the programs which call them:

- Link the shareable image as follows:

```
$LINK/SHARE = SHARELIB XFER,SUB1,SUB2,SY$INPUT/OPTIONS
GSMATCH = LEQUAL,1,0
PSECT__ATTR = ALPHA,NOSHR,LCL
PSECT__ATTR = BETA,NOSHR,LCL
CLUSTER = UPFRONT
COLLECT = UPFRONT,XFER
```

The PSECT__ATTR options override the usual page characteristics of the common areas, instructing the linker that they are not to be known outside the shareable image.

If the MAP/COM areas in your subroutines ARE to be referenced from the programs which call them:

- Link the shareable image as follows:

```
$LINK/SHARE = SHARELIB XFER,SUB1,SUB2,SY$INPUT/OPTIONS
GSMATCH = LEQUAL,1,0
PSECT__ATTR = ALPHA,NOSHR
PSECT__ATTR = BETA,NOSHR
CLUSTER = UPFRONT
COLLECT = UPFRONT,XFER,ALPHA,BETA
```

Common areas which are to be referenced by routines outside the shareable image should be "placed" where they aren't likely to be moved inadvertently by a code edit; hence, the new CLUSTER and COLLECT steps.

Unless you include corresponding PSECT__ATTR options in the \$LINK command stream for your main program, you will get bothersome but innocuous "CONFLICTING ATTRIBUTES" warning messages from the linker.

Note that, if the MAIN routine must reference common areas in your shareable image, any changes you make to the SIZES of the common areas WILL REQUIRE THAT REFERENCING MAIN PROGRAMS BE RE-LINKED.

Also, anticipate any new subroutines you will need to add to your shareable image and set up not-yet-implemented transfer vectors for them (as described above); otherwise, a new vector added some day may "push" your common regions further up in virtual memory and force main program re-linking.

In case you're wondering, shareable images may be linked to other shareable images, but "going crazy" can cause long image activation times and general confusion among software maintainers.

WHAT HAPPENS IF I DON'T BOTHER CHANGING THE MAP ATTRIBUTES?

By default, MAP and COM statements generate "global" program sections, which can be shared by multiple concurrently active processes; without overriding these defaults with PSECT__ATTR, VMS would assume that all programs linked to SHARELIB intend to read/write data from/to ALPHA and BETA at the same time! Trying to \$RUN MAIN against SHARELIB without PSECT__ATTR attribute overrides would confuse the image activator:

```
$DEFINE ...
$RUN MAIN
%DCL-W-ACTIMAGE, error activating image MAIN
-CLI-E-IMGNAME, image file dev:[directory]SHARELIB.EXE;1
-SYSTEM-F-NOTINSTALL, writable shareable images must be installed
```

WHAT'S THIS "NOT INSTALLED" STUFF?

So far, our experiments with SHARELIB have shown

how shareable images can be built to simplify program maintenance (using transfer vectors). You should note also that shareable images save disk space because subroutines used by many separate calling programs are stored in a single disk file (the shareable image .EXE file), rather than linked into every executable image which uses them. We can take these savings a step further by INSTALLing the shareable image:

```
$RUN SYS$SYSTEM:INSTALL
INSTALL> dev:[directory]SHARELIB.EXE/OPEN/SHARE/HEADER
INSTALL> /EXIT
$
```

In this example, INSTALL is used to add SHARELIB.EXE to the "known file list" in VMS and to make its shareable pages (i.e., its pages with the SHR attribute) "global." From this point on, VMS will manage only a single copy of these global pages in physical memory, no matter how many different process working sets "fault" after them. This, as you might imagine, reduces paging I/O and frees up some system resource.

The INSTALL step needs to be re-done at system start-up (of course, INSTALLED images survive a warm restart) and will require some special privileges (ask your system manager).

Since the default attributes for pages in common areas are SHR (shareable) and WRT (writable), images with such areas defined in them obviously MUST be installed to work properly; this explains our "writable images must be installed" error message.

HOW DO I SET UP A SHARED DATA AREA?

Simple.

1. Write a little BASIC routine with nothing but the MAP(s) and COM(s) you want to share (for this data sharing purpose, MAP and COM are equivalent):

```
SHAREIT.BAS:
10 MAP (SHARE__IT) STRING X = 1000
END
```

You can COM/NOSETUP to leave some unnecessary BASIC set-up code out of the image if you like, or you could code the common area definition in MACRO using the .PSECT directive (examples can be found in the BASIC on VAX/VMS manual). If you use MACRO, you can pre-load this data area with constants; otherwise, such sections are zero-initialized (note: they are NOT demand-zero-compressed, since this is not supported in shareable images).

2. Link the compiled object module into a shareable image, as before:

```
$LINK/SHARE = SHAREDAT SHAREIT,SY$INPUT/OPTIONS
GSMATCH = LEQUAL,1,0
```

3. Link the referencing routine (let's assume its called MAIN) to it:

```
$LINK MAIN,SY$INPUT/OPTIONS
SHAREDAT/SHARE
```

4. Before you run MAIN, INSTALL the shared data area:

```
$RUN SYS$SYSTEM:INSTALL
INSTALL> SHAREDAT.EXE/SHARE/WRITE
INSTALL> /EXIT
```

5. \$RUN MAIN, and you're in business! (Remember the \$DEFINE step)

The /WRITE switch in the INSTALL command instructs

SPSORT

ANOTHER TECHNICALLY SUPERIOR PRODUCT FROM

System Performance House, Inc.

SPSORT is an extremely fast, versatile RSTS/E sort utility.

SPSORT can sort a file into itself using no scratch space except for a 5 block control file.

SPSORT is extremely fast. In one test, SPSPORT sorted 100,000 64-byte records with 23-byte keys in 16 minutes on an 11/24 with RK07 disk drives.

SPSPORT comes in three forms, a SORTG/SORTM* emulator, a Sort-1 Plus* emulator and an FSORT3* emulator. These sorts provide a direct replacement for their namesakes - with no change in software.

In addition to the standard data types, SPSPORT'S emulators have been extended to sort DIBOL decimal, packed and EBCDIC data types.

SPSPORT runs under the RSX run-time system with any swap max from 16K to 31 K words. Front end modules require the BASIC-Plus run-time system or else a swap max of 26K words.

BEST OF ALL...

SPSPORT is priced to provide at least twice the performance-to-price ratio of any other third-party sort. The SORTM/SORTG emulator is priced at \$450.00 per copy, the Sort-1 Plus emulator is \$850.00 per copy and the FSORT3 emulator is \$1250.00 per copy. Attractive quantity discounts are available. Write or call now for the best bargain on the best sort!

* SORTM and SORTG are registered trademarks of Digital Equipment Corporation.
Sort-1 Plus is a registered trademark of Oregon Software, Inc. FSORT3 is a registered trademark of Evans, Griffiths and Hart, Inc.



System
Performance
House, Inc.

5522 Loch More Ct. E. • Dublin, Ohio 43017 • 614/265-7788

Please send more information to:

Name _____

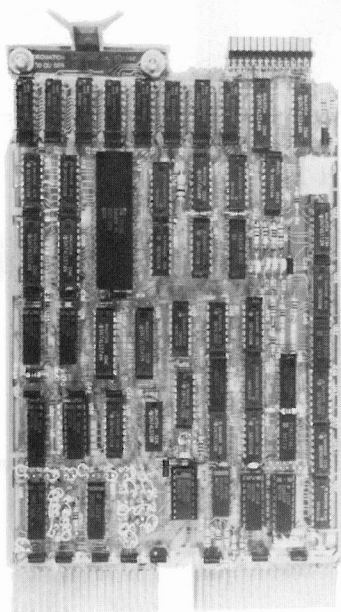
Company _____

Address _____

Phone _____

System Performance House, Inc.

5522 Loch More Court E. • Dublin, Ohio 43017 • 614/265-7788



AND NOW RSTS!

**CP/M® FOR YOUR LSI-11 AND PDP-11
RT-11 TSX+ RSX-11M**

Enter a New Software World

With our plug-in Z80 processor card you can run CP/M, the popular personal and business computer operating system. You can run thousands of low cost software packages available for CP/M. We include a powerful word processing package and a powerful electronic spreadsheet free! Everything you need is included starting at \$1,250.00.



DECMATION

3375 Scott Blvd., Suite 422
Santa Clara, CA 95051
408/980-1678

* CP/M is a registered trademark of Digital Research, Inc.

CIRCLE 198 ON READER CARD

VMS to write modified pages back to the image file periodically. On subsequent reference, the image file will contain the last updated information in the section, rather than initial zeros/null strings. You can also "force" VMS to update /WRITE-installed images by calling the "update section" system service.

WHAT CAN I USE THIS FOR?

Super-high-speed inter-process communication can be useful in a lot of applications. The following example program uses the LIB\$. . . self-relative queue instructions to establish and maintain a variable-length character string "stack," which could be INSTALLED and shared among several processes — or systems, if you're a 782 user.

Have fun!

STACK.BAS:

```

10  $TITLE      "VMS QUEUES IN SHARED COMMON MEMORY"
    $IDENT      "v1.0"
    $SBTTL      "Program Set-Up and Record Definitions"
    !
    ! Program to manipulate a VMS self-relative queue
    ! in a shared common memory area.
    !
    ! In this sample application, the queue structure is used to
    ! implement a LIFO stack with variable-length entries.
    !
    ! This "stack" structure can be made into a shareable image:
    !
    ! 1. Define the stack in a separate program,
    !    compiling and $LINK/$SHARE it
    !
    ! 2. INSTALL the stack shareable image
    !
    ! 3. Take the "initialize queue" logic in this
    !    program and run it as a separate step against
    !    the INSTALLED stack
    !
    ! 4. Separate processes or processors can now
    !    manipulate the stack; multiple processor
    !    access will be "interlocked" automatically
    !    (see the description of the queue functions
    !    in the run-time library manual)
    !
    OPTION TYPE=EXPLICIT &
    ,      SIZE = (INTEGER LONG)
    !
    ! Define queue record elements needed to support our "stack."
    !
    ! (The doubly-linked list queue structure used in this program
    ! is described in the VAX Architecture Handbook. The queue man-
    ! ipulation library services used in this program are described
    ! in the VAX/VMS Run-Time Library Reference Manual.)
    !
    RECORD STACK_LINKS_FMT
        LONG F_LINK
        LONG B_LINK
    END RECORD STACK_LINKS_FMT

    RECORD STACK_HEADER_FMT
        STACK_LINKS_FMT HEADER
        LONG DEPTH
        LONG NEXT_SLOT_OFFSET
    END RECORD STACK_HEADER_FMT

    RECORD STACK_RECORD_FMT
        STACK_LINKS_FMT HEADER
        LONG ENTRIESIZE
        LONG FILL
    END RECORD STACK_RECORD_FMT

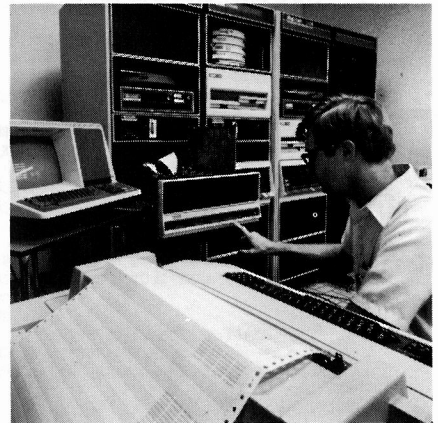
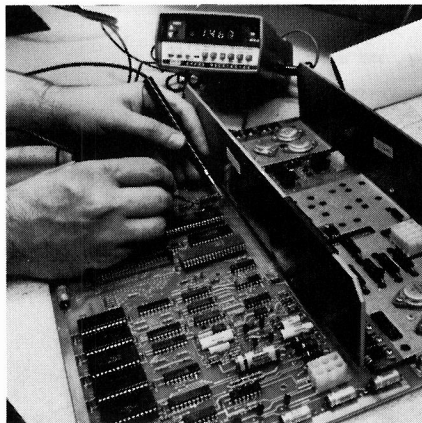
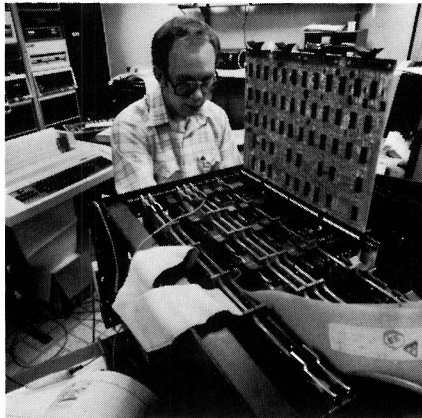
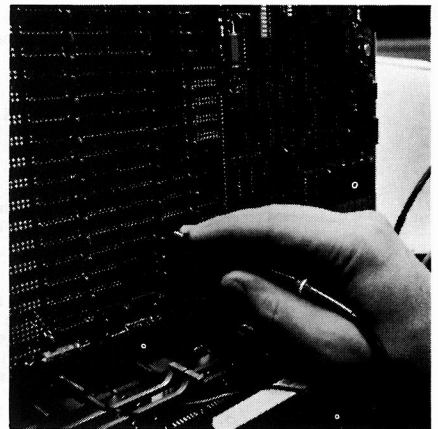
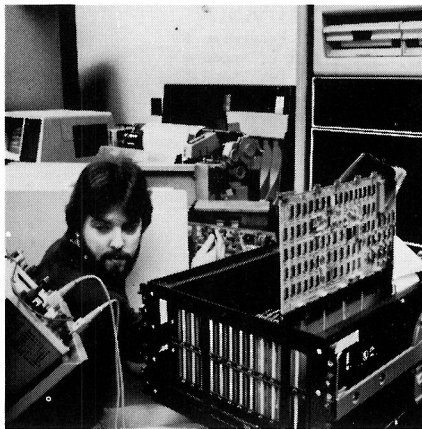
    $PAGE
    $SBTTL      "Define Internal/External Names and Allocate Storage"
    EXTERNAL INTEGER CONSTANT &
        SS$_NORMAL ! Normal successful completion of SYS$( ). &
        , LIB$_SECINTFAI ! Queue structured was interlocked by another &
        , ! processor. &
    !
    EXTERNAL INTEGER FUNCTION &
        LIB$INSQHI ! Hook queue entry to queue header. &
        (STACK_RECORD_FMT BY REF &
        , STACK_HEADER_FMT BY REF) &
        , LIB$REMQHI ! Unhook queue entry connected to &
        ! header. &
        (STACK_HEADER_FMT BY REF &
        , LONG BY REF) &
        , SYS$EXIT ! Exit program returning $STATUS. &
        (LONG BY VALUE) &
    !

```


The Single Source for Digital Products Service

DEC Module Repair, Exchange or Replacement

Photos show how we're fully equipped to repair and test DEC*, PDP-11* and TERMINAL MODULES. All modules we repair are 100% inspected.



Expert, Fast and Reliable Work

Serving the computer systems market since 1973, we are now troubleshooting, repairing and testing hundreds of modules weekly.

We can turn around a critical repair job in just 24 hours! Ten-day service is normal.

The modules we return to you are **guaranteed** to meet or exceed the manufacturer's specifications. And we prove it by submitting an inspection and test report.

For emergency service, 7 days a week, 24 hours a day, call

614/890-0939

DIGITAL PRODUCTS REPAIR CENTER

939 Eastwind Drive
Westerville, Ohio 43081
614/890-0939

*DEC and PDP are registered trademarks of Digital Equipment Corporation.


```

DECLARE INTEGER CONSTANT &
    STACK_SIZE          =      10000 ! This will determine &
                                ! The amount of space &
                                ! allocated for the &
                                ! stack. &
,
    STACK_HEADER_SIZE    =      16    ! The stack header is &
                                ! a quadword, defined &
                                ! as a RECORD above. &

!
DECLARE STRING &
    FUNCTION_OPTION      ! Stack function specified by user. &
,
    STACK_ITEM           ! String variable to store stack op- &
                                ! erands. &

!
DECLARE INTEGER &
    I, X                ! "I" for loops, "X" to call DEF's. &
,
    ENTRY_ADDRESS        ! Returned by LIB$REMQHI. &
,
    STACK_OFFSET         ! Index into stack structure, used &
                                ! in the "LIST" function and in the &
                                ! "POP" stack function. &

!
! Define shared memory .PSECT and list variable(s) which can be
! dynamically relocated therein at run-time.
!
MAP (STACK) &
    STACK_HEADER_FMT     STACK_HEAD &
    STRING              FILL=STACK_SIZE &
!
MAP DYNAMIC (STACK) &
    STACK_RECORD_FMT     STACK_RECORD &
    STRING              STACK_ENTRY &
!
%PAGE
%SBTTL      "Define Stack Manipulation Procedures"
DECLARE INTEGER FUNCTION &
    SETUP_STACK &
,
    PUSH_STACK (STRING) &
!
DECLARE STRING FUNCTION &
    POP_STACK &
!
DEF INTEGER STACK_EMPTY = (STACK_HEAD::F_LINK = 0%)
DEF INTEGER SETUP_STACK
    STACK_HEAD::F_LINK &
,STACK_HEAD::B_LINK &
,STACK_HEAD::DEPTH      =      0%

    STACK_HEAD::NEXT_SLOT_OFFSET = STACK_HEADER_SIZE
END DEF
DEF INTEGER PUSH_STACK(STRING STACK_ITEM)
! Position to next available stack slot, and update it:
REMAP (STACK) &
    BYTE FILL(STACK_HEAD::NEXT_SLOT_OFFSET) &
,
    STACK_RECORD &
,
    STACK_ENTRY = LEN(STACK_ITEM)
    STACK_RECORD::ENTRYSIZE = LEN(STACK_ITEM)
    STACK_ENTRY = STACK_ITEM
    X = LIB$INSQHI(STACK_RECORD, STACK_HEAD)
    X = LIB$INSQHI(STACK_RECORD, STACK_HEAD) &
        WHILE X = LIB$SECINTFAI
    X = SYS$EXIT(X) &
        IF (X AND SS$NORMAL) <> SS$NORMAL
    STACK_HEAD::NEXT_SLOT_OFFSET = STACK_HEAD::NEXT_SLOT_OFFSET &
        + STACK_HEADER_SIZE &
        + STACK_RECORD::ENTRYSIZE
    STACK_HEAD::NEXT_SLOT_OFFSET &
        = (STACK_HEAD::NEXT_SLOT_OFFSET*7%) AND (NOT 7%)
    STACK_HEAD::DEPTH = STACK_HEAD::DEPTH + 1%
END DEF
DEF STRING POP_STACK
! Don't bother if the stack is already empty.
EXIT DEF &
    IF STACK_EMPTY
    STACK_OFFSET = STACK_HEAD::F_LINK ! Save location of
                                ! POPped entry.
    REMAP (STACK) &
        BYTE FILL(STACK_HEAD::F_LINK) &
,
        STACK_RECORD &
,
        STACK_ENTRY = STACK_RECORD::ENTRYSIZE &
!
    POP_STACK = STACK_ENTRY
    X = LIB$REMQHI(STACK_HEAD, ENTRY_ADDRESS)
    X = LIB$REMQHI(STACK_HEAD, ENTRY_ADDRESS) &
        WHILE X = LIB$SECINTFAI
    X = SYS$EXIT(X) &
        IF (X AND SS$NORMAL) <> SS$NORMAL
    IF STACK_EMPTY THEN
! Next open stack slot follows the header.
    STACK_HEAD::NEXT_SLOT_OFFSET = STACK_HEADER_SIZE
    ELSE
! Next open stack slot is the address of the entry
! we just POPped.
    STACK_HEAD::NEXT_SLOT_OFFSET = STACK_OFFSET
    END IF
    STACK_HEAD::DEPTH = STACK_HEAD::DEPTH-1%
END DEF
%PAGE
%SBTTL      "Accept and Process User Commands"
X = SETUP_STACK
LINPUT "Option, please"; FUNCTION_OPTION
READ_LOOP:
UNTIL FUNCTION_OPTION = "" OR FUNCTION_OPTION = "EXIT"
    SELECT LEFT(EDIT$(FUNCTION_OPTION,-1%),4%)

```

```

CASE "PUSH"
    LINPUT " --String to Stack:"; STACK_ITEM
    X=PUSH_STACK(STACK_ITEM)
CASE "POP"
    IF STACK_EMPTY THEN
        PRINT " -- Can't - Stack is empty"
    ELSE
        STACK_ITEM = POP_STACK
        PRINT " -- POPped record:" &
            ; TAB(15%) &
            ; " Length=";LEN(STACK_ITEM) &
            ; TAB(35%) &
            ; STACK_ITEM
    END IF
CASE "LIST"
    PRINT
    PRINT "No. of Stack Entries = "; &
        STACK_HEAD::DEPTH
    PRINT
    PRINT "Stack" &
        ;TAB(15%) &
        ;"Entry" &
        ;TAB(25%) &
    PRINT "Slot#" &
        ;TAB(15%) &
        ;"Length" &
        ;TAB(25%) &
        ;" ---- Entry ----"
    PRINT " " &
        ;TAB(15%) &
        ;" " &
        ;TAB(25%) &
        ;" "
    PRINT
    STACK_OFFSET = STACK_HEAD::F_LINK
    FOR I=1 TO STACK_HEAD::DEPTH
        REMAP (STACK) &
            BYTE FILL(STACK_OFFSET) &
            STACK_RECORD &
            STACK_ENTRY = STACK_RECORD::ENTRYSIZE
        PRINT I &
            ;TAB(15%) &
            ;STACK_RECORD::ENTRYSIZE &
            ;TAB(25%) &
            ;STACK_ENTRY
        STACK_OFFSET = STACK_OFFSET &
            + STACK_RECORD::F_LINK
    NEXT I
    PRINT
CASE ELSE
    PRINT "Invalid function - try again."
END SELECT
LINPUT "Another option, please"; FUNCTION_OPTION
NEXT
END

```



MULTIFUNCTION BY DESIGN

**Spectra Logic delivers
single and multifunction
disk/tape controllers
for DEC users.**

Now you can improve system performance and reduce system cost with our popular family of DEC compatible disk and tape controllers.

We're Spectra Logic. The company that introduced the multifunction concept back in 1979. And we've been revolutionizing the controller market ever since.

Our multifunction disk/tape controllers provide the high-performance and added value you need to stay competitive. (Compare our \$2,900 SPECTRA 21 OEM price with separate disk and tape controllers.)

They eliminate the cost and need for separate disk and tape controllers, saving CPU slots, increasing reliability, and reducing power and spares requirements.

All of our controllers are smart, firmware-intensive, single board units. They support a wide range of independent disk and tape drives while emulating standard DEC subsystems. And they're backed by the industry's most comprehensive one year warranty.

Interested in single or multifunction controllers? Then contact the company that set the standard in the DEC market. Spectra Logic.

Call or write us today.

SPECTRA 12 PDP-11/VAX SMD Disk Controller

- Emulation of DEC RM02/05 & RK06/07 disk subsystems
- Attaches 4 SMD compatible disk drives
- Features 3 sector disk buffering, 32 bit ECC, overlapped seek support, dual port disk drives & automatic self-test microdiagnostics
- Media compatible with DEC RM02/05

SPECTRA 21 PDP-11/VAX Multifunction Disk/Tape Controller

- Industry's first multifunction disk & tape controller for DEC PDP-11 & VAX computers
- Simultaneous control of 4 SMD disk drives & 8 1/2-inch start/stop or "streaming" tape drives
- Emulation of DEC RM02/05, RK07 disk & TM11, TS11 tape subsystems
- Ideal for Winchester disk backup

The Multifunction Controller Company.

SPECTRA LOGIC

1227 Innsbruck Drive
Sunnyvale, CA 94086
Telephone (408) 744-0930
TWX 910 339-9566
TELEX 172524 SPL SUVL

Western Regional Sales Office:
(214) 934-9294
Eastern Regional Sales Office:
(216) 826-3137
New England District Office:
(914) 623-0502

©1983 Spectra Logic Corporation

SUMMER HEAT AND WINTER COLD

By Carl B. Marbach

At universities the first things air conditioned are mice and machines. Of the two, machines are less able to deal with environmental extremes. That may be the goal of controlling the computer room environment: keep away from all extremes.

One popular DEC computer lists the CPU operating specifications as:

Temperature: +10°C to +50°C

Humidity: 20% — 95% (without condensation)

Pretty wide limits. These are INTERNAL limits, and in order to keep the CPU at these temperatures you will need to provide quite a different ambient temperature and humidity range.

In the summer, an air conditioning system is necessary for most computers above the "micro" size. Some computers can operate comfortably in an office using the same air conditioning the office does; bigger computers will need a separate room and separate air conditioning.

In the winter, the heat from the computer can be used to warm a room, but you will need to continually supply cold air in some way or you will "cook" the computer up to unacceptable temperatures. Humidity is one of the biggest problems in winter, not enough of it. Winter air is dry and when you heat it, you lose more moisture. When the humidity is low, the static electricity is high and the shocks you get from pushing an elevator button are annoying to you but fatal to your computer.

There are several companies who specialize in air conditioning systems for computer rooms. They will work with raised floors and provide air up through these floors or through normal ductwork. The air handlers are located in the computer room and the compressors and heat exchangers are located away from the computer (on the roof or outside somewhere). These units will control temperature and humidity in both the winter and summer. Sizing an air conditioning system is important. If you choose one too small it won't cool the room. If you choose one too large, it will cycle on and off, and while it is off it will not be removing excess humidity. Too large a unit will result in too high humidity because it won't be on a cooling cycle long enough to effectively remove the water in the air.

If you are planning for growth, go for the modular approach. Plan on a separate air conditioner for each system or every two systems you install. This has the advantage of

giving you some redundancy. On all but the hottest days you will be able to squeeze by with only one working. I have had more computer downtime due to air conditioning than to the computers themselves in some years past. We actually installed a backup air conditioner in the computer room.

Air conditioning can be important in the winter. If the room is an inside room with no windows, it doesn't matter whether it's winter or summer, you'll need to air condition year 'round. If you can manage it, a computer room that could introduce outside cold air can save you some energy costs by using the (free) outside air to cool the room. Doesn't it sound silly to tell people that the computer is down because the air conditioning is broken when it is 15°F and the snow is 6 inches deep!

My choice for a computer room would be a room with an outside wall and no windows. I would mount two or more window units in the wall and keep a spare handy. In the winter I would draw the cold air directly from outside. If I added more computers I would add more window units in the wall. Window units are relatively inexpensive, can be fixed easily and are simple to swap. Our computer room air conditioning is under a service contract — with Sears.

There is one hazard connected with the use of window units. When the power fails very briefly, most window units will blow their breakers when the power returns, due to the back pressure in the compressor. They usually need a few minutes to bleed pressure before they are re-started. Some newer models take care of this for you. With older or less sophisticated units, it is necessary to provide a circuit that has a time delay relay and a contactor to provide the delay. Unattended computers get very hot with no air conditioning.

We have a humidifier that runs in the winter to keep the room from drying out too much. A raised floor or tile will keep the static under control while a carpet will help generate lots of electricity that will cause disks to go off-line and computers to crash. DEC loves to blame static electricity for the hardware problems they can't find, sometimes they are right and it is the static.

Heat is the enemy of microelectronics, keep them cool and they will provide long and faithful service. Let them get too hot and you will have hardware problems. I don't work with mice anymore, but I try to keep my computers comfortable.

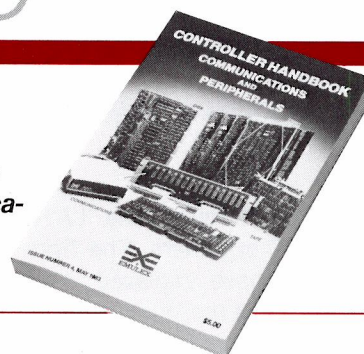


EMULEX TALKS DEC

5

WE WROTE THE BOOK ON DEC COMPATIBILITY...

The new Emulex Handbook (Issue No. 4) is hot off the press. It tells you everything you need to know about all Emulex disk, tape and communications controllers, including the celebrated new products introduced at NCC. Call or write to receive your free copy.



NEW PRODUCTS, PART I...

The UC01 has arrived! This single-board emulating host adapter provides full SCSI interfacing for a variety of controller/disk drives to DEC LSI-11 computers. The UC01 provides a parallel interface between the SCSI bus and the CPU. This allows up to seven controllers, usually resident in the peripheral, to be connected to each host adapter.

NEW PRODUCTS, PART II...

Emulex has something for PDP-11 and VAX-11/730 environments, too. Our new SC31 is a universal emulator for interfacing high-speed, high-density disk drives to these popular DEC computers. The SC31 handles almost any industry standard SMD disk drive in the 80 to 675 MByte range, with transfer rates up to 1.8 MBytes/second.

NEW PRODUCTS, PART III...

Don't think we overlooked VAX-11/750 and 780 users. For you, we've introduced two new disk controllers that will enhance your system's mass storage capability. The SC758 is a single-board controller that embeds directly into the backplane of a VAX-11/750. The SC788 is a single-board controller that plugs into the Emulex V-Master/780 Mass Storage Adapter, which installs as a sub-chassis within the VAX-11/780 CPU. Each controller board handles up to eight physical disk drives, ranging in capacity from 80 to 675 MBytes. Transfer rates up to 1.8 MBytes/second are supported by both controllers.

WE'VE GOT IT ALL FOR TAPE...

When you think of tape, think of Emulex. Our prices are competitive, our products reliable. We introduced the industry's first quad-sized TS11-compatible tape coupler series. For LSI-11 users, there's the TC02. For PDP/VAX-11s, try the TC12. Both models can handle every type of industry-standard formatted 1/2-inch tape transport. From conventional start/stop to streamers.

FOR THE RECORD...

The Emulex TC01 disk controller has a calculated MTBF of 41,000 hours. But in statistics compiled in field operations between 1980 and 1982, its actual MTBF was a whopping 164,930 hours! That's the equivalent of 31 years between failures, with the system in operation for 102 hours each week.



3545 Harbor Blvd., P.O. Box 6725,
Costa Mesa, California 92626,
Toll-Free (800) 854-7112, In Calif. (714) 662-5600.

RSTS/E VERSION 8.0

A FIRST REPORT: SURPRISE THE SYSGEN HAS A FATAL ERROR

By Mark E. Derrick, WAAY TELEVISION, 1000 Monte Sano Blvd, Huntsville, AL

The box arrived on Thursday, the "LATEST" distribution of RSTS/E Version 8.0 was here. The first thing was the documentation; they have replaced almost all of the manuals. My update kit is the W distribution, which means the changes have to be made to my existing documentation set. The fact that once updates to the documentation are made, I no longer have documentation for the software I am using has never bothered DEC. I open a few updates and glance through; well the changes aren't really much to worry about, mostly just rewrites to improve clarity. I decide to make the updates so the manuals can be read BEFORE the sysgen.

The binders have been reorganized, and somehow after everything is done I wind up with an extra one. The Volume 1 is now the Documentation Directory and the Maintenance Notes, now Volume 1A. Volume 2 is the Release Notes and a new Sysgen Manual. Volume 2A is the Managers Guide. The System User's Guide doesn't even have an update. Is that right? Yep, the READ ME FIRST says so. There is a new DCL manual; they are still on this idea of keyboard monitor command compatibility across operating systems. For some strange reason they moved the TECO manual from the Editing Volume to the Programming Utilities. Programmers must be the only people who use TECO any more. A new Task Builder Manual, maybe they fixed TKB! Nope, just updates for the new clustered libraries features. The new MACRO-11 manual looks the same as the old one. Why did they replace it? Nothing new for BASIC Plus, but a new Programming Manual to update the new SYS calls. A whole new RMS documentation set makes up Volumes 8 and 8A, with a new Introduction to RMS-11. The RMS User's Guide still requires intense concentration while reading. After the Maintenance Notes are combined with Software Dispatch Review the Volume One is bulging. Aha! I'll just take the spine tab from the 7.1 set labeled Volume 1A Maintenance Notes and put it on that empty binder which will be needed after the first Software Dispatch arrives. What is this little thing left over? It's a new Quick Reference Guide, but heavily oriented toward DCL. It doesn't matter, order three more — those quick guides will be a hot item around here.

Now for some homework, read the manuals. A "gotcha" first thing in the release notes; a new directory structure called RDS1. DEC had better have great plans for this because I can sure think of some things I would rather they spent time on. The V8 directory structure (RDS1) is not readable by version V7. That is going to be a pain. There is a program for an in place directory conversion from RDS0 to

RDS1, but it can't go backwards. Once it's run I'm committed to V8. That includes SAVRES, which means no RESTORE on V7 save set from V8. Other new goodies include prebuilt CUSPs. They are built with CSPCOM (.TSK) and so to make best use of them I'll have to take RSX as my default RTS. That's OK because the only reason I've avoided it in the past was it took so long to do a sysgen, and my users didn't get the familiar "Ready" prompt. With default KBMs and prebuilt CUSPs those reasons aren't valid anymore. Far out, it says that the maximum job size is 32kw! I'll have to make that huge swap file even bigger. An EMT logging feature to provide performance "tuning" information sounds nice until I read the fine print. It's on the same basis as monitor statistics, unsupported. I pay the same kind of penalties if I gen it in. The crazy part is only the raw data is available though FIRQB, the manual says it "requires a program (NOT SUPPLIED BY DIGITAL)." Digital giveth and Digital taketh away. The same with the new Micro-RSTS spooling package. It doesn't have batch and only provides one queue. With no way to see what's in it I'll pass on this one too. A lot of new things for DCL, which is starting to look attractive. It's a lot of trouble having to teach users how PIP works, and the new quick guide uses mostly DCL examples. A new RMS-11, which means re-taskbuilding everything, but I knew that was coming. They have changed the way to clean a dirty pack. The only way to do it under time sharing is with ONLCLN.SAV and it's not called a clean any more, it's called rebuild. They have done a few things to DSKINT that makes it nicer, and one thing that's a pain. It erases the disk when it's initialized, which will take a long time, but I suppose is for the best. Now some clown won't open a new file in his account and discover it comes complete with an old copy of ACCT.SYS or something worse. Speaking of ACCT.SYS, casual reading of the System Manager's Guide has a figure containing the listing of one for V8. I notice it's not the same format. That means entering 150+ accounts by hand! Not a word about that in the documentation. Well that seems to be about it for the stuff I need to know before the sysgen. Lots of changes to other CUSPs but that's not important right now.

Am I ready to do a sysgen? Seems like it . . OH NO! That damn new disk structure, what will it do to my third party software? I had better start calling people and asking. MANUS says that DSKBLD will not even fail gracefully, they are working on a new release. Seems like I remember hearing at DECUS that RMS DIBOL 4.5 might not work with RSTS V8 and RMS V2, they were planning to ship DIBOL 5.0

at the same time as RSTS V8. Quick call to Karen Hinds at DEC software maintenance; it's news to her — nothing she's got even mentions DIBOL V5. She'll make some calls and get back with me (yeah and my name is Ken Olsen). Incredible, she calls back in an hour. DIBOL V5 won't be around for at least 90 days, and I was right about 4.5 not working. She's got some unpublished patches on the way to her. At least DEC occasionally hires competent non-technical people who return calls. There's a software specialist in Birmingham who people in Huntsville swap tales about — "It took me eight calls to get him . . ."

So, let's take a summary of where I stand. I'll need new releases of my third party stuff before upgrading to V8 in the production environment. I need to upgrade because I'm still at V7.1 and I hear DIBOL V5 is going to make my programs run like the wind. When I do go to V8 everything had better work; I can't change back to V7. I think what I'll do is sysgen RSTS V8 now and that will give me a month or two to boot it up at nights and on weekends. I will have plenty of time to put in my performance and security feature patches, tailor my startup files, and delete the stuff I don't need like GRIPE, enter my accounts, even set up a few batch files to make the conversion

to a production environment easier. Then if something doesn't work I'm not in crisis mode until it gets fixed. This sounds pretty good, even if the real reason is that I can't stand waiting to see what it looks like.

I tried to save some money and bought my updates on 1600 BPI magtape even though I don't have a tape drive. So

that means trying to find someone who will let me borrow a machine. After calling in a few favors I have a site that will let me use their machine after hours if someone can stand over my shoulder and learn the new sysgen procedures. Fine, I'll be there Monday night at 5PM. What I plan to do is

create a "pseudo" RK07 distribution out of the magtape distribution. Then I can use disk for any future sysgens. I think I'll read some more until Monday.

The appointed day arrives. A receptionist says the "VP of software" will be out in a minute. A few minutes later and we are standing before an 11/34 with 128K, dual RK07s, dual RL02s, and a TS11. "Field Service just put some more TS11 ECOs in today, it seems to be better now," he says. I remember 7.1 wouldn't even boot up on a TS11. I mount up the SYSGNK tape on the "beloved" TS11 and thank god it BOOTS! I DSKINT two RK07 packs with packids of SYSGNK and SYSGEN, do a HARDWARE LIST to make sure V8 finds everything on the machine, and then a COPY/ALL to drive 0. Next comes an IN-STALL, a REFRESH to make a small SWAP.SYS, then a DEFAULT setting RT11 as my Run Time System, and finally a START. Half a page of stuff the sysgen monitor can't find and disables, then "?Can't find file or account"

and the RT11 "...", lookin' good.

Now run CREATE.SAV from the tape, and in come LOGIN, PIP, UTILITY, SYSGEN and a few other things with SAV extensions. (Question: Since they put all those pre-built CUSPs out on the distribution, why don't they make RSX the sysgen monitor, and do away with having SAV, TSK, and

More than just a word processor

Q'TEXT[®]

...a text management system for RSTS

Q'TEXT gives you

- proportional spacing
- phototypeset, letter- or draft-quality text
- compatibility with a wide range of printers (Qume, DEC, Diablo—as well as type-setting equipment)
- compatibility with standard RSTS files
- speed and efficiency—written in Macro assembler
- alternative character sets—for Spanish, French, Arabic, scientific writing
- extended screen display—lines up to 130 characters long
- cut-and-paste operations on columns as well as lines
- accounting with electronic spread sheets
- immediate screen access to an entire file

standard Q'TEXT features include

screen entry, editing and formatting of text • justified, flush-left, flush-right and centered text displayed on screen • underlining, boldface • deletion, transfer and copying of words, lines and passages • searches and changes • automatic carriage return • merging • sorting • variable line spacing • definable keys

Q'TEXT...was developed at Queen's University, Canada.
More than 1,000 units have now been sold worldwide.

Special introductory offer...

\$1995

until October 31, 1983

BERRN Research

Seaway Building
310 Bagot Street
Kingston, Ontario, Canada
K7K 3B5
(613) 549-7876

BERRN Research is a registered user of Q'TEXT

Also available for RT11, TSX, P/OS and RSX

DEC, RSTS, RT11, P/OS and RSX are trademarks of Digital Equipment Corporation

TSX is a trademark of S & H Computer Systems, Inc.

CIRCLE 197 ON READER CARD

BAC versions of these things?) CREATE chains to SYSGEN, but I'm cheating and control C out of it. Next PIP the entire SYSGNK and CSP180 tapes to disk. I would like to copy the PATCHA tape also, but there is no way to create a [200,200] account on the disk and no RTS to be able to run PATCPY from the tape. Finally SHUTUP, and return to option on my "pseudo" SYSGNK RK07 disk distribution.

Now I start the whole thing over again, except using the drive 0 (DM0:) pack as my RK07 distribution. COPY/ALL which boots me onto drive 1 (DM1:), INSTALL, REFRESH with a swap file for 40 jobs and a 300 block CRASH.SYS, DEFAULT, and a START. Then run CREATE from DM0:, this time however we are going to do a sysgen.

I take the long form questions, every little bit helps. In the setup questions I tell it that DM is my distribution medium, yes to delete files, patching from MS, yes to BASIC, and RSX will be my default RTS. In the hardware configuration questions things get a little more difficult because I want support for both the target machine plus the machine I'm running on. Some people gen in stuff for everything they might ever plan to buy or think they might need in an emergency. I have always avoided doing that, because SILs can be sysgened pretty quickly on-line if I buy something new. The SYSGEN SIL has the kitchen sink already and is the best choice if I need to support something at an emergency site on a foreign machine. Then come the SIL features questions. If I'm not careful I can gen a real performance dog. The new Sysgen Manual has been a real help here, giving me tables and worksheets to use to compute my answers. I decide to take 40 jobs, the default number of small buffers, 50 system logicals, I do take monitor statistics (if bit 15 of switch register is off it doesn't cost anything but monitor size), no to EMT logging, yes to directory and data caching, yes to resident libraries and RSX directives. (Why do they bother to ask?) My target machine has 384kw of memory so I make all the overlay code and sys call stuff resident. Next comes BASIC, seems like I remember that I can't take four word math and everything else, even though I can't find anything about it in the manuals. Anyway I take two word math, and everything else including MAT functions — I need that for STAR TREK Version 215.12A and some other important stuff. Done, it says to RUN \$SYSBAT, ok I'll go along — it starts working and I send out for a couple of burgers.

After dinner (dinner?) everything looks good, no errors it didn't warn me about, and nothing strange in the listings. So SHUTUP the system, INSTALL my new SIL, do some SETing, DEFAULT with RSX my new RTS and some XBUF, check the REFRESH listing to make sure CRASH.SYS is big enough, and then START. Now I run PATCPY from tape, but put the PATCHA files in [200,200] on DM0: my "pseudo" distribution disk. It's time to run BUILD. I built an RSX default RTS system ONCE before and it took hours and hours to compile all that stuff. I hope I've made the right decision and these prebuilt CUSPs are the way to go. Now how do I take the prebuilt CUSPs? Oh just read the manual — when it asks if I want to use CSPCOM for the compiles I answer yes. Evidently it knows to look for the pre-built TSK stuff. RUN DM0:BUILD and first I'm supposed to build RSX. It wants to know if I want to move LB: from [1,1]. Well with

V7 I almost had to, but with the new directory structure it doesn't matter so let it go as is. The patch files I have cleverly placed on the "pseudo" distribution. Finally do I want to do anything else ("Additional Control File?"); this first try I think not. Off it goes and here come the error messages, so control C and start looking at what's happening. Mostly protection violations, so I quickly discover that the protection codes on the "pseudo" distribution disk are strange: <10> and other weird stuff. It had to do with the way I copied from tape to disk, a minute with PIP/RE and everything is back to something normal. I rerun BUILD, and WONDERFUL — everything goes fine.

By now I'm feeling pretty good, so RUN DM0:BUILD with the BUILD control file for the system library and with patching. (They really ought to name that control file SYSLIB or something less confusing.) A strange thing happens, I get a couple of error messages while it's copying the command file to the BLD01.TMP control file. They look like:

```
?DM1:[200,200]PA1012.009/CS:39283—?Can't find file or account
?DM1:[200,200]PA1208.001/CS:7658—?Can't find file or account
```

However, I'm not concerned because they are just patch files. I can't find patches with those reference numbers; 10.12 would be LOGIN and 12.08 would be ERRBLD. So I just go on specifying control files: DCL, BIGPRG, HELP, SPLER, DEVTST, TECO, EDT, RMS11, UNSUPP. Any time it offers me a chance to put some part of a package anywhere other than [1,2] I take it and wind up moving the error, spooling, help and device testing packages to [1,3]. After about 45 minutes of this it finally starts the BUILD, and one minute into the BUILD control file I get:

```
File to patch — SY:[1,2]LOGIN.BAS=DM1:[1,2]LOGIN.BAS
#
*1Z
?eof
%Patch from KB:[1,2]CPATCH.CMD skipped.
#
?Error in job
?Command file aborted
!*** Processing ended on 16-MAY-83 at 11:39PM ***
```

I play around for a while and discover that I can't do a system library build. This problem took two hours to figure out and I won't bore you with the details of trying to find out what was happening when I had no system library to use to help me. The patch command files on the tape requested two patches be applied that are not on the tape. (Why is a mystery. Obviously nobody at DEC did a sysgen with PATCHA.) When BUILD cannot find the patches it becomes confused and places some strange things in the BUILD command file causing the build to be aborted. The solution is to build TECO first, then edit PA1012.CMD and PA1208.CMD to comment out the requests to apply PA1012.009 and PA1208.001 patches. Another work around would be to just build but don't patch, there doesn't appear to be anything in the patches that is absolutely required. When PATCHB comes out the problem will go away.

After the problem with the BUILD control file everything goes fine. I rerun BUILD (again!) and specify all those control files (again!). I can't believe what a difference the prebuilt CUSPs make. It's whizzing through the builds faster than if it were doing a BASIC-Plus default system. This is GREAT — until it gets to the UNSUPP control file. The

VAX DEC-10 DEC-20 VAX

SCOPE

FORMS MANAGEMENT MADE EASY

SCOPE is a forms management system which provides an interactive method of designing, creating and modifying video forms.



SCOPE
has been accepted
into DEC's EAS library

- **SCOPE** runs on VAX, DEC-10, DEC-20.
- **SCOPE** can operate on any terminal.
- **SCOPE** works with any language and any database.

SCOPE helps reduce programming and maintenance time of an on-line application using these features:

- Date Validation
- Data Duplication
- Automatic Screen Definition
- 80 or 132 Column Lines
- Line Mode for use on Networks
- All Video Attributes
- Interactive Screen Editor
- Hardcopy Generation of Screens
- Bounds Checking
- No Echo for Password Security

Send for free evaluation tape



INTERACTIVE
SYSTEMS
INC.

*Distributor
Inquiries Encouraged*

☐ Please send me information on financial software

Name _____

Title _____

Company _____

Address _____

City _____ State _____ Zip _____

Hardware _____

INTERACTIVE SYSTEMS, INC.

131 Middlesex Tpk., Burlington, MA 01803 • 617-273-4420

VAX, DEC-10, DEC-20 are registered trademarks of
Digital Equipment Corporation

build craps out again. It turns out that if I take the pre-built CUSPs they are just copied from the distribution. Except the pre-built CUSPs for the UNSUPP package aren't on the distribution, it says so in the release notes. (No one's perfect — I missed it.) There are but three missing, so I just compile them by hand — see section 5.4 of the Maintenance Notes for instructions. Then I copy the TSKs back to the "pseudo" distribution, and rerun the UNSUPP build.

That's it, done with the sysgen. I delete the SAV versions of some things off the sysgen disk, delete a few things left around such as TMPs and MAPs, a quick edit or two for some important stuff in the startup control files and SHUTUP the system. Image copy my fixed up "pseudo" RK07 distribution and START, a few linefeeds, a couple of minutes and then:

RSTS/E VERSION 8.0 IS ON THE AIR . . .

It's three o'clock the morning and I still have DIBOL and DATATRIEVE to install, but that's another story. I can't believe Mike Spencer at Information Systems sat through all this, thanks Mike.

ABOUT THE AUTHOR

Mark Derrick lives in Alabama with his hyperactive cat named "Streak," who loves keeping the author awake after all night sysgens. Mark works as a system analyst for a radio and television broadcaster not afraid to seek data processing solutions to problems using their inhouse PDP-11/44 CTS-500 system. He will be presenting a session at the 1983 Fall DECUS U.S. Symposium titled "EASYLINK: Communicate With The World." ♥

RSTS
PROFESSIONAL
AVAILABLE ON
MICROFICHE
DIRECT INQUIRIES TO:
MICRO PHOTO DIVISION
 **BELL & HOWELL**
OLD MANSFIELD ROAD
WOOSTER OH 44691
Contact Christine Ellis
Call toll-free (800) 321-9881
In Ohio, call (216) 264-6666 collect



WRITING AST ROUTINES IN VAX-11 BASIC

By Kevin Paul Herbert

Copyright © 1983 by Software Techniques, Inc.,
5242 Katella Ave., Suite 101, Los Alamitos, CA 90720.

1.0 Introduction

The purpose of this article is to describe a technique for declaring asynchronous system trap service routines (AST routines) from VAX-11 BASIC, and servicing them from VAX-11 BASIC.

2.0 Background

AST routines can be used for many purposes. One use is asynchronous I/O, which can be used to write high-performance software. Other uses include supporting mailboxes for inter-process communication and using the "out-of-band AST control character feature" in the terminal driver to write action routines for unused (by VMS) control characters.

Information on AST routines and their application can be found in the VAX/VMS System Services Reference Manual and the VAX/VMS I/O User's Guide.

3.0 Declaring an AST routine

The VAX/VMS system services that support ASTs expect to be passed the address of the AST routine. There is not a documented way in BASIC to pass the address of a routine.

To pass the address of an AST routine to a system service, it is necessary to declare the routine an EXTERNAL CONSTANT. This tells BASIC that it is concerned with the virtual address of the routine. Note that if an AST routine is declared as an EXTERNAL CONSTANT, it can not be called directly (but will be called when the AST occurs). When it is necessary to pass the address of the AST routine to the system service, it needs to be passed BY VALUE. This tells BASIC to pass the value of the global symbol associated with the start of the routine, which is what the system service expects.

4.0 Writing the AST routine

To write an AST routine in BASIC, declare it as a subroutine. VAX/VMS calls AST routines as if they are subroutines, passing special parameters. The SUB statement should be written as follows:

```
10 SUB ASTROUTINE ( LONG ASTPRM BY VALUE
    , LONG R0 BY VALUE
    , LONG R1 BY VALUE
    , LONG PC BY VALUE
    , LONG PSL BY VALUE )
```

The variable ASTPRM will contain the AST parameter that was specified in the system service. The variable R0 will contain the value of processor general register R0 at the time of the AST. The variable R1 will contain the value of processor general register R1 at the time of the AST. The variable PC will contain the value of the program counter at the time of the AST. The variable PSL will contain the value of the processor status longword at the time of the AST. Since these values are passed BY VALUE, they can not be modified by the subprogram.

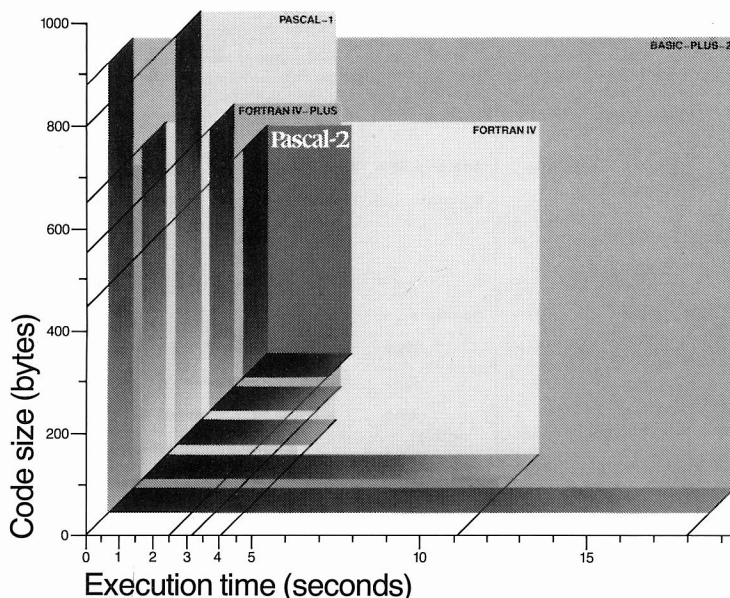
5.0 Conclusion

It is possible to declare and write AST service routines in BASIC. For further information on uses of the AST facility, see the VAX/VMS documentation.



Pascal-2TM

PDP-11 RSX,
RSTS/E, RT-11
and TSX-Plus



Quicksort benchmark executed on a PDP-11/45.

The Pascal-2 system consists of an optimizing compiler, a high-level debugger and other utilities.

Write us for benchmark details.

Oregon Software

The Pioneer in Performance Pascal

2340 S.W. Canyon Road
Portland, Oregon 97201
(503) 226-7760
TWX: 910-464-4779

FORTRAN IV-PLUS, BASIC-PLUS-2, PDP, VAX, RSX, RSTS/E and RT-11 are trademarks of Digital Equipment Corporation. TSX-Plus is a trademark of S&H Computer Systems.

CIRCLE 60 ON READER CARD

TAPE COPY

By W. Franklin Mitchell, Jr., Computer Operations Supervisor, Erskine College, Due West, South Carolina

TPECPY.BAS is a BASIC-PLUS program that enables users of RSTS systems with just one tape drive to make exact duplicates of DOS format magnetic tapes. TPECPY achieves this by moving all tape files to temporary disk storage under TPECPY names (TC1001.TMP, TC1002.TMP, TC1003.TMP, ...) and writing a PIP command file (WRITE.CMD) to move the disk files to a new tape using the original [p,pn] and file name. TPECPY also writes a PIP command file (CLEAN.CMD) that will erase all of the TPECPY temporary disk storage.

The procedure for making copies of tapes is outlined in line 130 of the program listing. Please note the TPECPY expectations that are listed in line 140. Edit line 210 if your tape drive is not named "MTO:".

Erskine College can provide a tape copy of TPECPY.BAS. If a copy is desired, send \$12.00 and a tape or \$18.00 and no tape to Tape Copy, Erskine College, PO Box 86, Due West, SC 29639. Please specify 800 or 1600 BPI.

After the program listing follows an example RUN of TPECPY.

```

1      EXTEND
2!
!
!      T A P E   C O P Y
!
3!      Program:      TPECPY.BAS
5!      Version:      7.0 - 1.0
6!      Date:         1-May-82
7!      Author:       W. Franklin Mitchell, Jr.
!                   Computer Operations Supervisor
!                   and Instructor in Mathematics
!                   Erskine College, Due West, South Carolina
8!
!      Copyright (c) 1982 by
!      Erskine College, Due West, South Carolina
!
! *****
! * This software is furnished without charge by Erskine College and *
! * may be copied only with the inclusion of the author's name and *
! * copyright notice. No title to or ownership of this software is *
! * hereby transmitted. Neither Erskine College nor the author assumes *
! * any responsibility for the use or reliability of this software. The *
! * author welcomes comments and/or bug reports mailed to Franklin *
! * Mitchell, Erskine College, Box 86, Due West, South Carolina 29639. *
! *****
!
10!     D I S T R I B U T I O N   L O G
!
!      Ver/Ed      When      To
11!      7.0-1.0      May 1982      DECUS Atlanta
12!      7.0-1.1      January 1983  RSTS/DEC PROFESSIONAL Magazine
!
!      M O D I F I C A T I O N   H I S T O R Y
!
!      Ver/Ed      Edit Date      Reason
21!      7.0-1.1      11-Jan-83      Handle "User data error on device"
```

```

100      !
!      G E N E R A L   D E S C R I P T I O N
!
110!     Tape Copy is designed to allow users of RSTS systems with just one
!     tape drive to make exact duplicates of DOS format magnetic tapes.
!     That is, to copy a tape to temporary disk storage and then to move
!     this data to a new tape preserving the original tape's [p,pn]'s,
!     ordering of files, and redundancy of files.
!
120!     Tape Copy
!
!     1) waits until the tape drive is ready.
!
!     2) asks which disk(s) (plus [p,pn] for a privileged user) to use
!     as temporary storage.
!
!     3) moves the data from the original tape to disk.
!
!     4) writes a PIP command file (WRITE.CMD) that will move the data
!     from disk to a new tape.
!
!     5) writes a PIP command file (CLEAN.CMD) that will remove the
!     temporary disk files.
!
!
130!     What to do:
!
!     1) Load the tape to be copied with no write ring and
!
!         RUN TPECPY
!
!     2) Load a new tape with a write ring and type
!
!         PIP @WRITE
!
!         Repeat step 2 until you have made as many copies as desired.
!
!     3) Type
!
!         PIP @CLEAN
!
!         to remove the tape files from the temporary disk storage.
!
140!     N O T E :
!
!     TPECPY.BAS expects
!
!     (a) DOS labeled magnetic tapes.
!
!     (b) enough free disk to store the whole tape.
!
!     (c) PIP V7.0-07 or equivalent.
!
!     (d) 8,999 files per tape or less.
!
!     (e) no files named TC?????.TMP on any of the disks/accounts
!         that are used to temporarily store the tape's files.
!
200      !
!      T A P E   D R I V E
!
210     TAPE.DRIVE$ = "MTO:"
!
!     Edit this line if your tape drive is named something else
!
300      !
!      I / O   C H A N N E L S
!
301!     Channel #      Used for
!
310!     CMD%      WRITE.CMD and CLEAN.CMD files.
!
!     TPE%      Magnetic tape drive.
!
!     DSK%      Temporary disk files which are copies of the tape's
!               files.
!
320     CMD% = 1%
!     TPE% = 2%
!     DSK% = 3%
!
800      !
!      F U N C T I O N   D E S C R I P T I O N
```


**What you
don't know
about your
DEC*
computer
can't
help you.**

DEXPO® West 83 can.

400 Reasons to Attend

Wouldn't you like to get just a little more out of your DEC* system? All it takes these days are the right DEC-compatibles. And you'll find them by the thousands at DEXPO West 83. Your most complete DEC-compatible resource.

Want *more* reasons? How about *more* compatible software, *more* compatible hardware, and *more* compatible services? DEXPO West features nearly 400 vendors with more for every DEC computer. Personals, Minis. Mainframes. All the latest DEC-compatibles designed to make your information systems work a little harder. And a lot smarter.

*DEC and DECUS are registered trademarks of Digital Equipment Corp.

**Attending DECUS*?
Then, DEXPO is Free**

If you're planning to be in Las Vegas for the DECUS meetings, make sure DEXPO West

is on your schedule for Sunday . . . before the meetings start. Of course, you'll probably want to come back on Monday, Tuesday and Wednesday, too. So be sure to use our five-minute shuttle bus services. It's free. And so is the Show when you bring your DECUS badge.

**Plus 60 In-Depth Sessions on How to
Put the Latest DEC-Compatibles
to Work**

DEXPO® West 83

The Fourth National
DEC-Compatible Industry Exposition

**Las Vegas
Convention Center
October 23-26, 1983
Sunday-Wednesday**

```

801! Function Use
810! FNPPN$(X$) extract PPN from DOS tape header
820! FNINTRAD$(X$) extract file name from DOS tape header.

900 !
! DIMENSION STATEMENT

910 Dim DISK.ACCOUNT$(32)

920 DISK.ACCOUNT.LIMIT% = 32%
!
! Limit of number of disks/accounts that can be used for temporary
! storage. Match limit in DIM statement

930 WATCHING% = -1%
!
! 0% = no watching, -1% = watching (like /W switch of PIP)

1000 !
! START

1020 On error goto 19000
\ Print if CCPOS(0%)

1040 Print
\ Print "Erskine College Tape Copy V7.0 - 1.1 ";
\ Print TIME$(0%); " "; DATE$(0%)

1060 PRIVILEGED% = 0%
\ PRIVILEGED% = PEEK(512%)
!
! If no error, we are privileged.

1080 Print
\ Print "Please mount the tape to be copied on "; TAPE.DRIVE$;
\ Print " with no write ring."
\ Print
\ Input "Tape drive ready <Yes>"; GO$
\ GO$ = CVT$(GO$, -1%)
\ GO$ = "Y" if LEN(GO$) = 0%
\ Goto 1080 unless ASCII(GO$) = ASCII('Y')

1100 Print
\ Input "Tape density (800 or 1600) <drive default>"; DENSITY$
\ DENSITY$ = CVT$(DENSITY$, -1%)
\ Goto 1100 if not (DENSITY$="800" or DENSITY$="1600" or LEN(DENSITY$)=0%)
\ E%, D% = 0%
\ If DENSITY$ = "800"
\ then
\ E% = 0%
\ D% = 3%
1120 If DENSITY$ = "1600"
\ then
\ E% = 256%
\ D% = 0%

1140 Open TAPE.DRIVE$ as file TPE$, mode 64% + E% + (D% * 4%)

1160 Field #TPE$, 512% as TPE.BUF$
\ Field #TPE$, 2% as FILE.NAME.FRONT$, 2% as FILE.NAME.REAR$,
\ 2% as FILE.NAME.EXT$, 1% as PROGRAMMER$,
\ 1% as PROJECT$

1180 I% = MAGTAPE(3%, 0%, TPE%)
!
! Rewind the tape.

1200 I% = MAGTAPE(7%, 0%, TPE%)
\ If (I% AND 1024%) = 0%
\ then
\ Print
\ Print "%Tape is write enabled."

1220 Goto 2000

1240 Print
\ Print TAPE.DRIVE$; " is off-line. Please fix."
\ Print
\ Input "Drive on-line <Yes>"; GO$
\ Goto 1140
!
! Handle tape off-line error.

2000 !
! GET WHICH DISK(S) TO USE

2020 Print
\ Print "Please define the disk(s)";
\ Print "/accounts(s)"; if PRIVILEGED%
\ Print " to use for temporary storage:"
\ DISK% = 0%

2040 Print
\ Print using "###. Disk", DISK% + 1%;
\ Print "/account"; if PRIVILEGED%
\ Print " <no more>";
\ Input line DISK$
\ DISK$ = CVT$(DISK$, -1%)
\ Goto 2100 if LEN(DISK$) = 0%
\ DISK.ACCOUNT$(DISK%) = DISK$
\ If INSTR(1%, DISK$, ".") = 0%
\ then
\ Print
\ Print 'Missing "." in device name. Try again:'
\ Goto 2040

2060 Open DISK$ + "TPECPY.TMP" as file DSK%
\ Close DSK%
\ Kill DISK$ + "TPECPY.TMP"
!
! If this causes an error, we don't have access to DISK$
! or DISK$ does not exist.

2080 DISK% = DISK% + 1%
\ Goto 2040 if DISK% <= DISK.ACCOUNT.LIMIT%

2100 NR.OF.DISKS% = DISK% - 1%
\ If NR.OF.DISKS% < 0%
\ then
\ NR.OF.DISKS% = 0%
\ DISK.ACCOUNT$(0%) = "SY:"

2120 Goto 3000

2140 Print
\ Print "You can't create files on "; DISK%; ". Try again:"
\ Goto 2040
!
! Handle illegal disk or private disk errors.

3000 !
! BUILD WRITE.CMD

3020 Open "WRITE.CMD" for output as file CMD%
\ Print #CMD%, TAPE.DRIVE$; "/ZE";
\ If LEN(DENSITY$) = 0%
\ then
\ Print #CMD%
\ else
\ Print #CMD%, "/DEN:"; DENSITY$

3040 !
! PROCESS TAPE DATA

3050 DISK% = 0%
\ FILE.NUMBER% = 1%
\ TOTAL.BLOCKS = 0.
\ Print
\ Goto 3060 if not WATCHING%
\ Print "Now processing:"
\ Print
\ Print "File # ----- File Name ----- Blocks"
\ Print

3060 Get #TPE%
!
! Read DOS header block

3080 TAPE.FILE$ = TAPE.DRIVE$
+ "[" + FNPPN$(PROJECT$) + ", " + FNPPN$(PROGRAMMER$) + "]"
+ FNINTRAD$(FILE.NAME.FRONT$) + FNINTRAD$(FILE.NAME.REAR$)
+ ". " + FNINTRAD$(FILE.NAME.EXT$)

3100 DISK.FILE$ = DISK.ACCOUNT$(DISK%)
+ "TC" + NUM1$(1000% + FILE.NUMBER%) + ".TMP"

3120 Print #CMD%, CVT$(TAPE.FILE$, 2%) "="; DISK.FILE$
\ If WATCHING%
\ then
\ Print using "###,###", FILE.NUMBER%;
\ Print TAPE.FILE$; SPACE$(1%);

3200 !
! MOVE TAPE'S FILE TO DISK

3220 Open DISK.FILE$ for output as file DSK%
\ DISK.BLOCK.COUNT, BAD.BLOCK.COUNT = 0.

3240 Get #TPE%
3250 Put #DSK% + SWAP$(TPE%)
\ DISK.BLOCK.COUNT = DISK.BLOCK.COUNT + 1.
\ Goto 3240

3260 Close DSK%
\ TOTAL.BLOCKS = TOTAL.BLOCKS + DISK.BLOCK.COUNT
\ If WATCHING%
\ then
\ Print using "###,###", DISK.BLOCK.COUNT;
\ If BAD.BLOCK.COUNT = 0.
\ then
\ Print
\ else
\ Print using " ###,### %bad blocks", BAD.BLOCK.COUNT

3280 FILE.NUMBER% = FILE.NUMBER% + 1%
\ If DISK% < NR.OF.DISKS%
\ then
\ DISK% = DISK% + 1%
\ else
\ DISK% = 0%
!
! Use disks/accounts in a round-robin fashion

3300 Goto 3060

4000 !
! CLOSE, BUILD CLEAN.CMD, AND QUIT

4020 Close CMD%, TPE%, DSK%
\ FILE.NUMBER% = FILE.NUMBER% - 1%
\ Print
\ Print using "### file", FILE.NUMBER%;
\ Print "s"; if FILE.NUMBER% < 1%
\ Print using " ###,### block", TOTAL.BLOCKS;
\ Print "s"; if TOTAL.BLOCKS < 1.
\ Print

```




MEETS THE DEC® DH-11 CHALLENGE

The challenge; increased communications capacity, increased reliability, and improved technology – at decreased cost.

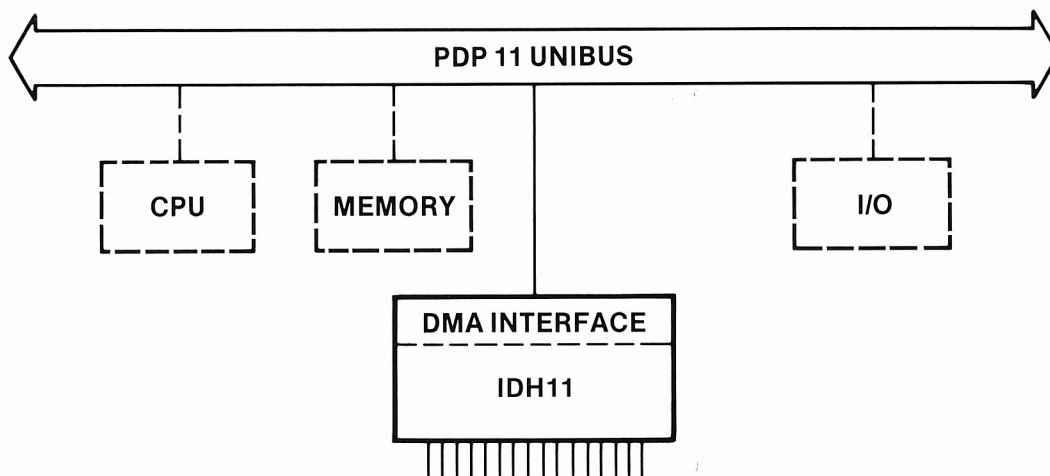
That's not a challenge for Intersil Systems.

- **Increased Capacity** – One Intersil DH-11 replaces nine DEC card slots
- **Increased Reliability** – Only one chance for a failure instead of nine.
- **Improved Technology** – The latest microprocessor based technology is employed.
- **Decreased Cost** – 66% less! One Intersil DH-11 costs 66% less than one DEC DH-11.

In addition the Intersil DH-11 offers these features:

- 16 Asynchronous local or remote channels on the UNIBUS®
- DMA output to free the CPU from Interrupt handling.
- On-board diagnostics.
- Complete software compatibility.

IDH11



16 ASYNCHRONOUS CHANNELS
TO LOCAL OR REMOTE TERMINALS

Get all the challenging facts. Call (408) 743-4300, TWX: 910-339-9369, or write Intersil Systems, Inc., 1275 Hammerwood Avenue, Sunnyvale, CA 94086

® Trademarks of Digital Equipment Corporation

CIRCLE 171 ON READER CARD

```

4040  Open "CLEAN.CMD" for output as file CMD%
4060  Print #CMD%, DISK.ACCOUNT$(1%);
      "TC????T.MP/DE/NO/W" for 1% = 0% to NR.OF.DISKS%
\    Print #CMD%, "WRITE.CMD/DE/NO/W"
\    Print #CMD%, "CLEAN.CMD/DE/W"
\    Close  CMD%

4080  Print
\    Print "Now mount a new tape on "; TAPE.DRIVE%; " and type"
\    Print
\    Print "PIP @WRITE"
\    Print
\    Print 'Repeat the above "mount" and "PIP" steps until all copies';
\    Print ' you want are made.'
\    Print
\    Print "Then type"
\    Print
\    Print "PIP @CLEAN"

4100  Goto 32767

15000 !
      !      F U N C T I O N      F N P P N $

15010 Def* FNPPN$(X%)
\    X% = NUM1$( ASCII(X%) )
\    FNPPN$ = SPACE$( 3% - LEN(X%) ) + X%
15020 Fend

15030 !
      !      F U N C T I O N      F N I N T R A D $

15040 Def* FNINTRAD$(X%) = RAD$( SWAP$( CVT$(X%) ) )
\
\    Convert a string to a Rad50 integer and then to characters.

19000 !
      !      E R R O R      P R O C E S S I N G

19020 Resume 3260 if ERR = 11% and ERL = 3240%
\
\    End of file on a tape file?

19040 Resume 4020 if ERR = 11% and ERL = 3060%
\
\    Are we done?

19060 Resume 1240 if ERR = 39% and ERL = 1140%
\
\    Tape drive off-line?

19080 Resume 2140 if ERL = 2060%
\
\    Can't write on that disk.

19100 Resume 1080 if ERR = 10% and ERL = 1060%
\
\    Can't do a PEEK() because we are not privileged.

19120 If ERR = 8% and ERL = 1140%
      then
        Print
        \    Print "?Sorry "; TAPE.DRIVE%; " is not available."
        \    Close CMD%, TPE%, DSK%
        \    Goto 32767

19140 If ERR = 13% and ERL = 3060%
      then
        BAD.BLOCK.COUNT = BAD.BLOCK.COUNT + 1.
        \    TAPE.FILE$ = TAPE.DRIVE$ + "[254,254]TPECPY.ERR"
        \    Resume 3100
\
\    User data error on tape drive prevents reading file name

19160 If ERR = 13% and ERL = 3240%
      then
        BAD.BLOCK.COUNT = BAD.BLOCK.COUNT + 1.
        \    Lset TPE.BUF$ = "%Bad tape block #" + NUM1$(BAD.BLOCK.COUNT)
        \    Resume 3250
\
\    User data error on tape drive makes us miss a block

19990 Print
\    Print "Unexpected Tape Copy error:"
\    Print
\    Print CVT$( RIGHT( SYS(CHR$(6%) + CHR$(9%) + CHR$(ERR)), 3% ), 4% );
\    Print " at line "; NUM1$(ERL); "."
\    Close CMD%, TPE%, DSK%

32767  END

PIP MT:[*,*]/L

Name .Ext      Size      Prot      Date      MT:[1,2]
README.1ST      3      <155> 10-Jan-83
TPECPY.RNO      3      <155> 10-Jan-83
TPECPY.BAS      30     <155> 10-Jan-83

Total of 36 blocks in 3 files in MT:[1,2]

Grand total of 72 blocks in 6 files in MT:[*,*]

Ready

RUN TPECPY

Erskine College Tape Copy V7.0 - 1.1 11:38 AM 12-Jan-83

Please mount the tape to be copied on MT0: with no write ring.

Tape drive ready <Yes>?

Tape density (800 or 1600) <drive default>?

Please define the disk(s) to use for temporary storage:

1. Disk <no more>? DM0:

2. Disk <no more>? DM1:

3. Disk <no more>? DM2

Missing ":" in device name. Try again:

3. Disk <no more>? DM2:

4. Disk <no more>?

Now processing:

File # ----- File Name ----- Blocks

1) MT0:[ 1, 2]README.1ST      3
2) MT0:[ 1, 2]TPECPY.RNO      3
3) MT0:[ 1, 2]TPECPY.BAS      30
4) MT0:[ 11, 7]README.1ST      3
5) MT0:[ 11, 7]TPECPY.RNO      3
6) MT0:[ 11, 7]TPECPY.BAS      30

6 files      72 blocks

Now mount a new tape on MT0: and type

PIP @WRITE

Repeat the above "mount" and "PIP" steps until all copies you want are made.

Then type

PIP @CLEAN

Ready

PIP @WRITE
*MT0:/ZE
Really zero MT0:/PARITY:ODD/DENSITY:800 ? Y
*MT0:[1,2]README.1ST=DM0:TC1001.TMP
*MT0:[1,2]TPECPY.RNO=DM1:TC1002.TMP
*MT0:[1,2]TPECPY.BAS=DM2:TC1003.TMP
*MT0:[11,7]README.1ST=DM0:TC1004.TMP
*MT0:[11,7]TPECPY.RNO=DM1:TC1005.TMP
*MT0:[11,7]TPECPY.BAS=DM2:TC1006.TMP

Ready

PIP @CLEAN
*DM0:TC????T.MP/DE/NO/W
DM0:TC1001.TMP erased and deleted
DM0:TC1004.TMP erased and deleted
*DM1:TC????T.MP/DE/NO/W
DM1:TC1002.TMP erased and deleted
DM1:TC1005.TMP erased and deleted
*DM2:TC????T.MP/DE/NO/W
DM2:TC1003.TMP erased and deleted
DM2:TC1006.TMP erased and deleted
*WRITE.CMD/DE/NO/W
WRITE .CMD erased and deleted
*CLEAN.CMD/DE/W
CLEAN .CMD deleted

Ready

PIP MT:[*,*]/L

Name .Ext      Size      Prot      Date      MT:[1,2]
README.1ST      3      <155> 12-Jan-83
TPECPY.RNO      3      <155> 12-Jan-83
TPECPY.BAS      30     <155> 12-Jan-83

Total of 36 blocks in 3 files in MT:[1,2]

Grand total of 72 blocks in 6 files in MT:[*,*]

Ready

```


"With...departmental budget cuts, high interest rates, hardware prices falling, upgrade-ability being vital..."

"But I need a VAX. They're the best."

"I didn't say don't USE a VAX...just don't BUY one!"

"Huh?"

"HAMILTON will rent you the latest in DEC systems-popular VAX 750s, hot new 730s and the full range of PDP11s or Personals and all on a simple no-deposit monthly payment basis. Great for fitting into tight budgets. You'll get prompt delivery, the systems configured the way YOU want them, AND they'll bundle in your choice from HAMILTON's extensive software and utilities library."

"That sounds great, but don't these rental schemes lock you in for 5 years?"

"No, not if you rent from HAMILTON. If you know you won't ever need to change so much as a terminal for 3-5 years, then go to your bank about a long term lease. But, if you want the flexibility of upgrading either all or any part of your system when you need to, then go talk to HAMILTON about rental. They'll listen."

"That's really interesting. I didn't know I had that kind of rental alternative."

"And what's more, you can talk to HAMILTON about their rental-with-purchase-option plan which gives you the right to buy the system when you're ready."

"But will DEC maintain my HAMILTON rental system?"

"DEC maintenance is already INCLUDED in your monthly rentals. Plus, HAMILTON handles all the shipping and guarantees you a really smooth installation."

"HAMILTON certainly has a comprehensive rental program."

"Of course. They have more experience in renting DEC systems than anyone else, with thousands of installations in North America and Europe."

"That's great, but I wonder if they would rent me applications software?"

"Sure, HAMILTON rents their full range of Word Processing, Graphics, Spreadsheet, Accounting, DBMS and other packages. Call them now..."

Toll Free 800-631-0298

In New Jersey 201-327-1444

In Canada 416-251-1166

or complete the coupon for an individual rental quotation on a system of your choice."

P.S. HAMILTON also offers great TIMESHARING
Ask them about it.

HAMILTON

HGL Software

HAMILTON Rentals

6 Pearl Court, Allendale, N.J. 07401

TOLL FREE 800-631-0298

In New Jersey 201-327-1444

415 Horner Ave., Toronto, M8W4W3
416-251-1166

TOLL FREE Ont. & Que. 800-268-2106

All other prov. 800-268-0317

NEW YORK • DALLAS • MONTREAL • CALGARY
LONDON • PARIS • DUSSELDORF

"Don't buy a VAX"



VAX, PDP, DEC are trademarks of Digital Equipment Corporation

HAMILTON

6 Pearl Court, Allendale, N.J. 07401

Please rush me information on the following:
(Circle and/or fill out items below)

Processor 11/23 11/24 11/34 11/44 VAX 11/730 VAX11/750

Disks RL02 RK07 RMO2 RM80 RA80 RA81 RA60

Software Word Processing Graphics Spreadsheet Accounting
DBMS Other _____

Timesharing Application: _____

Name _____

Position _____

Company _____

Address _____

City _____ State _____ Zip _____

Telephone Number _____

CIRCLE 2 ON READER CARD

BASIC-PLUS TECHNIQUES FOR TABLE ORGANIZATION AND LOOKUP

By Dennis Thibeault, DataCraft

Introduction

Table lookups are among the most frequently required procedures at most data processing installations. There is hardly an application that does not need to search for some information in a table.

This article surveys several techniques for organizing, maintaining, and searching tables. Although aimed specifically at the RSTS/E BASIC-PLUS community, the discussion may also be of interest to users working in other environments.

We begin with some definitions. Figure 1 shows a simple example of a table. Note that a table consists of a collection of entries, or records. In this example, each entry consists of two fields, a code and a description. Each table entry has a key which uniquely identifies it; in this case the key is state code. In addition, each entry may also have associated data fields — in this case, state name.

Of course a table key could be comprised of several different key fields concatenated, such as policy number and year. Likewise, the data portion could include a number of separate fields, like name, address, city, state, and zip. Or, there might be no associated data fields at all, such as in a table consisting simply of a list of valid transaction codes.

Two common uses for tables are validation and retrieval. Looking up "F-AA-2726D-TC" to see if it is a valid part number is an example of validation. Such a procedure might be found in a data entry module in order to prevent incorrectly coded data from entering a system.

Looking up an employee's name, given his or her employee number, is an example of retrieval. This kind of operation is typical of programs which produce printed reports. But retrieval operations are by no means restricted to report generation applications. For example, a salary program might look up an employee's rate of pay in a table in order to do a payroll calculation. Moreover, using tables to retrieve lengthy descriptive text can significantly reduce file sizes by reducing much redundant storage of data.

Some common examples of tables include: office, region, or division codes and locations; part numbers and descriptions; vendor or customer codes and names and addresses; employee numbers and names; menu options and program names; error codes and messages; cost center or department codes and names; and so on. This list of course is far from exhaustive in fact, the variety of applications for tables is limited only by the needs and imagination of the user.

Although table maintenance and lookup routines are frequently used, they are rarely standardized. All too often, when a new table is needed, a new program is written to maintain it, another new program is written to list it out, and new routines are developed to search it.

Recognizing that this continual re-inventing of the wheel is a waste of time and money, we at DataCraft developed Tables-11 in 1979. Tables-11 is a generalized software package that standardizes all table maintenance and lookup operations. It is based on a powerful and efficient data structure known as the B+ tree. We will discuss B+ trees and Tables-11 in more detail later in this article, but first let us review some of the more common techniques for table handling.

Internal vs. External Tables

Tables may be either internal or external. An internal table is totally contained within the program that uses it. Such "hard-coded" tables are typically arrays which are initialized via a READ/DATA loop or a series of LET statements.

On the other hand, an external table resides in a data file which is not part of the program. For some reason, external tables are quite often implemented as virtual arrays — perhaps because they began life as in-core arrays which outgrew their programs — but, for reasons we shall explore presently, they are frequently better implemented using RSTS/E Record I/O.

The chief advantage of internal tables is that they can be searched extremely quickly. In fact, almost any searching technique will yield acceptable performance for an internal table. Because the entire table resides in core, access to any entry is extremely fast. Since memory limitations mean that no internal table can ever be really large, it cannot take a very long time to search even the entire table. So, for internal tables, a straightforward sequential search is about as good as any. (Of course, none of these remarks applies to virtual memory systems.)

Against this advantage, however, there are several disadvantages. For one, the size of an internal table adds to the size of the program. For the BASIC-PLUS user, this means that as a table grows to a non-trivial size, its host program approaches the 16K-word limit.

Also, many tables are used in more than one program. For example, in a typical DP shop, a table of office codes and locations is apt to be used in several programs in every application system. If this is an internal table, then every single program that requires it must contain its own private copy. If — or, more accurately, when — the company opens a new office, or closes or moves one, then every program that uses an office table must be located, modified, recompiled, and tested.

The advantages of external tables are largely the negation of the disadvantages of internal tables. External tables can be virtually unlimited in size. The size of a table has no impact on the size of a program using it.

Also, external tables are sharable. Continuing our previous example, there need be only one copy of the office table. All programs that need to access office codes and names can use the same external table file and be assured of accessing consistent, up-to-date information. When the company opens a new office, only the office table needs to be modified; none of the programs that access it need to be changed, assuming of course that the design of the search algorithm is sufficiently general (e.g., doesn't have things like the total number of table entries hard-coded in the program). Table maintenance is thus separated from program maintenance.

Similarly, the main drawback to external tables is the opposite of the main advantage of internal tables, namely that access is slower due to the fact that disk accesses are required.

Because of the advantages of using external tables, and because of the relative unimportance of the method used to search internal tables, the remainder of this article will consider search techniques for external tables only. From a performance point of view, this means being concerned not only with the number of looks required, but with the number of disk accesses as well.

Sequential Searching

The simplest search technique is a straight sequential search. Table entries are examined one at a time until an entry is found whose key matches the desired value or until all the entries have been examined without finding a match.

Sometimes the desired entry will be found right away. Sometimes it will not be found until nearly every entry has been examined. On the average, a table with n entries will require $(n + 1)/2$ "looks" to locate a given entry. However, it takes a full n looks to determine that an entry is not in the table at all. N looks require n/b disk accesses where b is the number of table entries per block. (All estimates of the number of disk accesses consider "data" accesses only, and do not consider the effects either of window-turning, which will make the actual numbers worse, or of caching, which will make them better.)

Maintaining such a table is quite easy. Changes to data fields can be performed in place. Entries can be deleted simply by blanking them out or otherwise flagging them as deleted (of course the search algorithm must know to skip records that are marked for deletion). New entries can be added at the end, or they may re-use slots formerly occupied by deleted entries. The first record in the table — the zeroth element, if it is a virtual array — can be a header record containing the highest record number in use, so that the search routine will know when it has reached end-of-table.

With this technique, the actual order of entries in the table has no bearing on the results of the search. This is just a brute-force technique that makes no assumptions about how the entries are ordered. Ordering the entries, however, can have an impact on performance.

For example, putting the entries most likely to be accessed at the top of the table will reduce the average number of looks required to retrieve an existing entry. If a company does 80% of its business in the state of New York, and has programs that search a state table sequentially,

then making NY the first entry in the table can save significant time during lookups.

The determination as to what the most efficient order actually is usually must be made empirically. In fact it is possible to make such a table self-optimizing by incorporating in the search algorithm a routine to maintain a count of the number of times each entry is referenced, and then periodically sorting the table on descending count. However, this slows down the search itself due to the overhead of having to update the count fields and may also lead to complications involving concurrent multi-user updates. The slowdown is likely to more than offset the savings to be gained.


Another method of ordering a table which will be searched sequentially is to maintain it in sorted order by key. Assuming all keys are equally likely to occur, then ordering them would have no impact on the number of looks needed to retrieve an existing entry (still $(n + 1)/2$). But it halves the number of looks needed to determine that an entry does not exist, because the search need not proceed to the end of the table. Instead, it can stop as soon as a key higher than the desired value is encountered. So the average number of looks required is $(n + 1)/2$ for all key values, whether they exist or not.

... continued on page 38

Introducing

SORTC

THE FASTEST
MOST FLEXIBLE,
EASIEST—TO—USE
COMPOUNDING SORT
ON THE MARKET TODAY



For more information
Call **215-337-3138** or write:
332 W. Church Rd., King of Prussia, PA 19406

CIRCLE 191 ON READER CARD

TIME SHARING VAX 11/780 VMS


Basic/Pascal/Datatrieve FMS/EDT/WORD-11

Other Software Available - Short or Long-Term - Interactive or Batch

PRIME TIME	SECONDARY TIME
\$100.00/CPU hr.	\$60.00 CPU hr.
\$2.50/hr.	\$1.50/hr.
Connect Time	Connect Time
— Disk: \$40.00/MB Month —	
Other Charge Bases Available	

We Perform Full Daily Tape Back-Up Of All Data

Call or Write:
Computing Services Department

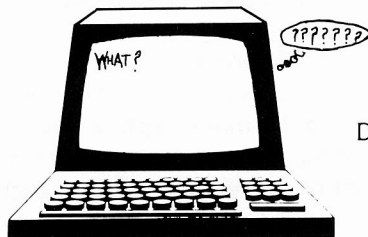


ALDEN PRESS, INC.

2000 Arthur Avenue
Elk Grove Village, IL 60007
(312) 640-6000, Ext. 542, 543
TLX 469799

CIRCLE 200 ON READER CARD

DEAR VAX/ RSTS MAN



Send questions to:
DEAR VAX/RSTS MAN
P.O. Box 361
Ft. Washington, PA
19034-0361

DEAR VAX/RSTS MAN:

I have come up with a patch to BATRUN for use on RSTS V8.0 systems using the new micro-Spooler. BATRUN will queue log files to the old spooler and if you do not have the old printer spoolers up, no log files will come out.

The following patch will cause BATRUN to queue files to the new micro-Spooler. Note that a \$JOB statement with a '/QUE' switch specifying a log device or a startup of BATCH processing specifying a log device will not output to that device, it will be ignored. The loss will always come out to the device (or devices) processing FORM=NORMAL. Note also that the patch DOES support the

'/DE' switch and will delete after printing a log.

Also, the CCL, 'DCL' must be supported; if not, see your RSTS manager's guide on how to do this.

The following patch goes into BATRUN.BAS which must be recompiled or task-built back into the spooler account. Just add the lines under the comment 'NEW SPOOLING PACKAGE', and remove (or comment as shown) the lines under 'OLD SPOOLING PACKAGE'. With this patch, the old print spooling package may be dropped and forgotten. (HOORAY!!!)

Philip Hunt

See Program A

DEAR VAX/RSTS MAN:

Do you have a good MACRO subroutine for dividing a 32-bit integer by a 16-bit integer? It may use EIS since all RSTS systems must have it, but please don't say, "Just use DIV." I need a routine which works even if the quotient has more than 16 bits.

Thanks very much.

J. Fred Stevens
Commonwealth Clinical Systems,
Charlottesville, VA

Dear Fred, Steven P. Davis of the Technical Services group at Software Techniques offers the following:

Within the system library, SYSLIB.OLB, there is a routine that very closely matches your needs. It, however, only allows a 15-bit divisor. An example of how to call this subroutine is below. If you use TKB to build your executable image it will automatically search this library to resolve the reference to \$DDIV.

See Program B

PROGRAM A

```
8010 INPUT LINE #11%, C$ %
      \ GOTO 8020 IF LEFT(C$,2%)<>"$1" %
      \ E9%=0% %
      \ NOREQUEZ=-1% %
      \ RQZ=Z0%(J9%,31%) %
      \ GOSUB 12100 %
      \ Z0%(J9%,31%)=RQZ %
      !=====NEW SPOOLING PACKAGE===== %
      \ C$="DCL PRINT " !setup del print start %
      \ CTEMP$ = FNU$(Z0%(J9%,38%),NULSTG$,4233Z,0%) !set log filename %
      \ CTEMP$ = CVT$(CTEMP$,38%) !cvt to drop spaces %
      \ C$=C$+CTEMP$+"/NOFEED/FORM=NORMAL" !setup as form normal %
      \ C$=C$+"/DELETE" IF QUEZ AND 4% !delete it if requested %
      !=====OLD SPOOLING PACKAGE===== %
      ! C$="RUN $QUE"+CHR13$+"Q " %
      ! C$=C$+RAD$(Z0%(0Z,32%))+RAD$(Z0%(0Z,33%))+":-" IF Z0%(0Z,32%) %
      ! C$=C$+FNU$(Z0%(J9%,38%),NULSTG$,4233Z,0%) %
      ! C$=C$+"/DE" IF QUEZ AND 4% %
      !===== %
      \ C$=C$+CHR13$ %
      \ GOSUB 13400 %
      \ GOSUB 12900 %
      \ GOSUB 13000 %
      ! SET UP THE FILE TO QUE AND LOG IN, QUE IT AND LOG OUT, %
```

PROGRAM B

```
.PSECT DATA,RW,D,LCL,REL,CON
REMAIN: .BLKW ;Remainder gets placed here
QUOLSB: .BLKW ;Quotient LSB gets placed here
QUOMSB: .BLKW ;Quotient MSB gets placed here

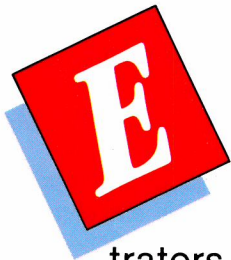
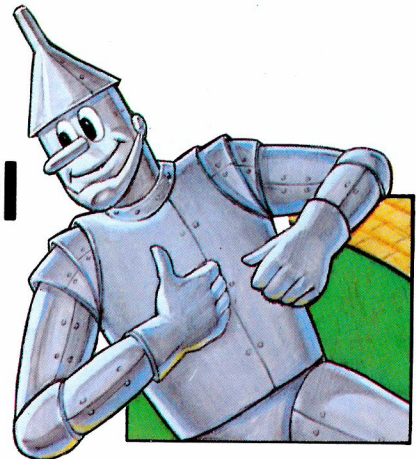
.PSECT CODE,R0,I,LCL,REL,CON
FOO:: MOV #10,R0 ;Set up number to divide by
      MOV #500,R1 ;Set up MSB of dividend
      CLR R2 ;And set up LSB of dividend
      JSR PC,$DDIV ;Go and do the divide
      MOV R0,REMAIN ;Now store the remainder
      MOV R1,QUOMSB ;And the quotient MSB
      MOV R2,QUOLSB ;And the quotient LSB
      RETURN ;And we're done

.GLOBAL $DDIV
.END FOO
```

... continued on page 43

PERSONAL and PROFESSIONAL

Produced for the Digital Personal and Professional Computer User



Each issue of PERSONAL and PROFESSIONAL is packed with the latest information on the world of personal computers in a high-quality, modern, attractive and easy-to-read format. The finest editors, writers, art directors, illustrators, cartoonists and photographers have been assembled to establish a new level of excellence and objectivity in personal computer magazines.

Simply stated, PERSONAL and PROFESSIONAL is the most substantial source on the Digital personal computer user's regular reading list. It's as if a team of computer experts came to your office or home every issue.

PERSONAL and PROFESSIONAL is an independent magazine, not sponsored or approved by or connected in any way with Digital Equipment Corporation.

Special Subscription Offer

Save \$14.⁰⁰ or More!

You can now become a special subscriber to PERSONAL and PROFESSIONAL with no obligation or risk.

Simply enter your subscription on this card and mail it to us. Our current issue will be sent immediately upon receipt.

Read your issue of PERSONAL and PROFESSIONAL, the independent magazine for Digital personal computer users. If it is everything you expected, honor our invoice. If it isn't, just write "cancel" across the invoice and mail it back. You won't be billed and the issue is yours at no charge.

SO ACT NOW. Become a PERSONAL and PROFESSIONAL subscriber today!

PERSONAL and PROFESSIONAL

P.O. BOX 114, SPRINGHOUSE, PA 19477-0114
215/542-7008

NAME _____
(please print full name)

ADDRESS _____ APT. _____

CITY _____ STATE _____ ZIP _____

- ☐ **YES!** I want to take advantage of your Special Subscription Offer **SAVE \$14.00**. Please bill me only \$28.00 for the next 12 issues of PERSONAL and PROFESSIONAL. Offer good until September 1, 1983 and is valid in the U.S. and Canada only.
- ☐ **PAYMENT ENCLOSED.** I want to save even more money! I'm enclosing a check or money order in the amount of \$20.00 — AN INCREDIBLE SAVINGS OF \$22.00 OFF THE COVER PRICE. Please send me the next 12 issues. Offer good until September 1, 1983 and is valid in the U.S. and Canada only.
- ☐ Foreign, air mail, payable in U.S. dollars, \$56.00 — Save \$20.00 over our normal foreign rate.
- ☐ Visa/Mastercard Exp. Date _____
Acct. # _____
Signature _____

Binary Searching

If we are going to maintain a table in order by key, however, then there is a much better technique than the sequential search. This better technique is the binary search. Although it is slightly more complicated to program than a sequential search, it is much faster.

Exploiting the fact that the table is known to be in order by key, with each look into the table the binary search either finds the desired entry or else eliminates half of the remaining entries from further consideration.

We begin a binary search by looking, not at the first table entry, but at the entry in the middle of the table. If that is the entry we are looking for, then the search ends successfully. Otherwise, if it is higher than the key we are looking for, then the entry we seek must be in the lower half of the table, if it exists at all. Similarly, if the table entry is lower than the desired key, then the entry we seek must be in the higher half of the table, if at all.

We can thus eliminate half of the table from further consideration and take our next look at the mid-point of the remaining half. This process continues until we either find the desired entry or run out of entries to look at.

For example, consider the table in Figure 1. If we wanted to determine whether OR were a valid state code, the sequential method described earlier would require looking successively at AK, AL, AR, and so on until, on the thirty-eighth look, we finally find OR.

Using a binary search, we would begin looking at the middle entry, MS. OR is higher than MS, so we can eliminate the first half of the table from further consideration and now look at the middle entry in the remaining half, which is PA. This time, OR is less than PA, so we eliminate all entries from PA on up and look now at the middle entry in the remaining portion (MT through OR).

Note that this portion of the table contains an even number of entries, so there really is no "middle" entry — the middle falls in between NJ and NM. We arbitrarily decide ahead of time that in such cases we will look at the entry just before the mid-point (we could just as well have decided the other way), so we now look at NJ.

OR is higher than NJ, so we look at the middle entry of the remaining six, which is NY. Again, OR is higher than NY so we look at the middle entry of the three remaining, OK. OR is higher than OK and there is only one entry higher than OK remaining under consideration, which indeed turns out to be OR. This search took only six looks, compared with thirty-eight for a sequential search.

A binary search on a table of n entries requires a maximum of m looks, where m is the smallest integer such that $2^{*}m \geq n + 1$, in order to find any given entry or to determine that it does not exist. Put another way, the maximum number of looks is $\log_2(n + 1)$. The average number of looks is somewhat less, since sometimes we will find the record of interest before the search narrows the table down to the last record.

The straight sequential search was indifferent to the arrangement of the table entries. But once we start using

search strategies that depend upon records being in a certain order, complications arise in the insertion and deletion routines. It will no longer do just to append new entries to the end of the file; we must somehow maintain the proper order.

One alternative for handling additions is to determine the new record's proper location in the table, "slide" all higher records down one position, then insert the new record in its correct place. This involves rewriting the entire contents of the table from the point of insertion to the end of the file every time a new record is added, which clearly is not a very efficient approach.

Another alternative is to batch all additions in a transaction file, and at the end of each maintenance session sort the transaction file and merge it with the original table file, creating a new table. This approach is only a slight improvement over the first alternative.

Yet another approach is to add new records to an overflow area at the end of the table and modify the search routines to do a binary search on the sorted part of the table, and then, if that is unsuccessful, do a sequential search on the overflow area.

This alternative is the least odious of the three, but if the tables are large and volatile and are not physically re-ordered regularly, then lookups frequently will be doing long sequential searches on the overflow area.

But it is expensive and time-consuming to maintain a large table in physical order. Is there, perhaps, some way in which the logical sequence of entries could be maintained without requiring massive physical reorganization? Of course, the answer is yes.

Linked Lists

Most programmers are familiar with the concept of a linked list, with which a file can be maintained in logical order by including in each record a pointer to that record's logical successor (which need not be the physically next record). For example:

Record #	Key	Next
1	DASHER	7
2	DANCER	1
3	PRANCER	4
4	VIXEN	0
5	COMET	6
6	CUPID	2
7	DONDER	3
8	BLITZEN	5

If somewhere in the header record we store the fact that record #8 is the first logical record, then we can process this table sequentially by following the pointers. A null pointer — such as we find in record #4 — signifies the last record. In addition to the "next" pointer, we could also have a "prior" pointer in each record to enable reading the table in reverse order.

Insertions can be made by adding the new record in an available slot (either at the end or by re-using a position previously vacated by a deletion) and updating the ap-

appropriate pointers. For instance, to add RUDOLPH, we would first determine that the new key logically belongs between PRANCER and VIXEN. When we add RUDOLPH in position #9, we must modify PRANCER's "next" pointer to point to RUDOLPH, and put PRANCER's old "next" pointer (to VIXEN) into RUDOLPH's "next" pointer.

Deletions are essentially the reverse process. The record to be deleted is blanked out and the contents of its "next" pointer replace the contents of its prior record's "next" pointer.

If "prior" pointers are also being maintained, these procedures are only slightly more complex.

The linked list technique described here allows us to maintain logical sequence without requiring frequent massive physical reorganization. But note that if a significant number of entries are out of physical order, then performance degrades because traversing the table sequentially will involve re-reading some blocks a number of times. Periodic reorganization, then, is still recommended to improve performance.

Although this file structure is adequate for doing a sequential search on an ordered table, it does not seem to help us at all if we want to perform a binary search. In a binary search, we do not want to find the next record, or the prior record. Instead, we want to find the middle record of a selected subset of the table. If the table is in physical sequence, then this is an easy calculation of a subscript (for a virtual array) or a block number and offset (for a Record I/O file). But how do we proceed if the table is implemented as a linked list?

Binary Trees

To answer that question, we now consider the concept of a binary tree. Physically, a binary tree is implemented in much the same way as the sequentially linked list structure described above, but the pointer fields carry a different meaning. Although we still have an ordered collection of records, instead of looking at it as a sequential file, we visualize it as shown in Figure 2.

Each node in Figure 2 represents one table entry, containing both key and data fields as well as two pointers — a "high" pointer and a "low" pointer, which are represented pictorially by the lines exiting each node from the right and left respectively.

The top-most node in a tree is known as the root. A node at the bottom level — that is, one with two null pointers — is known as a leaf.

All searches begin at the root — which node is the root can be stored in the header record. If the key field in the root is the key we are searching for, then the search ends successfully. Otherwise, if the value we seek is lower than the root key, we take the branch to the left; if it is higher, we take the branch to the right. In either case, we arrive at a node which is effectively the root of a smaller sub-tree, where we repeat the compare-and-branch process. Eventually we either find the entry we are looking for, if it exists, or end up at a node that has no pointer to follow further.

As an example, assume we are searching the tree in Figure 2 for CUPID. We begin at the root, with DANCER. CUPID is lower than DANCER, so we follow the branch to the

TABLES-11

? WHAT ?

TABLES-11 is a complete facility for creating and maintaining external tables and for accessing them from user-written BASIC-Plus programs.

? WHY ?

TABLES-11 can save your installation time and money by standardizing table maintenance and lookup procedures. Your application programmers or users don't have to reinvent the wheel each time they need to do a table lookup. Typical applications include data entry validation; retrieval of descriptive text based on a code; creation of flexible, table-driven systems; and more.

? HOW ?

An easy-to-use interactive utility allows you to create new tables; add, change, or delete table entries; and rename or delete existing tables. Via an APPENDable BASIC-Plus function (CSPCOM-compatible), a user program can easily and efficiently retrieve information from any number of tables. Full or partial key and forward or reverse sequential lookups are all fully supported.

? WHEN ?

TABLES-11 is **Ready** for immediate delivery.

Introductory Price: \$595 (single CPU license)

Trial Version Available

VMS version available soon

DataCraft

P.O. Box 292

Chestnut Hill, MA 02167

CIRCLE 193 ON READER CARD

left, and come to COMET. CUPID is higher than COMET, so we take the branch to the right and, indeed, find CUPID, and the search ends successfully.

On the other hand, say we are searching for NONAME. In this case, the search would proceed from DANCER to DONDER to PRANCER. From PRANCER we should take the branch to the left — but there is no such branch. Therefore we correctly conclude that NONAME is not in the table.

The tree in Figure 2 could be represented as a linked list as follows:

Record #	Key	Low	High
1	DASHER	0	0
2	DANCER	5	7
3	PRANCER	0	4
4	VIXEN	0	0
5	COMET	8	6
6	CUPID	0	0
7	DONDER	1	3
8	BLITZEN	0	0

Again, complications arise when we consider the question of table maintenance. We might be tempted to think we can simply add RUDOLPH as shown in Figure 3. But note that by doing so, the tree becomes unbalanced.

A binary tree gets out of balance when the longest path from the root to a leaf is more than one greater than the shortest path. This is what happens when we add RUDOLPH: the path from DANCER to BLITZEN has a length of 3, but the path from DANCER to RUDOLPH has a length of 5.

Although the searching algorithm we have described will continue to work, even on an unbalanced tree, performance degradation will occur. A perfectly balanced binary tree of n records never requires more than $\log_2(n+1)$ looks to find any key. But as the tree gets more and more unbalanced, the maximum number of looks required begins to approach n .

Not every addition unbalances the tree (adding NONAME would not have unbalanced Figure 2). But when an addition (or deletion) does cause the tree to get out of balance, rebalancing it "on the fly" is, fortunately, not difficult. Rebalancing requires nothing more than exchanging the logical positions of two or, at most, three nodes, which is accomplished by modifying appropriate pointers. Figure 4 shows the tree from Figure 3 after it has been rebalanced.

Rebalancing on the fly like this eliminates the need for periodic massive reorganization and ensures that searches will never require more than $\log_2(n+1)$ looks.

The main advantage of the binary tree method is that it combines the fast random lookup capabilities that are characteristic of a binary search on a physically ordered file with the low-overhead maintenance that is characteristic of linked lists.

The main disadvantage is that sequential processing is more difficult and less efficient since it involves much backtracking.

B-Trees

The time required to perform a sequential search, as we have seen, is directly proportional to n , the number of en-

tries in the table. The binary search is dramatically shorter because it is proportional to $\log_2(n)$, which increases much more slowly as n increases. It turns out that a generalization of the binary tree can speed search time even more dramatically by making it vary with the $\log_d(n)$, where d may be 3, 4, or even more.

This generalization of the binary tree is known as the B-tree. Recall that in the binary tree structure, each node contained a single table record and two possible branches. However, it is possible for each node to contain more than one record. Indeed, if a node is represented as a 512-character disk block — a convenient size for RSTS/E systems — then, depending on the record size, we can pack, say, four or eight or 12 or 32 records into each node.

If there are d records in a node, with $d+1$ possible branches from each node, then the maximum number of nodes that must be examined — or visited — during a search is $\log_d(n)$. The number d is called the order of the tree; thus a binary tree is effectively a B-tree of order 1.

Note that here, for really the first time, we must distinguish between the number of looks and the number of disk accesses. With each disk access in a B-tree we can look at a number of entries.

The end result is that B-trees of order greater than 1 are shorter — "bushier," if you will — than binary trees. The distance from root to leaves is accordingly less.

Figure 5 shows our earlier example re-cast as a B-tree of order 2. Now it only takes two accesses at most to retrieve any entry; in the earlier case it could take as many as four. On larger tables, and with larger values of d , the savings are even more impressive.

Searching a B-tree begins, as usual, at the top-most, or root node. Now, however, instead of choosing between two possible paths, we must choose among $d+1$. In Figure 5, if the key we seek is less than CUPID, we take the leftmost path; if it is greater than CUPID but less than DONDER, we take the middle path; and if greater than DONDER, then the rightmost path.

It is possible to maintain a balanced tree when adding or deleting records. In practice, we ought to choose a value of d such that $2d$ records can fit in a node (block). To add a new record, we first locate the node in which the record should reside. If there is room in that node to add the new record, it is simply inserted in its proper location, sliding higher records already present in the node (if any) one record position to the right.

If, however, there is no room for an additional record (i.e., it already contains $2d$ records), then we must split the node in two — the lowest d records remain in the original block, the highest d are inserted into a new block, and the record in the middle is moved up to its proper place in its "parent" node, where there will usually be room for it. If there is no room in the parent node, the splitting process occurs again at the parent's level. In the worst case, the root node splits and a new root is created, which increases the height of the tree by one. Note that, unlike binary trees, which grow downward by adding lower levels, B-trees only grow upward (i.e., at the root).

This procedure automatically maintains the balance of the tree.

On deletions, essentially the opposite is true. If a deletion would leave a node with less than d entries, we try to divide the remaining entries between its neighbors, or to borrow an entry from a neighbor, or to combine the node with one of its neighbors; one of these will always work. In any case, of course, an adjustment may have to be made at the parent level as well, where underflow may also occur. In the best case, the root node may disappear, decreasing the height of the tree by one.

B+ Trees

The reader may by now have noted several drawbacks to the B-tree file structure. One is that the value of d is constrained by the record size; it may not be possible to fit very many records into a single node.

Another drawback is that although random retrieval is quite efficient, sequential retrieval is not. We frequently need to process table entries in sequential order. To do so in a B-tree involves re-reading the same blocks a number of times.

Both of these disadvantages are overcome in a slight modification of the B-tree known as the B+ tree.

The main difference between a B-tree and a B+ tree is that in the B-tree, entire table records are stored in each node. In the B+ tree table records are stored only in the leaf nodes; all higher level nodes contain only table keys.

Since the higher level nodes contain only keys — and since the data portion of a table record is typically much longer than the key portion — B+ trees can be of a higher order than B-trees — i.e., they can be much bushier.

And since all the table records are in the bottom level, sequential processing is simple and efficient. In sequential processing, no block need ever be read more than once. The leaf nodes can be doubly linked to provide both forward and reverse sequential access.

So, B+ trees have two types of nodes, index nodes and data nodes. The higher level nodes are all index nodes. Not all keys in the table are in the index — only as many as are needed to direct the search to the appropriate leaf. And, in fact, keys appearing in the index need not even be in the table itself.

Tables-11

As luck would have it, the more efficient file structures and table maintenance and lookup techniques are more complex to program than the less efficient ones.

Designing such structures and routines anew each time a new table is created is an error-prone and labor-intensive procedure. On the other hand, a "quick and dirty" approach often yields inefficient algorithms that are wasteful of computer resources.

Tables-11 is our solution to this dilemma. Using a BASIC-PLUS Record I/O implementation of the B+ tree file structure, Tables-11 is an approach to table handling that is completely generalized and at the same time extremely efficient.

Tables-11 is a complete, easy-to-learn and easy-to-use facility for creating and maintaining external tables and for accessing them from BASIC-PLUS programs. The philosophy behind it is twofold: first, using external tables eases program maintenance by divorcing table maintenance from program maintenance. And second,

standardizing table updating and lookup procedures speeds program development and reduces possibility of error. As a beneficial side effect, the availability of an easy-to-use table maintenance and lookup facility also encourages the development of flexible, table-driven applications.

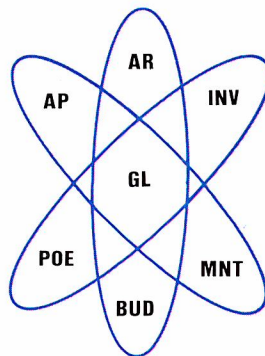
The Tables-11 package includes two programs, TABLES and TABLIS, and one APPENDable BASIC-PLUS function, FNLOOKUP\$.

TABLES is an interactive maintenance utility that is used to create new tables, to add, change, and delete table entries, and to delete obsolete tables. Each table can be given customized names for the key and data fields which are used to prompt the user during table update sessions and are also used for headings on table listings. For each ppn, a master table containing the names and titles of all tables in the account is maintained automatically. Since each table is a single RSTS/E Record I/O file, tables can be easily transported using PIP or any other file copy program.

TABLIS is a report program that



IBMS™ — FINANCIAL APPLICATION SYSTEMS FOR COMMERCIAL, GOVERNMENTAL AND EDUCATIONAL USERS.



™VAX is a registered trademark of Digital Equipment Corporation.

IBMS is the only COBOL based, financial system specifically designed for the VAX. Both the commercial and fund accounting versions of IBMS were designed with a layered architecture for greater efficiency, system integrity and application to networking environments. Of course, IBMS is interactive, integrated and user friendly.

the **FASBE** group inc.

92 Main St., Nashua, NH 03060
(603) 883-3212

Announcing DSKBLD V3.0 for RSTS/E V8.0

Improved RSTS/E performance

GUARANTEED

A recent
survey of
150 DSKBLD users produced these
overwhelming results:

100% of all respondents reported faster system response.

98% said it meets or exceeds expectations.

98% would recommend DSKBLD.

Here's why:

DSKBLD reduces overhead and file processor demands.

DSKBLD optimizes file clustersizes.

DSKBLD centers directories for faster seeks.

DSKBLD supports RSTS/E V8.0.

DSKBLD is easy to use and fully documented.

DSKBLD pays for itself within a few months.

For complete information about DSKBLD's specifications, applications and unconditional money-back guarantee, call or mail your business card.

MANUS

SERVICES CORPORATION

AUTHORIZED **digital**® COMPUTER DISTRIBUTOR

Pete Schnebele
1700 Westlake Ave. N.
Seattle, WA 98109
206/285-3260

RSTS/E is a trademark
of Digital Equipment Corp.

CIRCLE 39 ON READER CARD

C B E D I T The \$200 RSTS/E* WORD PROCESSOR

Version 3.0 includes list processing, optional print-time operator input, on-screen help and directory management plus total device independence. All files are standard ASCII with no control codes. Data loss protection on system crash or operator error.

Window edit includes add, insert, global locate, global change, block replace, block delete, block copy, block move, document merge, etc. Fully formatted output with page numbers, headers, footers, margins, indentation, justified text, underscore, super/subscript, center, multiple spacing, etc. Automatic or manual pagination with unlimited document length.

Table driven Video and Printer modules support any VDTs (including 132 column and 66 line) and any printers. Mixed units on same system are OK.

Price: \$200 on 9-track/800 bpi plus shipping. Other media available. User's manual and Basic-Plus* source are included.

T. F. Hudgins & Associates, Inc.
P.O. Box 10946 Houston TX 77292
Wes Seago 713/682-3651

*TM Digital Equipment Corporation

CIRCLE 9 ON READER CARD

produces an up-to-date, formatted, ordered listing of any table, or any specified portion of a table. If desired, the listing can also show the date that each entry was added or last changed.

The FNLOOKUP\$ function handles all table lookup tasks. FNLOOKUP\$ performs both validation and retrieval operations, using either full or generic (partial) keys. Forward and reverse sequential processing is also fully supported.

No prior setup is required in a user program before invoking FNLOOKUP\$ the function handles all housekeeping chores such as OPENing table files and initializing variables automatically. FNLOOKUP\$ is extremely easy to use, even for the casual "citizen programmer" in an end-user department. Figure 6 is an excerpt from the Tables-11 User's Guide which describes its use.

FNLOOKUP\$ adds approximately .75K to the size of a compiled BASIC-PLUS program, and is CSPCOM-compatible.

We at DataCraft have used Tables-11 extensively as a data management tool both in our own software and at our client companies since we developed it in 1979. To date, FNLOOKUP\$ has been used in over one thousand programs.

Other File Access Methods

Some readers may have concluded by this point that this is all much ado about nothing, and that all we really seem to need is some kind of indexed-sequential access method. This objection is not completely without merit, but the existing alternatives for the RSTS/E user are really not very satisfactory.

RMS, for example, is pretty much ruled out because it is unavailable to BASIC-PLUS users. Moreover, even if one is willing to endure the frustrations of BP2, the RMS solution falls short. RMS does not allow the flexibility that a truly generalized scheme requires, since record length, key length, key position and the like all must be specified at compile time. With BASIC-PLUS Record I/O, all these characteristics can be specified dynamically at run-time.

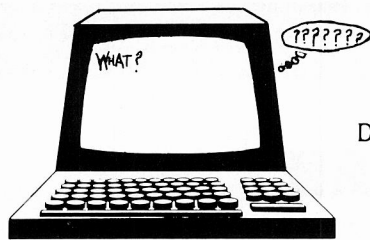
In addition, somewhat to our surprise, we have found in benchmarks that Tables-11 files are considerably more efficient than RMS; in terms of space utilization, Tables-11 files are typically half the size of the equivalent RMS files; in speed of loading, they are typically twice as fast; and in speed of lookup, a BASIC-PLUS program using FNLOOKUP outperformed a compiled BASIC-PLUS-2 program using indexed GETs.

A number of ISAM-like access methods do exist for BASIC-PLUS, most notably DMS-500 and FAM. But these, too, suffer from a number of drawbacks. They typically require separate I/O channels for index files and data files; a single "file" may require the use of as many as four I/O channels. This severely limits the number of indexed files a program can have open simultaneously, uses much more of the user's job space, and leads to difficulties in moving or restoring these files. In addition, DMS and FAM files do not dynamically restructure themselves during updates and must be periodically reorganized.

It is of interest to note that many ISAM schemes, including RMS and IBM's VSAM, are based on the B+ tree model.

... continued on page 46

DEAR VAX/ RSTS MAN



Send questions to:
DEAR VAX/RSTS MAN
P.O. Box 361
Ft. Washington, PA
19034-0361

... continued from page 36

DEAR VAX/RSTS MAN:

As a 'RSTS forever' enthusiast at a RSTS only site, I tend to skip over articles dealing with EDT. As a 'TECO forever' enthusiast working on an 11/34A (read 248 KB memory max) I tended to skip over EDT articles as well. Now I have changed positions and find myself in a VAX/VMS EDT site and am now reading some of the articles I once overlooked.

One of these is 'Further Feedback' from the October, 1982 issue of the *RSTS Professional*. It contained a very helpful command file to remember the last file edited. I had the same idea about two weeks before I finally read Mr. O'Nolan's article and worked up a command file that I feel handles the trick somewhat better:

EDT.COM

```
$ SET NOVERIFY
$ INPUT_STRING = P1
$ IF F$LENGTH(INPUT_STRING).NE.0 THEN $ EDT$MEMORY := 'P1'
$ DEASSIGN SYS$INPUT
$ ASSIGN TT: SYS$INPUT:
$ EDIT/COMMAND=[JEFFJEDTINI,EDT/NOJOURNAL 'EDT$MEMORY'
```

Some handy accessory commands (including one to execute this command file) should be in your LOGIN.COM. They are:

```
$ EDT := @[JEFFJEDT
$ PRINTP == 'PRINT &EDT$MEMORY'
$ LIST := @[JEFFJLISTING
```

EDT invokes your command file (above), PRINTP will queue the file you last edited, and LIST invokes another command file that generates a DIBOL listing and queues it to the SYS\$PRINT queue.

LISTING.COM

```
$ FILENAME = P1
$ IF F$LENGTH(FILENAME).EQ.0 THEN FILENAME := 'EDT$MEMORY'
$ DIBOL/NOOBJ/LISTING 'FILENAME'
$ FNAME = F$PARSE(''FILENAME'',',',NAME')
$ PRINT/DEL 'FNAME'.LIS
```

Jeff Corbett, Programmer
Business Systems, Inc.
Greenville, S.C.

Four Ways to Secure Your DEC System



with software
from McHugh, Freeman
and Associates, Inc.

- 1. Data Encryption:** Encrypts data, source and task image files
 - Prevents disclosure of sensitive data
 - Isolates sensitive programs (from privileged users)
 - Prevents unauthorized modifications
- 2. Menu/Access Authorization:** Allows only menu access to all applications programs
 - Limits display to only those items available to user
 - Prevents access to the monitor level
 - Traces user movements through the menu system
- 3. Keyboard Monitoring:** Monitors and interacts with any or all terminals on the system
 - Use at your discretion or as a function of the log-in sequence
 - Optionally creates a log of all user activity
- 4. Password Changing:** Changes passwords routinely and quickly
 - Choose from a list of over 3,000 encrypted English words or alphanumeric sequences

Contact us for full product specifications and demonstration packages.



McHugh, Freeman
and Associates, Inc.
1135 Legion Drive
Elm Grove, WI 53122
(414) 784- 8250

CIRCLE 57 ON READER CARD

RPM CAN DRAMATICALLY INCREASE THE PERFORMANCE OF YOUR SYSTEM

You can double or triple your system's performance without adding costly hardware. RPM provides everything you need to optimize your system and keep it running at peak performance.

RPM analyzes your system performance automatically, identifying problem areas. But, RPM doesn't stop there. It identifies the cause of the problems and makes suggestions for correction in plain English.

AUTOMATIC PERFORMANCE ANALYSIS

Instead of dumping columns of cryptic numbers, RPM gives you a plain English report that describes how your system is performing. It tells you where you have problems, what caused the problems, and how to fix them.

This report analyzes each of the resources that are common problem areas. Any resources that are not being used optimally are identified. The program or files that caused the problem are then identified and suggestions are made for correcting the problem.

DETAILED PROGRAM ANALYSIS

In addition to identifying problem programs, RPM can analyze the operation of individual programs, identifying problem areas.

RPM's detailed program analysis breaks the operation of the program down into CPU usage, input/output and system calls. Usage and directory overhead counts are displayed by channel and by system call.

---- File Processor (FIP) Usage ----

** The file processor (FIP) is being excessively used. It is in use by at least one job 67% of the time. In addition, an average of 2.7 other users are waiting to use FIP. Although the file processor is in use 67% of the time, it is waiting for information from the disk 72% of the time it is in use. The disk information that FIP is waiting for breaks down as follows:

- 91% Directory.
 - Reading or writing to the directory structure.
- 7% Disk Allocation Table (SAT)
 - Reading or writing information about free blocks on the disk.
- 2% Monitor Overlays.
 - Loading monitor overlays into memory.
- 0% Miscellaneous.
 - Loading disk cache information and other miscellaneous data.

The five programs that use the most FIP resources are:

OE047A	275.2
PAYREC	213.1
...TKB	158.0
EP2COM	110.3
LOGIN	78.7

FIP usage can be decreased by optimizing the clustersize of frequently used files (see section 4.4.1 of the RPM User's Guide), using contiguous files where possible, reordering the directories often and minimizing the opening

RPM> EXAMINE JOB 5 EVERY 5 MINUTES

Job: 5 Program: *ALL* CPU: 54% Sample Time: 297 Seconds

Chan	File	Count	Ovrhd	Chan	File	Count	Ovrhd
0 KB2:		3		1 * Closed *		4	
3 DM1: [1,3]CSPCOM.OLB		835	2	4 DM1: [5,1]TEMP05.TMP		39	42
6 DM1: [5,1]EXAMPL.STB		9	91	7 DM1: [5,1]EXAMPL.TSK		733	
15 DM1: [1,3]TKB.TSK		278					

EMT	Count	Ovrhd	EMT	Count	Ovrhd	EMT	Count	Ovrhd	EMT	Count	Ovrhd
CALFIP	149		.READ	1441	2	.WRITE	460	133	.CORE	7	
.TTECH	1		.TTRST	1		.DATE	1		.NAME	2	
.RTS	1		.LOGS	12		.CLEAR	2		.CCL	1	5
.FSS	42		.UUO	71		.RSX	1		.CLSFQ	62	28
OPNFQ	58	535	CREFQ	2	226	DLNFQ	1	23	RSTFQ	2	
LOKFQ	22	259	CRTFQ	1	64	CRBFQ	1	175	UU.ATR	70	250
UU.NAM	1	3									

EXTENDED PERFORMANCE STATISTICS

RPM adds extended performance statistics to your RSTS/E monitor. This monitor extension captures information about overall system performance plus information on individual programs and files. Using this information, you can improve system performance by minimizing disk head movement, reducing file processor (FIP) waits, reducing swapping, and optimizing disk cache operation.

DYNAMIC PLOTTING

In addition to comprehensive reports, RPM can generate a wide variety of graphs. It can plot curves, draw bar charts, and plot histograms using any combination of system information. By plotting one variable against another, you can immediately see correlations between variables. This is especially useful for determining critical resource usage points.

IDENTIFY PROGRAMS BY RESOURCE USAGE

With RPM, you can identify the programs that will provide the most benefit from optimization. You can determine which programs are used most often and which programs use the most of critical resources.

Once identified, these programs can be optimized. Overall system performance can be increased by making changes only where they will do the most good.

RPM> HISTOGRAM DLO_SEEK_DIST

	0	1	2	3	4	5	6	7	8	9	0
DLO: SEEK_DIST_1	*****										47.1
DLO: SEEK_DIST_4	*****										22.9
DLO: SEEK_DIST_16	*****										16.5
DLO: SEEK_DIST_64	*****										10.8
DLO: SEEK_DIST_100	**										2.7

RPM> HISTOGRAM DLO_DISK_USAGE

	0	1	2	3	4	5	6	7	8	9	0
DLO: DISK_USAGE_0	**										2.1
DLO: DISK_USAGE_10	**										2.0
DLO: DISK_USAGE_20	**										3.1
DLO: DISK_USAGE_30	**										3.6
DLO: DISK_USAGE_40	*****										17.6
DLO: DISK_USAGE_50	*****										42.2
DLO: DISK_USAGE_60	*****										19.8
DLO: DISK_USAGE_70	***										4.2
DLO: DISK_USAGE_80	**										3.9
DLO: DISK_USAGE_90	*										1.1

RPM> PLOT USER_CPU, MONITOR_CPU BY HOUR

	0	1	2	3	4	5	6	7	8	9	0	*	#
8.0	#											41.2	7.7
9.0	#											44.0	8.1
10.0	#											54.5	8.9
11.0	#											79.1	12.6
12.0	#											0.0	2.0
13.0	#											74.3	12.3
14.0	#											82.1	14.2
15.0	#											79.3	13.1
16.0	#											70.7	10.3

Variable	Description	Min	Max	Avg
X SAMPLE_HOUR	Hour of Day	8.0	16.0	12.0
* USER_CPU	User CPU Time	0.0	82.1	58.4
# MONITOR_CPU	Monitor CPU time	2.0	14.2	9.9

RPM> LIST TOP 5 PROGRAMS BY PRG_CPU_TICKS

PARREC	542
DAYEND	188
PAYROL	142
PAYANA	103
CLENUP	73

RPM> LIST TOP 5 FILES BY FILE_DIR_OVRHD

[3,10]PROFIL.JOU	33.1
[3,10]PROFIL.B2S	27.6
[2,0]MTHEND.TSK	19.6
[10,1]EOMREC.JOU	11.3
[2,0]PAYAUD.DAT	7.8

RPM>

NO-RISK GUARANTEE

If all this sounds too good to be true, here's your chance to find out for yourself. Try RPM for 30 days. Use it to tune your system. If RPM isn't everything we say it is and more, return it for a full refund. We're not worried. We know that once you see the difference RPM can make, you won't want to be without it.



RPM -- The Performance Revolution

NORTHWEST DIGITAL SOFTWARE

[100,199]STATES.TBL

Conclusion

As we have seen, a generalized table file maintenance and lookup facility such as Tables-11 has a number of advantages.

For one, application programmers are relieved of the chores associated with devising their own table initialization and searching routines, since these tasks are handled automatically by a standard function call, resulting in shorter program development time.

Secondly, since all table maintenance is handled by a single, generalized utility, new file maintenance and listing programs do not have to be developed every time a new table is required.

Third, there are the advantages, previously mentioned, of using external tables as opposed to internal tables. Table size does not affect program size. Because table maintenance is divorced from program maintenance, programs need not be revised, recompiled, re-linked, and retested every time a table changes.

If more than one program must access the same table, then using a single copy of an external table file ensures that all programs will use the most up-to-date information, which is not always the case when tables are "hard-coded" in a program.

In summary, using a generalized table handling package such as Tables-11 will save on installation time and money by standardizing table maintenance and lookup procedures, speeding program development and reducing program maintenance.

Further information on Tables-11 is available by contacting DataCraft at P.O. Box 292, Chestnut Hill, MA 02167.

Bibliography

Comer, Douglas, "The Ubiquitous B-Tree," *Computing Surveys*, Vol. 11, No. 2 (June 1979), pp. 121-137.

Knuth, Donald E., *The Art of Computer Programming, Vol. 1: Fundamental Algorithms*, Addison-Wesley Publ. Co., Reading, MA, 1968, Chapter 2.

Knuth, Donald E., *The Art of Computer Programming, Vol. 3: Sorting and Searching*, Addison-Wesley Publ. Co., Reading, MA, 1973, Chapter 6.

Price, C.E., "Table Lookup Techniques," *Computing Surveys*, Vol. 3, No. 2 (June 1971), pp. 49-65.

Postal Service State Codes and Names

Code	State
AK	Alaska
AL	Alabama
AR	Arkansas
AZ	Arizona
CA	California
CO	Colorado
CT	Connecticut
DC	District of Columbia
DE	Delaware
FL	Florida
GA	Georgia
HI	Hawaii
IA	Iowa
ID	Idaho
IL	Illinois
IN	Indiana
KS	Kansas
KY	Kentucky
LA	Louisiana
MA	Massachusetts
MD	Maryland
ME	Maine
MI	Michigan
MN	Minnesota
MO	Missouri
MS	Mississippi
MT	Montana
NC	North Carolina
ND	North Dakota
NE	Nebraska
NH	New Hampshire
NJ	New Jersey
NM	New Mexico
NV	Nevada
NY	New York
OH	Ohio
OK	Oklahoma
OR	Oregon
PA	Pennsylvania
RI	Rhode Island
SC	South Carolina
SD	South Dakota
TN	Tennessee
TX	Texas
UT	Utah
VA	Virginia
VT	Vermont
WA	Washington
WI	Wisconsin
WV	West Virginia
WY	Wyoming

FIGURE 1

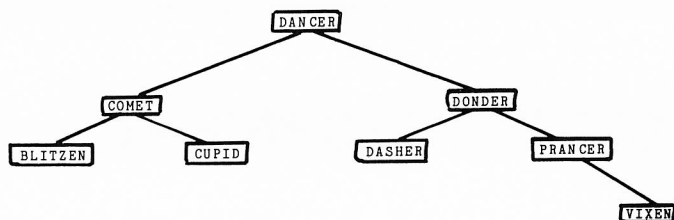


FIGURE 2

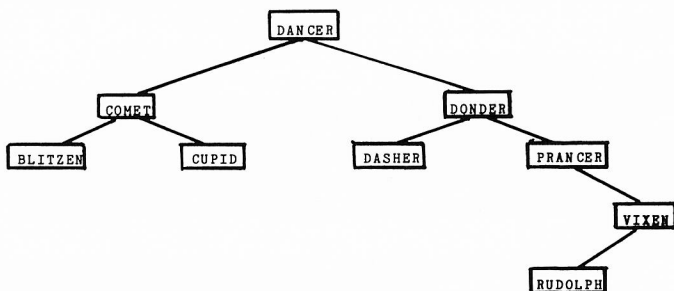


FIGURE 3

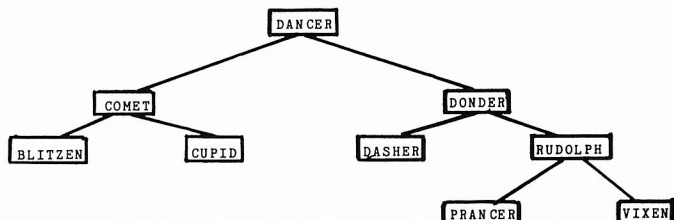


FIGURE 4

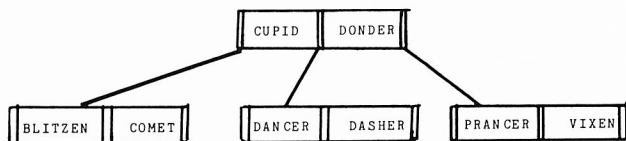


FIGURE 5

FNLOOKUP Reference Summary

Syntax: $v\% = \text{FNLOOKUP}\% (\text{key}\%, \text{tbl}\%, \text{ch}\% + \text{mode}\%)$

Data Passed: **key%** Search key. Not used if **mode%** is 32% or 64%.
tbl% Table name. Extension .TBL assumed. If null, uses the same table name as the previously executed function call on I/O channel **ch%**.
ch% I/O channel number, 1 thru 12.
mode% Type of lookup operation. Allowable values are:
 0% Normal (full key)
 16% Generic (partial key)
 32% Forward sequential
 64% Reverse sequential

Data Returned: **v%** Table data if **RC%**=0% or **RC%**=1%; else, null.
RC% Return code. Possible values are as follows:
 For **mode%** 0:
 0 Requested key was found
 -1 Requested key was not in table
 For **mode%** 16:
 0 Exact match for requested key
 1 Partial match for requested key
 2 No match for requested key; next higher entry was returned in **T1%** and **T2%**
 11 End-of-table encountered
 For **mode%** 32:
 0 Next table entry was returned
 11 End-of-table encountered
 For **mode%** 64:
 0 Previous table entry was returned
 -11 Top-of-table encountered
T1% Table key.
T2% Table data.

FIGURE 6

WANT NEW REPORTS FROM AN OLD APPLICATION?

YOU NEED OUR *REPORT-WRITER!*

GRS, The Generalized Reporting System,
runs on VAX and RSTS systems:

- More powerful than Datatrieve, but easier to use
- Much less expensive
- Compatible with most applications packages

etc ENTERPRISE
TECHNOLOGY
CORPORATION

305 MADISON AVENUE
NEW YORK, NY 10165
(212) 972-1860

CIRCLE 4 ON READER CARD

4TH GENERATION SYSTEM DEVELOPMENT

By D.G. Burnley

INTRODUCTION

This is a report on an actual system development effort using a Data Management System. There are several definitions of a DMS. A practical one might be an integrated collection of application system development tools.

DEVELOPMENT SYSTEM

The DMS used was ACT4-Data Management System-11 from the Toronto company of the same name. Features of ACT4 include an integrated data dictionary manager, and features for design, documentation and code generation. ACT4 was designed for development of the normal family of program types found in typical on-line, interactive commercial data processing systems. Program types supported include menus; master file programs with add, change, delete and inquiry; control file programs; on-line transaction processing; report; inquiry and merge-extract programs.

METHODS

As ACT4 DMS-11 is itself a menu driven, on-line interactive system, one uses it from a terminal—either screen or hardcopy. First, a selection is made from the main menu. All modules within the system follow the same general pattern. After a few identification questions, ACT4 engages the user in a plain English dialogue to ascertain the user's desires based upon the initial menu choice. Several of the modules contain menus within themselves for detailed choices.

The questions are virtually self-explanatory and normally include suggested defaults where appropriate. This method frees the user (system designer) from the constraints of language and data storage. The process allows one to consistently describe data files (once only), "paint" menus and screens, layout reports or cause selection or merging of data.

TIMING

One of the features of ACT4 is the speed with which the design, documentation and code can be generated. Elapsed time for any single step (dictionary or program) in the process is normally less than 20 minutes. This time, of course, will vary with one's terminal speed as well as familiarity with the questions being asked and complexity of file/program. CPU usage is very light. All our experience indicates less than 30 seconds CPU time. This result is consistent under all experienced system loads.

DESIGN MODE

ACT4 can itself be run in two similar modes. The first, design mode, produces documentation plus suggested program outlines. The documentation includes files, screen and report layouts as well as program narratives.

The documentation produced can be a very useful design tool. Documentation files are produced on disk and are also available for printing.

The program outlines are detailed to the code section level — much as a project leader might do (if he/she had the time) before handing over to a programmer for coding. In this mode, ACT4 produces only standard routine outlines plus section headings indicating detailed extended coding required.

DESIGN VERIFICATION

The products available at this stage can be easily used to verify the first design effort. File design, screen layouts and hence master file creation, transaction processing and report content can be easily verified for logic and completeness.

This verification procedure can be completed in much less time than one could possibly manually create this level of documentation.

ACT4 also includes an option to cross-reference and classify all usage of every data name in a system.

PROTOTYPING

The screen and report layouts along with, in particular, master and transaction program narratives, can be used to give your "customer" a static prototype of the proposed system. This documentation serves well to communicate the essence of the proposed solution to the user in an understandable manner no matter the customer's level of DP expertise. One can be reasonably sure of a successful design once the understanding and agreement of the user has been obtained for this documentation.

REDESIGN

Should either your or the customer's review indicate changes, they can be implemented quickly. As the documentation files are created on disk, and of course we have saved them, minor changes can be done with your system editor. We will probably repeat the affected steps under ACT4 in a matter of minutes for each step.

PRODUCTION MODE

The second method of running ACT4, besides producing all the documentation referred to above, creates source code and necessary control files to compile and build into runnable programs (bug and error free programs we should mention). If we have done our design work while running under production mode (very feasible if we are working from reasonable specifications or with a good understanding of system requirements), we can go directly to application system testing. Otherwise, we will need to repeat the design steps to generate the source code.

TRADE-OFFS

DMSs seem to come in two flavours; the interpretive that generate no code and hence are runnable right away; and the other type (ACT4 is such) which generate compilable source code.

The major trade-offs seem to me to be three:

1. Frequency of use of resulting application system. Normally the compile time required by ACT4 will be recouped many times over through better run times than that of interpretive systems.

2. Functional restrictions. With a code generation system, a programmer can apply a very few lines of his own code solutions to overcome any real or perceived limitations. Restrictions (real or not) in an interpretive DMS may not be so easily overcome.

3. Portability and security. Compilable systems, by definition, make use of a manufacturer's supported language thereby ensuring portability to any system supporting that language. In the case of ACT4 for example, its use of BASIC+2 (being a subset of VAX BASIC) is meant to facilitate both portability upwards and long-term viability.

The second part of this concern relates to both long-term solutions and operating system compatibility. It seems important to me that the actual data handler (for example RMS-11) also be supported by the manufacturer. While there have been and are good third party IO handlers, they tend over time to either become unsupported, receive lesser enhancements or simply be outclassed by later offerings from the manufacturer. The nightmare case, of course, is a new version of an operating system, or

even a new device's incompatibility with third party IO built into all your programs. In the case of ACT4, its dependence upon RMS-11 (the default data manager on VAX) provides about as long-term a solution with security as I can visualize.

ACTUAL CASE

The Work Order system example might well be taken as a fairly typical commercial type application subsystem. The only complexity, if you will, was caused by the fact that the existing environment required a tight two-way transaction level interface rather than the more typical one-way path. The system had to handle a fair transaction volume but volume was not sufficient to greatly impact design.

SCOPE

The resulting system is composed of 25 programs distributed between five menus, three master files, one control file, four separate transaction type programs, one major inquiry program, nine selective report/inquiry programs and two extract/merge programs. The system contains xxxx lines of source code and xxxx lines of documentation produced by ACT4.

INTERFACES

As indicated above, the system required a two-way transaction interface between the Work Order subsystem and existing GL, PO and AP systems.

ACT4 STATISTICS

Total ACT4 usage statistics were xxxx minutes of connect time and xxxx seconds of CPU usage. These totals are higher than a "perfect" usage of ACT4 would be, as we redid a few parts of the system to reflect after the fact customer changes.

The smoothest path between RSTS/E and VAX/VMS just got smoother: there's a major new release of

ROSS/V

ROSS/V has always provided:

- the fastest way to bring up RSTS/E applications on the VAX.
- the only way to do RSTS/E development on the VAX.
- an extensive subset of RSTS/E monitor calls and standard RSTS/E features, like CCLs, DOS-formatted magtape, and RSTS/E-style file update mode.

Now, in Version 3, ROSS/V supports:

- the "hidden" RSX run-time system (with 32 KW job size).
- resident libraries.
- job spawning and detached jobs.
- spooling to VMS print and batch queues.
- mailbox send/receive for communication with VAX-11 BASIC and other native mode applications.

How ROSS/V works:

ROSS/V is written in VAX-11 MACRO, and RSTS/E monitor calls are performed in VAX native mode. The rest of your PDP-11 code (in applications, run-time systems, TKB, etc.) is executed directly in the PDP-11 microcode that's present in every VAX. ROSS/V runs under VMS, not in place of it. Thus, some users may be working under the RSTS/E subsystem provided by ROSS/V while others are concurrently using any of the other VAX/VMS capabilities.

Call or write for the new ROSS/V technical summary, which describes all of ROSS/V's features.

Evans Griffiths & Hart, Inc.

55 Waltham Street
Lexington, MA 02173
(617) 861-0670

OnLine Data Processing, Inc.

N. 637 Hamilton
Spokane, WA 99202
(509) 484-3400

PDP, RSTS, RSX, VAX, and VMS are trademarks of Digital Equipment Corporation.

CIRCLE 176 ON READER CARD

ROSS/V

We found very few limitations in our use of ACT4. If measured by lines of hand written code, the answer is virtually none. Probably the greatest limitation is the system design discipline imposed by the use of ACT4. Most likely though, most of us could use the form and self-documentation features of ACT4.

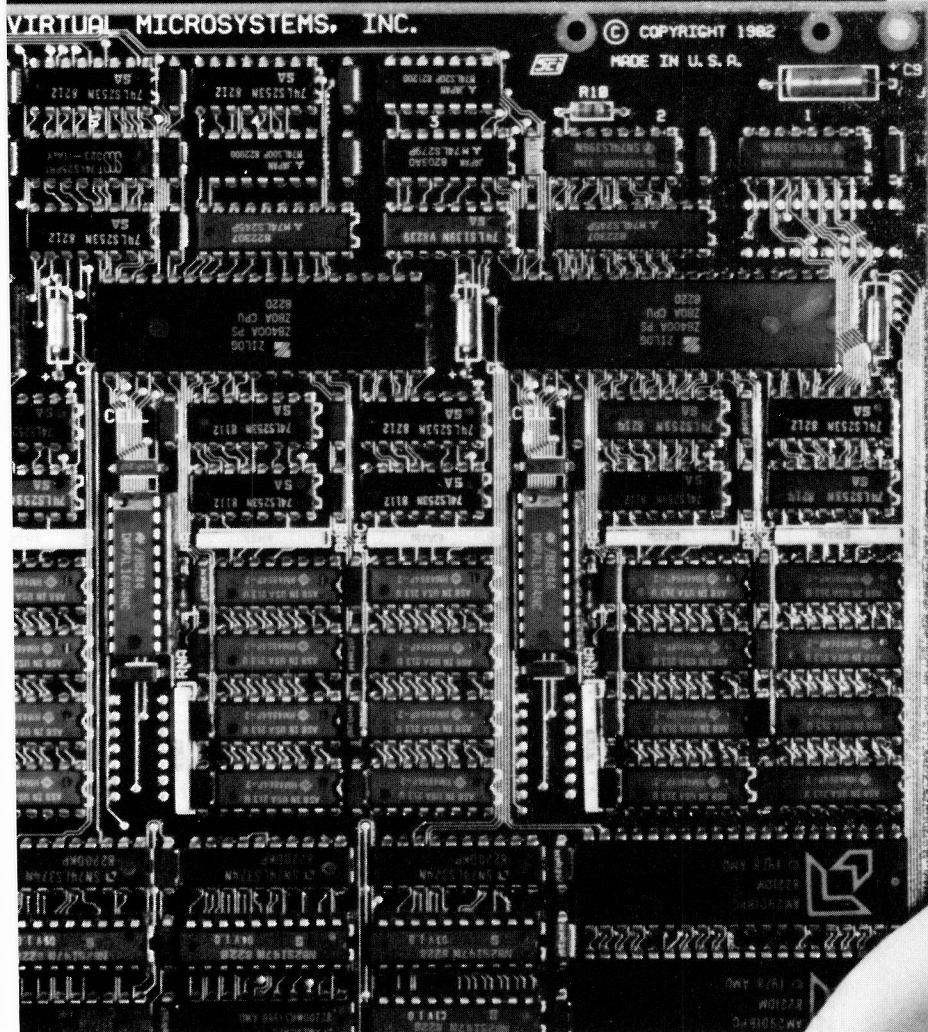
The difference between traditional system development and development with a DMS such as ACT4 is dramatic. The total design, development, documentation and system test effort for a complete subsystem might best be compared to that normally used to code one or two transaction programs of average complexity. It is literally possible to design, document and code several hundred (if not thousand) lines of error free code per day per person.

If your shop doesn't have a DMS, my advice is — get one soonest.

Field Name	Max. Size	Data type	Mask
TRAD.NO\$=10%, I SIS NUMBER	10	String	"XXXXXXXX"
TRAD.DATE\$=6%, I WORK DATE	6	String	"XXXXXX"
TRAD.DESCRPT\$=25%, I WORK DESCRIPTION	25	String	"XXXXXXXXXXXXXXXXXXXXXXXXXXXX"
TRAD.HOURS, I HOURS WORKED	10	Numeric	"##,#####.##"
TRAD.JOBNO\$=6%, I JOB/CONTROL NO	10	String	"XXXXXXXX"
TRAD.RATE\$=2%, I ST OT CODE	2	String	"XX"

All in all, we rate the MICRO-11 to be a fine machine. With more time, seasoning, a larger disk and the J-11 chip, it will go all the way and RSTS will go with it. ♥

How to give DEC users all the benefits of a CP/M based microcomputer network, without buying any microcomputers.



With the Bridge™ and the z-Board™, anyone at any terminal can run CP/M. That's fine. What's great is that they can run CP/M-based *software packages*. Like WORDSTAR®, dBase II™, or SUPERCALC™.

You get all the user benefits of these powerful, friendly, productivity packages.

Your users get access to *all* the great software written for CP/M computers.

It all runs on your DEC hardware.

And the z-Board handles the processing, so you get the fastest user response, all without tying up your DEC system's CPU resources.

You don't waste money on a lot of duplicate equipment. You don't have to inventory and maintain a fleet of floppy disk-based computers. And you don't have to tie up desk space with different terminals for different applications.

And because it all resides on your DEC system, you get a network, all without the hassle of network hardware and software.

It is quite simply the fastest, cheapest, neatest way to make your DEC UNIBUS™ or QBUS™ system run CP/M software.

If all this sounds wonderful, it is. And we can give you the references to prove it.

Just call (415) 841-9594 and ask for Tony Walker.

**virtual
microsystems**

2150 Shattuck Ave., Berkeley, CA 94704

Registered Trademarks: CP/M, CBASIC: Digital Research; WORDSTAR: MicroPro International Corporation. Trademarks: VAX, Q-Bus, UNIBUS, Professional: Digital Equipment Corporation; dBase II: Ashton-Tate; SUPERCALC: Sorcim Corporation; PERSONAL PEARL: Pearlssoft; Spellbinder: Lexisoft; MBASIC: Microsoft; Bridge, z-Board, z-Chip: Virtual Microsystems.

CIRCLE 157 ON READER CARD

Announcing MPR

A VERSATILE MACRO-LANGUAGE UTILITY FOR RSTS/E

Flexible enough for OEM's and End-Users

Functional highlights include:

- Customized code generation.
- Nestable macros and "INCLUDE" statements with arguments.
- Algebraic expression evaluation.
- Control file generation.
- Extensive debugging mode.
- Easy to use syntax.
- Complete documentation with sample macros.

Benefits:

- Reduce software maintenance costs.
- Increase software portability.
- Simplify operations and training.
- Compatible with your existing programming standards.

***Interested in significantly improving
programming productivity?***

CALL OR WRITE FOR DETAILED INFORMATION

Noah Dixon

Star Plan Data Processing, Inc.

2040 W. Wisconsin Avenue - Suite 354
Milwaukee, Wisconsin 53233 - (414) 933-0800

• *Soon to be released for RSX and VMS*

CIRCLE 178 ON READER CARD

DEC

SYSTEMS & COMPONENTS

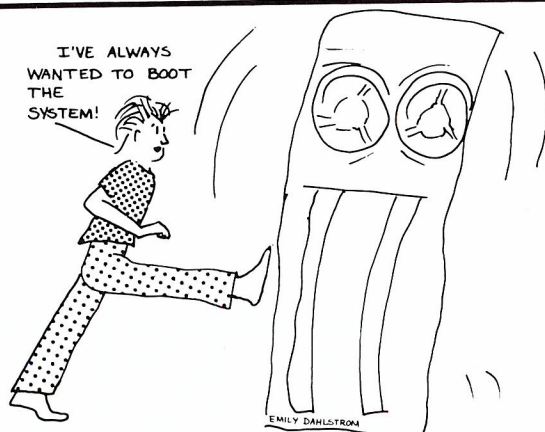
C.D. SMITH & ASSOCIATES, INC.

12605 E. Freeway, Suite 318

Houston, TX 77015

(713) 451-3112

CIRCLE 54 ON READER CARD



SMARTDCL

A DCL Alternative

(Does Anyonw Out Threr Type Az Porly As I Du?)

By Bob Stanley, Computer Methods Corporation

I used to type about 10 words a minute. Then my mother sent me to typing class in high school. After six weeks of intensive training during the summer session, I was up to about eight words a minute. Now, after spending the last five years working in front of a terminal, I have improved a little. I have managed to take the "Hunt" out of the "Hunt-and-Peck" method. However, being blessed with fat fingers, I could still chisel this article in stone faster than I could type it.

DEC has provided a nice interface to the VAX in its DCL language. The only thing they haven't provided for us slow typers is the ability to edit that last command that took five minutes to type and has two letters transposed. Or the ability to re-execute a command without typing the whole thing over again. What I needed was an interface to DCL that would let me avoid retyping as much as possible. This, therefore, is the story of SMARTDCL; my attempt to alleviate my typing problem.

SMARTDCL is a VAX BASIC V2 program that, when coupled with a small command procedure, lets you enter, execute, edit, and re-execute DCL commands. It emulates most of the features of DCL (we'll cover the exception in a minute) without any noticeable overhead.

How it Works

SMARTDCL inputs commands from the terminal and creates a DCL symbol containing the command typed by the user using the LIB\$SET__SYMBOL library routine. It then exits to the controlling command procedure via the SYS\$EXIT system service. It uses SYS\$EXIT so that it can set the \$STATUS flag to indicate to the command procedure that either the command typed should be executed, the previous command should be edited, the previous command should be re-executed, or the previous command should be displayed for reference purposes.

SMARTDCL utilizes two DCL symbols to perform command execution and command editing. The symbol "SMARTDCL" (not to be confused with the program name. Sorry about that!) contains the last command that was typed by the editor. It will have all continuation lines appended together with all control characters stripped out. The symbol "EDITDCL" will contain the last command exactly as it was typed including all control characters. We'll talk about continuing command lines in a second.

The program first gets the last value of the symbols "SMARTDCL" and "EDITDCL". It gets them via the LIB\$GET__SYMBOL library routine. It will then display the prompt "DCL>" (to distinguish between regular DCL and SMARTDCL) and will input a line typed at the terminal. It will ignore null lines and control Zs (just like regular DCL). If the user types a hyphen as the last character of the line, it will assume that a continuation line is coming and will prompt for it with the "DCL>_" prompt. As many continuation lines as needed can be entered this way.

Once the last line of the DCL statement has been typed (no hyphen at the end), SMARTDCL will place the command minus the hyphens and carriage returns into the symbol "SMARTDCL" and will place the command exactly as it was typed into "EDITDCL" via the LIB\$SET__SYMBOL library routine. The program then exits to the controlling command procedure in the normal fashion without setting the \$STATUS flag. The command procedure will then execute the command and return to run SMARTDCL again.

When it becomes necessary to exit from the command procedure back to normal DCL, the user must type EXIT at the "DCL>" prompt (you might have to type it twice depending on how many levels down into DCL you've gotten with prior commands).

Special Inputs

So far, SMARTDCL probably sounds the same as regular DCL, but now comes the fun part. The user has the option of typing several special input characters that give SMARTDCL added flexibility. If the user types an "S" (for Show) at the "DCL>" prompt, SMARTDCL will exit via the SYS\$EXIT system service while setting the \$STATUS flag to 7. The controlling command procedure will recognize the 7 and will display the last DCL command that was typed on the terminal.

If the user types an "R" (for re-execute), SMARTDCL will exit with the \$STATUS flag set to 9. This tells the command procedure to simply re-execute the last DCL command found in the "SMARTDCL" symbol.

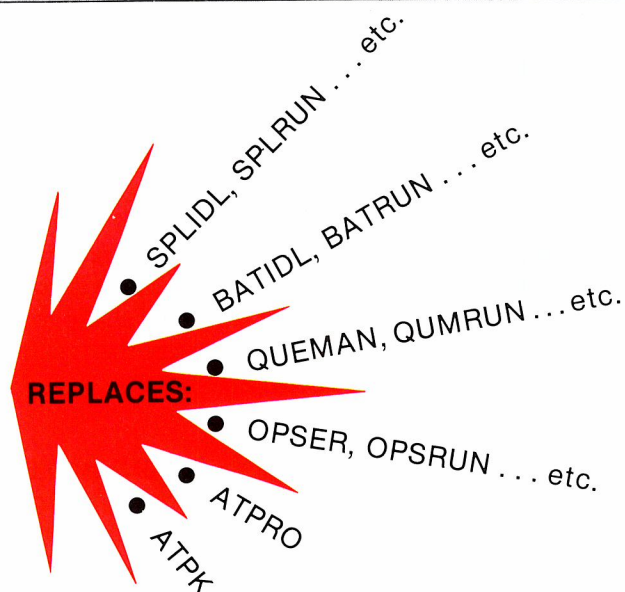
If the user types an "E" (for Edit), SMARTDCL will write the command found in the "EDITDCL" symbol out to the file SYS\$LOGIN:DCL.CMD and exit with a \$STATUS of 5 ("EDITDCL" contains the command as it was originally entered). The command procedure then invokes the editor and tells it to edit SYS\$LOGIN:DCL.CMD and to use a startup command file called RUNCOMAND.EDT (described later). After the user exits from the editor, the command procedure runs a program called EDITDCL and then re-executes the "SMARTDCL" symbol. The EDITDCL program is described below.

The last of the special input characters is a little trickier. If you've ever gotten to the end of a command and looked back and seen a mistake, you realize how frustrating it can be to delete all of those carefully and painstakingly typed letters to get back to where the error occurred. SMARTDCL, however, lets you avoid the old DELETE key (my favorite).

VERSION 2.2 NOW AVAILABLE

QUE.11 — V2.2

**ONE JOB SPOOLER
FOR RSTS/E CONTROLS
ALL SPOOLING**



QUE.11:

- DEC QUE Compatible
- Block letters on spooled header page
- One job controls all spooling
- Saves small buffers and job slots
- Spawns jobs as needed
- Handles line printer and keyboard spooling
- Controls as many BATCH JOBS as pseudo-keyboards
- Full parameter replacement in QUE
- calls "DO" command replaces indirect processors
- QUEMAN SYS call supported
- Program deliveries — NOW
- Only \$1500 single CPU license
- Trial Version \$100

For more information contact:

On Track Systems, Inc.

P.O. Box 245

Ambler, PA 19002-0245

Phone: 215/542-7008

In Europe:

Procyon Informatics, Ltd.

**19 St Kevins Road
Dublin 8, Ireland**

CIRCLE 11 ON READER CARD

If you are at the end of a DCL command, you look back and see an error and you haven't typed the RETURN key yet, you can type the ESCAPE key in place of the return key. This causes SMARTDCL to take everything you've typed up to that point (including all of the continuation lines), write it out to the DCL.CMD file and exit with a \$STATUS of 5. The command procedure will then invoke the editor as was described above. This gives you a chance to edit the current command before trying to execute it.

EDITDCL

Once a DCL command has been edited, it is necessary to re-create the "SMARTDCL" and "EDITDCL" symbols. The controlling command procedure, therefore, runs the EDITDCL program which does just that. It opens SYS\$LOGIN:DCL.CMD, reads all of the lines from it, creates the "EDITDCL" symbol in the exact form that it finds in the file, and creates the "SMARTDCL" symbol with all of the hyphens and carriage returns stripped out. This program is run only after the editor has been invoked. The logical name SYS\$LOGIN: is used so that the file is always placed in the user's default login directory. This eliminates the proliferation of little DCL.CMD files all over the system.

The Controlling Command Procedure

The controlling command procedure COMMAND.COM runs SMARTDCL and then, based on the value of the returned \$STATUS flag, executes, edits, displays, or re-executes the DCL command found in the "SMARTDCL" symbol. It sets control_Y trapping so that if a control_Y is

typed, it loops back to the top and re-executes the SMARTDCL program. If the re-execute option is chosen in SMARTDCL, it will first display the last DCL symbol and will then re-execute it.

RUNCOMAND.EDT

Whenever EDIT/EDT is invoked, it looks for a command file in the current directory called EDTINI.EDT. This file can contain startup editor directives such as key redefinitions or SET commands. You can also tell the editor to look elsewhere for this command file by including the /COMMAND = switch as part of the file specification. When COMMAND.COM invokes the editor, it does so via the statement

```
EDIT/EDT SYS$LOGIN:DCL.CMD/COMMAND=[DIRECTORY.SPEC]RUNCOMAND.EDT
```

This tells the editor to get its startup instructions from RUNCOMAND.EDT. In order to facilitate the edit step, this file should contain the following instructions:

```
DEFINE KEY 7 AS "EXT EXIT."
```

```
SET MODE CHANGE
```

The first instruction will redefine the key identified internally within the editor as key 7 (the 7 key on the keypad) as an extended exit command. This means that when you are editing in character or keypad mode, typing the 7 key on the keypad will cause you to exit the editor immediately with none of the normal exit messages being printed. If you edit a DCL command and then type the 7 key, you will exit the editor and the DCL command will then be re-executed without having to type control Z and EX as you normally would.

BACmac can do it all!

BAC into RTS / BAC into MAC / BAC into BAS

**Two Word Version
Now Available**

BACmac is a unique software tool, running under RSTS/E, which provides the following conversions:

- translation from Basic-Plus "compiled" back to Basic-Plus source code (only the comments will be missing)
- translation from Basic-Plus into Macro source code, which compiled under RSTS runs faster than Basic-Plus
- translation from Basic-Plus into Macro source code which may be compiled under RSTS for execution under RT11 — a migration facility
- translation from Basic-Plus into a RUN-TIME-SYSTEM. Now you can write an RTS in Basic-Plus. The ideal solution to memory thrashing due to "multi-copy" applications programs.

RSTS/E, RT11, Macro-11 and Basic-Plus are trademarks of Digital Equipment Corporation.

Western Distributor:
Telecom Computer Systems, Inc.
P.O. Box 03285
Portland, Oregon 97203
503/286-5122

ADOS | Advanced Digital
Office Systems

Eastern Distributor:
New England Micro Technology, Inc.
P.O. Box 767
Marblehead, Mass. 01945
617/631-6005

CIRCLE 138 ON READER CARD

The last instruction causes the editor to automatically enter character or keypad mode without the user typing the "C" command. When the editor is invoked with this startup command file, the user will be placed in character mode with the last DCL command waiting to be edited. Once the DCL command has been edited, he can type the 7 key on the keypad and the editor will exit and the command will be executed.

Exception to the Rules

As mentioned at the beginning of this article, there is at least one thing that you can do in regular DCL that SMARTDCL just won't handle. If you type part of a DCL command at the "\$" prompt such as "ASSIGN", DCL will come back with the prompt "\$_Device:". Once you enter a device name it prompts you with "\$_Log name:". SMARTDCL can't do this prompting. If you type "ASSIGN" to the "DCL>" prompt, you get the message "insufficient parameters." This is a result of the way DCL symbols work. If you type A:= ASSIGN and then A, you would get the same error message.

BASIC Version 2.0

The listings for SMARTDCL, EDITDCL, and COMMAND.COM can be found at the end of this article. SMARTDCL was written in VAX BASIC Version 2.0. An attempt was made to include as many of the new features of the language as was possible. Many of these features, although not necessary, demonstrate some of the things you can now do with the language. Several good articles describing BASIC Version 2.0 have already been included in this magazine. If you haven't tried it yet, do yourself the favor. I think you'll like it.

Conclusion

SMARTDCL can be used to facilitate the one computer process that never seems to be improved from one release of the operating system to the next: my typing. I'm sure that you have probably already thought of some extensions to it that would help you.

Now, if I can only find someone to type this article for me!

```

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
SMARTDCL.BAS
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!*****&
!&
! SMARTDCL - DCL LOOK-A-LIKE CONTROL PROGRAM&
!&
! Written by: R.S.Stanley (Computer Methods Corporation)&
! Date : 11-May-1983&
!&
! The purpose of this program is to allow a user to enter&
! DCL statements that can be executed immediately and can then&
! be re-executed or edited if need be. It will prompt the user&
! for a DCL statement (DCL> replaces the familiar $ prompt),&
! will write that command out to an output file, and will then&
! exit to a controlling command procedure which will execute&
! DCL statement. If any control statements are typed (S for&
! show the last statement, R for re-execute the last statement,&
! E for edit the last statement) the program will exit with a&
! predetermined status which can be checked by the command&
! procedure.&
!&
! This program is written in VAX BASIC Version 2.0.&
!&
!*****&

```

INTRODUCING

Business Modeller

The complete financial and numerical
modelling package for DEC computers.

■ Professional ■ PDP-11 ■ VAX ■

Business Modeller is a major advance on programs available for DEC computers. It is powerful, fast, easy to use — and ready to run.

Business Modeller is like an enormous worksheet. You enter words, numbers or formula; establish relationships between entries; and adjust the format of the worksheet at will.

Then you can alter any value and see how it affects all other — instantly and automatically! So if sales tax changes, or interest rates move, you can see what this does to your cash-flow or profit margins.

You can keep track of staff payrolls, salesmen's bonuses, loan repayments, key account purchases, research data and a host of other numerical information. Modifying and updating them at will.

And every time you want to know what would happen if...? You don't need to do hours of work — simply key in the changes and the computer will do all the recalculations for you!

So you spend your time making decisions. Not doing arithmetic.

And Business Modeller is above all simple to use. It is designed for the general manager with no computer experience.

The Unique Business Modeller Features:

- Instructions and detailed "help" keys are displayed on-screen.
- The screen — which acts like a "window" on the worksheet — can be used to view small or large work areas as required.
- 100 columns across
- 100 rows down
- 3 Dimensional
- The system has a wide range of inbuilt functions covering financial, statistical and scientific needs. These include:
 - MAXIMUM/MINIMUM/AVERAGE of a list
 - SUM/COUNT of items in a list
 - LOOK-UP/CHOOSE from a table
 - LOGIC expressions
 - TRIGONOMETRIC expressions
 - and many more.
- You can add to these functions by developing and permanently installing functions essential to your own business needs such as:
 - DCF, Compound Interest, Performance Ratios, complex mathematical formula, etc., etc.
- Prices from \$650

For details, etc., contact:

LOGSYS BUSINESS SYSTEMS

116 JOHN STREET, SUITE 1408 ■ NEW YORK, NY 10038 ■ TEL. (212) 608-5310

5 ALBERT ROAD ■ CROWTHORNE, BERKSHIRE, UK ■ TEL. (0344) 777877

```

DATA_DECLARATION_SECTION:
  OPTION TYPE=EXPLICIT
  DECLARE STRING
    DCL.COMMAND, &
    PROMPT.TEXT, &
    LAST.CHAR, &
    SHORTENED.DCL.COMMAND, &
    ORIGINAL.DCL.COMMAND, &
    EDIT.DCL.COMMAND, &
    LAST.DCL.COMMAND
  DECLARE STRING CONSTANT
    EDIT.DCL = "E", &
    SHOW.DCL = "S", &
    re-execute.DCL = "R", &
    HYPHEN = "-"
  DECLARE INTEGER
    E, &
    CONTINUATION.LINE
  DECLARE INTEGER CONSTANT
    TRUE = (1=1), &
    FALSE = NOT TRUE, &
    TRIM.TRAILING.BLANKS = 128%, &
    CNV.LOWER.TO.UPPER = 32%, &
    TRIM.EXCESS.CHARACTERS = 4%
  DECLARE LONG FUNCTION
    FNGET_FIRST_DCL_COMMAND, &
    FNEDIT_COMMAND, &
    FNCHECK_FOR_SPECIAL_INPUT, &
    FNGET_MORE_INPUT
  EXTERNAL LONG FUNCTION
    LIB$SET_SYMBOL, &
    LIB$GET_SYMBOL, &
    SYS$EXIT(INTEGER BY VALUE)
  OPEN "TT" FOR INPUT AS FILE 2%

PROGRAM_CONTROL_SECTION:
  E = LIB$GET_SYMBOL ("SMARTDCL",LAST.DCL.COMMAND)
  E = LIB$GET_SYMBOL ("EDITDCL",EDIT.DCL.COMMAND)
  E = FNGET_FIRST_DCL_COMMAND WHILE EDIT$(DCL.COMMAND,-1%) = ""
  E = FNCHECK_FOR_SPECIAL_INPUT
  DCL.COMMAND = EDIT$(DCL.COMMAND,TRIM.EXCESS.CHARACTERS + &
    TRIM.TRAILING.BLANKS)
  LAST.CHAR = RIGHT(DCL.COMMAND,LEN(DCL.COMMAND))
  SHORTENED.DCL.COMMAND = DCL.COMMAND
  ORIGINAL.DCL.COMMAND = DCL.COMMAND
  SHORTENED.DCL.COMMAND = LEFT(SHORTENED.DCL.COMMAND, &
    LEN(SHORTENED.DCL.COMMAND)-1%) IF LAST.CHAR = HYPHEN
  WHILE LAST.CHAR = HYPHEN
    E = FNGET_MORE_INPUT
    E = FNEDIT_COMMAND IF RIGHT(DCL.COMMAND,LEN(DCL.COMMAND))=ESC
    DCL.COMMAND = EDIT$(DCL.COMMAND,TRIM.EXCESS.CHARACTERS + &
      TRIM.TRAILING.BLANKS)
    SHORTENED.DCL.COMMAND = SHORTENED.DCL.COMMAND + DCL.COMMAND
    ORIGINAL.DCL.COMMAND = ORIGINAL.DCL.COMMAND + CR + LF + &
      DCL.COMMAND
    LAST.CHAR = RIGHT(DCL.COMMAND,LEN(DCL.COMMAND))
    SHORTENED.DCL.COMMAND = LEFT(SHORTENED.DCL.COMMAND, &
      LEN(SHORTENED.DCL.COMMAND)-1%) IF LAST.CHAR = HYPHEN
  NEXT
  E = LIB$SET_SYMBOL ("SMARTDCL",SHORTENED.DCL.COMMAND)
  E = LIB$SET_SYMBOL ("EDITDCL",ORIGINAL.DCL.COMMAND)
  CLOSE #1%, #2%
  GOTO END_OF_PROGRAM

DEF LONG FNGET_FIRST_DCL_COMMAND
  ON ERROR GOTO GFDC_ERROR
  CONTINUATION.LINE = FALSE
  PRINT IF CCPOS(2%) > 0
  10000 INPUT LINE #2%, "DCL> "; DCL.COMMAND
  FNEXIT
GFDC_ERROR:
  IF ERR = 11 THEN
    DCL.COMMAND = ""
    RESUME 10000
  END IF
  END DEF

DEF LONG FNCHECK_FOR_SPECIAL_INPUT
  E = FNEDIT_COMMAND IF RIGHT(DCL.COMMAND,LEN(DCL.COMMAND)) = ESC
  SELECT EDIT$(DCL.COMMAND,TRIM.TRAILING.BLANKS+CNV.LOWER.TO.UPPER+ &
    TRIM.EXCESS.CHARACTERS)
  CASE "E"
    OPEN "SYS$LOGIN:DCL.CMD" FOR OUTPUT AS FILE 1%
    PRINT #1%, EDIT.DCL.COMMAND
    E = SYS$EXIT(5%)
  CASE "S"
    E = SYS$EXIT(7%)
  CASE "R"
    E = SYS$EXIT(9%)
  END SELECT
  END DEF

DEF LONG FNGET_MORE_INPUT
  ON ERROR GOTO GMI_ERROR
  CONTINUATION.LINE = TRUE
  PRINT IF CCPOS(2%) > 0
  INPUT LINE #2%, "DCL> "; DCL.COMMAND
  FNEXIT
GMI_ERROR:
  IF ERR = 11 THEN
    DCL.COMMAND = ""
    RESUME 10100
  END IF
  10100 END DEF

DEF LONG FNEDIT_COMMAND
  OPEN "SYS$LOGIN:DCL.CMD" FOR OUTPUT AS FILE 1%
  PRINT #1%, ORIGINAL.DCL.COMMAND + CR + LF + EDIT$(DCL.COMMAND,132%) &
    UNLESS ORIGINAL.DCL.COMMAND = ""
  PRINT #1%, EDIT$(DCL.COMMAND,132%) IF ORIGINAL.DCL.COMMAND = ""
  E = SYS$EXIT(5%)
  END DEF

END_OF_PROGRAM:
  END

```

```

=====
EDITDCL.BAS
=====
10  EXTERNAL LONG FUNCTION LIB$SET_SYMBOL
    OPEN "SYS$LOGIN:DCL.CMD" FOR INPUT AS FILE 1%
    ON ERROR GOTO 19000

    INPUT LINE #1%, DCL.COM$
    UNTIL EOF%
      DCL.COM$ = EDIT$(DCL.COM$,132%)
      DCL.COM$ = LEFT(DCL.COM$,LEN(DCL.COM$)-1%) &
        IF RIGHT(DCL.COM$,LEN(DCL.COM$)) = "-"
      DCL.COMMAND$ = DCL.COMMAND$ + " " + DCL.COM$
      EDIT.COMMAND$ = EDIT.COMMAND$ + " " + CR + LF + DCL.COM$ &
        UNLESS EDIT.COMMAND$ = ""
      EDIT.COMMAND$ = DCL.COM$ IF EDIT.COMMAND$ = ""
      INPUT LINE #1%, DCL.COM$

    10000 NEXT
      DCL.COMMAND$ = EDIT$(DCL.COMMAND$,24%)
      EDIT.COMMAND$ = EDIT$(EDIT.COMMAND$,24%)

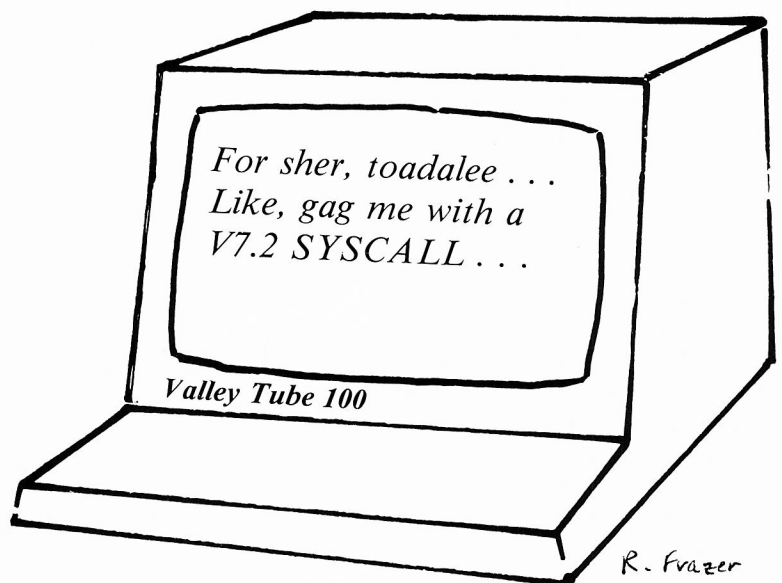
      E% = LIB$SET_SYMBOL ("SMARTDCL",DCL.COMMAND$)
      E% = LIB$SET_SYMBOL ("EDITDCL",EDIT.COMMAND$)
      GOTO 32767

    19000 IF ERR = 11 THEN
      EOF% = -1%
      RESUME 10000
    END IF

    32767 END

=====
COMMAND.COM
=====
$ SET NOON
$ SET CONTROL=Y
$ ON CONTROL_Y THEN GOTO TOP
$TOP:
$ VFY = 0
$ RUN PHODEV:[IOD.PROD.PROGUTIL]SMARTDCL
$ IF $STATUS .EQ. 5 THEN GOTO EDIT_STEP
$ IF $STATUS .EQ. 7 THEN GOTO TYPE_STEP
$ IF $STATUS .EQ. 9 THEN VFY = 1
$re-execute:
$ IF VFY .EQ. 1 THEN WRITE SYS$OUTPUT "'SMARTDCL'"
$ ASSIGN/USER SYS$COMMAND SYS$INPUT
$ 'SMARTDCL
$ GOTO TOP
$EDIT_STEP:
$ ON CONTROL_Y THEN GOTO TOP
$ ASSIGN/USER SYS$COMMAND SYS$INPUT
$ ASSIGN/USER NL: SYS$OUTPUT
$ EDIT/EDIT SYS$LOGIN:DCL.CMD -
  /COMMAND=PHODEV:[IOD.PROD.PROGUTIL]RUNCOMAND.EDT
$ PURGE SYS$LOGIN:DCL.CMD
$ RUN PHODEV:[IOD.PROD.PROGUTIL]EDITDCL
$ GOTO re-execute
$TYPE_STEP:
$ WRITE SYS$OUTPUT "'SMARTDCL'"
$ GOTO TOP

```



PUT A LITTLE MAGIC IN YOUR DEC!

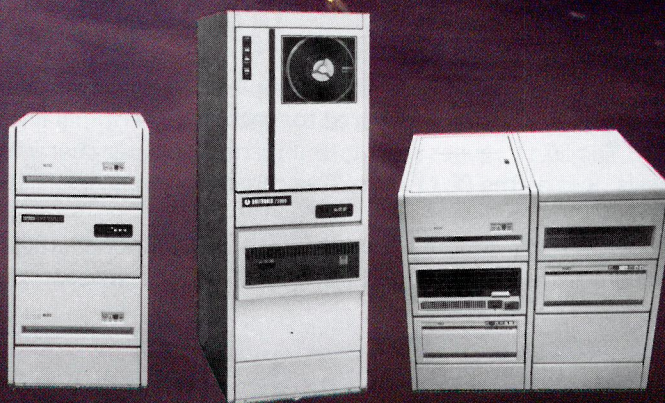


Unitronix introduces the DMT 3000 Magnetic Tape Drive — fully compatible with DEC Micro 11, PDP-11/23+, 11/24, 11/44 and VAX. . . running under RT11, RSX11, RSTS, TSX, UNIX and VMS.

- Connects directly to DEC Micro 11, PDP-11/23+, 11/24, 11/44 and VAX, as well as Data General Eclipse and Nova computers without program changes
- Only \$7,500 (single quantity) *with controller*
- Eliminates the need for costly disk packs. Eliminating 10 large disks pays for the system
- Safer, more reliable backup medium
- Accommodates ISO, ANSI, ECMA and IBM standards



197 Meister Avenue, Somerville, NJ 08876
(201) 231-9400 ■ TELEX: 833184



DEC, PDP, VAX, RT11, RSX11, RSTS and VMS
are trademarks of Digital Equipment Corporation
UNIX is a trademark of AT&T

CIRCLE 23 ON READER CARD

Understanding Terminal Interface Performance

1.0 Introduction

The purpose of this article is to discuss various terminal interfaces, how they function, and how they load a PDP-11 computer.

The devices that will be discussed are the DL11 compatible interfaces, the DZ11 compatible interfaces, the DH11 compatible interfaces, and the DMF32 compatible interfaces.

2.0 DL11 Compatible interfaces

The DL11 is a single-line interface. This means that one DL11 controller is required per terminal line.

The DL11 causes an output interrupt on a per-character basis. As a result, the PDP-11 must stop whatever it is doing to service the DL11. This causes a great deal of overhead, especially at high baud rates.

The DL11 causes an input interrupt on a per-character basis. As a result, every time any user types any key, the processor must be interrupted. In addition, the DL11 has only one character of internal buffering. If the processor is very busy, the DL11 can lose characters (called overrun errors). The DL11 is so slow, that on some processors, VT100 escape sequences can get "lost" when running the terminal at high baud rates.

The DL11 does not support software programmable baud rates. To change a speed on a DL11, it is necessary to change an option strap on the controller board.

The DL11 does not have hardware parity computation. The operating system's DL11 driver must perform parity computation if it is desired. This causes system overhead.

The DL11 does not offer modem control. DL11s are used for local lines only.

3.0 DZ11 Compatible interfaces

The DZ11 is an eight-line multiplexer. This means that one DZ11 controller is required for each eight terminal lines.

The DZ11 causes an output interrupt on a per-character basis, just as the DL11 does. This causes the same overhead as a DL11.

The DZ11 causes an input interrupt on a per-character basis, just as the DL11 does. Unlike the DL11, however, the

DZ11 has an input character buffer (called a SILO). It is very unusual for a DZ11 to have input overrun errors. It is possible to read in several characters from the input SILO if they are entered before the software has a chance to read them.

The DZ11 supports software programmable baud rates.

The DZ11 supports hardware parity computation. This means that there is no additional software overhead for parity computation.

The DZ11 has limited modem control. It is necessary for software to examine DZ11 status registers periodically to support a modem. In the case of RSTS/E, the DZ11 status registers are examined once a second, every second.

4.0 DH11 Compatible interfaces

The DH11 is a sixteen-line multiplexer. This means that one DH11 controller is required for each sixteen terminal lines.

The DH11 supports DMA output. This means that the terminal driver can place output data in memory, and instruct the DH11 to read it from memory, and interrupt the processor as soon as all the data is output. Although there is more overhead involved in setting up a DMA transfer, the transfer itself does not require processor overhead. As a result, the DH11 performs worse than a character-interrupt device on very small transfers (three characters or less is typical) but far better than character interrupt devices on transfers of four or more characters. Because of the design of the RSTS/E terminal driver, the DH11 will output as much as 30 characters without interrupting the processor.

The DH11 causes an input interrupt on a per-character basis. It has an input SILO just like the DZ11, and as a result performs almost identically on input. Multiple characters can be read with a single input operation if they are typed before software has a chance to read them.

The DH11 supports software-programmable baud rates.

The DH11 supports hardware parity computation. There is no software overhead involved in parity generation and checking on DH11 lines.

The DH11, assisted by the DM11BB modem control multiplexor, has full modem control. The DM11BB causes a processor interrupt on ring indications and carrier loss indications. This means that the software that supports modem control only runs when it needs to, rather than once a second as the DZ11 does.

5.0 DMF32 Compatible interfaces

The DMF32 is an eight-line multiplexer. This means that one DMF32 controller is required for each eight terminal lines.

The DMF32 supports both DMA output and programmed I/O output. As a result, it can be used in programmed I/O mode for short transfers, and DMA output for long transfers. In addition, the DMF32 contains an output SILO which allows programmed transfers of more than one character at a time. The ability to support both DMA and programmed I/O transfers causes DMF32 performance to exceed both the DH11 and DZ11 on output.

The DMF32 handles input differently from the previous multiplexers. Rather than interrupting the processor as soon as a character is input, it waits for a predetermined amount of time (programmed by software) before interrupting. This allows the DMF32 driver to read several characters from the input SILO with a single interrupt. The time delay is set to be short enough so that a user will not notice any degradation of response, yet long enough so that interrupts will be reduced when data is being entered quickly into the DMF32.

The DMF32 supports software programmable baud rates.

The DMF32 supports hardware parity computation. There is no software overhead involved in parity computation on DMF32 lines.

The DMF32 has full modem control. The modem control signals are given to software through the input SILO, so they may also be processed with input character interrupts. There is less overhead with DMF32 modem control than DM11BB modem control, since the modem control information may be read during the same interrupt as input character input.

6.0 Conclusion

The DL11 causes the most overhead and is the least reliable, since it is character interrupt and does not have an input SILO.

The DZ11 causes slightly less overhead since it has an input SILO.

The DH11 causes even less overhead since it supports DMA output.

The DMF32 has the least overhead since it supports both programmed transfers and DMA on output, and has a time-delay feature on its input SILO.

Ed. Note: Some DL-11 models will support modem control.

DEXOMP FALL '83 SHOW/SEMINARS ... YOU'RE INVITED TO TRADE WITH THE LEADERS IN THE DEC* COMPATIBLE MARKET.

The unique **DEXOMP** format gathers the leading DEC* compatible product competition at one location in an informal atmosphere. The personal time for practical in-depth discussions among **DEXOMP** principals and qualified clients is unrivaled.

Among those displaying their newest and best in DEC-compatible technologies are:

- ACC
- Cambridge Digital Systems
- Datasystems Corp.
- Dataram
- Distributed Logic Corp.
- Emulex
- National Semiconductor
- Western Peripherals

Whether you're a DEC system user or on OEM, you'll discover innovative ways to lift, extend, improve & enhance your DEC systems . . . most economically!

BE ONE OF THE WINNERS DURING THE DEXOMP FALL SERIES.

- Sept. 19 — Pittsburgh
- Sept. 22 — Rochester
- Oct. 11 — Philadelphia
- Oct. 13 — Washington, D.C.
- Nov. 15 — St. Louis
- Nov. 17 — Minneapolis

Technical seminars (net working concepts, & new innovations & applications) complement each day-long display/hospitality event.

Contact Dianne for your personal invitation and join the DEC compatible/competitive group.



2021-113 Business Center Drive
Irvine, California 92715

714/851-0623

*DEC is a trademark of the Digital Equipment Corporation.
No endorsement of DEXOMP by DEC is intended or implied.

CIRCLE 190 ON READER CARD

VAX SECURITY

By Terry C. Shannon

It is rumored that DEC will provide a System Manager's Security Guide and a new file protection scheme called an Access Control List (ACL) in a future major release of VMS. In the meantime, the following article will hopefully supplant the VMS security information currently available in DEC documentation and elsewhere.

Soon after gaining exposure to our new VAX 11/750 system, it became evident to me that the default security measures implemented in VMS 3.0 and at our installation were superficial at best. The purpose of this article is to point out potential weaknesses in system security and to suggest countermeasures that can be utilized to increase system security to an acceptable level. Although many installations may have no apparent need for enhanced security measures, consider the implications of the inquisitive employee who has gained access to your corporate accounting system, the larcenous individual who has just made a full backup of your entire system (including the \$20,000.00 software package that you just purchased) or the disgruntled computer operator who announces his or her resignation from employment by totally corrupting system files and backup media. Each of these potential EDP disasters is possible, but they all can be prevented in advance by invoking reasonable security conventions.

THE PROBLEM

System security may be compromised in two general ways: internally, where an individual has physical access to the computer and its terminals (and more than likely a valid username and password), and externally, where penetration is attempted from a remote location by somebody who has no legitimate right to access the system. Following are examples of both kinds of unauthorized access, how each may be employed to penetrate your system, and countermeasures that you can use to defeat them.

A TYPICAL INTERNAL COMPROMISE

The most elementary kind of internal compromise involves determining the password of a privileged user. This can be accomplished by guesswork, direct observation or through the use of a password snatcher program. A password snatcher program can be run from any terminal with sufficient privileges and can be written very easily in DCL (I do not profess to be a DCL expert, yet I was able to write and execute a successful password snatcher program in less than thirty minutes). The program works as follows:

When the snatcher program is executed, it allocates the target terminal and displays a simulated logout and the usual blinking cursor. When a victim attempts to log into the terminal, he receives the normal VMS welcome and is prompted for his username and password. After trapping the username and password, the grabber program returns a

user authorization failure, deallocates the terminal and goes away. The unsuspecting user believes that he merely mistyped his password, so he attempts another login. This time, he succeeds. The author of the grabber program has also succeeded in obtaining a username/password pair. A variant of this technique involves a simulated program crash during an interactive session. While running a program, the user is confronted with an unexpected program crash and stack dump. He is then prompted for username and password as if the crash had managed to log his process out. He dutifully logs back in and thus provides the perpetrator of the "crash" with his username/password pair.

One way to circumvent these techniques of penetration is to set the system parameters TTYOWNER and TTYPROT to prevent allocation of a terminal by another user. Additionally, users should report any irregularities or problems immediately to the System Manager. In a small installation, it might also be a good idea to enter the DCL command SHOW USERS on a regular basis — particularly just before quitting time or during other off-peak periods.

EXTERNAL PENETRATION

A technique which can be used to penetrate a system externally is a password guesser or generator program. Such a program will create its own passwords and use them to attempt a successful login. This is analogous to opening a combination lock by entering sequentially incremented numbers until the right combination is reached. Although this can be done manually, a program can accomplish the penetration at a much higher rate of speed. The best way to circumvent this form of compromise is to ensure the security of your modem — a dialup penetration would most likely be attempted. Obviously, this includes keeping your dialup number unlisted and confidential. It is also possible to write a program to keep track of logfiles which will issue an alarm or disable the offending account when logfiles reach a predetermined number. If you feel that you need more stringent security measures, there are commercially available add-on devices for modems which force you to enter a password during dialup before the modem will connect with the system.

WHAT CAN YOU DO?

As System Manager, you should analyze your existing system and procedures from a security standpoint. Look for potential weak points and utilize the appropriate suggestions contained in the VAX/VMS System Manager's Guide and this article to increase system security. Below are listed some factors and techniques to take into consideration when conducting a security audit and upgrading the security of your system:

Several relatively simple methods are available for the

System Manager to use to prevent unauthorized system utilization by normal users. User accounts can be disabled via the UAF by invoking login restrictions based on day of week and time. It is also possible to create and use an alternate UAF during weekends and non prime time periods. Normal user accounts could be omitted from this alternate UAF to effectively prevent logins outside of predetermined periods.

It is also possible to write a command procedure which prompts for a second password after the initial login sequence. Such a procedure, run as a privileged image, would entail having a different secondary password for each account and would serve as an additional measure of security.

Generally, privileges should be kept to a minimum in all accounts. In the event that a user with minimal needs requires special privileges to perform tasks like queue management or backup, you can write a captive login procedure and command procedure which will equate symbols with DCL commands. Using this technique, the user is able to perform privileged tasks without ever accessing the command language interpreter and receiving the DCL \$ prompt. As an example, an operator would be able to run system backups and other jobs requiring elevated privileges without having the ability to use these privileges for his own purposes.

When reviewing security at your installation, don't forget default accounts such as User and Field Service. These accounts should be disabled when not in use to eliminate the

possibility of penetration. When the accounts are needed, simply run Authorize to enable them.

WHAT CAN USERS DO?

Users can aid in maintaining security by changing their passwords on a regular basis and by avoiding very brief or

easily determined passwords. The use of one's middle name, telephone number or the name of relative should be prohibited. I would suggest changing passwords at least once a month and whenever an employee terminates or no longer needs access to the computer system. It is also a good idea to establish a minimum password length (at least seven characters) to decrease the probability of discovery of a password by guesswork. One method that can be used to keep users alert in this regard is to periodically attempt to guess their passwords and penetrate their accounts. You can also write a program which will inform a user who is logging into an account, the last time that a login or logout to that account took place. As previously mentioned, educate your users to inform you im-

mediately if they notice anything abnormal during their use of the system.

HARDWARE SECURITY

You should very carefully protect your operator console. If the UAF is missing, inaccessible or corrupted, it is possible to log into the system (UIC [1,4], all privileges) by

Software Tools for RSTS/E

Evans Griffiths & Hart, Inc., a pioneer in the development of RSTS and the winner of an ICP million dollar award for KDSS and TAM, offers packages that save you time and improve your productivity.

- **KDSS**, a complete multi-terminal key-to-disk data entry subsystem. Eliminates the need for keypunching and stand-alone key-to-disk systems. (Also available for VAX/VMS and RSX-11M.)
- **TAM**, an efficient multi-terminal screen-handling facility that provides complete support for the development of transaction-processing applications on a wide variety of terminals. (Also available for VAX/VMS and RSX-11M.)
- **FSORT3**, a very fast sort/merge package for RMS and non-RMS files. More economical of disk space than SORT-11 and much faster.
- **SELECT**, a convenient, very fast package for scanning files to extract records that meet user-specified selection criteria. Use as part of an online inquiry system and as a front end for building file indices and generating reports. SELECT and FSORT3 can save hours in nightly batch runs.
- **BSC/DV**, a RSTS/E device driver for the DEC DV11 synchronous multiplexer. Suitable for handling a wide variety of bisynchronous protocols. (Also available for VAX/VMS.)
- **COLINK**, a convenient, efficient link between two RSTS/E systems using DMC11s or DMR11s without the overhead of DECnet. Supports file transfers, virtual terminals, and across-the-link task communication.
- **DIALUP**, a comprehensive, efficient link between RSTS/E and other systems using asynchronous terminal lines. Supports file transfers, virtual terminals, auto-dialing, and the use of command files and macros. The premier RSTS/E package for remote support and reliable, CPU-efficient file transfers.

DEC, DECnet, RSTS, RSX, VAX, and VMS are trademarks of Digital Equipment Corporation.

Call or write for complete descriptions of features and benefits.

Evans Griffiths & Hart, Inc.
55 Waltham Street, Lexington, MA 02173
(617) 861-0670

EGH

CIRCLE 29 ON READER CARD

entering a username and password of one's choice at the operator console. One could also halt the processor with a CTRL P and run a conversational bootstrap directing VMS to an alternate UAF tailored to give systemwide access and all privileges. Don't rely on the console key as an effective safeguard — all 750 and 780 keys are identical.

You should also be careful about your backup procedures. Keep your backup media under lock and key and be sure to store a duplicate full backup at a secure off-site location.

CONCLUSION

This is by no means a complete review of security measures that can be implemented on a VAX system, but it does encompass a number of no-cost techniques that you can easily put into effect on your system to reduce the probability of unauthorized usage. I hope that you find at least some of these suggestions to be beneficial in increasing the security at your own site.

Terry Shannon is System Manager for Galson Technical Services, Inc., a prominent firm specializing in industrial hygiene, pollution analysis and abatement, and consulting engineering. He has more than ten years data processing experience which began with a tour of duty in the Army Security Agency as a Telecommunications Security Specialist and Signal Intelligence Analyst.

When he isn't working, Terry enjoys reading, camping, travel and technical writing.



VAX/VMS SOFTWARE



RESOURCE MANAGEMENT and CHARGEBACK SYSTEM

DP Managers use ARSAP for:

- User and Project Accounting
- Monitoring Usage and Trends
- Controlling Performance
- Billing for Services

Also Available for RSTS and RSX Systems



P.O. Box 188
Riverdale, MD 20737 (301) 864-3700
VAX, RSX, and RSTS are trademarks of the Digital Equipment Corp.

CIRCLE 182 ON READER CARD

ISCAN.BAS

By Davis S. Williams, Jr.
Yampa Valley Electric Association, Inc.
Steamboat Springs, CO

We are running RSTS/E V7.0-07 and our ISMUTL.TSK is V04C-01. The problem we had was that ISMUTL will not allow you to take a STATUS listing of any ISAM file that is open. We were having a difficult time keeping track of all our ISAM files and which ones needed to be reorganized.

This program reads the system monitor tables to find all the disk devices attached to the system. It then reads the MFDs and all the UFDs for all the disks looking for ISAM files. Each ISAM file is then listed on the video screen (VT100) showing the physical location of the file, the DEV:FILNAM.EXT of the file, the total number of records in the file, the total number of groups allocated to overflow, the number of groups used from overflow, the percentage of overflow usage, whether the file needs to be reorganized, and finally whether there exists sufficient free space on the disk to successfully reorganize the file.

If the program is executed from a CCL, you will get just the screen output. If the program is executed from the system prompt, you have the option of also creating a batch control file. This file will have all the necessary information to run an ISMUTL reorganize on all the files that need it and have enough disk free space to complete it. You can then run the control file using BATCH or let QUEMAN do it for you at night.

```

1      EXTEND
10     !
!      Program:      ISCAN.BAS
!      Date:         22-Feb-83
!      Version:      1.0
!      Author:       David S. Williams Jr.
!
!      Yampa Valley Electric Association, Inc.
!      Post Office Box 1218
!      Steamboat Springs, Colorado  80477
!      (303) 879-1160
!
! *****
!
! This program is furnished free of charge to any RSTS/E site.
! It may be distributed and copied so long as there is no
! direct profit derived from such distribution.
!
! Although this software has been tested and is believed to
! be error-free, neither Yampa Valley Electric or the author
! assume responsibility for the use or reliability of the
! software.
!
! Any problems with the software should be reported to the
! author at the address above, although support of the
! software is not guaranteed.
!

```



```

*****
!
! PROGRAM DESCRIPTION
!
! ISCAN.BAS looks for ISAM files by reading all the disk units
! connected to the system. A screen display (VT100) is created
! showing the physical location of the file, the DEV:FILNAM.EXT
! of the file, the total number of records in the file, the
! number of groups allocated to overflow, the number of groups
! used in overflow, the percentage of overflow used, whether
! the file should be reorganized, and whether there is enough
! free space on the disk to successfully reorganize the file.
!
! The system monitor tables are read using SYS calls so the
! program will access any type of disk unit connected to the
! system. The disk free space is also taken from the monitor
! tables.
!
! The program will open the ISAM files using MODE 4096% (read
! regardless) so the listing can be taken even when the files
! are in use.
!
! If a CCL is set up to start the program starting at line
! 30000, the program will show all the ISAM files on the system
! using a formatted screen display.
!
! If the program is executed by a RUN statement, the operator
! will be given the option to build a batch control file.
! Whether the batch file is to be built or not, the ISAM files
! will be shown on the screen using the formatted display.
! The control file is created as BAT:REORG.CTL but the file
! location and name can be changed on line 120 if desired.
! The control file will contain all the statements needed to
! reorganize all the ISAM files that have a YES or a MAYBE
! response to 'NEEDS REORG' and a YES response to 'ENOUGH
! SPACE' for the reorganize. This control file can be executed
! by using BATCH or it can be spooled for later execution by
! QUEMAN.
!
! The constants that determine whether a file needs to be
! reorganized can be changed on line 30510 to suit a particular
! need. Initially, they have been set so that under 25% of
! overflow use causes a NO response, from 25% to 40% use causes
! a MAYBE response, and over 40% use causes a YES response.
!
! The constants that determine whether enough free disk space
! exists to successfully reorganize a file can be changed on
! line 30530. Initially, they have been set so that free space
! under 1.5 times the total file size (index allocation plus
! all data file allocations) causes a NO response, free space
! from 1.5 to 2.0 times the total file size causes a MAYBE
! response, and free space over 2.0 times the total file size
! causes a YES response.
!
*****
!
! Print screen heading and find out if batch control file is
! to be built.
!
PRINT #KB%, CHR$(155%) + "<";
\ PRINT #KB%, CHR$(155%) + "[H" + CHR$(155%) + "[J" +
  CHR$(155%) + "#3" + CHR$(155%) + "[7m";
\ PRINT #KB%, "SCAN ISAM FILES FOR REORGANIZING"
\ PRINT #KB%, CHR$(155%) + "#4";
\ PRINT #KB%, "SCAN ISAM FILES FOR REORGANIZING"
110 PRINT #KB%, CHR$(155%) + "[Om" + CHR$(155%) + "[5;H" +
  CHR$(155%) + "[J"
\ INPUT #KB%, "Do you wish to build a batch control file";BAT.$
\ BAT% = ASCII(CVT$(BAT.$,-1%))
\ GOTO 120 IF BAT% = 89% "Y"
\ GOTO 30000 IF BAT% = 78% "N"
\ PRINT #KB%
\ PRINT #KB%, "Please <Y>es or <N>o only!!! " + CHR$(7%);
\ SLEEP 3%
\ GOTO 110
120 OPEN "BAT:REORG.CTL" FOR OUTPUT AS FILE #5%
\ PRINT #5%, "$JOB/RT11"
30000 !
! Define functions to read through MFD and UFD
!
DEF* FNMLNK$(L%) =
  (((L% AND 3584%)/512%) * MCLU% +
  (SWAP$(L% AND -4096%)/16%)) * 32% +
  ((L% AND 496%)/16%)
\ DEF* FNULNK$(L%) =
  (((L% AND 3584%)/512%) * UCLU% +
  (SWAP$(L% AND -4096%)/16%)) * 32% +
  ((L% AND 496%)/16%)
30110 !
! Dimension statements for SYS calls
! Dim SYS.3 array for monitor tables, part 1
! Dim SYS.12 array for monitor tables, part 2
!
DIM SYS.3%(30)
\ DIM SYS.12%(30)

```

From the Publishers . . .

. . . continued from page 4

1 MIP chips are appearing, and DEC is very quiet.

The current delays in delivering computers are certainly as painful to DEC as to the users. Fortunately DEC is a large enough company to withstand some failures. While the UDA50 is not a failure, the delay could have bankrupted a smaller company and it won't win DEC any friends for a while. Major projects that involve new ideas should be born with less trouble than this.

I first used a 36-bit DEC computer, and I am saddened by the announcement that there will be no more 36-biters from DEC. What is particularly upsetting is the fact that this was caused by the failure of the "Jupiter" team to design a computer that would work. The word at the DECUS symposium was that they built the prototype, turned it on and it wouldn't run. The whole concept didn't work due to improper or inadequate simulation in the design stage. A sad end to a great computer line.

IBM expects to build and sell more than 500,000 personal computers this year. This is the year DEC INTRODUCED its personal computer line. The DEC personal computers are well conceived and well built; my liking for them is well known. However, I must point out that there are several holes in DEC's offerings. Where is the Winchester disk for the Rainbow, and why is the Rainbow so expensive? Why will it be the end of the summer before a graphics option is available? Questions and more questions. In personal computers you can't wait five years to improve the product. DEC will have to be nimble if they are to succeed in this market.

DEC is number two in computers, but number three is gaining. I have always liked what the late pitcher Satchel Paige used to say: "don't look back, someone might be gaining on you." It's time for DEC to look ahead and start acting like the professionals they are.

RSTS SOFTWARE



RESOURCE MANAGEMENT and CHARGEBACK SYSTEM

DP Managers use ARSAP for:

- User and Project Accounting
- Monitoring Usage and Trends
- Controlling Performance
- Billing for Services

Also Available for VAX and RSX Systems



P.O. Box 188
Riverdale, MD 20737 (301) 864-3700
VAX, RSX, and RSTS are trademarks of the Digital Equipment Corp.

NEW—comprehensive guide to RSTS software.

THE SIERRA DIRECTORY: RSTS SOFTWARE

The latest, most complete list of RSTS software. Hundreds of software systems and software vendors *exclusively* for the RSTS Operating System. Product descriptions, prices, technical specifications, vendor support provisions, vendor addresses and phone numbers. Cross-indexed by software function and vendor.

- Financial modeling
- Accounting
- Statistics
- Electronic mail
- Word processing
- Languages and compilers
- Communications
- System utilities
- Programmers tools
- Database systems
- Report writers
- Games—and much more

Order your copy today for prompt delivery

Sierra Directory, 3304 Springhill Road, Lafayette, CA 94549
(415) 284-1610

Payment is enclosed for ____ copies of the Sierra Directory of RSTS Software at US\$29.00 each (US\$31.00 outside N. America), including handling.

NAME _____

ADDRESS _____

CITY/STATE/ZIP _____

COUNTRY _____

RSTS is a trademark of Digital Equipment Corp.

CIRCLE 192 ON READER CARD

Operate Your PDP-11 at Peak Efficiency

With SRF,

The PDP-11 Performance Monitor

- Improve System Response
- Reduce System Bottlenecks
- Eliminate Waste of Resources
- Optimize Hardware and Software

SRF for Peak Performance



P.O. BOX 188
RIVERDALE, MD 20737
(301) 864-3700

PDP-11 is a trademark of Digital Equipment Corporation

CIRCLE 201 ON READER CARD

```

30120 !
! Actual SYS call for monitor table, part 1
! Read maximum number of disk units table address
! Read disk free space table address
!
CHANGE SYS(CHR$(6%) + CHR$(-3%)) TO SYS.3%
\ SYS.3%(1%) = SYS.3%(1%) + SWAP$(SYS.3%(1% + 1%))
FOR I% = 5% TO 29% STEP 2%
\ DEVCNT% = SYS.3%(5%)
\ SATCTL% = SYS.3%(21%) !least significant word
\ SATCTM% = SYS.3%(25%) !most significant word

30130 !
! Actual SYS call for monitor table, part 2
! Read disk device name table address
! Read number of disk devices in device name table
!
CHANGE SYS(CHR$(6%) + CHR$(-12%)) TO SYS.12%
\ SYS.12%(1%) = SYS.12%(1%) + SWAP$(SYS.12%(1% + 1%))
FOR I% = 5% TO 29% STEP 2%
\ DEVNAM% = SYS.12%(5%)
\ DEVOKB% = SYS.12%(9%)

30200 !
! Open terminal on channel #1%
! Set terminal to VT100 mode
! Print headings on terminal
! Set scrolling region for row 6 through row 21
! Set scroll to smooth
!
KB% = 1%
\ OPEN 'KB:' FOR INPUT AS FILE #KB%
\ PRINT #KB%, CHR$(155%) + "<";
\ PRINT #KB%, CHR$(155%) + "[H" + CHR$(155%) + "[J" +
CHR$(155%) + "#3" + CHR$(155%) + "[7m";
\ PRINT #KB%, "SCAN ISAM FILES FOR REORGANIZING"
\ PRINT #KB%, CHR$(155%) + "#4";
\ PRINT #KB%, "SCAN ISAM FILES FOR REORGANIZING"
\ PRINT #KB%, CHR$(155%) + "[4;H";
\ PRINT #KB%, "FILE" + SPACE$(27%) + "TOTAL" + SPACE$(3%) +
"OVRFLW" + SPACE$(3%) + "OVRFLW" + SPACE$(2%) +
"OVRFLW" + SPACE$(2%) + "NEEDS" + SPACE$(2%) +
"ENOUGH"
\ PRINT #KB%, "LOCATION" + SPACE$(7%) + "DEV:FILENAM.EXT" +
SPACE$(3%) + "RECORDS" + SPACE$(2%) + "ALLOC." +
SPACE$(4%) + "USED" + SPACE$(3%) + "LOADED" +
SPACE$(2%) + "REORG" + SPACE$(2%) + "SPACE"
\ PRINT #KB%, CHR$(155%) + "[6;22r" + CHR$(155%) + "[?4h";
\ PRINT #KB%, CHR$(155%) + "[6;H" + CHR$(155%) + "[0m";

30300 !
! Read disk device count table for valid disk device types
! Get next disk device type if no units connected
!
FOR I% = 0% TO DEVOKB%/2%-1%
\ GOTO 30620 IF PEEK(DEVCTN%+(I%*2%)) < 0%

30310 !
! Read disk device name table when valid disk type found
! Load FREE with disk free space
! Load IN.$ with disk name and unit number
!
FOR J% = 0% TO PEEK(DEVCTN%+(I%*2%))
\ FREE = PEEK(SATCTL% + (J%*2%))
\ FREE = FREE + 65536 IF FREE < 0 !correct if neg.
\ FREE = FREE + (PEEK(SATCTM% + (J%*2%)) * 65536)
\ IN.$ = CHR$(PEEK(DEVNAM%+(I%*2%)) AND 255%) +
CHR$(SWAP$(PEEK(DEVNAM%+(I%*2%)))) +
NUM1$(J%) + ":"

30400 !
! Open MFD on disk
!
ON ERROR GOTO 31000 !check for error
\ OPEN IN.$ + "[1,1]" FOR INPUT AS FILE #2% !open MFD
\ ON ERROR GOTO 0 !turn error trap off
\ DIM #2%, M%(3583,7)
\ MCLU% = M%(31%,0%) !MFD cluster size
\ MPTR% = FNMLNK$(M%(0%,0%)) !get the first MFD link
GOTO 30610 UNLESS MPTR% !next unit if end of links
\ GOSUB 30430 IF M%(MPTR%,4%) AND 64% !if UFD entry
MPTR% = FNMLNK$(M%(MPTR%,0%)) !get next link in MFD
\ GOTO 30410

30430 !
! Get project/programmer number of UFD
! Open UFD if it is not empty
!
P% = NUM1$(SWAP$(M%(MPTR%,1%)) AND 255%) !project
Q% = NUM1$(M%(MPTR%,1%) AND 255%) !programmer
\ UDCN% = M%(MPTR%,7%) !first DCN of UFD
\ RETURN UNLESS UDCN% !quit if no actual UFD
\ OPEN IN.$ + "[ " + P% + ", " + Q% + "]" FOR INPUT AS FILE #3%
\ DIM #3%, U%(3583,7)
\ UCLU% = U%(31%,0%) !UFD cluster size
\ UPTR% = FNULNK$(U%(0%,0%)) !get the first UFD link
GOTO 30600 UNLESS UPTR% !exit if end of links in UFD
\ GOTO 30450 IF U%(UPTR%,4%) AND 64% !skip actual UFD entry
\ GOSUB 30460
UPTR% = FNULNK$(U%(UPTR%,0%)) !get next link in UFD
\ GOTO 30440

30440 !
! Load file name and file extension
! Return if file extension is not for ISAM file
!
FILNAM.$ = RAD$(U%(UPTR%,1%)) + RAD$(U%(UPTR%,2%))

```

VAX/RSTS PROFESSIONAL, August 1983


```

\ EXT.$ = RAD$(U$(UPTR$,3%))      !file extension
\ RETURN IF EXT.$ <> 'ISM'          !not ISAM file
30470 !
! Open ISAM file as READ REGARDLESS
! Read ISAM FCG for file information
!
  INFIL.$ = IN.$ + "[" + P$ + "," + Q$ + "]" +
    FILNAM.$ + "." + EXT.$          !input ISAM file
\ OPEN INFIL.$ FOR INPUT AS FILE #4%, MODE 4096%
\ DIM #4%, F$(65%)
\ O$ = NUM1$(F$(12%))               !allocated overflow
\ R$ = NUM1$(F$(11%))               !used overflow
\ REC = F$(63%)                     !low order total records
\ REC = REC + 65536 IF REC < 0       !correct for negative
\ REC = REC + (F$(64%)*65536)       !high order total records
\ REC$ = NUM1$(REC)
\ PER$ = NUM1$(INT(1000*VAL(R$)/VAL(O$)+0.5)) !percent X 10
30480 !
! Get total block allocation for the ISAM index and all the
! data files
!
  ALLOC = 0                          !initialize allocation
\ FOR A$ = 0% TO 7%
  A = F$(25% + A$)
  A = A + 65536 IF A < 0              !correct if neg.
  ALLOC = ALLOC + A                  !add to total allocation
\ NEXT A$
30500 !
! Load print line and print
!
  FILNAM.EXT$ = CHR$(F$(42%)) + CHR$(SWAP$(F$(42%))) +
    CHR$(F$(43%)) + "." + FILNAM.$ + "." + EXT.$
\ PRT.$ = IN.$ + "[" + P$ + "," + Q$ + "]" +
  SPACES$(8%-LEN(P$)-LEN(Q$)) + FILNAM.EXT$ +
  SPACES$(10%-LEN(REC$)) + REC$ + SPACES$(8%-LEN(O$)) +
  O$ + SPACES$(8%-LEN(R$)) + R$ + SPACES$(7%-LEN(PER$)) +
  LEFT(PER$,LEN(PER$)-1%) + "." + RIGHT(PER$,LEN(PER$)) +
  "% " + SPACE$(2%)
\ PRINT #KB$, PRT.$;
30510 !
! Make decision whether file needs reorg or not
!
  IF VAL(PER$)/10 < 25% THEN REORG% = 0%
  ELSE IF VAL(PER$)/10 < 40% THEN REORG% = 1%
  ELSE REORG% = 2%
30520 !
! Convert REORG% to decision
! Print decision
!
  PRT.$ = "NO" IF REORG% = 0%
  PRT.$ = "MAYBE" IF REORG% = 1%
  PRT.$ = "YES" IF REORG% = 2%
\ PRINT #KB$, PRT.$ + SPACE$(7%-LEN(PRT.$));
30530 !
! If reorg. is MAYBE or YES, check if enough free space
! to reorg. file
! If reorg. is NO, no need to calculate free space so just
! finish the print line and go for the next file
!
  PRINT #KB$ IF REORG% = 0%
  GOTO 30550 IF REORG% = 0%
\ IF FREE < ALLOC * 1.5 THEN SPACE% = 2%
  ELSE IF FREE < ALLOC * 2.0 THEN SPACE% = 1%
  ELSE SPACE% = 0%
30540 !
! Convert SPACE% to decision
! Print decision
!
  PRT.$ = "YES" IF SPACE% = 0%
  PRT.$ = "MAYBE" IF SPACE% = 1%
  PRT.$ = "NO" IF SPACE% = 2%
\ PRINT #KB$, PRT.$
30550 !
! Return to look for next ISAM file
!
  CLOSE #4%                          !close ISAM file
\ GOSUB 30700 IF BAT% = 89%           !build batch file
\ RETURN                              !return to get next file
30600 !
! Close the UFD and return to search MFD for next UFD entry
!
  CLOSE #3%
\ RETURN
30610 !
! Close the MFD and return to search the next disk unit
!
  CLOSE #2%
\ NEXT J$
30620 !
! Search for the next valid disk type
! End job when all disk types are checked
!
  NEXT I$
\ GOTO 32000
30700 !
! Build batch control file
! Be sure to remove all spaces from the file spec.
!
  RETURN IF REORG% = 0%              !no reorg needed
  RETURN IF SPACE% > 0%              !free space NO or MAYBE
\ PRINT #5%, "$RUN $ISMUTL"
\ PRINT #5%, "R"                     !reorganize
\ PRINT #5%, "R" + CVT$(FILNAM.EXT$,2%) !DEV:FILNAM.EXT
\ PRINT #5%, "B"                     !add to last data file
\ PRINT #5%, "E"                     !exit
\ RETURN
31000 !
! Print message that drive unit not found
! Then return to try next unit
!
  GOTO 31010 IF ERR <> 21
\ PRINT #KB$
\ PRINT #KB$, STRING$(22%,42%) + " Disk unit '" + IN.$ +
  "' is not mounted!!! " + STRING$(22%,42%) + CHR$(7%)
\ PRINT #KB$
\ RESUME 30610
31010 !
! Print message that drive unit disabled
! Then return to try next unit
!
  ON ERROR GOTO 0 IF ERR <> 22
\ PRINT #KB$
\ PRINT #KB$, STRING$(22%,42%) + " Disk unit '" + IN.$ +
  "' is locked out!!! " + STRING$(23%,42%) + CHR$(7%)
\ PRINT #KB$
\ RESUME 30610
32000 !
! End of program routine
! Close batch file if it is being built
! Turn off scrolling region and reset scroll to jump
! Print end of search message at bottom of screen
! Turn off all attributes and clear screen
!
  GOTO 32767 IF BAT% <> 89%          !no batch being built
\ PRINT #5%, "$EOJ"
\ CLOSE #5%
32767 PRINT #KB$, CHR$(155%) + "[1;24r" + CHR$(155%) + "[?41"
\ PRINT #KB$, CHR$(155%) + "[23;H" + CHR$(155%) + "[7m";
\ PRINT #KB$, "End of search... Hit <RETURN> to exit " +
  CHR$(7%);
\ INPUT #KB$, Z$
\ PRINT #KB$, CHR$(155%) + "[m" + CHR$(155%) + "[H" +
  CHR$(155%) + "[J";
\ END

```

Word Processing* VAX/VMS, RSTS/E, RSX-11M

*Word-11 by
Data Processing Design, Inc.
181 W. Orangethorp Avenue
Placentia, CA 92670

On Track Systems Provides:

- Sales
- Service
- Installation
- Demonstrations
- Training
- Consulting

*At your
convenience!*

*At your
office!*

On Track Systems. Inc.

P.O. Box 245
Ambler, PA 19002-0245

(215) 542-7008

CASTAT

AN OPTIMIZATION TOOL

By Joe Bigley, FOTOMAT Corp.

Have you ever wondered if you were using caching effectively? How hard FIP is working? Well, I have. The program below is the product of my curiosity. The program was written with the idea of getting caching and FIP statistics, but once I ran it I realized that it would also be nice to include some disk stats. Needless to say I ran out of room on the screen.

This program has been valuable to me in optimizing our systems. By looking at the disk statistics, for instance, you can tell how busy your disks really are. Shifting busy files to a less busy device will optimize disk access. Optimized disk access is one variable that can reduce response time, especially if your system is I/O bound. But enough about system optimization in this article and on to a brief explanation of the program.

The program is written to print the statistics in VT52 compatible mode, but is easily modified to print on any terminal. Since we primarily use VT52s and VT100s, this mode is all that was necessary for our use. The program documentation is pretty good, making it easily modified. It also contains some statistics that were easily gotten but I didn't have room to put them on the screen. I chose the stats that I found most useful to me to print, but I must admit that I change them from time to time to get a different 'view' of the system. You will probably want to do the same thing. I find it interesting to change something, such as the size of XBUFF, and be able to see the effects that it has on the system in real numbers.

This program was written and tested on RSTS/E version 7.1, but as far as I know it will work on version 7.2. We plan to have 7.2 up shortly and I'm sure this program will get a good workout then. I must also confess that I have only gotten disk stats on DR and DB type drives, since they are the only types we have. The program pretty much speaks for itself and after running it for awhile, you will notice some relationships between the different statistics gathered. Well, enough of my babbling and on to the code.

```
1      ! &
      !
      !      TITLE &
      !
      ! &
      !      CASTAT &
      ! &

2      !      PROGRAM      : CASTAT &
      !      VERSION      : 1.0 &
      !      EDIT         : A &
      !      EDIT DATE    : 13-AUG-82 &
```

```
!      AUTHOR      : JOE BIGLEY &
!                  FOTOMAT CORP. &

20    ! &
      !      MODIFICATION HISTORY &
      !
      !      VER/EDIT      EDIT DATE      REASON &
      ! &

50    ! &
      !      DESCRIPTION &
      !
      !      This program will print statistics for caching, FIP and &
      !      disk usage. It will print disk statistics for only four &
      !      units. This limitation is not on the program but on the &
      !      terminal width (80 columns) that the statistics print on. &
      ! &

60    ! &
      !      DISCLAIMER &
      !
      !      This software is provided without charge for non-commercial &
      !      use by FOTOMAT Corporation. The information in this soft- &
      !      ware is subject to change without notice and should not be &
      !      construed as a commitment by FOTOMAT Corporation. While all &
      !      efforts have been made to remove problems arising from the &
      !      running of this program, there can be no warranties nor &
      !      guarantees, expressed or implied, by FOTOMAT or the author &
      !      resulting from the use of this program on any system or &
      !      computer. &

100   ! &
      !      INPUT / OUTPUT &
      !
      !      CHANNEL #      USED FOR &
      ! &

400   ! &
      !      VARIABLE DEFINITIONS &
      !
      !      VARIABLE      USED FOR &
      !
      !      BASE.STAT%     STATISTICS BASE &
      !      DISK.BASE%     DISK STATISTICS BASE &
      !      JSTATS.BASE%   MONITOR STATISTICS BASE &
      !      CACHE.BASE%    CACHE STATISTICS BASE
401   !      EXEC()        1 DIM ARRAY - LAST MONITOR STATS &
      !      CACH()        1 DIM ARRAY - LAST CACHE STATS &
      !      DISKS()        1 DIM ARRAY - LAST DISK STATS &
      !      SYS.3%()       1 DIM ARRAY - MONITOR TABLES PART I &
      !      SYS.12%()      1 DIM ARRAY - MONITOR TABLES PART II &
      !      U%()           2 DIM ARRAY - DISK STATS WORKING VARIABLE &
      !      TOT.ACC()      1 DIM ARRAY - TOTAL DISK ACCESSES &
      !      READS()        1 DIM ARRAY - TOTAL READ ACCESSES &
      !      DSK.MNT%()     1 DIM ARRAY - DISK MOUNTED/DISMOUNTED FLAG &
      !      SWAPO()        1 DIM ARRAY - SWAP ACCESSES &
      !      USERIO()       1 DIM ARRAY - USER ACCESSES &
      !      SATTIO()        1 DIM ARRAY - SAT ACCESSES &
      !      OVERIO()        1 DIM ARRAY - OVERLAY ACCESSES &
      !      DIR.IO()        1 DIM ARRAY - DIRECTORY ACCESSES &
      !      BUFFIO()        1 DIM ARRAY - DEC TAPE BUFFER ACCESSES
402   !      FREES%        FREE SMALL BUFFER BASE &
      !      DEVCNT%        UNIT NUMBER TABLE &
      !      DEVNAM%        DEVICE NAME TABLE &
      !      DEVOKB%        NUMBER OF DISKS * 2 &
      !      JOBCNT%        CURRENT NUMBER OF JOBS &
      !      UNITS%         NUMBER OF DISK UNITS &
      !      MAX.UNITS%     MAXIMUM DISK UNIT NUMBER &
      !      UNTCNT%        DISK STATUS TABLE &
      !      HOME.CLEAR%    CLEARS THE SCREEN &
      !      IDENT.STG%     VERSION NUMBER &
      !      JOB.CNT        PRINTABLE JOB COUNT &
      !      SLEEP.T%       DEFAULT SLEEP TIME &
      !      SLEEP.TIME     INTERVAL (SLEEP TIME) &
      ! &

700   ! &
      !      FUNCTION / SUBROUTINE DESCR. &
      !
      !      NAME          USE &
      ! &
```




Meet Eddie.
He's learning everything
about the business.

READY
FOR
8.0

Everything.

LOCK-11 is a system security and management package for RSTS.

LOCK-11 gives you absolute control of access by keyboard or user-I.D.

LOCK-11 provides an optional MENU environment that keeps non-privileged users where they belong.

LOCK-11 offers the system manager powerful surveillance utilities that actually improve thruput.

LOCK-11 is very well documented, supported and enhanced regularly.

LOCK-11 is available right now. Circle the response number below for a full set of documentation, or call 215-364-2800.

 **LOCK-11**

CIRCLE 12 ON READER CARD

```

!      FNW()      COMPUTES DISK STATS AS REAL TIME &
!      FNX()      COMPUTES CACHE STATS AS REAL TIME &
!      FNY()      RETURNS A PEEK VALUE &
!      FNZ()      COMPUTES MONITOR STATS AS REAL TIME &
!      FNBARD$()  PRINTS CACHE STATS &
!      FNBARD$()  PRINTS DISK STATS &
!      FNBARDW$() PRINTS FIP STATS &
!      FNPENCT$() CALCULATES PERCENTAGES &
!      FNCURSOR$() DIRECT CURSOR FUNCTION (VT52 MODE) &
! &
!      10010      GATHER THE CACHE STAT DATA &
!      10030      GATHER THE MONITOR STAT DATA &
!      10050      GATHER THE DISK STAT DATA &
!      10100      DETERMINE DISK MOUNTED/DISMOUNTED &
! &
800  ! &
!      F U N C T I O N S &
! &
805  DEF FNW(C%) &
\      Q% = (C%*UNITS%+U%)%6% + 1% &
\      Q = FNY(DISK.BASE%+4%*Q%) &
\      FNW = Q - DISK(Q%) &
\      DISK(Q%) = Q &
\FNEND &
! COMPUTE DATA AS REAL TIME SINCE LAST SLEEP TIME INTERVAL &
! FROM DISK STATS TABLE... C% = ACCESS CODE &
! UNITS% = NUMBER OF DISK UNITS &
! U% = UNIT NUMBER &
! I% = REQUEST CODE &
! DISK.BASE% = DISK STATISTICS BASE &
! DISK(S) = STORAGE FOR LAST TIME AROUND &
! FNY() = FUNCTION TO RETURN A PEEK VALUE &
! &
810  DEF FNX(G%) &
\      G = FNY(CACHE.BASE% + G%) &
\      FNX = G - CACH(G%/2%) &
\      CACH(G%/2%) = G &
\FNEND &
! COMPUTE DATA AS REAL TIME SINCE LAST SLEEP TIME INTERVAL &
! FROM CACHE STATS TABLE... G% = OFFSET FROM BASE &
! FNY() = FUNCTION TO RETURN A PEEK VALUE &
! CACHE.BASE% = CACHE STATISTICS BASE &
! CACH() = STORAGE FOR LAST TIME AROUND &
! &
820  DEF FNY(Q1%) &
\      Q1 = PEEK(Q1%) &
\      Q1 = Q1 + 65536. IF Q1 < 0. &
\      FNY = PEEK(Q1% + 2%) * 65536. + Q1 &
\FNEND &
! RETURN A PEEKED VALUE FROM THE JOB OR CACHE TABLE &
! ... Q1 = THE VALUE TO BE PEEKED AT &
! &
825  DEF FNZ(Q%) &
\      Q = FNY(JSTATS.BASE% + Q%) &
\      FNZ = Q - EXEC(Q%/2%) &
\      EXEC(Q%/2%) = Q &
\FNEND &
! COMPUTE DATA AS REAL TIME SINCE LAST SLEEP TIME INTERVAL &
! FROM JOB STATS TABLE... Q% = OFFSET FROM TABLE &
! JSTATS.BASE% = MONITOR STAT BASE &
! EXEC() = DATA FROM LAST TIME AROUND &
! FNY() = FUNCTION TO RETURN A PEEK VALUE &
! &
830  DEF FNBARD$(T$,X,Y%,Z%) &
\      IF Y% = 0% &
\      THEN PRINT T$; TAB(24%); ':'; TAB(25%); &
\      PRINT USING '#####', X; &
\      IF Y% <> 0% &
\      THEN PRINT T$; TAB(24%); ':'; TAB(25%); &
\      PRINT USING '#####', X; &
\      PRINT TAB(34%); &
\      PRINT USING '###', Y%;
832  PRINT IF Z% &
\FNEND &
! PRINT THE CACHING STATS ... T$ = DESCRIPTION OF DATA LINE &
! X = REAL NUMBER FROM CACHE TABLE &
! Y% = CALCULATED PERCENT &
! Z% = CARRIAGE RETURN FLAG &
! &
835  DEF FNBARDW$(S$,V,W%) &
\      PRINT TAB(40%); S$; TAB(59%); ':'; TAB(60%); &
\      PRINT USING '#####', V; &
\      PRINT TAB(69%); &
\      PRINT USING '###', W%
836  FNEND &
! PRINT THE DATA LINE FOR FIP STATS...S$ = DESCRIPTION OF DATA LINE &
! V = REAL NUMBER FROM FIP TABLE &
! W% = CALCULATED PERCENT &
! (V : UPTIME) &
! &
840  DEF FNBARD$(T$,Y%) &
\      PRINT TAB(40%); T$; TAB(59%); ':'; TAB(60%); &
\      FOR U% = 0% TO MAX.UNITS% &
\      X% = SWAPIO(U%) IF Y%=0% &
\      X% = USERIO(U%) IF Y%=1% &
\      X% = SATTIO(U%) IF Y%=2% &
\      X% = OVERIO(U%) IF Y%=3% &
\      X% = DIR.IO(U%) IF Y%=4% &
\      X% = READS(U%) IF Y%=5% &
\      X% = TOT.ACC(U%) IF Y%=6% &
\      PRINT USING '#####', X%; &
\      NEXT U% &
\      PRINT &
\FNEND &
! FOR ALL UNITS &
! PRINT THE DISK STATS ... T$ = DESCRIPTION OF LINE &
! Y% = REQUEST CODE &
! U% = UNIT NUMBER &
! MAX.UNITS% = MAX # OF UNITS &
! &
850  DEF FNPENCT$(Q1,Q2) &
\      Q2 = .01 IF Q2 = 0.0 &
\      FNPENCT% = INT(Q1/Q2 * 100.0) &
\FNEND &
! CALCULATE THE PERCENTAGES...WE WON'T BE CONCERNED ABOUT ROUNDING &
! &
860  DEF FNCURSOR$(X%,Y%) = &
\      CHR$(155%)+Y%+CHR$(X%+31%)+CHR$(Y%+31%) &
! DIRECT CURSOR ADDRESSING FUNCTION &
! &
900  ! &
!      D I M E N S I O N S T A T E M E N T S & &
! &
910  DIM EXEC(70%), CACH(70%), DISKS(144%) &
! EXEC ARRAY FOR DATA STORAGE FOR REAL TIME &
! CACH ARRAY FOR DATA STORAGE FOR REAL TIME &
! DISK ARRAY FOR DATA STORAGE FOR REAL TIME (# OF UNITS*36) &
! &
920  DIM SYS.3%(30%), SYS.12%(30%) &
! SYS ARRAY FOR MONITOR TABLES PART I &
! SYS ARRAY FOR MONITOR TABLES PART II &
! &
930  DIM U(4%,6%), TOT.ACC(4%), READS(4%), DSK.MNT%(4%) &
! U(UNITS,6%) DISK STATISTICS WORKING VARIABLE &
! TOT.ACC(UNITS) TOTAL ACCESSES PER UNIT &
! READS(UNITS) TOTAL READS PER UNIT &
! DSK.MNT%(UNITS) DISK UNIT MOUNTED/NOT MOUNTED FLAG &
! &
940  DIM SWAPIO(4%), USERIO(4%), SATTIO(4%), OVERIO(4%), DIR.IO(4%), &
\      BUFFIO(4%) &
! SWAPIO(UNITS) SWAP ACCESSES PER UNIT &
! USERIO(UNITS) USER ACCESSES PER UNIT &
! SATTIO(UNITS) SAT ACCESSES PER UNIT &
! OVERIO(UNITS) OVERLAY ACCESSES PER UNIT &
! DIR.IO(UNITS) DIRECTORY ACCESSES PER UNIT &
! BUFFIO(UNITS) BUFFERING ACCESSES PER UNIT &
! &
999  ! &
!      M A I N C O D I N G A R E A &
! &
1000  ON ERROR GOTO 19000 &
\      Z$ = SYS(CHR$(6%)+CHR$(7%)) &
! ERROR AND CONTROL C TRAP &
! &
1010  DO% = -1% &
\      DISKS(I%) = 0.0 FOR I% = 0% TO 144% &
\      HOME.CLEAR$ = CHR$(155%)+CHR$(72%)+CHR$(155%)+CHR$(74%) &
! DEFINE COMMON VARIABLES FOR SCREEN &
! &
1020  BASE.STAT% = PEEK(156%) &
\      DISK.BASE% = PEEK(BASE.STAT%) &
\      JSTATS.BASE% = PEEK(BASE.STAT%+2%) &
\      CACHE.BASE% = PEEK(BASE.STAT%+6%) &
\      IF (DISK.BASE% AND JSTATS.BASE%) = 0% &
\      THEN PRINT 'STATISTICS FEATURE NOT AVAILABLE' &
\      GOTO 32767 &
! GET THE MONITOR STATISTICS TABLES BASE. &
! GET THE DISK STATS BASE, JOB/CPU BASE AND CACHE BASE. &
! DISK AND JSTATS ARE 0 IF MONITOR STATS NOT CONFIGURED. &
! ALL MUST BE CONFIGURED FOR THIS PROGRAM TO RUN &
! NOTE: CACHING STATS WILL BE GENERATED EVEN IF STATS ARE NOT &
! CONFIGURED AT SYSGEN TIME...TO GET CACHING STATS WHEN &
! STATS NOT CONFIGURED, JUST REMOVE THE TEST FOR STATS &
! &
1030  CHANGE SYS(CHR$(6%)+CHR$(12%)) TO SYS.12% &
\      SYS.12%(I%) = SYS.12%(I%)+SWAP$(SYS.12%(I%+1%)) &
\      FOR I% = 3% TO 29% STEP 2% &
\      CHANGE SYS(CHR$(6%)+CHR$(3%)) TO SYS.3% &
\      SYS.3%(I%) = SYS.3%(I%)+SWAP$(SYS.3%(I%+1%)) &
\      FOR I% = 5% TO 29% STEP 2% &
\      FREES% = SYS.12%(3%) &
\      DEVCNT% = SYS.3%(5%) &
\      DEVNAM% = SYS.12%(5%) &
\      DEVOKB% = SYS.12%(9%) &
\      JOBCNT% = SYS.12%(13%) &
\      UNITS% = PEEK(DISK.BASE%) AND 255% &
\      MAX.UNITS% = UNITS% - 1% &
\      DISK.BASE% = DISK.BASE%+2% &
! GET MONITOR TABLES, PART II AND I &
! GET THE FREE SMALL BUFFER BASE (FREES) &
! STORE HOW MANY UNITS PER DISK TABLE (DEVNT) &
! STORE DEVICE NAME TABLE (DEVNAM) &
! STORE HOW MANY DEVICES TIMES 2% (DEVOKB) &
! STORE CURRENT NUMBER OF JOBS (JOBCNT) &
! STORE NUMBER OF DISK UNITS (UNITS) &
! STORE HIGHEST INDEX IN DISK TABLE (MAX.UNITS) &
! INCREMENT THE DISK BASE TO THE NEXT ADDRESS IN THE TABLE &
! &
1040  IDENT.STG$ = 'V7.1A' &
! &
1050  PRINT HOME.CLEAR$; &
! &
1060  PRINT IF CCPOS(0%) &
\      PRINT &
\      PRINT 'CACHE STAT '+IDENT.STG$+' '+CVT$(RIGHT(SYS(CHR$(6%)+CHR$(9%)+
\      CHR$(0%)),3%),4%)+' '+DATE$(0%)+' '+TIME$(0%) &
! PRINT THE PROGRAM NAME &
! &
1070  PRINT &
\      SLEEP.T% = 30% &
\      PRINT 'Interval <';SLEEP.T%';> ': &
\      INPUT SLEEP.TIME &
\      SLEEP.TIME = SLEEP.T% UNLESS SLEEP.TIME &
! SLEEP TIME IS NUMBER OF SECONDS BETWEEN STATISTIC GATHERING &
! &
1075  PRINT HOME.CLEAR$; &
\      JOB.COUNT = PEEK(JOBCNT%) AND 255% &
\      GOSUB 10050 ! GATHER THE DISK STATS! &
\      GOSUB 10010 ! GATHER THE CACHE STATS! &
\      GOSUB 10030 ! GATHER THE MONITOR STATS! &
\      PRINT TAB(10%); 'CACHE STAT '+IDENT.STG$+' '+CVT$(RIGHT(SYS(CHR$(6%)+
\      CHR$(9%)+CHR$(0%)),3%),4%)+' '+DATE$(0%)+' '+TIME$(0%) &
\      PRINT FNCURSOR$(3%,40%); 'Jobs on System : ': 0 &
\      PRINT FNCURSOR$(3%,40%); 'Char/Sec In : ': 0 &
\      PRINT FNCURSOR$(4%,40%); 'Char/Sec Out : ': 0 &
\      PRINT FNCURSOR$(4%,3%); 'Interval : ': SLEEP.TIME &
\      PRINT FNCURSOR$(6%,3%); 'First Interval Prints NO Stats...' &
\      SLEEP SLEEP.TIME &
! GET THE INITIAL STATS &

```



```

1080  WHILE DO$ &
      PRINT HOME.CLEAR$; &
      JOB.COUNT = PEEK(JOBCNT$) AND 255$ &
      GOSUB 10050 ! GATHER THE DISK STATS! &
      GOSUB 10010 ! GATHER THE CACHE STATS! &
      GOSUB 10030 ! GATHER THE MONITOR STATS! &
      PRINT TAB(10$); 'CACHE STAT '+IDENT.STG$+' '+CVT$(RIGHT(SYS &
      (CHR$(6$)+CHR$(9$)+CHR$(0$)),3$),4$)+' '+DATE$(0$)+' '+TIME$(0$) &
      PRINT FNCURSORS(3$,3$); 'Jobs on System : ' &
      PRINT USING '#####', JOB.COUNT &
      PRINT FNCURSORS(3$,40$); 'Char/Sec In : ' &
      PRINT USING '#####', KB.CHAR.IN/SLEEP.TIME &
      PRINT FNCURSORS(4$,40$); 'Char/Sec Out : ' &
      PRINT USING '#####', KB.CHAR.OUT/SLEEP.TIME &
      PRINT FNCURSORS(4$,3$); 'Interval : ' &
      PRINT USING '#####', UPSECS, UPTIME &
      PRINT FNCURSORS(6$,15$)+'Caching Stats' &
      PRINT FNCURSORS(7$,26$)+'Number Per Cent' &
      PRINT FNCURSORS(6$,50$)+'FIP Stats' &
      PRINT FNCURSORS(7$,61$)+'Number Per Cent' &
      PER$ = FNPERCENT$(SIN.TOT.HITS,SIN.TOT.READS) &
      TER$ = FNPERCENT$(FIP.NEEDED,UPTIME) &
      F$ = FNBAR$( '# of Possible Hits',SIN.TOT.READS,0$,0$) &
      F$ = FNBAR$( 'FIP Needed',FIP.NEEDED,TER$) &
      TER$ = FNPERCENT$(FIP.DISK.WAIT,UPTIME) &
      F$ = FNBAR$( '# of Total Hits',SIN.TOT.HITS,PER$,0$) &
      F$ = FNBAR$( 'FIP Disk Wait',FIP.DISK.WAIT,TER$) &
      TER$ = FNPERCENT$(FIP.SAT.WAIT,UPTIME) &
      PER$ = FNPERCENT$(SIN.DIR.HITS,SIN.DIR.READS) &
      F$ = FNBAR$( '# of Directory Reads',SIN.DIR.READS,0$,0$) &
      F$ = FNBAR$( 'FIP SAT Wait',FIP.SAT.WAIT,TER$) &
      TER$ = FNPERCENT$(FIP.CODE.WAIT,UPTIME) &
      F$ = FNBAR$( '# of Directory Hits',SIN.DIR.HITS,PER$,0$) &
      F$ = FNBAR$( 'FIP Code Wait',FIP.CODE.WAIT,TER$) &
      PER$ = FNPERCENT$(SIN.DATA.HITS,SIN.DATA.READS) &
      TER$ = FNPERCENT$(FIP.WAITING,UPTIME) &
      F$ = FNBAR$( '# of Data Reads',SIN.DATA.READS,0$,0$) &
      F$ = FNBAR$( 'FIP Waiting Total',FIP.WAITING,TER$) &
      TER$ = FNPERCENT$(FIP.CPU,UPTIME) &
      F$ = FNBAR$( '# of Data Hits',SIN.DATA.HITS,PER$,0$) &
      F$ = FNBAR$( 'FIP CPU Time',FIP.CPU,TER$) &
      PER$ = FNPERCENT$(NOLOOK.READS,TOT.READS) &
      TER$ = FNPERCENT$(FIP.IDLE,UPTIME) &
      F$ = FNBAR$( '# of Reads Not Cached',NOLOOK.READS,PER$,0$) &
      F$ = FNBAR$( 'FIP Idle time',FIP.IDLE,TER$) &
      PER$ = FNPERCENT$(WRITE.HITS,WRITE.CHECKED) &
      F$ = FNBAR$( '# of Write Xfers',WRITE.CHECKED,0$,1$) &
      PRINT FNCURSORS(15$,50$)+'Disk Stats' &
      F$ = FNBAR$( '# of Write Hits',WRITE.HITS,PER$,1$) &
      PRINT FNCURSORS(16$,64$)+'Units (0-3)' &
      PER$ = FNPERCENT$(CACHE.TIME,UPTIME) &
      F$ = FNBAR$( 'Total Cache Time (ticks)',CACHE.TIME,PER$,0$) &
      F$ = FNBAR$( 'Swap Accesses',0$) &
      PER$ = FNPERCENT$(CHFCNT,TOTAL.TAGS) &
      F$ = FNBAR$( '# of Directory Tags',CHFCNT,PER$,0$) &
      F$ = FNBAR$( 'User Accesses',1$) &
      PER$ = FNPERCENT$(CHDCNT,TOTAL.TAGS) &
      F$ = FNBAR$( '# of Data Tags',CHDCNT,PER$,0$) &
      F$ = FNBAR$( 'SAT Accesses',2$) &
      F$ = FNBAR$( '# of Tags in XBUFF',CHENUE,0$,0$) &
      F$ = FNBAR$( 'Overlay Accesses',3$) &
      F$ = FNBAR$( '# of Tags in SMLBUF',CHENUM,0$,0$) &
      F$ = FNBAR$( 'Directory Accesses',4$) &
      PER$ = FNPERCENT$(MCLU.READS,CAS,LAR.XFER.TOTAL) &
      F$ = FNBAR$( 'Large Xfers Cut Up',MCLU.READS,CAS,PER$,0$) &
      F$ = FNBAR$( 'Total Read Accesses',5$) &
      PER$ = FNPERCENT$(MCLU.READS,NOTC,LAR.XFER.TOTAL) &
      F$ = FNBAR$( 'Large Xfers Not Cached',MCLU.READS,NOTC,PER$,0$) &
      F$ = FNBAR$( 'Total Disk Accesses',6$) &
      SLEEP SLEEP.TIME &
      NEXT &
      STOP &
      ! PRINT THE STATS AFTER EVERY INTERVAL IS UP - THIS IS THE MAIN LOOP &

10000  ! &
      ! SUBROUTINES &
      ! &

10010  SIN.TOT.READS = FN$(0$) ! # OF CACHE ATTEMPTS ! &
      \SIN.TOT.HITS = FN$(6$) ! # OF CACHE HITS ! &
      \SIN.DIR.READS = FN$(4$) ! # OF CACHE ATTEMPTS FOR DIRECTORY! &
      \SIN.DIR.HITS = FN$(12$) ! # OF CACHE HITS FOR DIRECTORY INFO! &
      \CHERST = PEEK(CACHE.BASE$+16$) ! # OF CACHER RESTARTS ! &
      \CHENUM = PEEK(CACHE.BASE$+18$) ! # OF MONITOR BUFFERS ! &
      \CHENUE = PEEK(CACHE.BASE$+20$) ! # OF XBUFF ELEMENTS ! &
      \CHFCNT = PEEK(CACHE.BASE$+22$) ! # OF DIRECTORY TAGS ! &
      \CHDCNT = PEEK(CACHE.BASE$+24$) ! # OF DATA TAGS ! &
      \CHEINV = PEEK(CACHE.BASE$+26$) ! # OF INVALID TAGS ! &
      \TOTAL.TAGS = CHFCNT+CHDCNT+CHEINV ! TOTAL # OF TAGS ! &
      \TOTAL.BUFFERS = CHENUM+CHENUE ! TOTAL # OF BUFFERS ! &
      \SIN.DATA.READS = FN$(28$) ! # OF CACHE ATTEMPTS FOR DATA ! &
      \SIN.DATA.HITS = FN$(32$) ! # OF CACHE HITS FOR DATA READS ! &
      \NOLOOK.READS = FN$(36$) ! # OF XFERS NOT CACHED ! &
      \INSTALL.READS = FN$(40$) ! FRACTION OF ABOVE DUE TO INSTALLS ! &
      \MCLU.READS.CAS = FN$(44$) ! LARGE XFERS CUT UP ! &
      \MCLU.READS.NOTC = FN$(48$) ! LARGE XFERS NOT CACHED ! &
      \LAR.XFER.TOTAL = MCLU.READS.CAS+MCLU.READS.NOTC &
      \SEG.READS = FN$(52$) ! SEG READS PRODUCED WHEN CUT UP ! &
      \WRITE.CHECKED = FN$(56$) ! TOTAL WRITE XFERS ! &
      \WRITE.HITS = FN$(60$) ! TOTAL WRITE XFERS CACHE HITS ! &
      \NOLOOK.TOTAL = (WRITE.CHECKED-WRITE.HITS)+NOLOOK.READS+ &
      (SIN.DATA.READS-SIN.DATA.HITS) &
      RETURN &
      ! GET CACHE INFO. &
      ! ...FN$( ) = FUNCTION TO COMPUTE CACHE STATS AS REAL TIME &

10030  TICKS = PEEK(JSTATS.BASE$) ! TICKS PER SECOND ! &
      \UPTIME = FN$(2$) ! UPTIME IN TICKS ! &
      \UPSECS = UPTIME/TICKS &
      \FREE.SMALL.BUFF = PEEK(FREE$+2$) &
      \SYS.CHARGED = FN$(8$) &
      \LOST.TIME = FN$(12$) &
      \SYS.UNCHARGED = FN$(16$) &
      \NULL.TIME = FN$(20$) &
      \USER = UPTIME-NULL.TIME-LOST.TIME-SYS.CHARGED- &
      SYS.UNCHARGED &

```

```

\CHARGED = USER + SYS.CHARGED &
\UNCHARGED = NULL.TIME+LOST.TIME+SYS.UNCHARGED &
\FIP.NEEDED = FN$(24$) &
\FIP.IDLE = FN$(28$) &
\FIP.WAITING = FN$(32$) &
\FIP.CPU = FIP.NEEDED - FIP.WAITING &
\FIP.CODE.WAIT = FN$(36$) &
\FIP.DISK.WAIT = FN$(40$) &
\FIP.SAT.WAIT = FN$(44$) &
\FIP.OTHER.WAIT = FN$(48$) &
\CPU.PRI.0 = FN$(52$) &
\CPU.PRI.1 = FN$(56$) &
\CPU.PRI.2 = FN$(60$) &
\CPU.PRI.3 = FN$(64$) &
\CPU.PRI.4 = FN$(68$) &
\CPU.PRI.5 = FN$(72$) &
\IO.TIME = CPU.PRI.4 + CPU.PRI.5 &
\CACHE.PRI.0 = FN$(76$) &
\CACHE.PRI.1 = FN$(82$) &
\CACHE.PRI.2 = FN$(86$) &
\CACHE.PRI.3 = FN$(90$) &
\CACHE.PRI.4 = FN$(94$) &
\CACHE.PRI.5 = FN$(98$) &
\CACHE.TIME = CACHE.PRI.0+CACHE.PRI.1+CACHE.PRI.2+ &
      CACHE.PRI.3+CACHE.PRI.4+CACHE.PRI.5 &
\KB.CHAR.IN = FN$(110$) &
\KB.CHAR.OUT = FN$(114$) &
\RETURN &
! GET JOBS INFO. &
! ...FN$( ) = FUNCTION TO COMPUTE MONITOR STATS AS REAL TIME &
! CPU.PRI.0 = CPU EXECUTING USER PROGRAM &
! CPU.PRI.1 = CPU EXECUTING THE NULL JOB &
! CPU.PRI.2 = NOT USED BY RSTS &
! CPU.PRI.3 = CPU EXECUTING MONITOR CODE &

```

```

10050  SWAPIO, USERIO, SATTIO, OVERIO, DIR.IO, BUFFIO, TOT.NOC = 0.0 &
      \TOT.READS = 0.0 &
      \GOSUB 10100 ! CHECK TO SEE IF DISKS ARE MOUNTED! &
      \FOR U$ = 0$ TO MAX.UNITS ! UNIT NUMBER LOOP ! &
      \TOT.ACC(U$), READS(U$), SWAPIO(U$), USERIO(U$), SATTIO(U$), &
      \OVERIO(U$), DIR.IO(U$), BUFFIO(U$) = 0.0 &
      \GOTO 10080 IF DSK.MNT$(U$) &
      \FOR I$ = 0$ TO 4$ STEP 2$ &
      \CLEAR BUFFERS &
      \CHECK TO SEE WHICH DISKS ARE MOUNTED (GOSUB 10100) &
      \ACCESS CODE ACCESS TYPE EXPLANATION &
      ! ----- &
      ! 0 SWAPIO SWAP DATA &
      ! 1 USERIO USER DATA &
      ! 2 SATTIO SAT DATA &
      ! 3 OVERIO OVERLAY DATA &
      ! 4 DIR.IO DIRECTORY DATA &
      ! 5 BUFFIO BUFFER DATA &

```

... continued on page 72

AMERICAN MANAGEMENT & INFORMATION SYSTEMS
announces

MCBA software for DEC systems at prices you can afford

All application packages available
at these tremendous discounts:

- CTS-300/TSX PLUS packages 60% off MCBA list price
- CTS-500/RSTS packages 40% off MCBA list price
- VAX/VMS packages 20% off MCBA list price

End users only

Complete DEC systems available
PDP-11/23, 11/23 PLUS, 11/34A, 11/44, VAX 11/730, VAX 11/750

Call us immediately
and receive our special summer bonus

American Management & Information Systems

1531 Ashford Parkway
Houston, TX 77077
Serban Constantin
713-496-7584

CIRCLE 194 ON READER CARD

PRO TALK

... continued from page 6



Send letters to:
PRO TALK
P.O. Box 361
Ft. Washington, PA
19034-0361

circulating our desks.

Robert S. Gregory
Baha'i World Centre
Haifa, Israel

I really appreciate the magazine you put out. When I read your articles on lowering the priority of a job hogging the system (Vol 4, #5), I tried it out on our PDP-11/44. I've enclosed a listing of how I did it.

Thanks again for your great magazine.

Danny Kumamoto, Assistant Manager
Loma Linda University
Riverside, Calif.

tape for 1200 baud (one wire-wrap jumper), switch the TU-58 port to match the console (DIP switches on the M7096), and away you go (slowly, but it is a TU-58, after all...).

Since making the change, we have logged no DD: errors, no matter what the system load was. I have not experimented with any other baud rates, but both the 11/44 and the TU-58 may be set for 600, 2400 or 4800. Running the TU-58 at a slower console speed is possible, but it might not be practical (20 seconds per block at 300 baud).

James Van Bokkelen
Manager, Software Development
Perception Technology Corp.,
Canton, Mass.

I have not renewed my subscription because I do not think it provides me with \$35 worth of useful information. I find maybe 25% of its contents useful but my main gripe is what's not in each issue. As of my last issue, Feb. 83, I had not seen anything on 8.0. Information on 8.0 was presented at the December DECUS, I have a copy of the information presented. Now I understand delays in printing but for a group that is supposed to be so close to RSTS I would expect something. The change to the MFD and disk structure is pretty major and advanced information such as this is what I want.

Daniel Corbishley
Princeton, N.J.

Dear Daniel: V8.0 is reviewed in this issue.

I have been working with PDP-11s and RSTS for the last four years. My previous background is as an Electrical Engineer with 15 years experience in various aspects of hardware design, software engineering, and data communications. I am currently serving on a voluntary basis with the Data Processing Office of the Baha'i

World Centre. Our installation presently includes an 11/70, and 11/34, and several 11/23s, all running under RSTS.

The "RSTS Pro" is received with much enthusiasm by the data processing staff. At least it seems to disappear longer and more often than the myriad other publications

```

TO HALT A JOB HOGGING THE SYSTEM

Step 1 Type ^P ; Enter the Console State.
; System will respond with "CONSOLE<CR>" and ">>>".

Step 2 Type H<CR> ; Halt the CPU.
; The system will display the address of the PC and
; the halt address (i.e. "17777707 102301").

Step 3 Type E 1006<CR> ; Check the current job number the system
; was running.
; The system will display "00001006 000xxx" where xxx
; is the job number * 2 (in octal). If xxx = 0 or
; xxx is the wrong job number then type "C<CR>" and
; goto step 1.

Step 4 Type E 1010<CR> ; We need to locate the job's Job Data
; Block (JDB). The location "1010"
; contains the current JDB.
; The system will display "00001010 xxxxxx"
; where xxxxxx is the address of the current job's JDB.

Step 5 ; Add 34(8) to xxxxxx (= yyyyyy) to get the offset
; to the runburst/priority word in the JDB.

Step 6 Type E yyyyyy<CR> ; Find out the job's runburst/priority.
; The system will display "00yyyyyy zzzzzz"
; where zzzzzz is the runburst/priority word
; of the offending job. Possible zzzzzz's are:
; 003200 = -128 / 6 ! suspended job
; 003370 = -8 / 6 ! normal job
; 003000 = 0 / 6 ! slight boost
; 003010 = +8 / 6 ! a bad case
; The high byte is the runburst and the low byte
; is the priority.

Step 7 Type D yyyyyy ZZZZZ<CR> ; Type in the desired runburst/priority
; word in the format shown in the list
; above. (000600 to suspend the job.)
; System should return to ">>>".

Step 8 Type C ; To exit console emulator and go back
; to RSTS/E KBM.
; At this point the system should be "unhung."

; The following is a sample "run."
^P ; Type Control P.
CONSOLE
>>>H ; Type H<CR>.
17777707 114404
>>>E 1006 ; Type E 1006<CR>.
00001006 000034
>>>E 1010 ; Type E 1010<CR>.
00001010 021000
>>>E 021034 ; Add 34 to 21000 and type E 021034<CR>.
00021034 003000 ; Runburst = 6; priority = 0.
>>>D 21034 003370 ; Change the priority to -8; type D 21034 003370<CR>.
>>>C ; Type C<CR>.
; All done!

```


RSTS/E INTERNALS MANUAL

The RSTS community has been clamoring for years for a book that details the inner workings of RSTS/E. Well, clamor no more. Michael Mayfield of Northwest Digital Software, and M Systems, the publisher of The RSTS Professional and The DEC Professional Magazines, have teamed up to produce the RSTS/E Monitor Internals Manual.

This manual describes the internal workings and data structures of the RSTS/E monitor. It also notes differences in the internal structures between version 7.1 and earlier versions of the monitor. Future updates will include changes for new versions of the monitor.

Information is available for all levels of users:

- Gain a basic understanding of the workings of the monitor for optimizing system performance.
- Information on disk structures allows recovery of data from corrupted disk packs.
- Special uses of runtime systems and resident libraries allow complex applications to be developed without degrading system performance.
- Write your own custom device drivers for that "foreign" device you need to add but thought you couldn't.

CONTENTS:

Chapter 1 describes the structures used by the monitor that are resident on disk. These include the directory structure, disk allocation tables, Save Image Library (SIL) formats, bootstrap formats and bad block mapping.

Chapter 2 describes the tables used within the monitor to control system resources and provide program services. These tables provide job, memory, file and device control, as well as program services such as interjob communication.

Chapter 3 contains information on writing and installing a custom device driver. It describes the entry points and information the driver must provide to the monitor as well as the subroutines and macros the monitor provides for the driver.

Chapter 4 contains information that enhances information already provided by Digital on writing custom resident libraries and runtime systems. It concentrates mainly on non-standard uses of resident libraries and runtime systems to increase system performance and functionality.

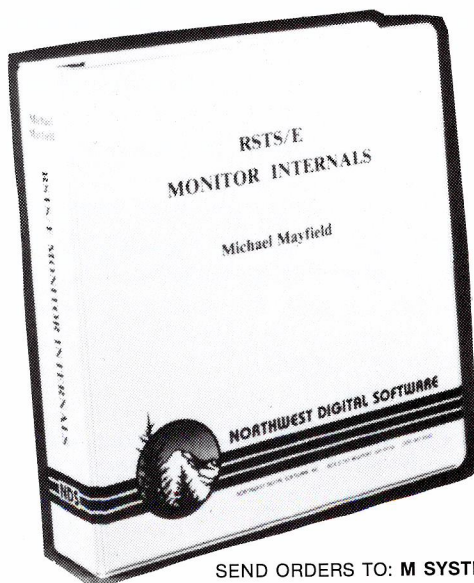
Appendix A provides six quick reference foldout charts:

- The directory structure.
- The monitor tables.
- Fixed memory locations and common data structures.
- Monitor subroutines.
- Device driver entry points.
- Device driver macros.

Appendix B provides examples of the peek sequences required to access most of the monitor tables. It also contains an example program that uses many of the monitor tables to display a job and open files status.

Appendix C provides an example device driver.

Appendix D provides an example runtime system that doubles as a menu system for restricting specified users to a menu of options.



\$95⁰⁰

SEND ORDERS TO: M SYSTEMS, INC., BOX 361, FORT WASHINGTON, PA 19034-0361

```

! &
!
! REQUEST CODE(1%) EXPLANATION &
!
! 0 UNIT READ MISS ACCESSES &
! 1 UNIT READ MISS BLOCKS &
! 2 UNIT WRITE ACCESSES &
! 3 UNIT WRITE BLOCKS &
! 4 UNIT READ HIT ACCESSES (CACHED) &
! 5 UNIT READ HIT BLOCKS (CACHED) &
!

10060 SWAPIO = FNW(0%) \USERIO = FNW(1%) &
      SATTIO = FNW(2%) \OVERIO = FNW(3%) &
      DIR.IO = FNW(4%) \BUFFIO = FNW(5%) &
      U(US,1%) = SWAPIO+USERIO+SATTIO+OVERIO+DIR.IO+BUFFIO &
      SWAPIO(US) = SWAPIO(US) + SWAPIO &
      USERIO(US) = USERIO(US) + USERIO &
      SATTIO(US) = SATTIO(US) + SATTIO &
      OVERIO(US) = OVERIO(US) + OVERIO &
      DIR.IO(US) = DIR.IO(US) + DIR.IO &
      BUFFIO(US) = BUFFIO(US) + BUFFIO &
      NEXT 1% &
      TOT.ACC(US) = TOT.ACC(US)+U(US,0%)+U(US,2%)+U(US,4%) &
      HEADS(US) = HEADS(US)+U(US,0%)+U(US,4%) &
      TOT.READS = HEADS(US)+TOT.READS

10080 NEXT US &
      \RETURN &
      ! FOR EACH DEVICE &
      ! FOR EACH REQUEST CODE &
      ! CALCULATE ACCESS VALUES &
      ! CALCULATE TOTAL ACCESS PER UNIT (TOT.ACC) &
      ! CALCULATE TOTAL READS PER UNIT (HEADS) &
      ! CALCULATE TOTAL READS (TOT.READS) &
      ! NOTE: TO OBTAIN ONLY PHYSICAL ACCESSES MAKE THE REQUEST FOR LOOP &
      ! (1%) FROM 0 TO 2% - TO INCLUDE CACHING 0 TO 4%

10100 D% = 0% &
      \UNTUNT% = SYS.3%(1%) &
      \FOR DEV% = 0% TO DEVOK%+2% STEP 2% &
      ! FOR UNIT% = 0% TO PEEK(DEV%+DEVUNT%) &
      ! DSK.MNT%(D%) = 0% &
      ! DSK.MNT%(D%) = -1% IF PEEK(UNTUNT%) < 0% &
      ! D% = D% + 1% &
      ! UNTUNT% = UNTUNT% + 2% &
      ! NEXT UNIT% &
      \NEXT DEV% &
      \RETURN &
      ! FOR EACH DISK DEVICE. &
      ! FOR EACH UNIT IN THIS DEVICE &
      ! DSK.MNT% = -1 IF THE DISK IS DISMOUNTED &
      ! DSK.MNT% = 0 IF THE DISK IS MOUNTED &

18999 GOTO 32767
19000 ! &
      ! ERROR ROUTINES &
      ! &

19100 IF ERR = 28% &
      THEN F% = SYS(CHR$(6%)+CHR$(-7%)) &
      \ RESUME 32767 &
      ! CONTROL C TRAP &

19900 PRINT ERR; ' @ ' ; ERL &
      \STOP &
      ! PRINT UNEXPECTED ERRORS &

32767 END &

SAMPLE RUN
-----
RUN %CASTAT
CACHE STAT V7.1A RSTS V7.1-11 SAN DIEGO - A 08-Mar-83 07:04
Interval < 30 > ? 20

FIRST SCREEN
-----
      CACHE STAT V7.1A RSTS V7.1-11 SAN DIEGO - A 08-Mar-83 07:04
Jobs on System : 15 Char/Sec In : 0
Interval : 20 Char/Sec Out : 0

First Interval Prints NO Stats...

SECOND SCREEN
-----
      CACHE STAT V7.1A RSTS V7.1-11 SAN DIEGO - A 08-Mar-83 07:04
Jobs on System : 15 Char/Sec In : 1
Interval : 20 1197 Char/Sec Out : 24

      Caching Stats FIP Stats
      Number Per Cent Number Per Cent
# of Possible Hits : 48 FIP Needed : 13 1
# of Total Hits : 48 100 FIP Disk Wait : 3 0
# of Directory Reads : 48 FIP SAT Wait : 3 0
# of Directory Hits : 48 100 FIP Code Wait : 0 0
# of Data Reads : 0 FIP Waiting Total : 6 0
# of Data Hits : 0 FIP CPU Time : 7 0
# of Reads Not Cached : 0 FIP Idle time : 1189 99
# of Write Xfers : 3
# of Write Hits : 2 66
Total Cache Time (ticks): 1 Swap Accesses : 0 0 0
# of Directory Tags : 48 87 User Accesses : 1 0 0
# of Data Tags : 7 12 SAT Accesses : 3 0 0
# of Tags in XBUFF : 55 Overlay Accesses : 3 0 0
# of Tags in SMLBUF : 0 Directory Accesses : 44 0 0
Large Xfers Cut Up : 0 Total Read Accesses: 48 0 0
Large Xfers Not Cached : 0 Total Disk Accesses: 51 0 0

```

TRAPPED BY CONTROL-C

By Philip G. Anthony, Fidelity Bank, Philadelphia, PA

A user on one of our PDP-11/70s encountered an unusual problem a few weeks back: Two copies of the user's BP2 menu program plus an audit trail print program had managed to lock one another up completely. Obviously, multiple record locks in two files had prevented any of the programs from completing the jobs they had set out to do.

No problem, said the user. We'll kill the menu programs, permitting the audit trail job to run. But a series of control-Cs left the menu programs still sitting on their terminals — glaring disdainfully, according to the user.

"I don't understand it," the user complained the following morning. "We aren't trapping for control-C in the menu program. Something must be wrong with the system."

It took a little bit of research to uncover the root of the problem. The user was right — the menu program didn't have a single SYS(CHR\$(6%)+CHR\$(-7%)) to be found. However, the menu program calls a terminal handling subroutine, which calls a password entry and checking routine, which — you guessed it — sets a control-C trap. Control-C traps aren't disabled as the system follows its way back from a subroutine, so the user arrived back in the calling program with the trap still set.

Compounding the mess, the user's error handler (which knew not from ERR = 28%) checked for specific errors, decided anything else was a temporary glitch, and RESUMED — right back into the code that called the subroutine that called the subroutine that set the control-C trap that confused the dickens out of the house that Jack built. And so the loop progressed.

After I had put my finger on the offending code, I started thinking. A programmer writing a subroutine can never be sure of the context in which it will be used months or years later. On the other hand, even the sharpest coder throwing a program together will forget sooner or later the internals of subroutines that are part of his or her favorite .OLB file. So while cleaning up the error handler in this program solved the immediate problem, it did nothing for my future plans, which involve immense amounts of laziness.

Finding out how to disable a control-C trap only took a couple of hours. It seems that the main thing BP2 does to set the trap is to stuff location 24 (octal, of course) with the value of the global symbol \$CCHDX — the entry point to BP2's control-C trapper. If trapping isn't enabled, location 24 contains a zero. My first disabler looked like this:

```

.TITLE CCDIS
.PSECT
CCDIS:: CLR @#24
RTS PC
.END

```


A little thought persuaded me that that wasn't good enough. All I had done was to reinvent $J\% = RCTRLC$. It would take care of a program that used no subroutines and set a control-C trap, then wanted to disable it. But it wouldn't do much for a subroutine; if trapping had already been enabled in the main program and the subroutine enabled it and then disabled it, the main program would be left with no trap in place. And even the sharpest coder . . .

The second cut was a little bit better. It was a goodie that set up a local stack and pushed the current value at location 24 onto it, then zeroed out location 24. The user then issued a control-C trap `SYS()` call. On the way out of the subroutine, he or she called the routine again to restore the previously saved value.

No matter how clever the systems programmer (me), he's going to miss a bet or two in his time. This time I'd forgotten one of the brighter spots in BP2: the `ONERROR GO BACK` statement. By the time control gets to the main program's error handling routine, all the control-C trap states that have been saved so carefully on the local stack are about as useful as vacuum tubes in a Professional 350.

Additionally, I was making the applications programmers work too hard — after all, they're lazy too. Why should they have to issue a `SYS()` call or invoke $J\% = CTRLC$ as well as call a subroutine when the whole `SYS()` call involves only stuffing a value in a location?

The result of all my pondering appears as the accompanying MACRO-11 code. Five routines are defined: One that sets the trap and saves the state, one that clears the trap and saves the state, one that restores the previous state, one for use after control-C has been trapped in the same program where the trap was set, and one for use after control-C has been trapped by an `ONERROR GO BACK`.

Incidentally, this set of routines works only for the CSPCOM that accompanied RSTS V7.x and for BP2 V1.6. For systems that have the new BP2 V2.0 — and, unlike ours, that have gotten the compiler to work — the taskbuild fails with an undefined symbol, `$CCHDX`. I haven't been able to test it, but it looks as if editing the MACRO source to replace `$CCHDX` with `$CCHDL` (which is different from the `$CCHDL` in V1.6) should compile correctly and run perfectly well.

So far, it works. The programmer has complete control within his or her module over whether control-C is trapped. Trapping can be restored to its previous state on exit from a subroutine. And depending on where a control-C came from (which can be determined by examining the variable `ERN$`), the programmer can take appropriate action using the reset and abort subroutines.

But I'm not afraid. In the real world out here — unlike Maynard, Mass. - if it can happen, sooner or later it will. Eventually one of my users, or maybe some inventive RSTS Pro reader, will come up with some wonderful, wild, wooly use of control-C trapping that'll make me throw up my hands in despair. And then rewrite the whole thing.

RSTS

PROFESSIONAL AVAILABLE ON MICROFICHE

DIRECT INQUIRIES TO:

MICRO PHOTO DIVISION

 **BELL & HOWELL**

OLD MANSFIELD ROAD
WOOSTER OH 44691
Contact Christine Ellis
Call toll-free (800) 321-9881
In Ohio, call (216) 264-6666 collect



REORG

Finally one product which handles
all of your reorg and backup needs

MAJOR FEATURES

RSTS/E V8 Compatible

High Speed Disk to Disk Reorganization
Speeds Similar to DEC's IMAGE copy

Disk to disk process can optionally
produce a tape backup

Tape backup utility
Tapes use extended DOS labels

For more information call or write:

NORDATA LTD



4433 27th Ave W
Seattle, WA 98199
(206) 282-1170

CIRCLE 159 ON READER CARD

```
.FNABL LC
.TITLE CCHND
.IDENT /M1.01E/
.SBTTL Documentation
.PAGE
```

```
; Five MACRO-11(TM) routines are defined for use with
; BASIC-Plus-2(TM) or CSPCOM(TM) programs under RSTS/E(TM).
; They are:
```

```
CCSET - Set control-C trap, saving previous state
CCCLR - Disable control-C trap, saving previous
       state
CCRES - Restore previous state
CCRST - Reset control-C trap after detection of a
       control-C (for use when control-C is trapped
       by an ON ERROR GOTC xxxxxx statement within
       a module)
CCABO - Disable control-C trap, purge previous-state
       information (this is useful after control-C
       is trapped by an ONERROR GO BACK statement
       in a subroutine)
```

```
; These routines require no argument(s), and any arguments
; specified by the caller are ignored. Possible errors are:
```

```
32.    "?No buffer space available" - The user is
       nested more than 31. calls deep to CCSET
       or CCCLR. This overflows the local buffer.

248.   "?Illegal subroutine return" - The user
       has called CCRES without having previously
       called CCSET or CCCLR. There is no state
       to restore.
```

```
; This set of routines is available free of charge to readers
; of The RSTS Professional. No guaranty is made by the author,
; by Fidelcor Inc. nor any of its subsidiaries, nor by the
; editors of The RSTS Professional as to its functionality or
; its reliability. It is not supported by any of the above nor
; by Digital Equipment Corporation for use under current or
; future releases of BASIC-Plus-2, CSPCOM, RSTS/E, or the RSX(TM)
; runtime system.
```

```
; MACRO-11, BASIC-Plus-2, CSPCOM, RSTS/E, and RSX are trade
; marks of Digital Equipment Corporation, Inc.
```

```
; Author: Philip G. Anthony
;          Technical Systems
;          Fidelity Bank
;          Broad & Walnut Streets
;          Philadelphia, Pa. 19109
;          (215) 985 8489
;          10-Mar-83
```

```
.SBTTL Data Declaration
.PAGE
.PSECT SUBR.D RW,D,LCL,CON,REL
```

```
BOT:   .BLKW 37      ; 31. words of local stack space
TOP:   .WORD .       ; Local stack pointer
.EVEN
```

```
.SBTTL Code
.PAGE
.PSECT SUBR.I RO,I,LCL,CON,REL
```

```
CCSET:: MOV    #CCHDX,R0      ; BP2/CSPCOM CTRL/C trap handler
CCCHG:  MOV    @#TOP,R4       ; Local stack pointer
CMP     R4,#BOT              ; Is there available space?
BHI     10$                  ; If so, proceed
TRAP    40                   ; "?No buffer space available"
RTS     PC                   ; This emulates ONERROR GO BACK
10$:    MOV    @#24-(R4)      ; Current setting onto stack
CCEXI:  MOV    R0,@#24        ; Set the low-core trap vector
MOV     R4,@#TOP             ; Save the local stack pointer
RTS     PC

CCCLR:: CLR     R0            ; Set for no CTRL/C trapping
BR      CCCHG               ; Go do it

CCRES:: MOV    @#TOP,R4       ; Local stack pointer
CMP     R4,@#TOP             ; Anything to restore?
BLO     10$                  ; If so, proceed
TRAP    370                  ; "?Illegal subroutine return"
RTS     PC
10$:    MOV     (R4)+,R0       ; Get previous state
BR      CCEXI               ; Go restore it

CCRST:: MOV    #CCHDX,@#24    ; BP2/CSPCOM CTRL/C trap handler
; address into low-core vector
RTS     PC

CCABO:: CLR     @#24          ; Disable CTRL/C trapping
MOV     @#TOP,@#TOP          ; Reset the local stack pointer
RTS     PC                  ; Tha-tha-tha-that's all, folks!
```

.END

TICKETS PLEASE: SETUP/RESERVE

By Don Buhman
Blue Mountain Community College
Pendleton, Oregon

I was recently confronted with the interesting problem of reserving theater seats. Our local college/community theater group was having a terrible time handling reservations for a given performance. They didn't know how many or what seats were available on which nights. As you can imagine, this does not make for good public relations.

The first program (SETUP) establishes a Block I/O disk file which is the seat configuration of the theater for each night a play is to run. The arrays which establish the seats can easily be modified for different theaters as long as they fit within a 24 x 80 VT100 screen.

The second program (RESRVE) is the most interesting of the two. I really enjoyed exploring the VT100 escape sequences in writing this program which is why I believe it may be of interest to you. Seats are reserved by moving the cursor to the desired seats with the "arrow" keys and entering #'s. Other commands are self explanatory. Give it a try. I think you'll agree that it's fun.

These programs are the first two of a system of seven which complete the cycle of printing tickets for both reserved and unreserved seats. These other programs are nothing special, therefore, I don't think they would be of interest to you. Also, they (and the working version of RESRVE) incorporate a copyrighted DBMS to store data like names, addresses, seats, etc. in another disk file. These first two programs illustrate where such a system could lead.

```
1      ! SETUP
      ! ESTABLISH DISK FILES FOR SEAT RESERVATIONS

10     EXTEND
15     ON ERROR GO TO 32000
20     OPEN 'THEATR.RES' FOR OUTPUT AS FILE 1%, RECORDSIZE 2048%
25     PUT#1%, BLOCK 97%      ! PRE-EXTEND THE FILE
30     PRINT
\      INPUT 'WHAT SHOW ARE YOU SETTING UP';SHOWNAME$
\      IF LEN(SHOWNAME$)>20% THEN PRINT
\      'TOO MANY CHARACTERS.....20 IS MAXIMUM'
\      GO TO 30
35     INPUT 'HOW MANY PERFORMANCES ';NPER%
\      PRINT
\      BLK%=5%
\      NO.SEAT%=0%
\      DIM SEAT$(24%,80%), PER.DATE$(24%), PER.DAY$(24%),
\      PER.TIME$(24%), SHOW.NAME$(24%), NO.SEATS$(24%)
44     ! BLANK THE INDEX BLOCK
45     FIELD#1%, 2048% AS BLANK$
\      LSET BLANK$=SPACE$(2048%)
\      PUT#1%, BLOCK 1%
49     ! ESTABLISH SEAT MATRIX
50     FOR J%=1% TO 24%
\      FOR K%=1% TO 80%
\      SEAT$(J%,K%)=9%
\      NEXT K%
\      NEXT J%
60     READ A$
\      ROW%=VAL(LEFT(A$,2%))
\      J%=VAL(MID(A$,3%,2%))
\      K%=VAL(MID(A$,5%,2%))
\      LIMIT%=J%+K%-1%
```



```

70   FOR K%=J% TO LIMIT%           &
    SEATS%(ROW%,K%)=8%           &
    NEXT K%                       &
    GO TO 60                      &

80   ! NUMBER THE ROWS
    SEATS%(2%,10%)=7%             &
    SEATS%(2%,71%)=7%             &
    SEATS%(4%,10%)=6%             &
    SEATS%(4%,71%)=6%             &
    SEATS%(6%,10%)=5%             &
    SEATS%(6%,71%)=5%             &
    SEATS%(8%,10%)=4%             &
    SEATS%(8%,71%)=4%             &
    SEATS%(10%,10%)=3%            &
    SEATS%(10%,71%)=3%            &
    SEATS%(12%,10%)=2%            &
    SEATS%(12%,71%)=2%            &
    SEATS%(14%,10%)=1%            &
    SEATS%(14%,71%)=1%            &

99   ! BUILD & STORE A PLAY DATE
100  FOR IX%=1% TO NPER%
103  INPUT 'IS THIS THE SAME SHOW <Y-N>';AN$
    IF LEFT(AN$,1%)='Y' THEN 110
105  INPUT 'NEW SHOW';SHOWNAME$
    IF LEN(SHOWNAME$)>20 THEN PRINT
        'TOO MANY CHARACTERS.....20 IS MAXIMUM'
        PRINT
        GO TO 105
110  PRINT 'DATE OF PERFORMANCE #';IX%;
    INPUT '<MODA>';PER.DATE$(IX%)
    IF LEN(PER.DATE$(IX%))<>4% THEN
        PRINT 'BAD DATE'
        GO TO 110
120  INPUT 'WHAT DAY OF THE WEEK IS THAT';PER.DAY$(IX%)
    INPUT 'WHAT TIME IS THE PERFORMANCE';PER.TIME$(IX%)
    SHOW.NAME$(IX%)=SHOWNAME$
    ! PUT AWAY A PLAY DATE
130  FOR J%=1% TO 24%
        FIELD#1%, (J%*80%)-80% AS POINTER$,
            80% AS DAT.OUT$
        A$=""
        FOR K%=1% TO 80%
            IF SEATS%(J%,K%)=9% THEN A$=A$+' '
            GO TO 138
            IF SEATS%(J%,K%)=8% THEN A$=A$+'O'
            NO.SEATS$(IX%)=NO.SEATS$(IX%)+1%
            GO TO 138
            A$=A$+NUM1$(SEATS%(J%,K%))
        NEXT K%
        LSET DAT.OUT$=A$
    NEXT J%
    GO SUB 200
    PUT#1%, BLOCK BLK%
    BLK%=BLK%+4%
    PRINT
    PRINT
NEXT IX%

150  ! ESTABLISH INDEX BLOCK (1)
160  FOR J%=1% TO 24%
    FIELD#1%, (J%*80%)-80% AS POINTER$, 4% AS PDAT$,
        10% AS PDAY$, 4% AS PTIME$, 3% AS SEAT$,
        3% AS SOLD$, 20% AS SHOW$, 36% AS DUMMY$
170  LSET PDAT$=PER.DATE$(J%)
    LSET PDAY$=PER.DAY$(J%)
    LSET PTIME$=PER.TIME$(J%)
    NO.SEAT$=NUM1$(NO.SEATS$(J%))
    LSET SEAT$=NO.SEAT$
    LSET SOLD$='000'
    LSET SHOW$=SHOW.NAME$(J%)
    LSET DUMMY$=SPACE$(36%)
NEXT J%
180  PUT#1%, BLOCK 1%
    PRINT
    PRINT
    GO TO 32767

200  ! WHICH SEAT CONFIGURATION
    INPUT 'NORMAL SEATING <Y-N>';AN$
    IF LEFT(AN$,1%)='Y' THEN 210 ELSE RETURN
210  FIELD#1%, (2%*80%)-80% AS POINTER$, 80% AS DAT.OUT$
    LSET DAT.OUT$=SPACE$(80%)
220  FIELD#1%, (14%*80%)-80% AS POINTER$, 80% AS DAT.OUT$
    A$=LEFT(DAT.OUT$,32%)+SPACE$(14%)+RIGHT(DAT.OUT$,47%)
    LSET DAT.OUT$=A$
    NO.SEATS$(IX%)=NO.SEATS$(IX%)-72%
    RETURN

9990 DATA ' 21258'
10000 DATA ' 41312'
10010 DATA ' 42824'
10020 DATA ' 45512'
10050 DATA ' 613 5'
10060 DATA ' 618 8'
10070 DATA ' 62922'
10080 DATA ' 654 8'
10090 DATA ' 662 5'
10120 DATA ' 813 6'
10140 DATA ' 83020'

&
10150 DATA ' 853 8'
10160 DATA ' 861 6'
10170 DATA ' 819 8'
10190 DATA '1014 6'
10210 DATA '103118'
10220 DATA '1052 8'
10230 DATA '1060 6'
10240 DATA '1020 8'
10260 DATA '1218 3'
10280 DATA '123216'
10290 DATA '1251 8'
10300 DATA '1259 3'
10310 DATA '1221 8'
10330 DATA '142010'
10340 DATA '143314'
10350 DATA '1450 8'
32000
32010 ! ERROR HANDLING
    IF ERL=60% AND ERR=57% THEN RESUME 80
&
32767 CLOSE 1%
    END
&

1 ! RESERVE
    CCT SEAT RESERVATIONS
&
10 EXTEND
20 OPEN 'THEATR.RES' FOR INPUT AS FILE#1%, RECORDSIZE 2048%
25 ESC.SEQ$=CHR$(155%)+'[ '
30 DIM SEATS%(24%,80%),NO.SEATS%(24%),SOLD.SEATS%(24%)
35 DIM#5%, PER.DATE$(24%)=4%, PER.DAY$(24%)=16%,
    PER.TIME$(24%)=4%, SHOW.NAME$(24%)=32%
    OPEN 'INDEX.DAT' FOR OUTPUT AS FILE 5% !VIRTUAL ARRAY-INDEX
    DIM ROW.CROSS$(14%)
    ROW.CROSS$(2%)=7%
    ROW.CROSS$(4%)=6%
    ROW.CROSS$(6%)=5%
    ROW.CROSS$(8%)=4%
    ROW.CROSS$(10%)=3%
    ROW.CROSS$(12%)=2%
    ROW.CROSS$(14%)=1%
40 ! INITIALIZE THE INDEX
    GET#1%, BLOCK 1%
    FOR J%=1% TO 24%
        FIELD#1%, (J%*80%)-80% AS POINTER$, 4% AS PER.DAT$,
            10% AS PER.DA$, 4% AS PER.TIM$, 3% AS NO.SEAT$, &
            3% AS SOLD$, 20% AS SHOW.NAM$, 36% AS DUMMY$
        IF LEFT(PER.DAT$,1%)=' ' THEN
            NO.PERF%=J%-1%
            GO TO 60 ! END OF PLAYDATES
        PER.DATE$(J%)=PER.DAT$
        PER.DAY$(J%)=PER.DA$
        PER.TIME$(J%)=PER.TIM$
        NO.SEATS$(J%)=VAL(NO.SEAT$)
        SOLD.SEATS$(J%)=VAL(SOLD$)
        SHOW.NAME$(J%)=SHOW.NAM$
    NEXT J%
60 ! DISPLAY INDEX FOR ALL SHOWS
    OPEN 'KB:' AS FILE 2%, MODE 256% ! ESCSEQ OPEN MODE
    PRINT#2% ESC.SEQ$+'2J'+ESC.SEQ$+'H';
    PRINT#2% CHR$(155%)+'#6';
    PRINT#2% 'CURRENT PLAYDATES:'
    FOR I%=1% TO NO.PERF%
        SEATS=NO.SEATS%(I%)
        SOLD=SOLD.SEATS%(I%)
        PER.CENT%=(SOLD/SEATS+.005)*100.
        PRINT#2% USING ' \\\\ \\ 'SPACE$(8%)+'\ ' \\ '
            SPACE$(18%)+'\ ' ### AVAILABLE ### SOLD',
            LEFT(PER.DATE$(I%),2%), RIGHT(PER.DATE$(I%),3%),
            PER.DAY$(I%), SHOW.NAME$(I%),
            NO.SEATS%(I%)-SOLD.SEATS%(I%),PER.CENT%
    NEXT I%
70 ! START WITH WHICH SHOW?
    PRINT#2% ESC.SEQ$+'19;10H';
    PRINT#2% CHR$(155%)+'#6';'SELECT STARTING PLAYDATE'
    PRINT#2% ESC.SEQ$+'2;3H';
    PRINT#2% CHR$(155%)+'#'; ! ENABLE ESCSEQ INPUT
    NORTH%=1%
    SOUTH%=1%
    GET#2%
    FIELD#2%, 3% AS MOVE$
    DIR$=MID(MOVE$,3%,1%)
    IF DIR$='B' THEN SOUTH%=SOUTH%+1% ELSE
    IF DIR$='A' THEN NORTH%=NORTH%+1% ELSE
    IF LEFT(MOVE$,1%)=CHR$(13%) THEN 100 ELSE GO TO 80
    PRINT#2% CHR$(155%)+MOVE$; ! MOVE CURSOR
    GO TO 80
    SHOW%=SOUTH%-NORTH%
    BLK%=1%
    FOR I%=1% TO SHOW%
        BLK%=BLK%+4%
    NEXT I%
110 ! DISPLAY SPECIFIED PLAYDATE SEATS
    PRINT#2% ESC.SEQ$+'2J'+ESC.SEQ$+'H';
    PRINT#2% USING ' \\ 'SPACE$(18%)+'\ ' \\ \\ \\ '
        SHOW.NAME$(SHOW%),LEFT(PER.DATE$(SHOW%),2%),
        RIGHT(PER.DATE$(SHOW%),3%), PER.DAY$(SHOW%)

```

```

120 GET#1%, BLOCK BLK%
\ FOR J%=1% TO 16%
\ FIELD#1%, (J%*80%)-80% AS POINTER$, 80% AS A$
\ PRINT #2% A$
130 NEXT J%

140 PRINT#2% ESC.SEQ$+'16;2H';
\ PRINT#2% USING '### AVAILABLE',NO.SEATS$(SHOW$)-
\ SOLD.SEATS$(SHOW$);
\ PRINT#2% TAB(65%);
\ PRINT#2% USING '### SOLD',SOLD.SEATS$(SHOW$)
\ PRINT#2% ESC.SEQ$+'23;H';
\ PRINT#2% 'COMMANDS: ';
\ PRINT#2% TAB(55%);ESC.SEQ$+'7m';'DUMP'+ESC.SEQ$+'m';
\ PRINT#2% '=DUMP TO PRINTER'
\ PRINT#2% ESC.SEQ$+'7m';'NEW';ESC.SEQ$+'m';
\ PRINT#2% '=CHANGE PLAYDATE '
\ PRINT#2% ESC.SEQ$+'7m';'RES';ESC.SEQ$+'m';
\ PRINT#2% '=SPECIFY ROW & SEATS '
\ PRINT#2% ESC.SEQ$+'7m';'#';ESC.SEQ$+'m';
\ PRINT#2% '=SOLD SEAT '
\ PRINT#2% ESC.SEQ$+'7m';'END';ESC.SEQ$+'m';
\ PRINT#2% '=TERMINATE';
\ PRINT#2% ESC.SEQ$+'16;40H';

150 I COMMAND ENTRY OR MOVE CURSOR TO SEATS
NORTH%, SOUTH%, EAST%, WEST% =0%
160 GET#2%
\ FIELD#2%, 10% AS COM$, 5% AS OVRFL0$
\ UNLESS LEFT(OVRFL0$,1%)=' ' THEN 235 I 10 MAX CHARS.
162 IF LEFT(COM$,3%)='END' THEN 5000 ELSE
IF LEFT(COM$,3%)='RES' THEN 300 ELSE
IF LEFT(COM$,3%)='NEW' THEN 400 ELSE
IF LEFT(COM$,4%)='DUMP' THEN 500 ELSE
IF LEFT(COM$,1%)='#' THEN 200 ELSE
A$=LEFT(COM$,3%) I ESC SEQ - MAYBE?
165 I GARBAGE TRAP
DIR$=MID(A$,3%,1%)
IF DIR$='A' THEN 170 ELSE
IF DIR$='B' THEN 170 ELSE
IF DIR$='C' THEN 170 ELSE
IF DIR$='D' THEN 170 ELSE 235 I START OVER-NOT ESCSEQ
170 I CURSOR MOVE
IF DIR$='A' THEN NORTH%=NORTH%+1% ELSE
IF DIR$='B' THEN SOUTH%=SOUTH%+1% ELSE
IF DIR$='C' THEN EAST%=EAST%+1% ELSE
WEST%=WEST%+1%
180 PRINT#2% CHR$(155%)+A$;
\ GO TO 160

200 I WHERE WE AT ??
ROW%=15%-(NORTH%-SOUTH%)
\ COL%=40%+(EAST%-WEST%)
\ I HOW MANY SEATS
\ NS%=0%
\ FOR I%=1% TO 10%
\ IF MID(COM$,I%,1%)='#' THEN NS%=NS%+1%
210 NEXT I%

215 I STILL WITHIN THE BALLPARK ?
IF ROW%>1% AND ROW%<17% AND COL%>1% AND COL%<80% THEN 220
ELSE 235

220 PRINT#2% ESC.SEQ$+'18;25H';
\ ROW.PRT$=ROW.CROSS$(ROW%)
\ PRINT#2% 'ROW=';ROW.PRT%;' COL=';COL%;'#SEATS';NS%;
\ PRINT#2% ESC.SEQ$+'16;40H';
\ I VALIDATE SEATS & STORE
230 FIELD#1%, (ROW%*80%)-80% AS POINTER$, 80% AS ROW$
\ FOR I%=COL% TO COL%+NS%-1%
\ IF MID(ROW$,I%,1%)='0' THEN 240
235 I MADE AN ERROR - RESTART
PRINT#2% ESC.SEQ$+'21;10H';CHR$(155%)+'#3';
PRINT#2% 'WON'T WORK!';
PRINT#2% ESC.SEQ$+'22;10H';CHR$(155%)+'#4';
PRINT#2% 'WON'T WORK!';
PRINT#2% CHR$(7%);CHR$(7%);
\ SLEEP 2%
\ PUT#1%, BLOCK BLK%
\ GO TO 110
240 ROW$=LEFT(ROW$,I%-1%)+'#'+RIGHT(ROW$,I%+1%)
\ NEXT I%
\ FIELD#1%, (ROW%*80%)-80% AS POINTER$, 80% AS ROW2$
\ LSET ROW2$=ROW$
250 I SEATS ARE VALIDATED - UPDATE COUNT
SOLD.SEATS$(SHOW$)=SOLD.SEATS$(SHOW$)+NS%
\ I NOW GET NAME & ADDRESS
\ FOR I%=1% TO 22%
\ I$=NUM1$(I%)
\ PRINT#2% ESC.SEQ$+I$+'0H';
\ PRINT#2% ESC.SEQ$+'0K';
\ NEXT I%
\ PRINT#2% ESC.SEQ$+'18;2H';'NAME';TAB(16%);
\ PRINT#2% ESC.SEQ$+'7m';SPACE$(25%);ESC.SEQ$+'m';
\ PRINT#2% ESC.SEQ$+'19;2H';'STREET';TAB(16%);
\ PRINT#2% ESC.SEQ$+'7m';SPACE$(20%);ESC.SEQ$+'m';
\ PRINT#2% ESC.SEQ$+'20;2H';'CITY';TAB(16%);
\ PRINT#2% ESC.SEQ$+'7m';SPACE$(20%);ESC.SEQ$+'m';
\ PRINT#2% ESC.SEQ$+'21;2H';'ZIP';TAB(16%);
\ PRINT#2% ESC.SEQ$+'7m';SPACE$(5%);ESC.SEQ$+'m';
270 PRINT#2% ESC.SEQ$+'18;11H';
\ PRINT#2% ESC.SEQ$+'7m';
\ INPUT NAMX$

& \ PRINT#2% ESC.SEQ$+'19;11H';
& \ INPUT STREET$
& \ PRINT#2% ESC.SEQ$+'20;11H';
& \ INPUT CITY$
& \ PRINT#2% ESC.SEQ$+'21;11H';
& \ INPUT ZIP$
& \ PRINT#2% ESC.SEQ$+'m';
& \ PRINT#2% ESC.SEQ$+'18;40H';'ROW';ROW.PRT%;' SEATS: ';
& \ SEATX$=COL%
& \ FOR I%=1% TO NS%
& \ PRINT#2% USING '##',SEATX%;
& \ SEATX$=SEATX%+1%
& \ NEXT I%
& \ GO TO 140

300 I RES COMMAND
PRINT #2% ESC.SEQ$+'16;44H';
PRINT#2% ESC.SEQ$+'1K';
PRINT#2% ESC.SEQ$+'18;1H';
PRINT#2% 'RESERVE SEATS:'
INPUT 'ROW';ROW$
INPUT 'STARTING SEAT#';COL$
INPUT 'HOW MANY SEATS';NS%
IF ROW$='1' THEN ROW$='14' ELSE
IF ROW$='2' THEN ROW$='12' ELSE
IF ROW$='3' THEN ROW$='10' ELSE
IF ROW$='4' THEN ROW$='8' ELSE
IF ROW$='5' THEN ROW$='6' ELSE
IF ROW$='6' THEN ROW$='4' ELSE
IF ROW$='7' THEN ROW$='2' ELSE ROW$='0'
310 FOR I%=1% TO 21%
ROWX$=NUM1$(I%)
PRINT#2% ESC.SEQ$+ROWX$+'18H';
PRINT#2% ESC.SEQ$+'1K';
NEXT I%
ROWX$=NUM1$(VAL(ROW$)+1%)
PRINT#2% ESC.SEQ$+ROWX$+'';'+COL$+'H';
FOR I%=1% TO NS%
PRINT#2% '#';
NEXT I%
ROW$=VAL(ROW$)
COL$=VAL(COL$)
GO TO 215

400 I MOVE TO A NEW PLAYDATE
PUT#1%, BLOCK BLK%
GO SUB 2000
GO TO 60

500 I DUMP STATUS TO LINE PRINTER
OPEN 'LP:' AS FILE 6%
BLK%=1%
FOR I%=1% TO NO.PERF%
SEATS=NO.SEATS$(I%)
SOLD=SOLD.SEATS$(I%)
PER.CENT%=(SOLD/SEATS+.005)*100.
PRINT#6% CHR$(12%)
PRINT#6% USING ' \\\ \\' +SPACE$(8%)+'\ ' +
SPACE$(18%)+'\ ' ### AVAILABLE ### SOLD',
LEFT(PER.DAT$(I%),2%), RIGHT(PER.DAT$(I%),3%),
PER.DAT$(I%), SHOW.NAME$(I%),
NO.SEATS$(I%)-SOLD.SEATS$(I%), PER.CENT%
PRINT#6%
BLK%=BLK%+4%
GET#1%, BLOCK BLK%
FOR J%=1% TO 16%
FIELD#1%, (J%*80%)-80% AS POINTER$, 80% AS A$
PRINT#6% A$
NEXT J%
NEXT I%
CLOSE 6%
GO TO 60

2000 I SUBROUTINE TO STORE UPDATED INDEX
FOR J%=1% TO NO.PERF%
FIELD#1%, (J%*80%)-80% AS POINTER$, 4% AS PER.DAT$,
10% AS PER.DA$, 4% AS PER.TIM$, 3% AS NO.SEAT$,
3% AS SOLD$, 20% AS SHOW.NAM$, 36% AS DUMMY$
LSET PER.DAT$=PER.DAT$(J%)
LSET PER.DA$=PER.DA$(J%)
LSET PER.TIM$=PER.TIM$(J%)
LSET SHOW.NAM$=SHOW.NAM$(J%)
SOLDS$=NUM1$(SOLD.SEATS$(J%))
IF LEN(SOLDS$)<3% THEN SOLDS$='0'+SOLDS$
GO TO 2020
2020 LSET SOLDS$=SOLDS$
LSET NO.SEAT$=NUM1$(NO.SEATS$(J%))
NEXT J%
FOR I%=J%+1% TO 24%
FIELD#1%, (I%*80%)-80% AS POINTER$, 80% AS BLANK$
LSET BLANK$=SPACE$(80%)
NEXT I%
PUT#1%, BLOCK 1%
RETURN

5000 PUT#1%, BLOCK BLK%
GO SUB 2000
PRINT#2% ESC.SEQ$+'2J'+ESC.SEQ$+'H';
GO TO 32767

32767 CLOSE I% FOR I%=1% TO 10%
END

```


ALL BACK ISSUES OF THE RSTS PROFESSIONAL



**Includes a Holiday Special of
2 Library Cases**

GREAT GIFT IDEA - ACT NOW - SUPPLY IS LIMITED

SIMPLY FILL IN COUPON BELOW AND SEND IT ALONG WITH YOUR CHECK TO:
RSTS PROFESSIONAL, P.O. BOX 361, FORT WASHINGTON, PA 19034

☐ Please rush my order for _____ sets of ALL BACK ISSUES OF RSTS PROFESSIONAL
@ \$100.00 per set. TOTAL ENCLOSED \$ _____

☐ I have all the Back Issues but would like the Library Case(s) @ \$7.50 each.
Send me _____ Library Case(s). [Library Case holds 12 magazines.] TOTAL ENCLOSED \$ _____

— DON'T FORGET YOUR FRIENDS AND BUSINESS ASSOCIATES —

NAME _____

ADDRESS _____

CITY/STATE/ZIP _____

PLEASE ENCLOSE YOUR CHECK WITH YOUR ORDER.

EFFECTIVE USE OF THE VT-100 PRINTER PORT

By Lawrence F. Koolkin, Systems Alternatives, Montpelier, VT

At Systems Alternative, Inc. (SAI), all users of our PDP-11/44 RSTS/E system are equipped with VT-100 compatible CRTs and printers, for use as remote workstations. The SAI system has layered menus over RSTS/E so that our users are neither aware nor concerned with any operating system dependencies in their work. In order to fully utilize such an environment, each remote user needs the ability to print files; either on his CRT, to the central site spooled printer, or on his own remote printer (if one exists). To allow these remote users to have complete local printing capabilities, using only one telephone connection, SAI had to develop a complete PRINT function on its menu of choices that would allow these users to have complete printing functionality and flexibility.

The following program listing is extracted from the SAI menu processing software PRINT function, and edited to function as a stand alone program (GOSUB). This routine may also be easily included into your own menu processing, if that is a requirement.

This software makes use of many of the VT-100 ANSI escape codes, and is currently operating successfully on C.Itoh CIT-101 terminals with both letter quality and dot matrix printers attached. The program requests a print file name (in our case, limited to certain required extensions), and first ensures that the file exists and is readable by the user. Once this is ensured, the user has the option of printing the file through his/her printer port, on a spooled printing device, or simply on the CRT screen.

If the printer port option is chosen, then the user is queried as to whether a pause is required between pages; i.e., to insert new sheets of stationery for each page. If the answer is affirmative, then the program will pause and request a <CR> each time a Form Feed character is encountered. If the print file is being spooled, the user is asked whether or not the file should be deleted after printing (i.e., /DE switch), but in any case, the program is currently set up to spool to LP: and always include the /NH switch on filenames.

With this type of printing function available, any CRT with a printer attached through the printer port can become a complete word processing as well as general data processing workstation. Good luck and happy printing!

```
1  | PPRINT.BAS - GOSUB for printing from ASCII files to CRT, Spooled Printer,
   | or VT100 type printer port, currently used with C.Itoh CIT101
   |
   | Line 10722 can be used to limit to only certain extensions
   | current setup only allows *.PRT and *.RPT
   | Stopping between pages requires <FF> between pages
   | Some SYS calls may be PRIV, depending upon your system setup
   | You may need to adjust TTYSET parameters for proper operation
   |
10  | Normal start of program, CTRL/C traps back to this line...
   |
20  | DIM JUNK$(30)
   | For SYS calls
   |
30  | ON ERROR GOTO 32000
   | JUNK$=SYS(CHR$(65)+CHR$(-7))
   | CTRL/C trapping
   |
```

```
100  EXTEND
   | \ ESC$ = CHR$(27)+128%
   | \ NORMAL$ = ESC$+"[0m"
   | \ BOLD$ = ESC$+"[1m"
   | \ BLINK$ = ESC$+"[5m"
   | \ REVERSE$ = ESC$+"[7m"
   | \ UNDERLN$ = ESC$+"[4m"
   | \ CLEAR.SCR$ = ESC$+"[2J"
   | \ HOME$ = ESC$+"[1H"
   | \ SAV.CUR$ = ESC$+"7"
   | \ RES.CUR$ = ESC$+"8"
   | \ VT52$ = ESC$+"[?21"
   | \ VT100$ = ESC$+"<"
   | \ PP.ON$ = ESC$+"1"
   | \ PP.OFF$ = ESC$+"2"
   | \ PP.FLUSH$ = ESC$+"[5z"
   | \ ERASE.EOL$ = ESC$+"[OK"
   | \ WIDTH.80$ = ESC$+"[731"
   | \ WIDTH.132$ = ESC$+"[73h"
   | VT-100 line and screen attributes
   |
200  GOSUB 10700 / GOTO 32767
   | Request print GOSUB, then end
   |
10700 |
   | ***** GOSUB FOR VARIOUS PRINT AND *.PRT FUNCTIONS *****
   |
10702 | PRINT VT100$+CLEAR.SCR$+HOME$;
   | Clear screen
   |
10720 | PRINT \ PRINT BOLD$+REVERSE$+"Filename to Print"+NORMAL$;
   | \ INPUT LINE FTP$
   | \ FTP$ = CVT$(FTP$,-1)
   | \ GOTO 10799 IF LEN(FTP$) = 0%
   | Collect the name, return if <CR>
   |
10722 | JUNK$=INSTR(1$,FTP$,".")
   | \ FTP$ = FTP$+"*.PRT" IF JUNK$=0%
   | \ JUNK$=INSTR(1$,FTP$,".")
   | \ JUNK1$=RIGHT(JUNK$,JUNK$)
   | \ GOTO 10745 IF JUNK1$="*.PRT" OR JUNK1$="*.RPT"
   | \ PRINT "Invalid Filename, must be *.PRT or *.RPT"
   | \ GOTO 10720
   | VERIFY FILE EXTENSION AS *.PRT OR *.RPT
   |
10745 | OPEN FTP$ FOR INPUT AS FILE 10%, MODE 4096#+8192%
   | \ CLOSE 10%
   | Check for existence of file
   |
10747 | JUNK$=SYS(CHR$(65)+CHR$(-10%)+FTP$)
   | \ CHANGE JUNK$ TO JUNK$
   | Do Filename String Scan
   |
10750 | PRINT
   | \ PRINT +REVERSE$+"Printer Port (PP), Spooled Printer (SPL), or CRT (KB)";
   | \ PRINT NORMAL$;
   | \ INPUT LINE FTP.WHERE$
   | \ FTP.WHERE$ = CVT$(FTP.WHERE$,-1)
   | Request location for printed output
   |
10752 | GOTO 10786 IF FTP.WHERE$="SPL"
   | \ GOTO 10754 IF FTP.WHERE$="PP" OR FTP.WHERE$="KB"
   | \ PRINT "Invalid Print Destination"
   | \ GOTO 10750
   | Validate destination
   |
10754 | PG.STOP$="N"
   | \ IF FTP.WHERE$="PP"
   | THEN PRINT REVERSE$+"Stop Between Pages (Y or N)";NORMAL$;
   | \ INPUT LINE PG.STOP$
   | \ PG.STOP$ = LEFT(CVT$(PG.STOP$,-1),1)
   | Request manual stop between pages (<FF> only!!)
   |
10756 | OPEN FTP$ FOR INPUT AS FILE 10%, MODE 4096#+8192%
   | \ IF FTP.WHERE$="KB" THEN PRINT HOME$+CLEAR.SCR$;
   | \ PRINT REVERSE$+"Use 'NO SCROLL' Key to Halt/Continue, CTRL/C to Abort"+NORMAL$;
   | \ PRINT \ PRINT \ SLEEP 2
   | \ GOTO 10764
   | Open file read-only regardless
   |
10760 | SLEEP 9% \ PRINT PP.OFF$;
   | \ PRINT REVERSE$+"Align Paper; RETURN To Continue, CTRL/C to Abort..."+NORMAL$;
   | \ INPUT LINE JUNK$
   | \ PRINT PP.ON$;
   | Request set-up, enable Auxiliary Control Mode
   |
10764 | INPUT LINE #10%, JUNK$
   | \ PG.FF$ = 0%
   | Read a line from the file, trim parity
   |
10765 | JUNK$=INSTR(1$,JUNK$,CHR$(27))
   | \ GOTO 10766 IF JUNK$=0%
   | \ JUNK$=LEFT(JUNK$,JUNK$-1)+ESC$+RIGHT(JUNK$,JUNK$-1)
   | \ GOTO 10765
   | Make sure all ESCs are CHR$(155%), not CHR$(27%)
   |
10766 | JUNK$=INSTR(LEN(JUNK$)-2%,JUNK$,CHR$(13))
   | \ GOTO 10767 IF JUNK$=0%
   | \ JUNK$=LEFT(JUNK$,JUNK$-1)+RIGHT(JUNK$,JUNK$-1)
   | \ JUNK1$ = JUNK$
   | \ GOTO 10766
   | Strip out <CR>s at end of line
   |
```


A PROPER BALANCE BETWEEN MEMORY AND DISK

By
M. Christopher Getting
JBM Group, King of Prussia, PA
and
Philip G. Anthony
Fidelity Bank, Philadelphia, PA

VAX/VMS has come, if not of age, at least into its adolescence with the release of Version 3.2 of the operating system. This may, then, be a good time to analyze the major failing of the system on which DEC has staked its corporate reputation in hopes that future efforts on the part of our favorite computer company may ameliorate a significant problem.

The concept of virtual memory was introduced to the computer world by a certain big blue corporation in its OS operating system. In its day it was a major advance, dealing with the problem of limited program size. Despite this, a great number of Big Blue users remained loyal to the original operating system on which their mainframes were introduced, DOS, even going so far as to spend the relatively large sums necessary for users of their day to expand machine memory.

Today, of course, the situation is reversed. As our DEC customer representatives are fond of telling us, memory is cheap. The development of VAX has demonstrated ad-

mirably that there is nothing inherently difficult about 32-bit addressing of that memory, due in part to the falling prices of chips and the remarkable advances made in that field. In fact, since the 16-bit-register barrier has been breached, there is nothing in the way of developing 64- or even 128-bit register addressing. Disk prices, on the other hand, remain high relative to the costs of CPUs and memory boards, while disk access times though improved have not been reduced by so much as one order of magnitude.

Therefore the entire philosophy on which VAX/VMS is based has been brought into question. What is needed is not a virtual memory machine capable of using expensive disk space as an extension of inexpensive memory, reducing execution time as a byproduct by slowing it to disk access speeds when paging is required. Rather, future development should address the possibility of using inexpensive memory as an extension of the disk, particularly for large data files such as are common in the commercial user environment.

What we propose, then, is the development of an operating system based on the concept of the virtual disk. Under this concept a minimum of disk space is required, perhaps just enough to contain the operating system and a modicum of system programs. One RLO2 would probably suffice for any operational system that does not use RMS, supplemented perhaps by one RX01, if DEC can locate the floppy on which its driver is stored, for developmental systems. The second disk would be desirable only to hold the editor — here we recommend TECO, because of its small size and low system overhead, rather than more costly memory manipulators such as EDT — and a few scratch files. The slow speed of the RX01 would also effectively decrease the priority of development work, reducing user complaints about the programming staff.

All other files would be written to memory when disk space was exhausted, which it tends to be rather frequently in the commercial environment. Swap files (obviously there need be no paging files) could also be held in memory, in case there is need to swap out files from disk for more efficient handling. Minimal disk configurations would be highly cost-effective, since the more that could be forced to memory, the better would be the resultant access times.

Alternatively, all disks could be eliminated. The monitor, CUSPs, and ancillary executable files such as the editor could be loaded via such low-cost devices as paper tape, card reader, floppy, or TU58 cassette. No backup mass storage device would be required if sufficiently powerful battery backup, such as is available on VAX, were provided for memory. Failure of the memory subsystem itself could be guarded against by writing duplicate copies of all files to a secondary computer via DECnet, and restoration could be done at network-transfer speeds after the defective card had been replaced.

The authors have been working on such a system for some months now and have created a prototype that may be interesting to our readers. For the moment, we have restricted ourselves to designing the system for use on PDP-11s under RSTS/E, since we are most familiar with these machines and have no particular interest in learning a totally foreign system such as VAX/VMS until we can find a good reason for doing so. The addition of DCL.RTS, named directories, and an inefficient, inoperable BASIC-Plus-2 V2.0 is not reason enough; these are purely cosmetic changes at best that increase the complexity while slowing the operation and wasting the resources of native RSTS. We further believe that once RSTS/E is given virtual-disk capability, which is significantly more powerful than virtual memory, DEC may reconsider its options and modify VMS until it becomes a lookalike for "RSTS/V" instead.

Of course, PDP-11s are restricted to 16- or 22-bit addressing, so we have not been able to add the memory we would prefer for a full-scale virtual-disk device. However, we have adapted the RSTS/E NL: handler in such a way as to permit it to address APRs 8 through 4,294,967,295 once proper registers are provided, while maintaining its current functionality. The new mnemonic for this device is VD:, standing for Virtual Disk.

When available disk space has been exhausted during the writing of a file — say, SY:[255,255]USRFIL.DAT — the modified monitor changes modes and writes the remainder (in a manner completely transparent to the user) to VD:. At the system manager's discretion, this may be part of user program space, in which case program termination may introduce certain file access problems. Alternatively, a read-write resident cluster library, VD:USRFIL.LIB, may be created dynamically and mapped to the user I/O buffer. Other programs may then map to the same file by locating VD:USRFIL.STB and issuing the appropriate .PLAS directives.

Thus far we have been completely successful in our approach, with no negative feedback whatsoever — at least via GRIPE, which by default writes to VD: — from our users. In fact, we have had virtually no feedback from this source. This may or may not be explained by the transparency of the process. We have, however, experienced some difficulty in accessing our own system files, such as [1,2]PNOCHL.ODD and [0,1]ACQUE.DCT, and therefore are waiting with eager anticipation for DEC to realize the error of its ways and provide us with 1024-bit memory addressing, which would make our access algorithm infinitely more effective.

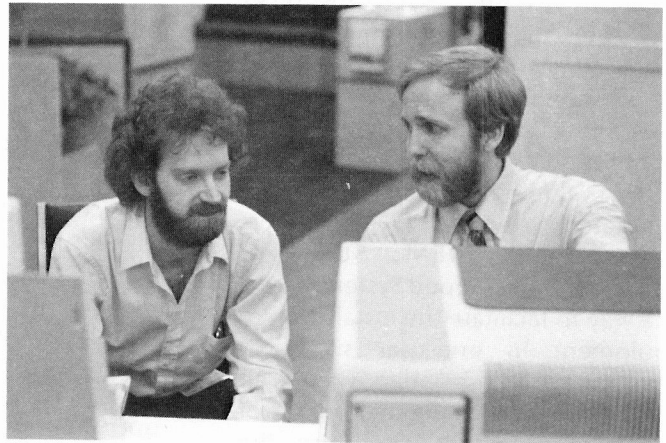
If any of our readers have an interest in modifying their systems to include the VD: driver, they may access our system and request further information via GRIPE. As soon as DEC allows larger memory mapping registers, we will look for the messages and respond accordingly.



SCENES FROM OUR WINDOW



One antique delivers another . . .



Famous authors BOB "Macro Man" MEYER and MIKE MAYFIELD exchange ideas.



"WHATEVER YOU DO, DON'T GET THEM STARTED TALKING ABOUT COMPUTERS!"

NEW PRODUCTS

AUTOMATED SYSTEM MANAGER PACKAGE FOR VAX NEW FROM SI

Strategic Information (SI) announces the release of SystemMaster, the automated system manager package for VAX systems running VMS release 3.1 or higher. Designed for the person responsible for the operation and management of a VAX system, SystemMaster represents the newest in innovations for system management. For end-users, SystemMaster is an alternative to lengthy formalized training; for OEM suppliers, SystemMaster is a classic value-added feature to any integrated system.

As a VAX OEM, Strategic Information developed SystemMaster as a way to facilitate the installation of equipment in end-user sites. No previous VMS knowledge is required to perform most of the major tasks of system management. Therefore, the novice user can readily assume day-to-day management of any VAX/VMS system.

For the more experienced user, SystemMaster serves as a productivity tool by combining many single commands into one screen. The result is more free time to concentrate on a wide range of more productive activities. SystemMaster allows for consistent procedures for operation of one or many VAX/VMS systems.

SystemMaster features:

- **EXTENSIVE ON-LINE HELP**

SystemMaster employs an extensive internal help facility. Each task is fully documented with help screens, providing the information required for task completion.

- **USER INTERFACE**

SystemMaster is completely menu-driven with screen oriented data entry. Most of the system management tasks require the user to do little more than complete one input screen to effect the desired change. The function keys of a

VT100 or compatible terminal are used to advance, backup or get help throughout the user's session.

- **SYSTEM MANAGEMENT TASKS**

The Disk Utilities menu mounts, dismounts, and initializes disk volumes. In addition, the quota file is created and maintained on each volume. The User Authorization menu defines groups and individual accounts on your system, and controls the disk space available to each user. The Print Queue Management menu creates and controls all aspects of print queue operation. The Batch Queue Management menu creates and controls all aspects of batch queue operation. The System Backup menu performs full and incremental backup of the contents of selected disks to tape. The File Restore menu copies specific files from a backup tape to disk. The Terminal Characteristics menu sets or modifies the characteristics of a terminal. The System Shutdown menu schedules shutdowns of your system. The Software Update and Installation menu performs maintenance updates of the VAX/VMS operating system and the installation of layered products.

SystemMaster is a customer installed product. The installation kit comes with software, installation guide, and user's guide. SystemMaster is completely supported by Strategic Information.

SystemMaster is currently available for installation. The perpetual license fee is \$3500 for VAX-11/782 and VAX-11/780 systems; \$2500 for VAX-11/750 systems; and \$1500 for VAX-11/730 systems. Multiple machine discounts and site licenses are available. Educational institutions are eligible for a discounted license fee. Rates for OEM suppliers will be negotiated. SystemMaster is available for a 30-day trial. For further information, please contact: Abigail H. Turner, Strategic Information, Computer Services Group, 80 Blanchard Road, Burlington, MA 01803, (617) 273-5500 ext. 418. SystemMaster will be demonstrated at DEXPOWEST 83, Booth #1424.

VAX and VMS are registered trademarks of Digital Equipment Corporation, Maynard, Massachusetts.

MULTIPLE ACCESS CONTROL SYSTEM FOR VAX VMS USERS

System Industries announced the availability today of a multi-access control system for VAX VMS users. Entitled 9920 SIMACS (S/I Multi-Access Control System), this system enhances the availability of on-line storage products to users with more than one VAX CPU. Specifically, this product provides:

1. Common storage availability for up to eight local CPUs
2. File integrity for both read and write update operations
3. VAX hardware (780, 750, 730) compatibility
4. User transparent VMS software compatibility.

The SIMACS product has been designed to minimize system overhead while allowing multiple processors to access the data base. The data base is comprised of S/I proprietary storage system products and utilizes special file control software and hardware which permits multiple access data paths. This approach can be contrasted with the typically high system overhead approach which employs network hardware and software techniques to move data among CPUs.

Configurations are expandable to include many multiple common access paths and well over 20 gigabytes of storage. In addition, the system can be attached through the VAX processor to the MASSTOR network product that allows storage increments of 55 gigabytes and completely transparent data movement to IBM and other manufacturers mainframes.

The SIMACS product will be available for delivery 30 days after receipt of order starting in the third quarter 1983 and can be installed on any combination of VAX CPUs using VMS software. A special upgrade kit has been developed to allow the more than 2,000 present S/I VAX users to add SIMACS capability to their systems.

System Industries designs, manufactures, markets and services high

performance disk and tape storage systems for use with minicomputers. For more information contact Systems Industries, Inc., 1855 Barber Lane, Milpitas, CA 95035, 408-942-1212.

**RADLEX ANNOUNCES
VAX-11 COBOL, FMS
SOFTWARE PACKAGES**

Radley Business computers, has announced the recent development of application software packages written using VAX-11 COBOL, VAX-11 FMS.

The packages are available in RMS file structure or VAX-11 data base management system formats.

The systems include payroll, accounts payable and general ledger. They are stand-alone systems which can optionally interface with each other. The general ledger system optionally interfaces with the FIN-IPULATOR financial modeling package, also available from Radley. Contact David Ley at Radley Business Computers, 5600 W. Maple, Ste. B212, West Bloomfield, MI 48033, (313) 855-6181.

**VAX DBMS RELEASED
BY SOFTWARE HOUSE**

A relational VAX DBMS designed to satisfy the demands of both casual users and experienced programmers has been released by Software House of Cambridge, Massachusetts.

Experienced programmers can control efficient queries, updates, and output using an integrated structure of commands and system options. Yet novices learn the DBMS fast using a scheme of system defaults, flexible command entry, and on-line user assistance.

The DBMS, called System 1032, employs an inverted file structure for fast retrievals, a built-in block-structured programming language for quick application development, asynchronous I/O for efficient resource utilization, and a host language interface for COBOL, FORTRAN, BASIC, MACRO, C, PL/I and

PASCAL application programmers. System 1032 runs on any model of Digital's VAX computer family using the VMS operating system.

As a relational DBMS, System 1032 stores data in easy-to-understand tables called relations or data sets. Data relationships are established simply by value, not by embedded pointers, using System 1032's dynamic mapping facility. Since programs need not be recompiled when new relationships are established, the data base structure can grow whenever the users' needs expand.

Despite its ease of use and flexibility, System 1032 does not sacrifice fast data access. A proprietary B-tree structure, supporting inverted file structures, makes it possible to index multiple fields.

"This permits impressively fast data retrieval regardless of the request's complexity or data base size," says Patrick Johnson, project director. "Event-driven operation maximizes I/O overlap, while data and indexes are compressed for efficient storage of large data bases."

System 1032 supports seven familiar data types as well as special data types for date/time information and time spans, multi-dimensional arrays, and hierarchical groups.

Its built-in programming language is similar to PASCAL with the added power of data management commands. Application programmers can also access System 1032 from a variety of high-level languages through its host language interface.

"System 1032 is a completely new software product," says Marketing Director Eugene Shklar. "It is not a conversion of an existing program or the outgrowth of the theoretical research project. We've designed System 1032 to be a high-performance product, extremely adaptable to end-user needs."

In 1974 the company introduced the most widely used DBMS for DECsystems-10/20, System 1022. The new system's sophisticated design is "the synthesis of ten years of experience developing and marketing a DEC-compatible DBMS," says Shklar.

A typical System 1032 license for the VAX 11/780 costs \$40,000. Discounts are offered for educational institutions and multiple machines. For more information, contact Paul Perkovic at Software House, 1105 Massachusetts Avenue, Cambridge, Mass. 02138 U.S.A., telephone (617) 661-9440.

**MENU-DRIVEN VAX/VMS
ACCOUNTING PACKAGE**

Computer Information Systems of Mass., Inc. (CIS) has announced REACT, a menu driven VAX/VMS resource accounting package that monitors and bills for system resources and software usage on single VAX/VMS systems or networks connected with DECnet. Billable resources include: connect time, CPU time, page faults, buffered I/O, direct I/O, volumes mounted, pages printed, disk storage, and software license fees. Additionally, REACT maintains statistical information on interactive jobs, subprocesses, detached, batch, and print jobs, login failures, peak working set size, and peak page file usage.

REACT allows for rate assignment to be set per resource and per node on a user by user basis. Extraction of resource usage information can be set to run automatically at a user-specified time of day, or it can be executed interactively. Extensive and flexible reports allow data selection by date range, node, billing account, username, UIC group or UIC member number. Comprehensive invoicing features, a budget subsystem, and an on-line help function are integral parts of REACT.

REACT costs \$2995.00 and is available from CIS at 165 Bay State Drive, Braintree, MA 02184.

**NEW PREPROCESSOR
FROM SOLUTIONS DECK**

The Solutions DECK Transaction preprocessor is now available for the quick and low cost production of 'uniquely' written transaction programs.

It almost eliminates programming for development of interactive applications that use screen displayed forms to guide data entry and processing. Keyboard input is reduced by 95% while producing 90% plus code required.

The process is itself interactive. The user 'paints' the screen by answering questions as to column and row and any special handling required for each data element. Full screen formatting, data mapping and processing logic are produced.

This approach allows for development without programming and retains the flexibility of procedural programming language.

Solutions DECK supports the use of Digital's BASIC+2 and RMS-11 if desired. The introductory price is \$795, making it affordable and ensuring a very quick payback even in low usage situations. For more information write Solutions DECK, P.O. Box 684 Postal Station "A", Fredericton, NB E3B534, Canada.

BASIC+2, RMS-11, DEC and RSTS are trademarks of Digital Equipment. SOLUTIONS DECK is a trademark of SOLUTIONS DECK.

SUMMER CATALOG NOW READY FROM MISCO

The new 80 page full color Summer Catalog of computer supplies and accessories is now available from MISCO Inc., the direct response specialists in computer supplies and accessories.

The catalog contains over 1,000 different products in 86 categories including several new items. All are available for immediate shipment upon written order or by phone call.

"Our catalog has become a standard reference for computer users around the country," says Joseph Popolo, President, MISCO Inc. "We work with all the major manufacturers of computer supplies in the country so we can offer the very latest state-of-the-art supplies and modifications as soon as they become available. Our sales representatives are experts in matching the needs of customers with products that fit their computers and word processors."

All products in the MISCO catalog are fully guaranteed and warranted. They may be returned any time within 30 days for replacement or refund. A copy of the new summer catalog is available by phoning: (800) 631-2227, in New Jersey (201) 946-3500, or by writing to MISCO Inc. at 404 Timber Lane, Marlboro, New Jersey 07746.

NEW HOST ADAPTER FROM EMULEX

A new single-board emulating host adapter has been announced by Emulex Corporation. This new product, designated the UC01, provides full SCSI (Small Computer Standard Interface) interfacing for a variety of SCSI controller/disk drives to the LSI-11 series computers made by Digital Equipment Corporation.

The UC01 Emulating Host Adapter completely supports standard SCSI interface features defined by the ANSI X3T9.2 specification. It provides a parallel interface between the SCSI bus and the CPU; up to seven controllers, usually resident in the peripheral, can be connected to each host adapter. Currently, drives supported include the economical Iomega cartridge disk drive as well as a variety of 5¼ Winchester drives which can be connected to the SCSI bus through SCSI controllers.

The initial model of the UC01 Series, designated the UC01/LX, emulates the RLV11/RLV12 controller with multiple RL01 and RL02 disk units to support drives in the range of 5.2 to 83.2 megabytes. Operation of the UC01 is transparent to applicable DEC diagnostics and all operating systems which support the RL02.

A significant feature of the UC01/LX is its support of 22-bit bus addressing to provide full 4 MByte memory capacity for the new DEC LSI-11/23+.

The UC01/LX host adapter is unique in that it emulates two independent RLV11/RLV12 register sets within a single quad-sized board. Each register set handles up

to four RL01/RL02 type drives to provide for a total capacity of 80 MBytes storage per host adapter. This approach saves the expense of an additional host adapter, requiring considerably less space and power than any other equivalent system. Built-in clock control, plus optional 512 word bootstrap and bus terminators, enable users to eliminate a complete separate board (such as a BDV11) from the system, for added economy.

The UC01 Emulating Host Adapter has a list price of \$1900 per unit. It may be combined with other Emulex controllers in a mix/match OEM and large end-user volume purchase plan to qualify for attractive price discounts.

As with all Emulex products, the UC01 is constructed of pre-tested, pre-aged components and tested at least twice at complete subsystem level. Each unit is environmentally cycled from 40 to 125 degrees F for ninety-six hours prior to shipment. The product is backed by the Emulex standard one year warranty and is supported worldwide by the company's technical services network.

Emulex Corporation, located in Costa Mesa, CA, is a leading manufacturer of disk, tape, and communications controllers for computer equipment made by Digital Equipment Corporation. For further information on the UC01, please contact Mr. Flip Begich, Director of National Sales, at Emulex Corporation, P.O. Box 6725, 3545 Harbor Boulevard, Costa Mesa, CA 92626, Telephone: (800) 854-7112; in California (714) 662-5600.

PRODUCT UPDATES

EXPANDED DIGICAL™ FROM WHY SYSTEMS

WHY Systems, Inc. of Redmond, Washington, has expanded their business spreadsheet DIGICAL™ to

an advanced version DIGICALC™ V 1.2 released this month. DIGICALC™ V 1.2 runs on DEC, VAX/VMS, PDP-11 with RSTS/E. This software package is for professionals involved with budgeting, financial modeling, project management and other complex calculations. The advanced version includes goal seeking procedures, 112 character per cell, three dimensional consolidation, sort capability, variable column widths, on-screen down to zero width and built in training. The system allows multi-user access, external file interface and business quality reports. DIGICALC™ V 1.2 is "user-friendly" engineered, with extensive built in "HELP". In addition, WHY Systems Customer Support Package provides automatic updates, telephone support, and monthly training classes for beginners and advanced users.

The software is now available for \$6000.00 on VAX/VMS and \$4000.00 on RSTS/E installed in over nine countries with over 400 sites and 8000 users. Current customers with Support Agreements will be sent DIGICALC™ V 1.2 free. For more information contact WHY Systems, Inc. at 206-881-2331.

AROUND-THE-CLOCK COMPUTER ROOM PROTECTION TOOL

Innovations In Control, Inc. is pleased to announce the availability of the third generation of its Computer Room Alert series, the DPU 360, which has proved to be an effective protection tool for computer centers in installations across the country.

The DPU is a self contained alarm/control system which interfaces directly to DEC computers and incorporates sensors to monitor all aspects of operating conditions in a computer room. The DPU alerts personnel and if necessary powers down the computer when the operating environment becomes hostile to safe computer operation.

The factors monitored by the DPU are: Temperature, Humidity, Line Voltage, Line Frequency, as

well as Smoke, Water, Blackout, and Brownout.

The front panel displays temperature and humidity and pinpoints the reason for an alarm via latching indicators. This provides information at a glance and eliminates the confusion of multiple alarms.

An optional dialer dials a pre-programmed phone number and transmits a recorded message in case of an alarm.

The DPU is available stock to 4 weeks ARO. For further information contact Innovations In Control, Inc., 409 Reynolds Circle, Suite 24, San Jose, CA 95112, 408-298-7218.

UNIX SUPPORT FROM VOELKER-LEHMAN

Voelker-Lehman Systems, a specialist in VAX-based turnkey systems, announces that it will offer and support UNIX™ because of the software's over-all price/performance/reliability advantages.

"UNIX brings so much more capability to DEC's VAX™ computers, especially those handling multi-user software development projects. It enables us to provide customers with superior total value," said Joseph Voelker, vice president-marketing.

Voelker-Lehman analyzes customer needs, then designs and configures systems around VAX computers, installs and brings them on-line. It trains user personnel, provides continuing hardware and software service and other support to its customers. According to Voelker, VLS is one of the very few VAX/UNIX system houses to offer this complete one-stop capability to the huge DEC market.

"The results are highly efficient systems design, important single vendor responsibility, significant cost savings, and a reliable, timely supply of upgrades and enhancements from DEC and Bell Labs," Voelker stated.

Some of the advantages of UNIX, according to Voelker, are portability from microcomputers to superminis, which protects software investments; its ability to work with

a broad range of hardware configurations; and its compatibility with a variety of software development languages and tools.

Voelker-Lehman was founded in 1981 and maintains offices in Fremont and Sacramento, California. It also provides third party servicing to DEC computers and peripherals of other manufactures. Some of its customers include 3-COM, Plantronics, the University of California, and LucasFilm, Ltd. For more information write Voelker-Lehman Systems, 44160 Warm Springs Blvd., Fremont, CA 94538.

(VAX and DEC are trademarks of Digital Equipment Corp.) (UNIX is a trademark of Bell Laboratories.)

ABLE INTRODUCES PDP-11/23 UPGRADE

Able Computer announces the QNIMAP, a memory expansion module for PDP-11/23 computer systems manufactured by Digital Equipment Corporation. The module, consisting of two dual width boards, increases the addressing capability of 18-bit Q-Bus DMA controllers from 256K bytes up to four megabytes on the 22-bit Q-Bus.

QNIMAP provides a PDP-11/23 with the I/O mapping strength of PDP-11/70, -11/44 or PAX-enhanced -11/24 computers. System performance improvements are significant, and the product is specifically designed to meet the needs of -11/23 systems applications involving timesharing with more than eight users, large applications programs, and large data array manipulation.

The QNIMAP is software compatible with RSTS/E, RSX-11 and UNIX operating systems. It is software transparent to the PDP-11/23 system, and any diagnostic appropriate for a device attached through QNIMAP to the arbitrating system can be used to verify proper installation.

Another feature of QNIMAP is the on-board, LED display, activity indicator. When lighted, each LED reports that a basic function of the board is active. For example, one LED indicates DMA is in progress

from the 18-bit Q-Bus; another reports when QNIMAP is receiving a DMA grant from the CPU.

QNIMAP can easily be installed into any dual slot of an LSI-11/23 or PDP-11/23 system and interfaces to the Q-Bus via the A and B connectors. Supplied with it is a comprehensive user's guide, which provides detailed installation and operation information.

Delivery is 30 days after receipt of order, and the domestic list price is \$1,495 f.o.b. Irvine, California. Contact Able Computer, 1732 Reynolds Ave., Irvine, CA 92714, 714-979-7030.

VAX UPGRADE KIT AVAILABLE FROM EMC

EMC Corporation announced a packaged main memory upgrade kit for the original style DEC VAX 11/750 system which *allows the CPU to be expanded up to 8MBytes.*

Users of the original style VAX 11/750-AA were formerly constrained by the 2MByte limit of their system. Now, users can convert their machines to a new one and add much more memory. The benefit of increased residual value that comes with the new 8MByte expansion capability of the VAX 11/750-CA Model is another plus. Further, users can trade in their old 256KByte memory cards to EMC for credit when they upgrade.

"Many of the early VAX-11/750 Users are at the 2MByte limit and need more memory. This must have been apparent to DEC inasmuch as the VAX-11/750s that are being shipped now are all capable of going to 8MBytes," stated Jack Egan, EMC's DEC Product Manager.

When this simple field conversion is done, the User is usually left with essentially excess *1/4MByte cards which EMC will buy back.* "The number of DEC manufactured cards we will take in-trade depends upon the number of new 1MByte cards the customer needs from us."

The average price of the minimum 2MByte upgrade kit designated the EMC Model VX-2MB-750CA is \$11,500 without trade-in. This includes 2MBytes of

EMC VX-1MB Memory contained on two single hex plug-in cards. Additional 1MByte Memory Cards are \$2,450 as compared to DEC's \$4,900 for the same card. Delivery is from stock to three weeks. Further information may be obtained from EMC Corporation, 385 Elliot St., Newton, MA 02164, 617-244-4740.

NEW TOOLKIT EDITION FOR RSTS/E V 8.0

Software Techniques Incorporated has announced the latest version of its disk management "tool kit," DISKIT. Designed for use with RSTS/E version 8.0, this release will support the new RSTS directory structure, RDS1.

DISKIT, which improves system performance and speeds disk access, has also been enhanced with some new features that make disk optimization even easier than before. Now RDR, the high-speed alternative to REORDR, contains a qualifier which will disable its output log but still print any error messages. In addition, all patches to DISKIT can be installed using the RSTS automated patching facility.

DISKIT, version VO 7.01, includes these utilities:

- DSU-The utility which restructures the information on your disk, making data fast and easy to access.
- DIR-The directory tool that finds files at an incredible rate.
- RDR-Reorders disk directories 30 times faster than ever before possible.
- OPEN-Displays complete job statistics and file activity so you can see what your system is doing.

The new DISKIT is priced at \$1,350.00 with additional CPU

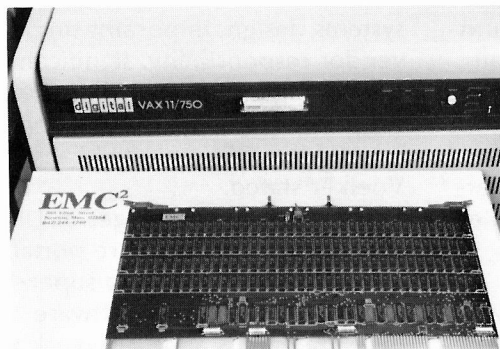
licenses for \$350.00 and an update service for \$295.00. OEM discounts are available. For further information, contact Software Techniques Inc., Software Product Sales, 5242 Katella Ave., Los Alamitos, CA 90720, 714-994-0533.

PLYCOM HAS VAX PAYABLES SYSTEM

Plycom Services, Inc. has an Accounts Payable System designed to work on any Digital Equipment Corporation VAX computer running the VMS operating system.

Features include easy-to-use menus, batch entry of invoices, on-line maintenance of transactions and master files, accounting period orientation and automated period and year-end closing procedures. Query program and numerous informational reports provide excellent control over the disbursements function. Aging reports and cash requirements forecasts make managing cash flow easy. This system is designed to handle a multidivision or multicompany corporate structure. Automatically produces expense and disbursement entries for the general ledger. Optional companion products include Fixed Assets, General Ledger, Accounts Receivable and Financial Reporting packages.

Written in VAX BASIC, the programs are available in source code along with excellent documentation. Complete technical and accounting support is available. Prices start at \$7,500. Further details may be obtained from Plycom by calling (512) 734-4366 or writing to 4243 Piedras East, Suite 150, San Antonio, TX 78228.



Main memory upgrade kit
for VAX available from EMC.

VAX/RT-11 AVAILABLE
FROM CONTEL

The Small Computer Systems Group of Contel Information Systems has announced the availability of Virtual RT-11 (VRT) for users of Digital Equipment Corporation's VAX computers.

VRT is a low-cost, high-efficiency emulation of DEC's most widely installed operating system, RT-11. VRT provides an environment which supports both runtime applications and program development. All RT-11 single-job applications and utilities will run under VRT. An RT-11 license is not required unless DEC RT-11 utility programs are used.

RTFILE, Contel's interactive relational DBMS can be run on the VAX computers under VRT without an RT-11 license. Program development would normally require an RT-11 license because DEC RT-11 utilities (like KED, MACRO, LINK, LIBR, etc.) would probably be used.

VRT includes a Virtual Interchange Program (VIP) which allows users to move files between RT-11 and Files-11 formats. VRT programs may access files in both formats.

A version of VRT for RSX-11M users will be released in the near future.

VRT is an extension of VAMP, the Virtual Access Monitor Program, which allows RT-11 users on STAR-eleven satellites to access files and programs on a VAX/VMS system. STAR-eleven is a high-performance local area network for multiple PDP-11 and/or LSI-11 computers. VRT is being offered at an introductory price of \$1250 until August 1, at which time the list price of \$1750 will apply.

VRT, VAMP, and STAR-eleven were developed by Hammond Software.

More information on all of these products is available from Contel Information Systems, Small Computer Systems Group, 4330 East-West Hwy, Bethesda MD 20814, 301-654-9120. ♥

CLASSIFIED

Send Classified Ads to: VAX/RSTS Classified, P.O. Box 361, Ft. Washington, PA 19034-0361. \$1⁰⁰ per word, first 12 words free with one year's subscription. [Be sure to include a phone number or address in your message.]

DEC BEST VALUES

PRE-OWNED DEC EQUIPMENT

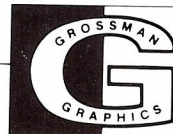
SYSTEMS • CPU's • PERIPHERALS • TERMINALS
OPTIONS • MEMORY • COMPATIBLES

BUYING OR SELLING CALL (305) 771-7600

dataware
incorporated

1500 Northwest 62nd Street
Suite 512
Fort Lauderdale, FL 33309
Telephone (305) 771-7600

Dealers in computer equipment since 1974.



5041
Frankford
Phila., Pa.
19124

215
537-0782

*Everything you always wanted
to know about RSTS but were
afraid to ask.*

*The RSTS Internals Manual
tells all.*

REPRINTS REPRINTS REPRINTS REPRINTS REPRINTS!

All content in this publication is
copyrighted.

All reprints must be purchased from
M Systems, Inc. No other reprints are
authorized.

All reprints shall contain both a
cover and a subscription blank.

Price quotation available on request.

Buy, Sell, Trade: DEC Systems, Parts, Peripherals. Call Paul, Digital Computer Exchange, Inc., 27892 Adobe Court, Hayward, CA 94542. (415) 886-8088.

FOR SALE: PDP11/45, 256KB Core, RH11 controller, two cabinets, currently under maintenance running RSTS/E. Best Offer. Call Richard (617) 232-9111.

INVENTIONS, ideas, technology wanted: Industry presentation/national exposition. 1-800-528-6050, X831.

RSTS

PROFESSIONAL AVAILABLE ON MICROFICHE

DIRECT INQUIRIES TO:

MICRO PHOTO DIVISION

BELL & HOWELL

OLD MANSFIELD ROAD
WOOSTER OH 44691
Contact Christine Ellis
Call toll-free (800) 321-9881
In Ohio, call (216) 264-6666 collect



Accounting Software For Vax

Long time a supplier to DEC RSTS users, PLYCOM now has software for VAX. Easy to learn. Simple to use. Complete support and training. Excellent documentation. In Basic for VAX and VMS. Applications now available include:

- General Ledger
- Financial Reporting
- Accounts Payable
- Fixed Assets Reporting

Plycom services, inc.
P.O. Box 380465
San Antonio, TX 78280
(512) 734-4366

AUTHOR! AUTHOR!

**The VAX/RSTS PROFESSIONAL
wants you to be an author!**

The VAX/RSTS PROFESSIONAL is your magazine. You can make it better by contributing articles, programs or comments directly to us. Our authors are paid honoraria for published works, which because of their hard work, they deserve. We ask you to contribute, send us your manuscripts for possible publication (we prefer machine readable tapes or floppies in PIP, RNO, WORD-11, or ?? format) to VAX/RSTS PROFESSIONAL, P.O. Box 361, Ft. Washington, PA 19034-0361, Attn: Editors. Thank you.

I have left the RSTS PROFESSIONAL for a warmer climate in SW New Mexico, and I will seek temp. contract work in the fall of '83. 10+ years of DP, 5 of them in RSTS, mostly Basic+; please ask publisher Dave Mallory if I'm any good. Rob Frazer, Programmer-at-Large, c/o THE VAX/RSTS PROFESSIONAL, P.O. Box 114, Springhouse, PA 19477.

RSTS RESCUE SQUAD

We salvage all kinds of disasters:

- unreadable disks
- ruined UFDs and MFDs repaired
- immediate response
- telephone DIAL-UP
- on-site
- software tools
- custom recovery
- 90% success to date
- more than 1 GB rescued to date

Brought to you by
On Track Systems, Inc.
and a well known (and read)
RSTS expert.

CALL 24 HOURS
215-542-7008

RSTS/E

DATASAFE

Tape Library System

ONLINE
UTILITY
and
TAPE
ARCHIVE

415-595-5595

Sofprotex

PO BOX 271, BELMONT, CA. 94002

REPAIR Field & Shop

PDP11 - All Models

VAX - 11750 & 11780

Terminals - All Models

Micro Computer Systems -
XEROX, IBM, APPLE, KAYPRO,
OTHERS

Offices In L.A. and SAN DIEGO

Quality work at reasonable rates
contract or per call

CALL TODAY FOR FREE QUOTE

**ACE ASSOCIATED
COMPUTER
ENGINEERS**

A SUBSIDIARY OF COMPUTER SERVICES GROUP, INC.

1250 Union Street, San Diego, CA 92101
(619) 233-0103

Software
Techniques
Incorporated

**Want to work in a shirt
sleeve environment?
Advance the state of the art
in VAX/VMS and RSTS/E?**

If you have experience in analysis
and design using BASIC-PLUS-2
and RMS-11, send resume and
salary history to:

Steve Davis
Software Techniques, Inc.
5242 Katella Avenue
Suite 101
Los Alamitos, CA 90720

VAX AND RSTS/E DEVELOPMENT TIME

NO KILOCORE TICK CHARGES / NO CPU CHARGES

\$7/\$14

RSTS/E

VAX

PER HOUR
CONNECT TIME

**BUDGET
BYTES**
212-
944-9230

by **Omni**computerTM

1430 Broadway, New York, N.Y. 10018

EXCEPTIONAL OPPORTUNITY

Nationally respected, well-established Hos-
pital Pharmacy Package written in RSTS/
Basic Plus is available for outright purchase
or co-marketing arrangement.
Priority given to firm with technical
background.

Send your inquiries to:

HOSPITAL PHARMACY SYSTEM

c/o VAX/RSTS Professional

P.O. Box 361

Fort Washington, PA 19034

DISPLAY CLASS IFIEDS

Classified ads are priced at \$1.00 per word.
Display ads are \$35.00 per column inch, plus
\$1.00 per word. If we set, this includes
border and 2 lines in bolder and/or larger
type size, if desired. — please specify.

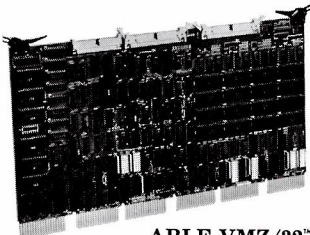
LIST OF ADVERTISERS

ABLE ComputerI.B.Cover
ADOSp.54
Alden Press, Inc.p.35
American Management & Information Systemsp.69
Berrn Research Ltd.p.23
C.D. Smith & Associatesp.52
DataCraftp.39
Data Processing Design, Inc.	..B.Cover
Dataram Corp.p.11
DEC, Software Servicesp.13
Decmationp.16
Dexcompp.59
DEXPOp.29
Emulex Corp.pp.7,21
Enterprise Technologyp.47
Evans, Griffiths & Hart, Inc.	pp.49,61
Fasbe Group, Inc.p.41
Gejac, Inc.pp.62,63,64
Hamilton Rentalsp.33
Interactive Systems, Inc.p.25
Intersil Systemsp.31
JBM Group, Inc.p.35
Logsys Business Systemsp.55
M Systemsp.71
Manus Services Corp.p.42
McHugh, Freeman & Associates	p.43
Nationwide Data Dialogp.67
Nordata Ltd.p.73
North County Computer ServicesI.F.Cover
Northwest Digital Software, Inc.pp.44-45
On Track Systems, Inc.pp.52,65
Oregon Softwarep.27
Personal & Professionalp.37
Reliance Electricp.17
Ross Systemsp.1
RSTS Professionalp.77
Sierra Systems Consultingp.64
Software Techniques, Inc.p.9
Spectra Logic Corp.p.19
Star Plan Data Processingp.52
System Performance House, Inc.pp.5,15
T.F. Hudgins & Associates, Inc.	p.42
Unitronixp.57
Virtual Microsystemsp.51
WHY Systems, Inc.p.2

If you're in the market for communications modules, make the ABLE connection now. And join the thousands who already have.

We are known as the innovators. Most of our products are industry "firsts" which become popular quickly, then settle into a stage of steady long-term acceptance. These four DEC-compatible, communications devices fit the pattern perfectly. They are ABLE originals. They achieved instant success worldwide. They provide top performance. And they are very reliable. Read on to find the one for you.

INCREASED VAX THROUGHPUT.



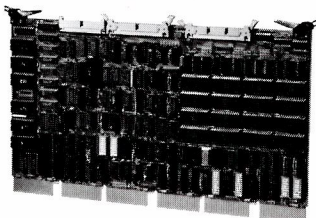
ABLE VMZ/32™
16-line DMF/32 subset

Here's an asynchronous microcontroller with programmable DMA, fully transparent to VAX/VMS as two 8-line DMF 32's and contained on a single board. Priced

below the DZ11-E, it outperforms DZ or DH devices under VMS v.3, has interrupt-driven modem control on every line, and includes an output throttle which lets peripheral devices optimize their own data rate.

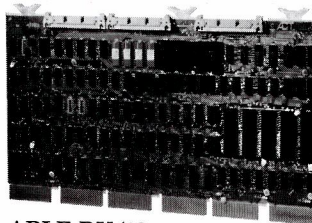
#1 UNIBUS DMA.

Then there's our DH/DM, the original multiplexer which puts 16 lines with modem control on a single board. This popular device meets UNIX VAX system needs for DMA communications requirements, serves UNIBUS systems equally well, and beats them all for MTBF, throughput and



ABLE DH/DM™
16-line combination DH11
& DM11 replacement

price. Other features include on-board diagnostics, modem control on all lines, superior on-board silo depth and variable prom-set. **SYNC/ASYNCH FLEXIBILITY.**



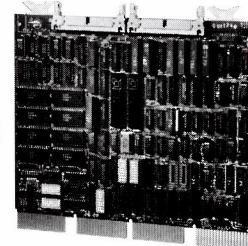
ABLE DV/16
16-line DV11 replacement

A controller for the PDP-11 user, the DV/16 contributes microprocessor-derived flexibility, which permits mixing of sync and async lines in combinations

of 4 or 8 lines with modem control and full system software compatibility. It takes less than half the space of a DV11 and uses word transfer instead of byte DMA to gain a 2 to 1 speed advantage or permit operation in half the bandwidth required for data transfers.

Q-BUS DMA.

The Q/DH is an asynchronous controller which makes DH-class performance possible on PDP-11/23 and LSI-11/23 Q-BUS systems. It connects the standard Q-BUS to as many as 16 async lines with DMA output capabilities and allows optimum Q-BUS utilization. Features include software compatibility with RSTS/E and RSX operating systems, large input silo, modem control on all lines.



ABLE Q/DH™
8 or 16-line DH/DM
for Q-BUS

Write for details on our complete line of DEC-compatible products. Be on the lookout for exciting new ABLE communications products soon to come.

For Immediate, Toll-Free Information, Dial 800 332 ABLE.



CORPORATE OFFICES
ABLE COMPUTER
1732 Reynolds Avenue
Irvine, CA 92714 • (714) 979-7030

NATIONAL OFFICES
Burlington, MA (617) 272-1330
Irvine, CA (714) 979-7030

INTERNATIONAL OFFICES
Canada (Toronto) (416) 270-8086
England (Newbury) (0635) 32125
W. Germany (Munich) 089/463080

DEC, PDP, UNIBUS, Q-BUS, LSI, VAX and VMS are trademarks of Digital Equipment Corporation.



IB Graph for DEC users. The most cost-effective way to get in touch with your data.

Meet the latest Used Software package from Data Processing Design, Inc. It's called IB Graph.™ Nothing less than the most complete, most cost-effective system for business graphics on Digital PDP-11™ and VAX™ processors.

Of course, you could buy more expensive graphics software, but you'd probably end up with more capabilities than you'd actually need—which wouldn't be cost-effective. Or you could buy *less* expensive graphics software, that simply can't do the job. Again, not very cost-effective. IB Graph is the ideal compromise, giving you literally everything you need, and no more.

IB Graph is a multi-user, interactive business graphics system that lets you generate graphs in minutes instead of days. And get quality, full-color bar charts, line charts, or pie charts for immediate publication or presentation. With IB Graph, you can better visualize and analyze your data. You can make quicker decisions and increase your efficiency—not to mention your company's profits. Best of all, IB Graph is easy to learn and simple to use, whether you have no experience in computers at all, or are an experienced programmer.

IB Graph outputs to a variety of graphic CRTs and plotting

devices. And it will continue to be compatible with future hardware announcements by DEC.

If you'd like more information, call or write to us at DPD. We'll be happy to tell you more about IB Graph.

IB Graph. You can buy a more expensive system. Or a less effective one.

You can't buy a better one.

IB Graph



Data Processing Design, Inc.

AUTHORIZED DIGITAL COMPUTER DISTRIBUTOR