

RSTS PROFESSIONAL

Volume 3, Number 4

Second Anniversary Issue

December 1981

\$10⁰⁰/issue, \$25⁰⁰/year



INSIDE:

- A Top-Down Look at Interactive Software, Part 1
- DEC Data Security, The Software Encryption Solution
- TTYSET Optional Patch for VT100 Width Changes
- Software Protection: What Is It Really?
- ATPKED: At-pee-kay Basic-Plus Line Editor
- TIMER.BAS
- BLINK: A Basic Plus Preprocessor
- \$REORDER — Sorting Alphabetically
- Tips & Techniques
- RSTS Defector
- Don't Bubble — Quick Sort
- The VAX-SCENE
Transportable Software on RSTS/E and VAX/VMS Using a Support Library Technique
- Building a Born-Again 40
- Using Sort-11 from Basic Plus-2 on RSTS/E Version 7.0
- Optimizing Space Allocation of Literals in Basic-Plus-2
- VTEDIT.TEC
- The RSTS/E System Manager
- Dungeon Map
- CALLER.BAS
- System Management for RSTS/E
- Benchmark DIBOL vs. BASIC + 2
- Over The I/O Page
- RSTS/E Monitor Internals, Part 3
- The RSTS Extension
- TSXplus An Alternative to RSTS
- More . . .

Financial Decision Support Canned For Easy Use!



Fast Flexible Information

MAPS™ IS FRESH! It's the ideal decision support software product for today's financial manager. With MAPS you get fast, fresh, flexible information handling the moment you want it. Plans, forecasts, models, reports can all be created and manipulated right in your own department. Changes that affect your company can be reflected and evaluated on the spot. And best of all, MAPS is canned for easy use! Call Ross Systems right now. Ask for a demonstration of MAPS. You'll be amazed what a little fresh decision support can do. Currently available on worldwide timesharing, or purchase for DEC's,* RSTS/E and VMS operating systems. Contact Ross Systems for more information today.

*DEC is a trademark of Digital Equipment Corporation.

ross systems

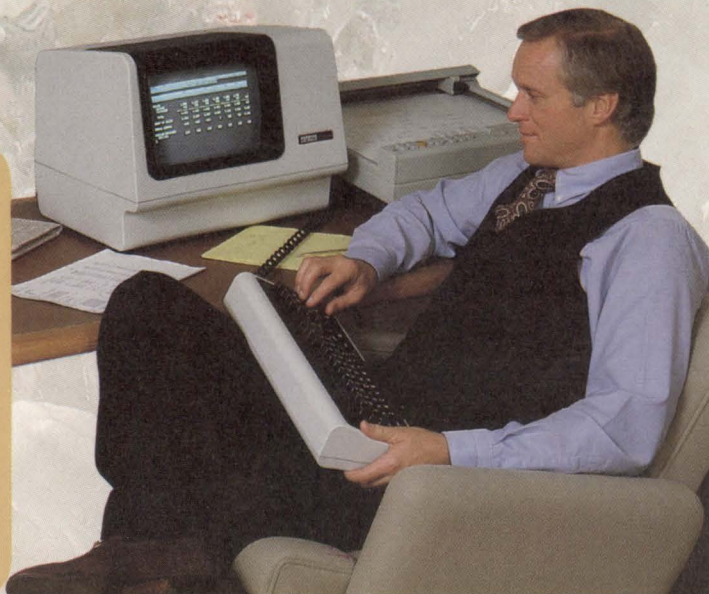
1800 Embarcadero Road
Palo Alto, CA 94303
(415) 856-1100

Regional offices: New York,
Dallas, San Francisco, Los Angeles

"Now that's fresh decision support."

MAPS meets your daily requirements:

Financial reports	100%
Financial planning	100%
Consolidations	100%
Financial modeling	100%
Performance forecasts	100%
Full color graphics	100%



Looking for a quiet, tough printer system? Southern Systems has it. The QT Family.

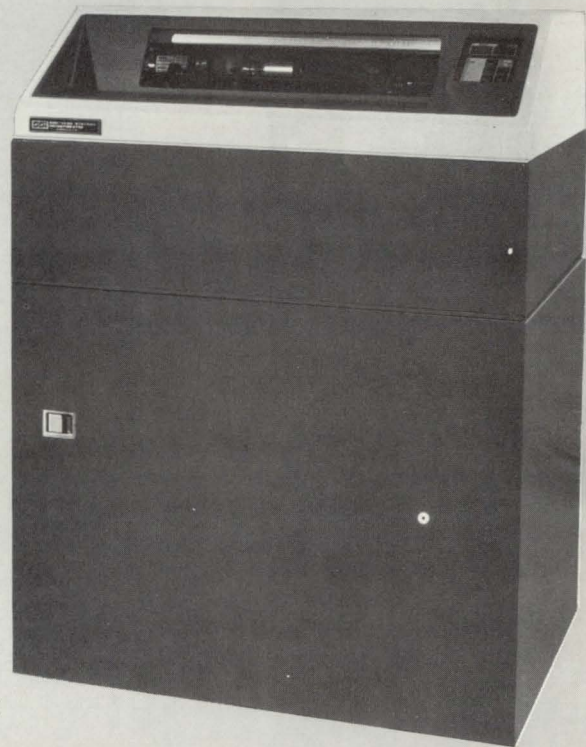


People have been talking about The QT for a long time. They didn't call it The QT, of course. They just talked about the ideal line printer system — one that would be both durable and easy to operate, a low-cost system with great versatility and dependability. Give us a line printer system, they said, with one answer to my most important needs:

YES

- Do the QT printer systems give me a choice of 300, 600, 1000 & 1200 lpm?
- Does the QT family set the standard for quietness?
- Are they built for production DP applications?
- Are the bands heavy duty steel?
Operator changeable?
- Are they easy to maintain?
- Is the print quality top-caliber?
- Is there a diagnostic display? Internal self test?
- Towel ribbon? Forms alignment?
Membrane touch control?
- Does SSI guarantee compatibility with my computer?
- Is service available nationwide?
- Is the QT line printer family affordable and cost-effective?

There's just one other answer you might need:
"Delivery in approximately 30 days."



Tell me about your printers from:
 200-300 lpm 600-900 lpm 1000-plus lpm
 Parallel interfacing Serial interfacing (Synchronous or asynchronous)

My computer system is _____
 Name _____
 Title _____
 Company _____
 Address _____
 City _____ State _____ Zip _____
 Telephone _____ RS



Southern Systems
 The Printer System Problem-Solvers.
 2841 Cypress Creek Road
 Fort Lauderdale, FL 33309
 (305) 979-1000
 (800) 327-5602 • Telex 522135

"we write everything with Ambase"



Don Chick and Greg Jones, Automation, Inc., Omaha, Nebraska, Service Bureau, using the power of AMBASE in an integrated environment with AMCOR's Accounting and Business Control Application products.

"AMBASE training was well presented and made our use of AMBASE easier.

We are using AMBASE to write custom programs and customize modules within other products. It takes less time and we write everything exclusively with AMBASE.

We are able to put out a report in a matter of hours as opposed to days. We couldn't live without it."

AMBASE is a revolutionary state-of-the-art system for application development and data base management. AMBASE is increasing programmer productivity worldwide from 100-900%. In addition to the data management system, AMBASE includes a

report generator, query language, screen formatter and automatic code generator.

If you would like to find out more about AMBASE, just clip and mail the coupon, or give us a call, TOLL FREE at 1-800-626-6268. We will send you free information immediately.

Mail to:
AMCOR Computer Corp.
1900 Plantside Drive
Louisville, Kentucky 40299

Please send more information
on AMBASE DEC RSTS
Software to:

Name _____

Company _____

Address _____

City _____

State _____

Zip _____

Phone _____

amcor computer corp.
a Kaneb company



From the editors...



Mr. Howie Brown Chairman
IRUS
Box 3043
Pawtucket, RI 02861
November 2, 1981

Dear Howie,

CONGRATULATIONS! With lots of hard work, dedication, late nights and plenty of inspiration you had a great ACCESS-11 conference. The RSTS community and all DEC users salute you and your group for their efforts.

We have hoped for some time that there could be a forum for suppliers of services for DEC users to display their products at a dedicated show. INFO, NCC, COMDEX and other shows have their places but somehow the DEC suppliers seem to get lost in the vastness of the expositions. The ACCESS-11 show was big enough to keep people busy, but not too busy to allow lots of one-on-one talking, demonstrations and personalized information.

We were pleased to participate in the seminar part of ACCESS-11 for the second year in a row, and from the extra time we spent after the formal sessions talking about the topics we know that they attendees were rewarded for coming to Hartford.

Just a word about Hartford, Co.: The Civic center was large, friendly, well placed, responsive (our booth had the right telephone for our modem, and electricity to run it and the terminal) and next to the Sheraton hotel. The hotel managed to feed us when required, gave us a good room to sleep in and had the reservations correct; hard to beat.

The RSTS PROFESSIONAL promises to participate in the next ACCESS-11 and hopes that all the vendors and participants concur.

Best wishes and thank you.

Carl B. Marbach Editor, RSTS PROFESSIONAL

I have printed a letter we sent to Howie Brown and his IRUS (Independent RSTS User Society) about their conference and exposition held at the end of October. I hope all of us spend a moment to think about what has transpired. A RSTS show! RSTS people came, RT-11 people came ('we are thinking of moving to RSTS'), RSX people came ('what's a RSTS'), TSX people came, VAX people came and I even met some non-DEC users who came to see what DEC was all about.

I am not sure what the attendance actually was, but we think it was good for all the RSTS vendors. We saw more RSTS users than we will see in one place (except for DECUS) and we had full access to them, and they had ACCESS to our products and services. The non-RSTS vendors, such as RACAL-VADIC (modems) or BASF (magnetic media) saw fewer people than they ever had in a show before. Did they see enough? They will decide by seeing how many sales leads were generated and what kind they were (lookers? buyers?). As a community we want these vendors back, and we want more; here is what we can do:

**SUPPORT THE VENDORS WHO
COME TO YOUR SHOW!
SUPPORT THE VENDORS WHO
ADVERTISE IN YOUR MAGAZINE!**

Your support means: Ask for information from them (sign up at the show or use the reader service card in the magazine). Give them a chance to show you their products and why they think you should buy them, then make your own decision — we want you to buy what is right for you. If these vendors generate enough qualified action from our show and magazine you will benefit. For example, there were three Statistical (TDM's too) multiplexer companies there. It was a great opportunity to find out what each one has to offer.

The Vendors will be here if we show them we want them; only you can do that. There were so many good things at ACCESS-11, but if you contribute there will be still more at the next.

See you there!

FIRING UP OLD #71 A Case Study

Dave Mallery

The 11/70 started volume production in the summer of 1975. My first machine, #104, was delivered in December '75 and has been running like the engine it is, ever since. We recently heard that a local insurance company was selling off its old 70 and by a variety of luck and instinct, go it in the low \$50's. As delivered, it had 1/2 MB of core, 4 DH11 AE's, a TU16 and RWS04(!) and a pair of RP04's plus a RSTS license and other gems. It had been powered down in March of 1981 and had been sitting in a clean computer room ever since. We sold off the RP04's and controller for a tidy sum and proceeded to install the rest (less the swapper) in our computer room. My goal was to have the machine shipshape (?) when DEC came in to take it on service. After all, there are 3 or 4 different independent depot repair services today — surely they can fix anything.

Once I had the electrical outlets in place, I re-cabled the TU16's controller and powered her up. Wonderful—she can load address!! Quick—17765000 load address, 60—start—she BOOTS! EUPHORIA. Load a memory diagnostic — EMJA?? — after a few minutes, reality—parity errors—hard.

Now, fixing a computer without a spares kit requires some creativity and courage, fortunately old memory boxes have many repetitions of the same boards — just move-em around and you can move the location of the error. At this point, we also discovered a bad regulator in one of the boxes. I used Reliance Electric in Columbus for fixing my regulator and bad memory boards—they seemed to turn them around almost immediately. Now that the memory seemed O.K., on to the major processor diagnostic EQKC??. This is the one that

... continued on page 36

RSTS PROFESSIONAL

Editors

R.D. Mallery
Carl B. Marbach

Assistant Editor/Advertising
Helen Marbach

Controller
Peg Leiby

Administrative Assistant
Hope Makransky

Subscription Fulfillment
Kathi B. Campione

United Kingdom Representative
Pauline Noakes
RTZ Computer Services, Ltd.
P.O. Box 19, 1 Redcliff Street
Bristol, BS997JS
Phone: Bristol 24181

Contributors

William L. Baker
Al Cini
Steven L. Edwards
Steven Fahey
Dave Froble
Jeffrey R. Harrow
Kevin Paul Herbert
Steve Holden
Ed Judge
Michael H. Koplitz
George T.A. May
Mike Mayfield
Frank Metcalf
Bob Meyer
Gary W. Miller
Neil Robertson
Robin Robinson
Joel Schwartz
Michael B. Schwartz
Brad Smith
David Spencer
Ron D. Troy

Cartoons

Douglas Benoit

Photographic Consultant
Bill Marbach

Design & Production
Grossman Graphics

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 publication is not promoted, not authorized, and is not in any way affiliated with Digital Equipment Corporation. Material presented in this publication in no way reflects specifications or policies of Digital Equipment Corporation. All materials presented are believed accurate, but we cannot assume responsibility for their accuracy or application.

We've got the VT100TM market seeing red,



Introducing the VT100-code compatible Colorscan 10 for just \$3,195. It's the first affordable display terminal with the dramatic high-resolution color that businesses need.

and blue, and yellow...

Inside, the Colorscan is packed with performance. Like smooth or jump scrolling and split screen/regional scroll. Double-high/wide and double-wide characters. Key-

The new Colorscan 10 features eight independently selected foreground/background colors (red, green, blue, yellow, cyan, magenta, black and white) crisply displayed in both 80- and 132-column formats. You also get a 128-character set, plus 128 special graphic symbols.

board selected set-up parameters, self testing, even a CRT saver. All standard. If your business needs more effective financial and sales graphs, charts, histograms, or drawings in 80- or 132-column formats, ask about the Colorscan 10.

The Colorscan boasts state-of-the-art ergonomics too. You get a tiltable screen, high resolution display, and detachable keyboard with separate numeric pad.

It covers the whole spectrum of business applications.

Write or call the smarter terminal maker: (609) 665-5400. Datamedia Corp., 7401 Central Highway, Pennsauken, NJ 08109.

Colorscan 10
 Make me see colors. Please:
 Send complete product information.
 Have your representative call.
 Return to: Datamedia Corporation, 7401 Central Highway, Pennsauken, N.J. 08109. (609)665-5400.

Name _____
 Company _____
 Address _____
 City _____
 State _____ Zip _____
 Phone _____

RS1281



From the smarter terminal maker.

VT100 is a registered trademark of Digital Equipment Corporation.

Better than Datatrieve!

- Easier to understand
- Much more powerful
- Better manuals
- Better support

Not Limited to RMS-11!

- Can access most file types and DBMS
- Include your own file routines
- Mix file types in a single report

THE
ULTIMATE
REPORT WRITER

GRS

GENERALIZED REPORTING SYSTEM
QUERY LANGUAGE and REPORT WRITER

for PDP-11/RSTS systems



Call or write for complete details:

305 Madison Avenue • New York, NY 10165
 (212) 972-1860

Incredibly Flexible!

- Complete formatting flexibility
- Powerful built-in calculations
- User-specified calculations
- Include your own special routines

Easy to use!

- Usable by non-programmers
- English query-language
- Very user-friendly
- Nonprocedural language

A TOP-DOWN LOOK AT INTERACTIVE SOFTWARE

By Al Cini, Computer Methods Corporation

THE CRAFT OF SOFTWARE DESIGN

In "Segmenting BASIC-PLUS-2 Applications" (RP Vol. 3 No. 1), we reviewed the advantages of modular (many parts) over monolithic (all-one-piece) software systems. Functionally organized modules, i.e. modules which exist to perform one isolated independent function, make the "best" systems when they are interconnected using simple coupling mechanisms (preferably, using argument lists with a minimum number of arguments). To many, the derivation of a system's modules is an arbitrary, artistic process — a personal matter between programmer and computer which is closed to objective study. The scientists (we prefer not to think of ourselves as mere "craftsmen," and yearn for the day they grant Nobel Prizes to coders) among us, however, prefer to think of software design as a convergent rather than divergent intellectual endeavor. A number of competent software engineers independently implementing the same set of user specifications will, we believe, produce very similar rather than entirely different results.

Philosophically, this sort of thinking raises an interesting hypothetical question:

"For a given set of perfectly detailed specifications, how many different software designs are possible?"

Basically, there are two different answers to this question: "infinite" and "a few."

If you think "infinite" is correct let me clarify that by "design" I mean the representation of a system in a functionally diagrammatic form, such as the hierarchy charts we discussed in "Segmenting BP2 Applications" (see figure 1). Selection of a programming language and actual coding are not included.

If you still say "infinite" let me further stipulate that all synonymous symbols in the design process (such as different variable names or graphic techniques that mean the same thing) are not to be considered as significant in this exercise.

If you still believe "infinite" is the correct answer, please note that an infinite space is a very comfortable one in which to solve problems for a living. After all, if the solution you select is wrong who can fault you given the astronomical odds against picking the right one?

If you bravely answered "a few," allow me to carry this one step further and assert that the answer is ideally "one." The design of a software system is an essential re-statement of its specifications; where two or more equally plausible design alternatives are possible, the specifications are in some respect incomplete.

Of course, all "real world" specifications are incomplete, so we're back to "a few." Software analysis and design — the iterative processes by which we gradually converge on agreement between specifications and final product — are still human activities which elude "cookbook" description (if we could define them precisely, then we could program our machines to do them).

Just the same, a few program design **heuristics** which facilitate the development of software systems have evolved and become accepted over the years. One of these, "top-down software development," pursues program construction by defining the higher levels ("main routine") of a system and proceeding through the design of the lowest levels ("subroutines") by a process called **stepwise refinement**.

TELLING TOP FROM BOTTOM

Simply stated, the "top" of an application is the part a user sees, while the "bottom" communicates directly with the machine. Information understood at the top of an application, such as names, addresses, social security numbers, etc., are very abstract from the computer's point of view. By the same token, cylinders, tracks, sectors, buffers and the like seem pretty foreign to the average user. The purpose of a software system is to convert a user's input symbols into the proper output symbols by transparently translating them to and from forms the computer can deal with.

This translation process occurs in discrete steps at specific levels of the software design sometimes referred to as **levels of data abstraction**. Consider the very simple example of a program which reads a sequential file and prints its contents on a terminal (figure 1). At the highest level of the application, a user requests a printed listing of, say, all the **RSTS Professional** subscribers. The computer, which can't deal with entire logical files and complete output reports (setting aside PIP and Datatrieve for the moment), can however manage individual input and output records. At the next level of the application, we deal with the next-lowest abstraction of all **RSTS Pro** subscribers, i.e., **individual subscribers**.

In a stepwise fashion, we have "decomposed" the user-world data into a smaller form by refining the overall file listing process into individual record handling operations, which are executed iteratively for each record in the subscriber file. Since our computer (really, our programming language/operating system) offers instructions to deal with these decomposed data records (GET and PUT/PRINT), no further refinement is required and our design is complete.

CALOUT

POOR MAN'S COMPUTER LINK UTILITY

When you run CALOUT from your terminal you may dial up another computer by simply entering the phone number at your keyboard. You are then able to work as a remote user to the other computer with the additional ability to move files between your system and the other computer. All types of files may be transferred between the computers in either direction. This includes binary program files. Complete communication error checking guarantees that the files are transferred correctly even if the telephone lines are noisy. The package may also be used to automate incoming and outgoing TWX systems which then become directly attached to your computer.

CONTRL

REMOTE INTERACTIVE TRAINING AND USER SUPPORT

When you run this utility program at your terminal you are able to capture any keyboard on the system. Thereafter, every keystroke entered by the user, together with every response from their job is also presented to your terminal. In addition, any keystrokes at your terminal will appear at the user's screen and the system will respond exactly as if they were entered by the user. A complete log file of the session may be kept. The process of capture and release is invisible to the user. The user may be engaged in any task whatsoever, indeed the user keyboard may be logged off and turned off when CONTRL captures or releases it.

DOC

UTILITY PACKAGE MAKES FOUR TERMINALS OUT OF ONE

With this package you may run up to four independent, interactive jobs of any kind from your terminal with context switching between them at any point in a job session. Each of the four logical terminals generates its own log file of all that passes between you and the computer. This includes straight paranthetical text which you may insert anywhere in the middle of a session without interrupting the job. In the session log file, each key stroke is underlined to distinguish it from computer responses to the job.

- ★ No Installation, Simple to Use
- ★ Clean, Powerful, Certified
- ★ Inexpensive, Low Overhead

Call: Janet (617) 275-6642

Clyde Digital Systems, Box 348, Bedford, MA 01730

DEC DATA SECURITY THE SOFTWARE ENCRYPTION SOLUTION

By Michael B. Schwartz, Prime Factors, Oakland, CA

If you've opted to read this first sentence it most likely means you've keyed on the word "security." The reason: Within the last year the DP community has been bombarded with articles describing known computer crimes, estimates of the much larger number of undetected and undisclosed frauds, and a gloomy outlook at computer privacy in the 80's.

There's good reason for concern. Information is becoming a currency in itself — it has value, sometimes inestimable. Computers are the new concentrators of the information asset. They are the banks of the future. Conventional banks use physical security to separate their currency from those who would like to own it illegally. But the physical security of computers themselves plays a minor roll. A dial-up port is a 24 hour a day open invitation and so is absolute access to on and off line magnetic media. Computer crime rarely means physical theft. A simply copy or off-site data draw usually leaves no trace. Similarly, a simple wire tap and data capture into a microcomputer is undetectable. In fact, a number of crimes involved **adding** information: consider the ramifications of a furtive entry into an accounts payables file to write checks to a dummy company (owned, of course, by the white collar criminal).

The mainframe world has acknowledged these real and potential hazards and is now, rather hastily, trying to patch the breaches in the security wall. The minis are rapidly assuming many of the functions of the mainframe and are encountering the same risks. Risks multiply with the proliferation of distributed processing networks, word processing and electronic mail, minis in use by banks and other financial institutions, and corporations who are replacing their monthly time sharing bill with their own mini-computers.

Yet, until now, there has been no level of data security for the minicomputer user other than the minimal effectiveness of standard operating system passwords and simple access control commands.

This article discusses how a new concept, the GSDEU (Generalized Software Data Encryption Utility) developed by Prime Factors of Oakland, California, and implemented in its PSYPHER™ system, can solve these problems. This first section is a primer on data encryption. The second will address encryption's roll in security on DEC computers.

As with most techniques of secrecy, the foundations of data encryption grew out of military necessity. A couple of its earliest known forms are the "skytal" device, used by the Spartans against the Athenians; and the Caesar Cipher, used by Julius Caesar against almost everybody. Two world wars brought cryptography out of its adolescence and the emer-

gence of the electronic computer kicked the discipline into maturity.

Types of encryption can be broken down into 2 major categories: ciphers and codes. The most basic cipher, the substitution, works exactly as it sounds. During the encryption process individual letters in an alphabet are consistently substituted for others. The result can appear confusing, but can easily be broken by frequency analysis. Each letter in every alphabetic language has a distinctive frequency of occurrence. These frequencies are the joy of the cryptanalyst (one who delights in cracking codes and ciphers. Edgar Allen Poe was notorious for his addiction — read "The Gold Bug"). Take, for example, the following substitution scheme:

ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789. becomes
0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ

Substitution of the word "CLEARTEXT" (means unencrypted information) produces the ciphertext (encrypted information): "2B40HJ4NJ."

It looks cryptic but it isn't. Because of the multiple occurrences of 'J' and '4' in the ciphertext one could safely assume that these characters represent the most frequently encountered letters in the English alphabet: 'E' and 'T'. Given a minimal amount of ciphertext even an inferior cryptanalyst can break this cipher.

The second type of encryption, the code, is usually based upon substitution, except groups (not letters) of elements are replaced by an alternative notation. Stenography, where words and syllables are replaced by symbols, is a well known code. Codes based on substitution are as insecure as substitution ciphers and because of their design are particularly unsuitable for computer automated encryption.

Such is the insecurity of substitution techniques. But in one simple step a substitution cipher or code can be made **absolutely** unbreakable. Consider the following harmless series of "random" numbers:

3 7 4 12 2 6 6 5 1

Now, using the substitution scheme shown above these numbers can be used to shift the lower sequence of characters to the right just prior to substituting. Thus the cleartext 'C' is no longer '2' but is shifted 3 digits to the right and the ciphertext character becomes '5'. This operation transforms "CLEARTEXT" into "5I8CJPASK." The randomness of the shift has eliminated the tell-tale letter frequencies. Repeated characters in the cleartext are no longer represented by uni-

niques. Only the most careful analysis has determined that the pseudokey generation processes are virtually untracable and are without period degeneration.

Here I've described only **one** of the algorithms employed in the PSYPHER system. The system offers a spectrum of security algorithms designed to allow the user to select a trade-off between runtime efficiency and desired level of security. Included is the U.S. Government's DES algorithm, an extremely secure cipher that does not employ random number generation techniques. The PSYPHER system is a complete ready to load and run module that performs simultaneous file compression, "in place" encryption of user defined fields within file records and incorporates many other security and convenience features.

Thus far I've discussed some basics of encryption and have described the rudiments of stream ciphering (vs. the government's NBSDES block ciphering technique. The difference is profound on all levels; from theoretical design to program implementation.)

But on to the second part: Just how can the abstract concepts of data encryption be applied to the practical world of automated security? The first step is to more closely examine some reasons for data protection. The following list is a non-comprehensive summary of situations calling for the reduction of data exposure risks.

- (1) Efficient banking practice has evolved an interbank network that transfers \$400 billion daily over 99% insecure communication channels.
- (2) The rapid move towards distributed data processing has necessitated constant data communications.
- (3) The automation of previously manually executed business functions has caused a dramatic increase of access to computer systems by company personnel.
- (4) Electronic mail systems, growing proportionally with word processing systems, are, and will be, in extensive use.
- (5) The information glut has considerably increased the volume of offline and off-site storage of magnetic media.
- (6) Public time sharing installations are widespread and on the increase. This includes companies who, for economic reasons, are opening their computers to outside users.
- (7) In the absence of telecommunication facilities, companies are using the postal and courier services to transport magnetic media.

This list is certainly not complete, but is sufficient to enable a classification of the 3 major areas of data exposure:

- Off-site and offline storage
- Online storage
- Communicated information (telecommunicated or physically transported)

Presently the possible solutions to securing information that fall into these classes are limited to products that aim at shielding only one of the 3 exposure areas.

In mainframe computers online information may be secured by "access control systems," complicated systems

that define who can access what files, when they may be accessed, and from which terminals. This is fine for mainframe installations, but does nothing for the DEC user. Hardware encryption devices have been introduced to protect telecommunications; but that's all they do. And there is no off-the-shelf product for protecting offline and off-site media. The usual solution is physical security.

This leaves the DEC security scenario in a sorry state. Online access control is at best limited and simplistic (password protection) and the cost of the last 2 targeted solutions can be steep if not prohibitive. Encryption communication hardware can cost \$5000 per node and the cost of good physical security, including magnetic locks, secured structures and guard personnel is astronomical.

PSYPHER services the 3 areas of exposure via a **single** product. The guiding principles behind the development of the PSYPHER system were cost effectiveness and generality. Previous to the design of the system Prime Factors determined a set of encryption system characteristics that would meet the needs of generalized security. The result was the aforementioned GSDEU.

The PSYPHER GSDEU is a marriage between specialized encryption algorithms and generalized I/O capabilities. In order to accomplish its full spectrum of security tasks, the GSDEU must be software and have the following characteristics:

- It should run standalone. There should be no need to modify existing software. Installation and full operation is accomplished in hours.
- It should encrypt on a file by file basis servicing virtually all file structures supported by an operating system.
- It should offer the user the option of employing more than a single encryption algorithm. This offers the data manager trade-offs between degree of security and run time efficiency depending on the nature of the data to be protected.
- It should have the ability to partially encrypt file records. Under most circumstances only a fraction of a file's contents (e.g. names and addresses) contain proprietary information. Partial, or "field" encryption saves on run time and provides the distinct advantage of allowing the user to operate (sort, merge etc.) on the unencrypted portions of file records.
- It should offer additional useful and money saving services, such as file compression, that should be performed within any utility that manipulates every byte in a file.
- It should provide more than one kind of encrypted output to facilitate compatibility with communication hardware. In particular, encrypted output should be available in binary stream or character coded forms.
- It should be available for use in batch or interactive modes.

Now, let's examine how the PSYPHER GSDEU can accomplish protection in the 3 areas of data exposure. First, the GSDEU can be used to encrypt all off-line media. On-site physical security may be relaxed because the threat of illegal copy is nullified. The result: reduced costs. In addition, file compression significantly reduces the amount of physical

storage space required because of the reduced number of volumes to be stored.

Second, the GSDEU protects on-line files from accidental or purposeful disclosure. It compensates for the inadequate security of standard operating system passwords. A file that is encrypted has its access **implicitly** controlled. That is, the file itself may be accessible, but for all practical purposes its contents aren't.

Third, the GSDEU can virtually replace hardware encryption in telecommunications. In some ways the software solution is better. The absence of hardware means no installation costs, no down time, and the inevitable hardware malfunctions are avoided.

The GSDEU's compression facilities perform functions beyond the scope of hardware devices. Compressed data not only reduces transmission times and costs, but offers greater security. This is due to the fact that repetitive information in encrypted files makes the encoded text more susceptible to being broken.

The GSDEU's encryption algorithms can produce output in binary or character coded forms and records may be fixed or variable length. This allows tailoring the encrypted information to the computer's communications gear rather than the other way around.

This completes the discussion about how the GSDEU can successfully service the 3 major areas of data insecurity; but what about the security of the encryption algorithms and their run time efficiencies? Presently there is considerable debate within academic circles about the degree of security inherent in each of many possible algorithms. This is all debated with little or no concern for the pragmatics of production data processing. Obviously, one does not employ an algorithm that encrypts at 500 bytes per CPU-second to continually encrypt and decrypt a 5 megabyte file. In general, it appears that the relationship between processing time and subjectively estimated degree of security is geometric; processing time increases at a very fast rate with the "more complex" algorithms.

The solution to the security-efficiency problem is to make more than 1 algorithm available to the user. Then, in most cases where information is of extremely sensitive nature and relatively short length, the user employs a slower, more "secure" algorithm. But when it comes time to encrypt a mailing list of 100,000 names, the algorithm with the faster runtime is the likely choice. The ideal GSDEU has a number of user selectable algorithms of different design running from 500 to 10,000 bytes per CPU-second. It is important to allow the user to determine the proper trade-off between runtime efficiency and degree of security.

Let's take a brief look at the application of the PSYPHER GSDEU to the RSTS system. Two of the major data exposure areas, offline storage and data communications are independent of the system and any protection it may provide. Therefore if an installation has weak physical security or is communicating data via any medium, PSYPHER is the **only** protection against data copy and capture.

RSTS can only offer limited protection to online files via sign-on passwords and rudimentary access control pass

codes. Just how effective are these "protections"? Breaches of sign-on passwords are well known and published. High School students have breached both sign-on passwords and file access pass codes on much more elaborate mainframe operating systems. And studies have shown that these illegal penetrations have occurred with much greater frequency than have been reported. An estimated 1 in 10 security breaches resulting in breaking federal or state laws go unreported. This is because such occurrences would frighten corporate stock holders, drive away time sharing business, and embarrass management.

It is this writer's opinion that RSTS security will be breached (if it hasn't been already) when the "work-break cost," the cost of researching and writing the break code, is sufficiently low compared to the value of the information to be fraudulently added, altered or replaced. Computer crime is a business, the work-break cost is simply the break even point in the cost of perpetrating the fraud versus the expected financial returns. With the increased use of RSTS and VMS systems by large financial institutions, the work-break cost is now **very** low compared to the illegal gains.

Another problem is the case of the privileged user. Unlike most mainframe installations, minicomputer sites tend to be lax in freely assigning privileged accounts to too many individuals. If proprietary data is present, the risk of someone gaining illegal access grows significantly with the number of privileged users. It's simply too easy for an outsider to learn the proper pass codes for global access, especially when the privileged user often works in common user rooms. The privileged account is an obvious convenience. But it takes only seconds to obtain the proper pass codes with a wire tap or a glance over a shoulder.

Finally there's the problem of the operators console. Any unauthorized individual who gains access to the main console has unlimited browsing power, usually **without** the need for any pass codes. Highly restricted physical access (other than proper encryption) is the only solution.

In summary, the need for data security is rising. There must be a cost effective tool for maintaining privacy and data integrity. In many cases the law **requires** (personal, medical and government data) via the Privacy Act that an installation take all available measures to insure data privacy. The PSYPHER system gives the user virtually complete control over his own files. PSYPHER GSDEU encryption insures the user that his proprietary files will remain secure and undisclosed independent of the degree of physical security, the operating system, illegal system penetrations, and misused privileged user access. An investment in encryption is simply paying a one time insurance premium against a theft or accident that one hopes will never occur.

Prime Factors is a software and consulting firm specializing in the area of software data security. Its primary product, PSYPHER, is presently in use on IBM, DG, and DEC computers. Other products available are callable DES modules (DESCRYPTTM), an exact implementation of the National Bureau of Standard's Data Encryption Standard; and PSYSUBSTM, a callable version of the stream ciphering algorithms in PSYPHER.

Michael Schwartz is owner and founder of Prime Factors. His staff consists primarily of mathematical and computer scientists out of the University of California. Interested parties should contact:

Prime Factors, 6529 Telegraph Ave., Oakland, CA 94609
(415) 654-5090



TTYSET Optional Patch for VT100 Width Changes

By David Spencer, Infinity Software, Inc.

One of the great features of the VT100 is the ability to switch between 80 and 132 column mode. When the VT100 was put on the market, I couldn't wait to get my hands on one. No longer, I thought, would I have to hassle with trying to read wide reports on a VT52.

Well, I finally got a VT100, and things didn't work out like I thought they would. In order to examine my wide listing, I had to hit SET-UP, hit the 132-COLUMN key, hit SET-UP again, and then run TTYSET to change my width to 132 columns. Of course, after about doing this once or twice, it was time to get more sophisticated.

There are several alternatives. One is to always run the VT100s in 132 column mode. Another is to write two little programs as CCLs that do the switch and SYS() call themselves. My solution was to patch the TTYSET program to send an escape sequence to the terminal when you change its width.

The following is a patch to the TTYSET program to switch VT100 terminals automatically to 132 or 80 column mode when the width is changed. This can be done easily because the TTYSET "VT100" macro sets "XON"; a feature no other devices utilize. (Using "XON" to check terminal type is used by DEC software like EDT version 2.)

One new variable has been added to TTYSET. WIDTH% is a flag to prevent sending VT100s the width change sequence each time a parameter is changed. If you wish to leave your VT100 in VT100 mode after switching widths, remove the last escape sequence to switch back to VT52 (<esc>[?2l]). One note, this patch has been implemented to send the width change sequence to terminals other than your own if they are specified.

The TTYSET patch can be applied with the CPATCH auto-

patch program. (If in doubt, see the RSTS/E V07.0 System Manager's Guide.)

```
*G/2/V<cr>
2l<tab><tab>PROGRAM<tab><tab>; TTYSET.BAS<cr>
*H/1040<tab>/V<cr>
1040<tab>GOTO 1530 IF C$="HELP" &<cr>
*I<cr>
WIDTH% = 0% &<cr>
<tab> \ <esc>
*V<cr>
<tab> \ GOTO 1530 IF C$="HELP" &<cr>
*H/1240<tab>/V<cr>
1240<tab>GOSUB 12100 &<cr>
*AI<cr>
<tab> \ WIDTH% = V% &<cr>
<esc>
*V<cr>
<tab> \ IF E%=0% AND V%>1% AND V%<256% THEN &<cr>
*H/1430<tab>/V<cr>
1430<tab>GOTO 1500 UNLESS LEN(Fs) &<cr>
*OI<cr>
1425<tab>M%(4%) = ASCII(MID(SYS(CHRS(6%)+CHRS(9%)),2%,1%))/2% &<cr>
<tab><tab><tab>IF M%(4%) = 255% &<cr>
<tab> \ IF WIDTH% = 81% AND M%(9%) = 255% THEN &<cr>
<tab><tab>C1$ = SYS(CHRS(6%)+CHRS(5%)+CHRS(M%(4%))) &<cr>
<tab><tab>
<tab><tab>! GET KB #, GET OUR OWN TO SET UP COLUMN CHANGE FOR VT100 &<cr>
<tab><tab>! IF WE HAVE XON AND WIDTH OF 80 THEN &<cr>
<tab><tab>! BECOME VT100, SWITCH TO 80 COLUMNS, BECOME VT52 AGAIN &<cr>
<cr>
1427<tab>IF WIDTH% = 133% AND M%(9%) = 255% THEN &<cr>
<tab><tab>C1$ = SYS(CHRS(6%)+CHRS(5%)+CHRS(M%(4%))) &<cr>
<tab><tab>
<tab><tab>! IF WE HAVE XON AND WIDTH OF 132 THEN &<cr>
<tab><tab>! BECOME VT100, SWITCH TO 132 COLUMNS, BECOME VT52 AGAIN &<cr>
<cr>
<esc>
*V<cr>
1430<tab>GOTO 1500 UNLESS LEN(Fs) &<cr>
*EX<cr>
```

David Spencer's VTEDIT.TEC, discussed in the last issue, can be found on page 70.

DEC BEST VALUES

PRE-OWNED DEC EQUIPMENT

BUYING AND SELLING

SYSTEMS • CPU's • PERIPHERALS • TERMINALS
OPTIONS • MEMORY • COMPATIBLES

CALL DICK BAKER (305) 979-2844

dataware
incorporated

Carico Center
2845 NW 62nd Street
Ft. Lauderdale, Florida 33309
Telephone (305) 979-2844

CIRCLE 49 ON READER CARD

DUMPER-FILE BACKUP UTILITY

- Much faster than BACKUP
- Supports RSTS Large-File feature
- Ideal for big databases
- Proven and reliable, easy to use

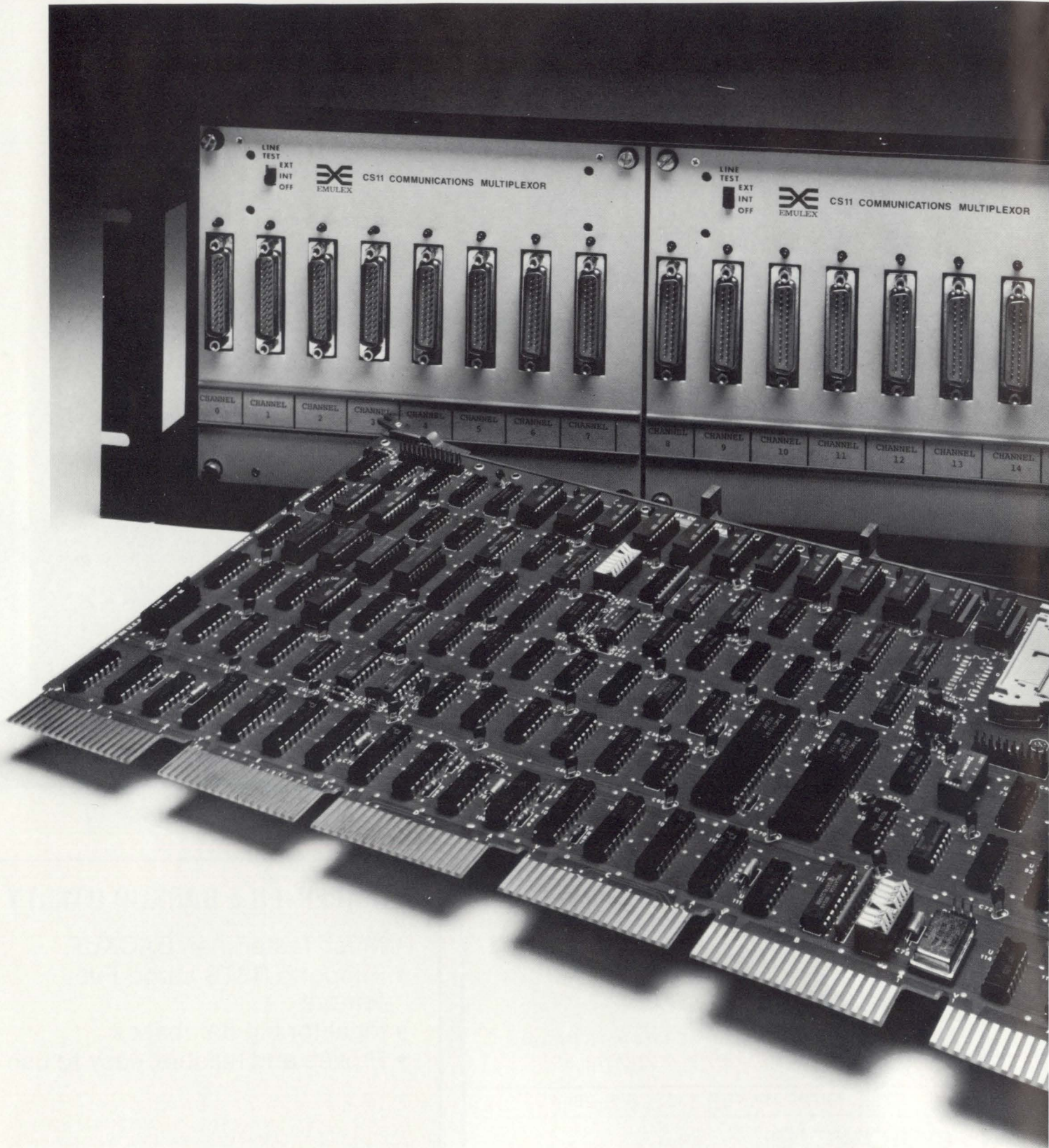
etc

ENTERPRISE
TECHNOLOGY
CORPORATION

305 Madison Avenue
New York, N.Y. 10165
(212) 972-1860

CIRCLE 48 ON READER CARD

Improve your communication



with DEC. Just concentrate.

We've earned our reputation by designing and delivering dependable, easy to live with tape and disk controllers for DEC cpus. Products that really showed the industry how to do it.

Now we've cleaned up the communications I/O morass, with a new DH-11 type multiplexer for PDP-11 and VAX systems. It duplicates all the features of DEC's DH-11. And it goes well beyond, delivering improved line handling capabilities in a far smaller package and at significantly lower cost.

You might say we're now doing for communications controllers what we've already done for tape and disk controllers —

applying the latest microprocessor technology and some sexy design to give our customers the performance and reliability they deserve.

Our newest achievement proves the point. To wit: just one hex size board, occupying a single Unibus slot, provides up to 64 lines for terminals, printers or other communications I/O. That's eight times more efficient than the DEC approach.

And our unique system organization means maximum flexibility. You can daisy-chain up to four remote distribution panels, expanding in 8 or 16 line increments all the way to 64 lines. You can configure any 8 line group for an RS-232 interface, with modem control a standard feature. Or you have the option of selecting a current loop interface. And because all your interface variations are made

at the distribution panels, no additional hardware or cpu restructuring is needed when you add channels.

You get everything you got from DH-11, such as DMA on transmit operation and line format flexibility. Naturally, we're fully software transparent and run existing DEC diagnostics.

But we like to do everything we do just a little bit better than the other guy. So we boosted performance to 19.2 K baud per line. For peak traffic, we can "double fifo" to 128 characters per 16 lines. Multiple controllers can be added for increased capacity. You get full 16-bit word transfers on DMA operation — not just byte transfers. And, of course, we've taken advantage of our system's high speed bipolar microprocessor to build-in extensive controller and channel self-test and fault indicators.

We're concentrating on DEC, making the world's best CPUs even better with some ingenious controllers. We've done it for tape and disk interfacing. Now for communications I/O.

For off-the-shelf delivery (U.S.A.), contact: First Computer Corporation, Corporate Square, 825 N. Cass Ave., Westmont, IL 60559. Telephone: (312) 920-1050.

For complete information (U.S.A.), call or write: Emulex Corp., 2001 East Deere Avenue, Santa Ana, CA 92705. Telephone: (714) 557-7580; TWX: 910-595-2521

In Europe: Emulex Corp., 10th Floor, Cory House, The Ring, Bracknell, Berkshire, ENGLAND. Telephone: 0344-84234; TELEX: 851-849781.



The genuine alternative.

The new Emulex CS-11/H really concentrates your communications I/O. Just one hex size board multiplexes up to 64 lines.

**ANNOUNCING NEW
CS21 16-LINE MODEL
AND COMPLETE VAX SUPPORT.
SEND FOR MORE INFORMATION.**

SOFTWARE PROTECTION: WHAT IS IT REALLY?

By Robin Robinson, Mini-Computer Business Applications, Inc.

Mention software protection to most people and they'll say "What?" Mention it to lawyers and they'll say "Right . . . copyrights, trade secrets, patents, contracts . . ." Mention it to the guy who spends much of his time actually trying to protect the stuff and he'll groan.

There are relatively few in the software industry who've faced the day-to-day battles straightening out the messes their software got into. A software house is fortunate if they have just one full-time employee chasing down offenders—who run the gamut from not knowing any better to the out-and-out thieves. There's a quiet insanity that accompanies his job, as no sooner does one violation get resolved than another one shows its ugly head. And the truth is, with a large vendor, it's normally not just one at a time. Few understand the dilemma because they don't face it eight hours a day. Hence, it isn't discussed much. But put a few of these "software protectors" in a room together and the chatter will go on for hours. Far more than the safely distant and pedantic recital of "copyrights . . . trade secrets . . . patents, etc." the software protector is put in the trying position of making others understand what these **mean** sufficiently enough to prevent the need for a courtroom judge to do the educating. Courts and lawyers are, after all, expensive.

THE REAL PROBLEM

The problem faced by vendors and software protectors is that software gets stolen. Surrounding all such thievery is an ignorance of what's supposed to accompany software, and how it's supposed to be protected by those who've been entrusted with it. Users of all types often do not realize that software is usually the proprietary property of a vendor, and that the use of the packages is a **licensed** use. Frequently, a "seller" represents the transaction as a "sale" and the user proceeds with his use of the software with this understanding in mind. Hence, a consultant or OEM (or even the guy next door) may request to purchase a copy of the users "property" and the user, not knowing differently, will be happy to provide him with one for a small fee. It's not unlikely that the vendor will hear about this years after the fact, and after the software has passed through the hands of three or more companies—with the first two or three of them now out of business. The rhetorical question presents itself to the software protection "Where do you suppose our software is now?"

WHO GETS HURT?

The difficulties encountered are shared by all segments of the industry. Unauthorized copying of computer software

is a contagious disease, compounded by the fact that many of the recipients of the now "illegal" packages don't always know they've been exposed to the potential losses involved. A rather devastating example: Company A purchases a one-use license from a vendor, and then issues multiple copies of the packages. That's violation number 1. Company A then does a few other things: he sells one of his users a leased machine; he fails to deliver packages he's sold; he fails to ensure the packages are covered by a license or sublicense agreement; he fails to support those packages he **does** issue; and then he skips town. This example is not too far off the beaten track. What Company A has done is he left a large handful of users holding the bag. The vendor is faced with a group of understandably irate users, many of whom may have source code. Despite an excellent contract designed to prevent this sort of thing, the problem still faces a vendor as to what to do now? Ignoring it may be the easiest choice, but to do so runs the risk of weakening its trade secret protection. Besides, it's not unlikely that the users have formed a sort of modern-day vigilante committee who seek justice where there apparently is none. They've already paid so much money to Company A, they aren't about to pay the vendor for the packages and support they still need, at the vendors' usual rate. And in the wings are probably one or two consultants ready to take advantage of the situation. Try asking any of these users what they think of the software industry now and you're likely to get run out of town. Ask the vendor, and the other consultants involved, while you're at it. Situations like these don't leave any fond memories.

EFFECTS ON END-USERS

On a smaller scale, end-users are frequently left in the cold when it turns out that their supplier (OEM) didn't have the rights he claimed to have when the packages were issued to the users. Often, the user doesn't receive a written agreement for use of the proprietary software and therefore doesn't realize it belongs to some vendor located miles away in some other state or country. It usually isn't until negotiations between the OEM and end-user breakdown that the user discovers he's gotten hold of some "hot" software — information that's often given him by the new consultant he calls in. The vendor gets a phone call, which is frequently how he finds out the original licensee was messing around, and now the software protector has two problems on his hands: an end-user who maybe has his source code and no license agreement (and a bitter taste in his mouth); and an OEM or licensee who's been violating his license agreements. The details always vary, but can range from a user being out anywhere from \$100 to \$50,000 for the software alone. Sometimes more. The software protector will often hear the

a businessman's ears is a miraculous development requiring hours and hours of sweat and brilliance to produce. The vendor who has cared enough to produce a package (or 50 of them) that will make the lives of others easier is to be commended. True, he's in the business to make a living for himself. Maybe he'd even like to get rich. But if he's any good at all, he's not riding on the wings of someone else's creativity and hardwork. He's producing something new, and hopefully better. And it's costing him a good chunk of money to do so. To then release this into a marketplace full of users who see the value (threatened with thieves who also see the value) places the vendor in an unenviable position. His software protection people (or person) have to be sharp; have to know where to look at what to listen for; and then have to act fast when something's amiss, without having to resort too often to outside attorneys and courtrooms who could very well eat whatever profits come in.

Software protection individuals and efforts could well use some organization. The marketplace (end-users and OEM's alike) are in dire need of understanding the situation and its ramifications. Without adequate protection, both by the courts and on a day-to-day basis, the honest and productive vendor could be forced to close his doors. An Association for Software Protection (ASP) has long been needed, but has only recently been formulated by three software vendors who share the same problems. It is a bright idea needing support and contributions. Theoretically, it could prove to be a vehicle which introduces software protectors to one another and help them to create a marketplace in which a blackmarketeer wouldn't dream of trading and in which end-users know the rules up front; before the troubles begin. The form and activities of such an association would obviously need to respond to the input received from other software vendors and protectors; information which, due to the present alienation, has previously lacked any organization. Those interested in participating

PDP-11® & VAX® users...

ComDesign's TC-3 Concentrates Multiple Terminals and Printers on a Single Telephone Line Cost-Effectively for the PDP-11 and VAX.

- | | |
|--|---|
| ■ CONCENTRATION
Many terminals—one line | ■ COST SAVINGS TO 40%
Totally eliminates DZIs and cables |
| ■ PROVEN RELIABILITY
TC-3s are in use world-wide | ■ ERROR CORRECTION
Data checked with all errors eliminated automatically |
| ■ STATISTICAL MULTIPLEXING
Efficiencies to 400% | ■ FAST ECHO RESPONSE
—at last! |
| ■ BUILT-IN SYSTEM MONITORS
Real-time data and signal displays | ■ POWERFUL DIAGNOSTIC PANEL
Modem loop-back, unit self-tests, and terminal diagnostics |

Put a TC-3 on Your UNIBUS® and Get a DZ11 with Stat Muxing too!

"We are impressed . . . have already recommended the unit to a number of other RSTS/E users and shall continue to do so . . ."
Intercontinental Forwarders, Inc.

See us at DEC USER SHOW, Booth #123

For further information and the sales office for your area
Call **toll-free (800) 235-6935** or in Calif. (805) 964-9852

ComDesign

ComDesign, Inc
751 South Kellogg Avenue
Goleta, California 93117

*Registered Trademarks of DEC
CIRCLE 46 ON READER CARD

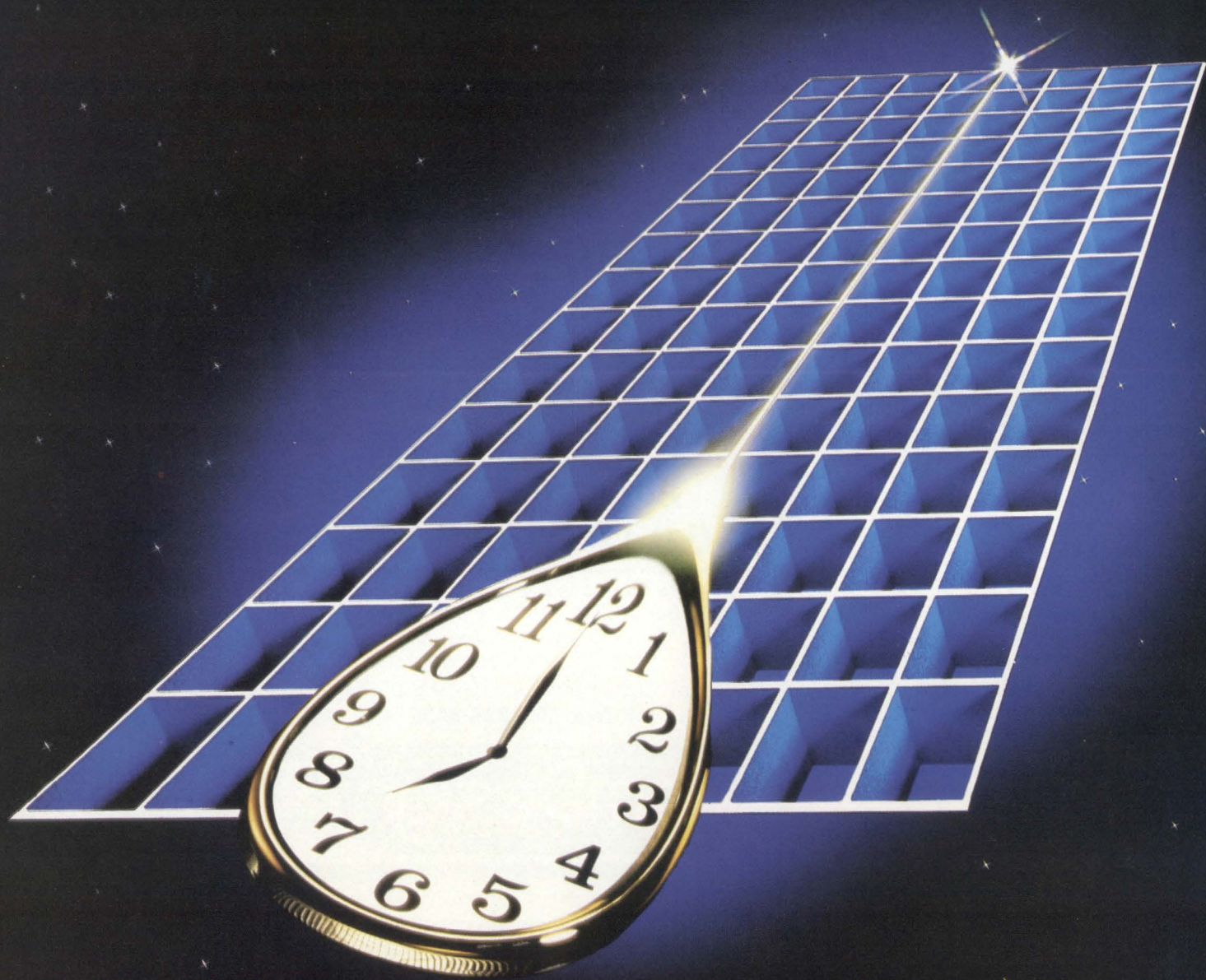
at the mini-computer level, are invited to contact the author (in writing) at 117 South Brand Boulevard, Glendale, Calif. 91204; or Ms. Adrienne Webb, Digital Information Systems, Inc., 6247 Fair Oaks Boulevard, Carmichael, Calif. 95608; or Mr. Dale Coleman, S&H Computer Systems, Inc., 1027 17th Avenue South, Nashville, Tennessee 37212. ♥

LOOK at the "tear-out" cards in this issue.

There's a "HOT to COLD" form for rating our articles. Let us know what is most and least interesting to you.

There's a FREE gift for you. Bring in new subscribers and collect rewards. See "BOUNTY HUNTERS" card.

There's a **READER INQUIRY** card for your convenience.



Introducing, for PDP-11 RSX, RSTS/E, and RT-11 users

Pascal-2TM

The Time Machine

The First Dimension: Performance

Pascal-2 performs.

Pascal-2 programs run as fast as FORTRAN IV-PLUS programs—or faster. (FORTRAN IV-PLUS is Digital's fastest PDP-11 high-level language.) Pascal-2 code is as small as code generated by any Digital PDP-11 compiler or interpreter—or smaller. And Pascal-2 typically compiles at 1000 lines per minute.

The Second Dimension: Structure and Portability

As a programmer, you can write in a language close to your thoughts. With Pascal's structured methods, you can do the job right the first time. It's easier to *design* in Pascal than it is to *debug* in FORTRAN, assembler, BASIC, or COBOL.

As a software manager, you will see the value of Pascal in improved communication among team members: they can understand one another's code. Pascal's portability will protect your software investment: your programs will outlive your current hardware.

The Third Dimension: Tools, Tools, Tools

The compiler precisely reports typographic or syntactic errors. The interactive, source-level debugger helps detect deep-rooted logical errors. The profiler helps identify code that can be rewritten to speed program

execution. Also included are formatters, index generators, and documentation aids—a total of 70,000 lines of Pascal code.

Our 2,000 customers use Pascal for such diverse applications as general ledger and payroll, integrated circuit design graphics, word processing, typesetting, and off-track betting; for trimming integrated circuits, monitoring particle accelerators, real-time ballistics modeling, and controlling saws in a lumber mill.

The Fourth Dimension: Our Past and Future

The core of our technical group has been together more than a decade. Our Pascal-1 compiler entered commercial use in 1975. Before releasing our PDP-11 product, we delivered Pascal-2 under contract to two major computer manufacturers for three different processors. Now we're moving Pascal-2 to Motorola's MC68000, to Digital's VAX-11, and to the UNIX operating system. We're committed to Pascal for the long term.

Call or write. We'll send benchmark details, a product description, and a free copy of the Pascal-2 manual (specify RSX, RT-11, RSTS/E).

PDP, VAX, RSX, RSTS/E, RT-11, and FORTRAN IV-PLUS are trademarks of Digital Equipment Corporation. MC68000 is a trademark of Motorola Inc. UNIX is a trademark of Bell Laboratories.

Pascal-2: The Dimensions of Performance

Ask for a free 18" x 24" poster of this photograph.

Network Computer Services
6 Cunningham Street
Sydney 2000, Australia
Telephone: 211-2322
TLX: 25523

Valley Software, Inc.
6400 Roberts Street, #390
Burnaby, BC Canada V5G 4G2
Telephone: 604-291-0651

Hours Computing, Ltd.
7-8 Mill Street
Stafford ST16 2AJ, England
Telephone: 0785-44221
TLX: 36540

Real Time Products
1 Paul Street
London EC2A 4JJ, England
Telephone: 01-588-0667
TLX: 884971

Rikei Corporation
1-26-2 Nishi Shinjuku
4316 Shinjuku-ku
Tokyo 160, Japan
TLX: 24208

AC Copy
Kurbrunnenstrasse 30
D-5100 Aachen
W. Germany
Telephone: 0241-506096
TLX: 832368

Periphere Computer Systems GmbH
Pfalzer-Wald-Strasse 36
D-8000 Munchen 90
W. Germany
Telephone: 089/681021
TLX: 523271

Oregon Software

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

ATPKED: At-pee-kay BASIC-PLUS LINE EDITOR

By William L. Baker, Pig Improvement Corporation

1.0 ABSTRACT

This article describes the implementation of new commands for RSTS/E version 7.0 utility ATPK. The commands will take some of the unfriendliness out of the otherwise friendly BASIC-PLUS environment by combining the functionality of ATPK with editing of basic without the use of an editor program.

2.0 BACKGROUND

The principal advantage to BASIC-PLUS is programmer productivity through the interactive program development. Productivity is sacrificed when the programmer must constantly switch back and forth between basic and an editor. In the native BASIC-PLUS environment the programmer has the choice of retyping the line or using an editor after saving the program. The inefficiency of the procedure is quite pronounced when learning or modifying large programs with multi-line basic lines. The ATPKED modification provides a simple substitute command to improve programmer productivity.

3.0 MAKE NEW FRIENDS

The ATPKED modification provides the programmer with an extended program development environment including: editing, control-C interruption, logging, run-time system switching, and indirect command files.

3.1 Commands

3.1.1 STANDARD ATPK COMMANDS

The programmer should be familiar with ATPK. See "RSTS/E V7.0 Release Notes," September, 1979 Sequence 17.1.1N. Pay close attention to the section on control characters, quoted control characters, the use of control-M on page 4 of 9, and "Other features" on page 9 of 9.

3.1.2 NEW COMMANDS PROVIDED BY ATPKED

\$EXIT Abort ATPK and close the PK job. This command replaces the use of control-C to end a command file from the KB: and simulates the native environment for control-C.

\$X Execute to last \$Substitute command. This command provides the ability to change the same string several times.

\$S line-number Xstring-1Xstring-2Xinteger-1Xinteger-2

Substitutes the integer-1 occurrence on integer-2 line of string-1 with string-2; X is a delimiter and may be any printing character; the space(s) are optional between the "\$S" and the delimiter. The basic line is listed upon successful edit. The "?" is removed from error lines.

The basic line number may be modified by the \$Substitute. This will cause a clone to be generated and listed.

3.2 Examples of Operation.

ATPKED EXAMPL = KB:

Ready

NEW EXAMPL

Ready

10 ! THIS IS SOME TEXT &
! AND SOME MORE

Ready

\$S 10 /E /E MORE /

10 ! THIS IS SOME MORE TEXT &
! AND SOME MORE

Ready

\$X

10 ! THIS SIS SOME MORE MORE TEXT &
! AND SOME MORE

Ready

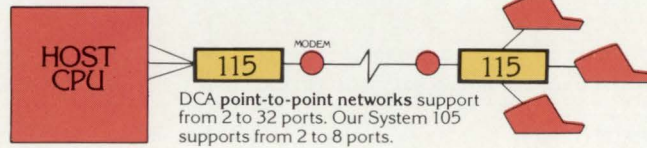
\$S10/ MORE///2

10 ! THIS IS SOME MORE MORE TEXT &
! AND SOME &

FROM STATISTICAL MULTIPLEXORS TO COMPLETE DATA COMM NETWORKS, EVERY DCA COMPONENT IS ENGINEERED TO EXPAND

DCA protects your initial investment in statistical multiplexors with the lowest-cost network growth in the industry. So you can start with a small datacomm network today, and expand or modify it to meet your needs tomorrow.

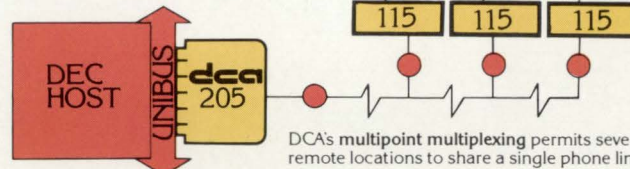
Our System 115 statistical multiplexor can be used in point-to-point networks to support from 2 to 32 asynchronous terminals at a remote site. DCA's statistical multiplexing assures excellent response time, character transparency and error-free transmission.



DCA point-to-point networks support from 2 to 32 ports. Our System 105 supports from 2 to 8 ports.

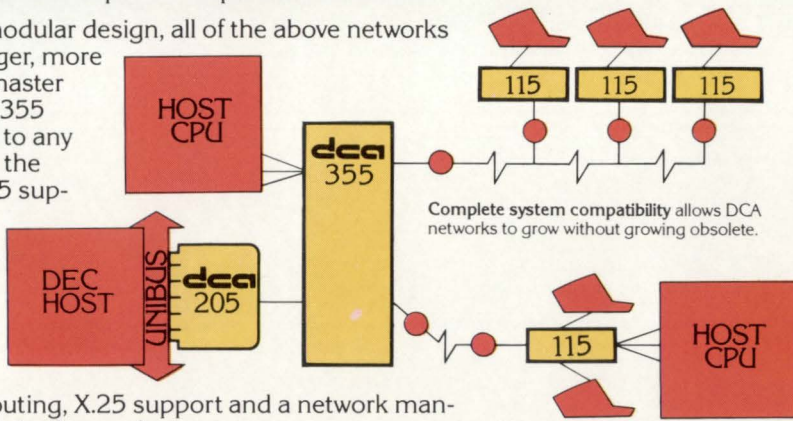
DCA's multipoint multiplexing configuration serves a number of remote terminal locations with just a single telephone line, for substantial savings in phone-line and hardware costs. DCA users have benefited from multipoint multiplexing since January 1979.

DCA's System 205 is a statistical multiplexor designed for DEC UNIBUS*-based computers. The 205 requires only one UNIBUS slot to emulate up to 16 DEC DZ11 modules and a 128-port stat mux. This greatly lowers hardware costs and improves response time as well.



DCA's multipoint multiplexing permits several remote locations to share a single phone line.

Because of DCA's unique modular design, all of the above networks could easily expand into larger, more powerful networks. As the master network processor, System 355 gives terminal users access to any host computer anywhere in the network. In addition, the 355 supports up to 126 ports, 44 of which can be high-speed synchronous trunk links. Several 355's can be combined to greatly expand this support. Features include port contention, switching, unlimited routing, X.25 support and a network management system built right into the product.



Complete system compatibility allows DCA networks to grow without growing obsolete.

Total system access and unlimited network growth at very low cost — that's the DCA advantage. For complete information, call or write for our 16-page brochure.



ENGINEERED TO EXPAND

*TM Digital Equipment Corporation

CIRCLE 61 ON READER CARD

RSTSPROFESSIONALRSTSPROFESSIONALRSTSPROFESSIONALRSTSPROFESSIONALRSTSPROFESSIONALRSTSPROFE

```

16360  GOSUB 16500
\      GOTD 16320
\
\          ! GO SAVE THE LINE
\          ! REPEAT PROCESS UNTIL NO COMPLETE LINE ARE FOUND
\
16380!  **** ROUTINE TO CHECK FOR NOTEXT PROBLEM IN NO EXTEND MODE ****
16385  IF INSTR(1%,SUB.TEXT,REM$,"Ready")=0% THEN 16010
\
\          ! CHECK FOR READY PROMPT EXTEND LINE IN NO EXTEND MODE
\          ! FALL THRU TO REJECT ROUTINE BECAUSE WE MISSED THE
\          ! CR-LF BY SUBING AN EXTEND LINE IN NOEXTEND MODE
\
16389!  *** ROUTINE TO REJECT REQUEST IF EXTEND LINE IN EXTEND MODE ***
16390  SUB.NDEX,ERRZ=1%
\      SUB.SWZ=0%
\      GOTD 16010
\
\          ! SET ERROR SWITCH
\          ! RESET SAVER SWITCH
\          ! RETURN SO SUB ROUTINE CAN ABORT REQUEST
\
16499!  **** TEXT SAVER SUBROUTINE ****
16500  SUB.TEXTZ=SUB.TEXTZ+1%
\      SUB.TEXT$(SUB.TEXTZ)=SUB.LINE$
\      RETURN
\
\          ! BUMP COUNT OF LINES SAVED
\          ! SAVE THE LINE
\
19030  IF ERL>14000 AND ERL<14999 THEN RESUME 14900
\
\          ! TRAP $S INTERPRETER ERRORS
\
19031  IF ERL=13660 THEN RESUME 13670
\
\          ! TEST FOR VAL ERROR WHEN RESETTING BASIC LINE NUMBER
\
19032  IF ERR=28 THEN RESUME 4100
\
\          ! TRAP ^C AND ATTEMPT A RESTART
\
32767  END

```

Patch to ATPK.BAS (V7.0-07B 08-JUL-80) line 16000:

```

*H/16000<tab>!</U<cr>
16000<tab>!<cr>
*H/DEF*/U<cr>
<tab>DEF* FNTD.LOGZ(T0$,T0%)<cr>
*AV<cr>
<tab>\ IF (NOT LOG.OFFZ) THEN <cr>
*1<cr>
<cr>
16003<tab>GOTO 16030 IF SUB.SWZ<tab>&<cr>
<tab>\ L.CPZ=INSTR(1%,T0$,"^C+CR.LF$)<tab>&<cr>
<tab>\ T0%=LEFT(T0$,L.CPZ-1%)+RIGHT(T0$,L.CPZ+4%) IF L.CPZ<tab>&<cr>
<tab><tab><tab><tab><tab><tab><tab><tab><tab><tab><cr>
<tab><tab>! CHECK IF SUB COMMAND ACTIVATED THE BASIC READER<tab>&<cr>
<tab><tab>! CHECK FOR CTRL C<tab><tab><tab><tab><tab><tab>&<cr>
<tab><tab>! REMOVE CTRL C SO ONLY ONE IS DISPLAYED<tab><tab>&<cr>
<cr>
<esc>*U<cr>
<tab>\ IF (NOT LOG.OFFZ) THEN <cr>
*2C/16007/U<cr>
16007 IF (NOT LOG.OFFZ) THEN <cr>
*EX<cr>

```

Checksum: 65395

Magnetic tape format: D05/ODD:800 BPI

Directory:

Name	.Ext	Size	Prot	Date	MM:[2,1]	(Copy #1)
RTCULL	.DOC	51	<155>	17-Sep-81		(Complete article)
ATPKED	.RND	9	<155>	17-Sep-81		(Section 1.0 thru 4.*)
ATPKPA	.DOC	2	<155>	17-Sep-81		(ATPK Patch)
ATPKED	.BAS	38	<155>	17-Sep-81		(ATPKED.BAS append code)

Total of 100 blocks in 4 files in MM:[2,1]

Name	.Ext	Size	Prot	Date	MM:[2,2]	(Copy #2)
RTCULL	.DOC	51	<155>	17-Sep-81		
ATPKED	.RND	9	<155>	17-Sep-81		
ATPKPA	.DOC	2	<155>	17-Sep-81		
ATPKED	.BAS	38	<155>	17-Sep-81		

Total of 100 blocks in 4 files in MM:[2,2]

Grand total of 200 blocks in 8 files in MM:[*,*]

WHY TAKE RISKS?

NPI proves purchasing Disk Sub-systems does not have to be

HIGH RISK VS BIG SAVINGS

- Disk drives manufactured by the same vendor DEC buys them from
- No operating systems software changes
- One day installation
- 100% software transparency
- DEC's RM05-RM03-RM02 drivers are used
- 50% cost savings
- Single vendor maintenance in major cities
- Uses DEC's diagnostics
- Timely delivery
- DEC RP06-RP04-RM03 RM02 accepted on trade

AT LAST!!!
 The RSTS PROFESSIONAL will be published six times a year. February, April, June, August, October, December

NPI National Peripherals, Inc.

41 Chestnut Lane Westmont, IL 60559 (312) 325-9700

CIRCLE 62 ON READER CARD

TIMER.BAS

By Michael H. Koplitz, Allis-Chalmers

```

10 EXTEND
20 *****
30 DIM JOB.NUMBER%(32%),PROJ%(32%),PROG%(32%),
   ENTRY.TIMES$(32%),REMOVE%(32%)
40 PRINT "TIMER V1.0 ALLIS-CHALMERS HTD ";DATES(0)
50 PRINT
60 INPUT "How many minutes to wait ";MINUTES.WAIT%
70 PRINT
80 PRINT "Timer will be detaching in job slot ";PEEK(518)/2
90 XS = SYS(CHRS$(6%)+CHRS$(7%))
100 *****INIT VALUES HERE*****
110 FOR X = 1 TO 32
120   JOB.NUMBER%(X),PROJ%(X),PROG%(X),REMOVE%(X) = 0
130   ENTRY.TIMES$(X) = ""
140   NEXT X
145   TABLE.COUNTER% = 0
   XS = SYS(CHRS$(6%)+CHRS$(3%))
   JOB.MAX% = ASCII(MID(XS,4%,1%))
150 *****VARIABLE TABLE*****
160 ON ERROR GOTO 32000
170 *****MAIN PROGRAM CODE STARTS HERE*****
180 FOR X% = 1% TO JOB.MAX%
185 GOTO 220 IF X% = PEEK(518)/2
190 XS = SYS(CHRS$(6%)+CHRS$(26%)+CHRS$(X%)+CHRS$(1%))
200 STATUS.JOB% = CVT$(MID(XS,21%,2%))
210 GOSUB 1000
220 NEXT X%
225 GOSUB 6000
230 FOR Y% = 1 TO TABLE.COUNTER%
240 GOTO 270 IF REMOVE%(Y%) = 0%
250 GOSUB 5000
260 Y% = Y% - 1%
270 NEXT Y%
280 SLEEP 3%
290 GOTO 170
300 *****END OF MAIN CODE*****
310 *****IMPORTANT NOTE: ONLY A ^C WILL REMOVE THE PROGRAM OUT OF ITS PROCESSING.
   A ^C TRAP IS UNNECESSARY BECAUSE THERE AREN'T ANY FILES OPEN. IT DOES
   NOT MATTER WHERE THE CODE FINISHES OUT. IF A ^C IS ENTERED WHILE IN
   THE MIDDLE OF A SYS CALL, THE SYS CALL WILL BE COMPETED BY THE FP
   BEFORE THE PROGRAM RETURNS TO ^C READY STATE.*****
1000 *****TABLE SEARCH AND JOB FORCE LOG OUT*****
1100 This routine will search the job table, IF the job is found in the
   table and it is not in ^C status (STATUS.JOB% >= 0) then the entry in
   the table will be flagged for deletion (REMOVE%(Z%) = 1). If the
   job is found and it is in ^C status the time will be examined to
   determine whether the job has been in ^C status for more than
   MINUTES.WAIT% minutes, if so send a message and BYEF. If the job
   is not found in the table and it is in ^C status, enter it into
   the table with the current time (TIMES(0)).
1200 Note that accounts [1,*] are skipped.
1010 Y$ = SYS(CHRS$(6%)+CHRS$(26%)+CHRS$(X%))
1020 TEST.PROJ% = ASCII(MID(Y$,22%,1%))
   GOTO 1230 IF TEST.PROJ% = 1%
   TEST.KB% = ASCII(MID(XS,4%,1%))
   GOTO 1230 IF TEST.PROJ% AND 128%
1030 TEST.PROG% = ASCII(MID(Y$,21%,1%))
1040 TEST.JOB.NUMBER% = ASCII(MID(Y$,3%,1%))/2%
1050 FOR Z% = 1% TO TABLE.COUNTER%
1060 GOTO 1080 IF TEST.PROJ% = PROJ%(Z%)
   AND TEST.PROG% = PROG%(Z%)
   AND TEST.JOB.NUMBER% = JOB.NUMBER%(Z%)
1070 GOTO 1190
1080 GOTO 1180 IF STATUS.JOB% >= 0
1090 HOURS = LEFT(ENTRY.TIMES$(Z%),2%)
   HOURS1$ = LEFT(TIMES$(0),2%)
1100 MINUTES$ = RIGHT(ENTRY.TIMES$(Z%),4%)
   MINUTES1$ = RIGHT(TIMES$(0),4%)
1110 TIME.CREATED% = VAL(HOURS$) * 60% + VAL(MINUTES$)
   TIME.NOW% = VAL(HOURS1$) * 60% + VAL(MINUTES1$)
1120 IF HOURS$ = "23" AND HOURS1$ = "00"
   THEN TIME.NOW% = TIME.NOW% + 60% * 24%
1130 TIME.CHANGE% = TIME.NOW% - TIME.CREATED%
1140 GOTO 1230 IF TIME.CHANGE% < MINUTES.WAIT%
1150 KB% = MID(Y$,4%,1%)

```


BLINK: A BASIC PLUS PREPROCESSOR

By Steve Holden, Manchester University, Dept. of Computer Science, Manchester, England

ABSTRACT

A preprocessor is described for use in a BASIC programming environment. The available directives, and some techniques which can be used to overcome weaknesses of BASIC PLUS, are presented. Experience with the preprocessor as a tool for producing commercial software is described.

INTRODUCTION

About eighteen months ago my company was charged with developing the software for a small PDP-11 bureau, using the RSTS/E operating system.

With an 11/34, and only a small development team, we had to come up with a way of increasing the productivity of our programmers. We decided to base our development effort on BASIC PLUS, which offers quite fast program development and a relatively easy upgrade to the compiled language BASIC PLUS 2. This could later be used to make more effective use of system resources.

The major software tool has been a preprocessor, which accepts a superset of BASIC PLUS.

What BLINK Does

The BLINK preprocessor reads a source file which looks like BASIC PLUS with certain things added (BLINK directive lines, labels, set symbols), and certain others left out (line numbers).

From this source file BLINK produces a BASIC PLUS file which can be loaded and compiled in the usual way.

Reasons for Development

BLINK is a response to some of the problems of programming in BASIC PLUS. Whilst the base language is adequate for expressing algorithms, there are few facilities which encourage the programmer to develop suites of programs in a neatly modular way.

The BLINK project was started because our standards, based on those published by DECUS, called for certain types of code to be allocated line numbers within certain ranges. We found it difficult to move routines from applications libraries into our standard libraries, which both started out as APPENDable files.

An obvious solution was to get the computer to enforce the standards for us. BLINK allows us to delegate to the system those tasks which we find least tasteful and most time-consuming.

The BLINK Language

BLINK is mostly BASIC PLUS. Indeed, as an aid to migration we have ensured that BASIC PLUS will pass through the preprocessor without change.

A line of BLINK may be in one of six types.

Numbered Lines — The line number is used to set BLINK's program counter, so that subsequently generated line numbers will be higher in sequence and thus follow on in the usual way.

Labelled Lines — A label is recognised when a normal extend-mode identifier begins in column one. It must be followed by a colon, which makes it look much like a label in more conventional programming languages.

The label is entered into the label table, a line number is allocated to it by incrementing the program counter, and a new line of output code is commenced.

Comments — To encourage liberal use of comment as a documentation aid, any line beginning with an exclamation mark in column one is considered to be a comment, and completely ignored by the BLINK preprocessor.

Directives — All lines beginning with a period are considered to be requests for the preprocessor to take some special action. The various directives are discussed at some length below.

Code Lines — Any line which starts with a blank or a tab is considered to be code. Such lines are passed through to the output stream after certain symbol substitutions have been made.

Blank Lines — These are not considered significant when they precede a directive line, allowing the conscientious programmer to improve the readability of his code. For simplicity of processing, we assume that all code continuation is indicated by ampersands as in BASIC PLUS 2 and the extend-mode convention.

A blank line thus triggers incrementation of the program counter and the start of a new output statement. They are often used to delimit the scope of a multi-statement THEN or ELSE group.

DILOG NUMBER 1 FOR DEC 11*

With LSI 11/PDP 11 Software Compatible Disc/Tape Controllers Offering Single Board Low Power μ P Based Design and Low Cost... Plus Many Other Good Reasons!

The reasons start with DILOG'S (Distributed Logic Corp's.) full time engineering and design staff. *Not outside suppliers.* That means when you contact DILOG for product selection or after sale service, you'll get "first hand" assistance... along with years of experience manufacturing μ P based controllers that interface with DEC 11 CPUs.

The intelligent products you'll discuss all utilize common proprietary architecture and DILOG automated design techniques—products with exceptional reliability and cost efficiency... mostly available from stock. And

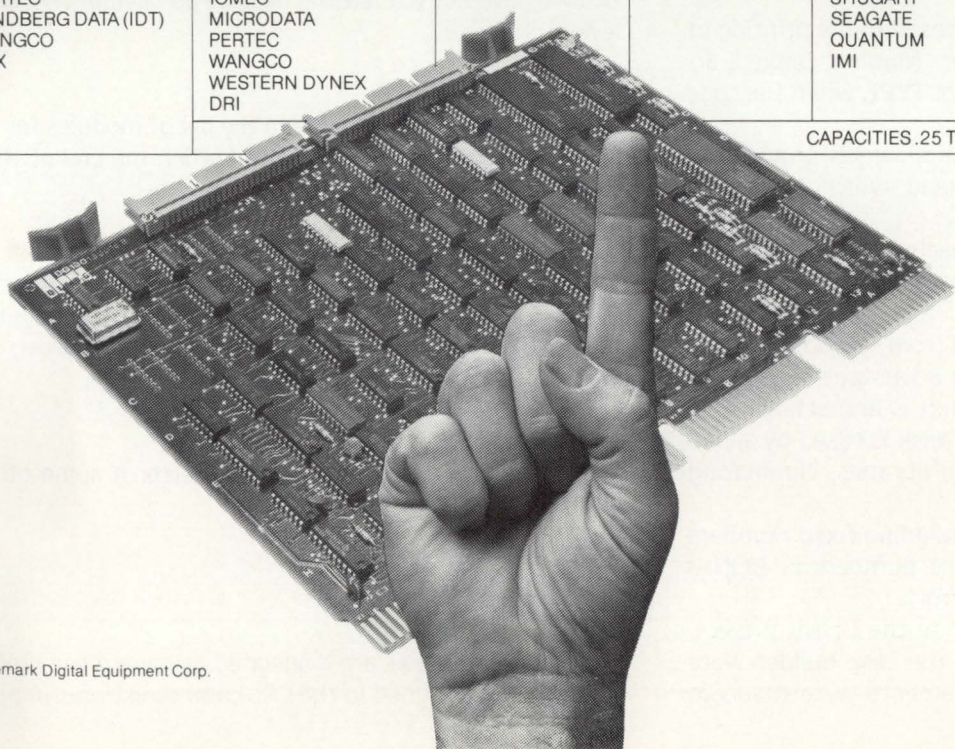
when you plug a DILOG controller into your DEC CPU it's ready-to-run because it's *fully operating system software compatible.*

These high performance data storage interface products also feature • minimum bus/space requirements • up to 60% less power • 10 to 50% lower cost • automatic self-test... and numerous other features for easy system integration.

Consult the DILOG/disc-tape compatibility table for your needs. Then ask for detailed data on existing, or future products from DILOG... #1 in single board DEC 11 compatible disc/tape controllers. Distributed Logic Corp., 12800-G Garden Grove Blvd., Garden Grove, CA 92643, Phone: (714) 534-8950 • 64-A White Street, Red Bank, New Jersey 07701 Phone: (201) 530-0044

DISC/TAPE DRIVE MANUFACTURER COMPATIBILITY CHART

MAGNETIC TAPE		DISC				
1/2" REEL-TO-REEL STD. & STREAMER	2315/5440/RK05 CARTRIDGE CLASS	CMD CARTRIDGE MODULE	SMD STORAGE MODULE	WINCHESTER 5 1/4", 8" OR 14"	1/4" TAPE CARTRIDGE	FLOPPY DISC DRIVE
AMPEX CIPHER CONTROL DATA DIGI-DATA KENNEDY MICRODATA PERTEC TANDBERG DATA (IDT) WANGCO TDX	AMPEX CAELUS CENTURY DATA CONTROL DATA DEC DIABLO IOMEC MICRODATA PERTEC WANGCO WESTERN DYNEX DRI	AMPEX CONTROL DATA	AMPEX CENTURY DATA CONTROL DATA BALL COMPUTER MITSUBISHI	BASF CONTROL DATA FUJITSU KENNEDY MEMOREX PRIAM SHUGART SEAGATE QUANTUM IMI	DEI KENNEDY PEREX QUANTEX	BASF DECITEK MICROPOLIS PERTEC REMEX SHUGART
CAPACITIES .25 TO 300 MB						



**DISTRIBUTED
LOGIC CORP.
DILOG**

*Trademark Digital Equipment Corp.

BLINK SYMBOLS

The preprocessor handles two types of symbols not normally a part of the BASIC PLUS language.

Labels are substitute for line numbers, which we feel 'harden' BASIC PLUS code, and make it less likely that a piece of code will be usable elsewhere.

A reference to a label in the code is preceded by the commercial at sign (@) because this makes them easy to detect by the preprocessor and also flags them to the eye of the human reading the code.

It is interesting to note that the "at" sign is one of the few with no syntactic significance to the BASIC PLUS run-time system. It was chosen in preference to the others, after several false starts, largely because it is available without use of the shift key on most of our locally available keyboards.

The second type of symbol is the so-called 'set symbol,' which may hold an arbitrary string. Several of the BLINK directives are concerned with setting and testing the value of these symbols.

They can thus be used

- to control the inclusion or exclusion of a given piece of code from the source file, allowing systems to be configured by feature selection.
- to generate different literal strings or variable names during different preprocessing runs, allowing the same source to produce code which is tailored to the circumstances.
- to modify the actions of library routines, or the code that such routines generate, giving a more flexible applications library without undue duplication of code.

The value of a set symbol is obtained in the program text by flagging the reference in the way shown below.

```
PRINT "@/TYPE/Ledger"
```

The above code fragment could result in the printing of "Sales Ledger", "Purchase Ledger" or "Nominal Ledger" according to the value of the set symbol TYPE when the code is processed.

To assist in such configuration a means is provided of setting symbols in the CCL command which triggers a BLINK preprocessing run.

It is interesting to note the parallels between programming in BLINK and programming in an assembly language followed by link-loading.

In both cases, the code produced from symbolic sources is absolute, although BASIC has the advantage that insertion of new code is much easier than it is in machine code.

In both cases the programmer's task is eased by allowing him to make use of pre-written libraries, eliminating duplication of effort.

Furthermore, it is the task of assigning fixed numbers (addresses in assembly language, line numbers in BLINK) which prompted the development effort.

This parallel was the main spur to the BLINK project, and the facilities of MACRO-11 and the task builder were used a model when various enhancements were made to

BLINK. It is a sad reflection on the state of commercial programming that many of the features for long taken for granted remain unavailable in one of the most popular languages. Naturally the introduction of BASIC PLUS 2 has changed the situation somewhat, but we feel that even in a PLUS 2 programming environment many of the configuration abilities of BLINK would continue to be useful.

BLINK DIRECTIVES

Apart from the symbol substitution features mentioned above, BLINK enriches the base language by processing directive lines.

A BLINK directive is flagged with a period in column one, followed by a keyword to specify the required action, and optionally a list of arguments to modify the action.

There are several types of directive, and they are split for the purposes of discussion into several different classes.

Code Sharing Directives

One of the best ways to improve programmer productivity is to make it easier to use someone else's code than to write a routine yourself.

BLINK has facilities to allow a section of code from another file to be inserted in line, which is similar to the action of an APPEND command. It also has library searching facilities, which help to reduce the number of files on the disk, and reduces file processing overhead during preprocessing.

.INCLUDE file

takes the text contained in the named file (with default of SY:(SELF).SRC) and processes it as though it replaced the INCLUDE directive in the source program. INCLUDE directives may be nested up to a depth of nine, although in practice we have not yet felt the need to go beyond two or three levels.

.LIBRARY module

adds the given module name to the list of modules for which library search must be performed when the end of the top-level source file is encountered.

.SEARCH file

adds the named file (with default SY:(SELF).LIB) to the list of files to be searched for requested modules during library processing. The libraries are searched in the order of the SEARCH directives met during the source scan.

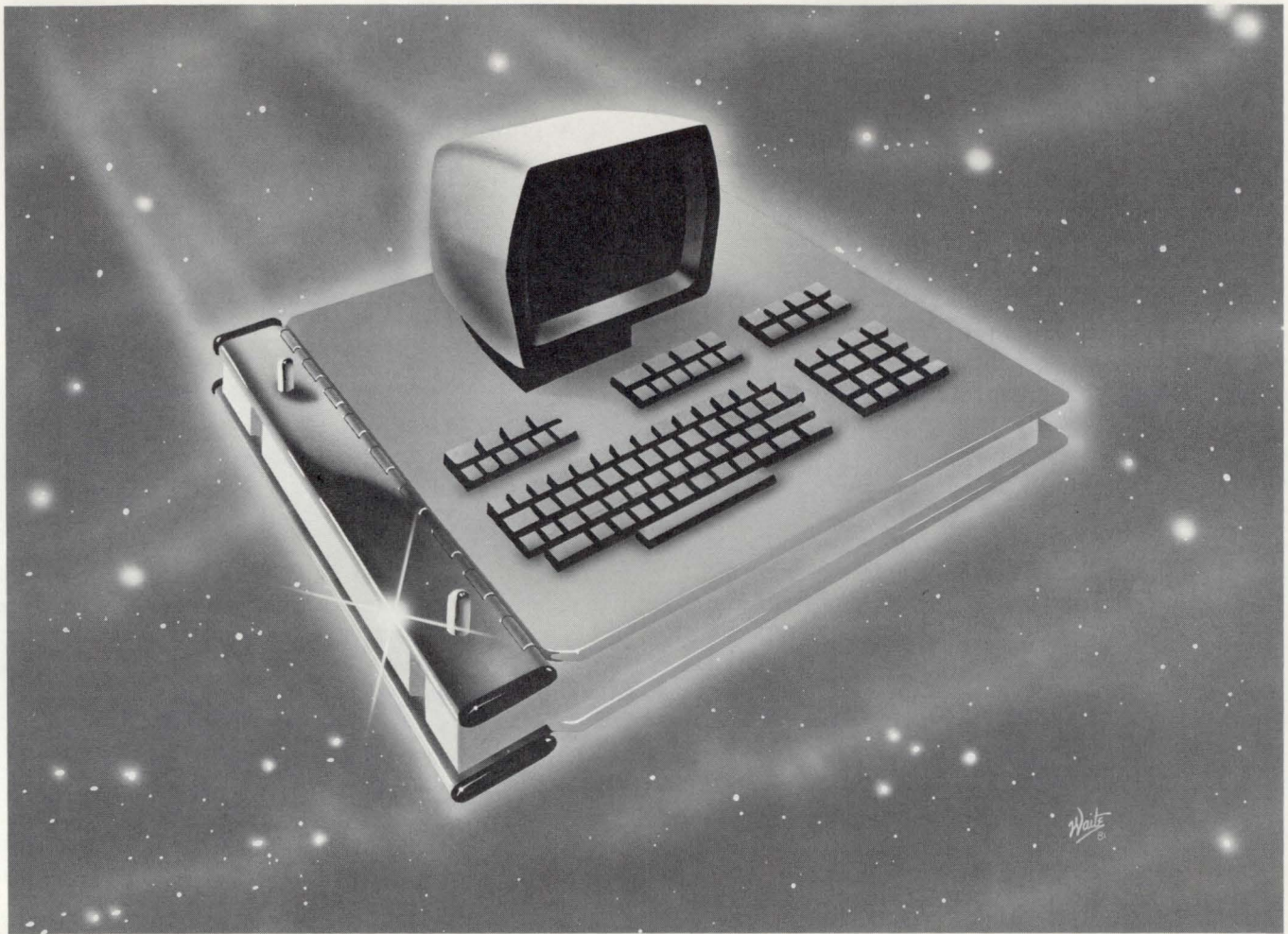
.NEEDS module

allows a library module to cause an error if some other required module was not requested.

Symbol Setting Directives

The set symbols are translated as having the null value if no value is assigned in the CCL command or during source

Pascal POWER for your RSTS Machine



LEAP INTO THE 80's with Accounting Software in Pascal. The TBS accounting systems, in use internationally for the past 2 years, are now available for the RSTS user. The systems are designed to fulfill your accounting needs for the years to come, using the speed and flexibility of RSTS and Pascal.

FIVE INTEGRATED FINANCIAL PACKAGES: A/R, G/L, A/P, Payroll, and Order Entry with Billing, Inventory Control, and Sales Analysis.

- INTERACTIVE
- EASY-TO-USE
- COMPREHENSIVE
- WELL DOCUMENTED
- MENU-DRIVEN
- VERSATILE
- SOURCE CODE AVAILABLE
- EXTENSIVE REPORTING CAPABILITIES
- CUSTOMIZATION SERVICES
- HIGHLY PARAMETERIZED FOR RAPID MODIFICATIONS

PASCAL DEVELOPMENT SYSTEM: A complete multi-user ISAM-type file handler and an extensive package of external procedures and utilities, designed to greatly reduce software development time. Standard maintenance and print programs can be created in less than an hour.

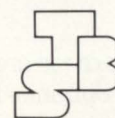
Pascal programs are easily maintained, updated, and modified. The systems are also available on RT-11, TSX-Plus, and RSX-11M. A UNIX and VAX-VMS version will be available soon. For pricing information, product descriptions, or demos, contact Theta Business Systems or our U.S. Distributor, Secteur Corporation.



SECTEUR CORPORATION

BAINBRIDGE PROFESSIONAL BLDG., SUITE 103
BAINBRIDGE ISLAND, WA 98110
(206) 842-5612

CIRCLE 63 ON READER CARD



THETA BUSINESS SYSTEMS

1110 SONORA AVENUE, SUITE 106
GLENDALE, CA 91201
(213) 242-7981 or 245-0917

processing. Thus a reference to an undefined set symbol is not considered to be an error.

.SET symbol = value

assigns the value on the right of the equals sign to the symbol named on the left. The assignment is unconditional.

.DSET symbol = value

acts like a SET, except that the assignment is only made if the symbol currently has the null value. This allows an INCLUDED or library text to assert default values for its generation parameters.

.ASK symbol

prompts for the value from the terminal if the symbol currently has the null value. This is a useful directive when generating a highly-configured piece of software, when there are two options open to the programmer.

He may generate the file as it stands, when he will be prompted at the terminal for the system generation parameters.

Alternatively, he may write a module which SETs some or all of the system generation parameters and then INCLUDEs the source file. Because of the null action when the symbol is already defined, he will then not be prompted for the parameters he has set when he generates this higher-level module.

Conditional Assembly Directives

When writing turnkey software it is quite often desirable to include or exclude certain sections of code according to whether particular features are required (and will be paid for!).

BLINK has facilities to optionally skip sections of the source code, depending on the value of one or more arguments to the directives.

.ENDIF

delimits the scope of all conditional assembly directives given below. If conditional skipping is not taking place then it has no action.

.IFB string

.IFNB string

assemble the lines up to the next ENDIF if the remainder of the directive line is (or is not) blank. These directives are most often used to test the value of a set symbol, as in

```
.IFNB @/TEST/
```

where the following code is assembled only if the set symbol TEST has a non-null value.

.IFEQ string1,string2

.IFNE string1,string2

continue processing if the two strings are (or are not) identical. This is useful where one or more options exist during system generation, as in

```
.IFEQ @/TYPE/,P
```

(Purchase ledger code)

```
.ENDIF
```

```
.IFEQ @/TYPE/,S
```

(Sales ledger code)

```
.ENDIF
```

At present there are no facilities to nest the conditional directives, although this is an obvious extension to the current language. Since an ENDIF has no action if skipping is not taking place, however, as long as the scope of two conditionals ends at the same place such an action can be simulated in a way which will be upwards-compatible with nesting when it is implemented. We can write after the above code

```
.IFNE @/TYPE/,P
```

```
.IFNE @/TYPE/,S
```

```
.PRINT ?Invalid ledger type @/TYPE/
```

```
.ENDIF
```

```
.ENDIF
```

Since the first ENDIF delimits the scope of both conditionals, skipping can never be taking place when the second one is encountered, and it will be ignored. When nesting is implemented, however, it will delimit the scope of the second IFNE directive.

Program Section Directive

Many installations, ours included, have adopted standards which call for restrictions on the line numbers which may be used for certain types of code.

It is common, for example, to put array dimensions in at 900-998, initialisation code at lines 1000-1999, and so on.

Whilst BLINK has to a large degree removed the need to consider line numbers, it is still useful to know roughly whereabouts in the program a piece of code will be found — during interactive debugging, for example.

Furthermore, it often happens that a single source module must ensure that certain variables are preset during initialisation, or that a particular error at a certain line is catered for in the error handler.

This is catered for by splitting the range of permissible line numbers into named program sections, or PSECTs, which will be a familiar concept to the MACRO-11 programmer. Each PSECT has its own program counter, and so BLINK will resume output at the next available line number within that section when it is resumed.

```
.PSECT {section-name} {,LD:n} 1,HZ:m
```

is the directive which causes the preprocessor to switch between different sections. We will deal with the most complex case first, where all arguments are specified.

```
.PSECT INIT,LD:1000,HI:1998
```

declares a program section, with limits on the line numbers which may be generated from within it. If a PSECT named

Do you own a PDP-11?

Do you need to move files between PDP-11's?

Link command sets up communications port (in this case TT12)

File transfer commands, where: output file = input file (note \$ or £ sign indicates remote computer)

STS command requests status of file transfer

NOTE - the command control/Z releases the terminal to run other jobs. File transfers then continue as background tasks. IPL-11 can be re-entered at any time by typing IPL. An END command causes IPL-11 to exit on completion of transfers.

```

IPL>LINK TT12
IPL>
LINK ESTABLISHED TO X800011
↑R
IPL>DK2:DAYFILE.LST=£LOGFILE.DAT
IPL>DK2:LONGLAS.MAP=£LONGLAS.MAP
IPL>£IPL800011.TSK=DK2:[100,14]IPL800011.TSK
IPL>STS
*** TRANSFERS IN PROGRESS ***
£IPL800011.TSK=DK2:[100,14]IPL800011.TSK
...FILE SIZE: 48. CURRENT BLOCK: 4.
DK2:DAYFILE.LST=£LOGFILE.DAT
...FILE SIZE: 13. CURRENT BLOCK: 12.
*** TRANSFERS PENDING ***
DK2:LONGLAS.MAP=£LONGLAS.MAP
IPL>
COMPLETED DK2:DAYFILE.LST=£LOGFILE.DAT
↑R
IPL>_
    
```

When the package is run it prompts with IPL

Indicates link to remote computer established

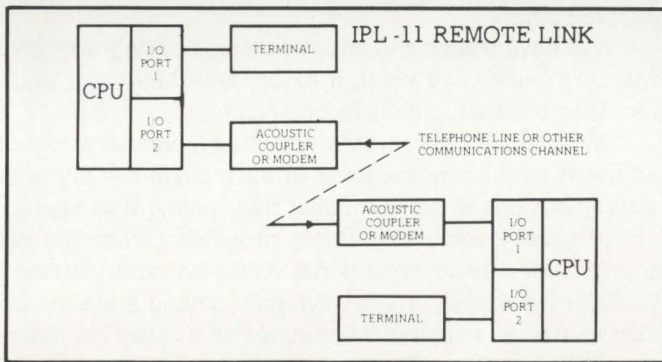
Indicates status of file transfers in response to STS command

Indicates this file transfer is complete

XOREN IPL-11

The simple way to transfer PDP-11 files

XOREN IPL-11 is a software package which enables files to be transferred over a communications link between two DEC computers each of which may be a PDP-11, an LSI-11 or a VAX-11. Using the package, files can be transferred over a telephone line or a direct line.



- INTERFACE HARDWARE -

No special interface hardware is required other than (in the case of remote computers) modems or acoustic couplers. Package operates via standard DEC terminal interface cards—DL11, DZ11, DH11, etc.

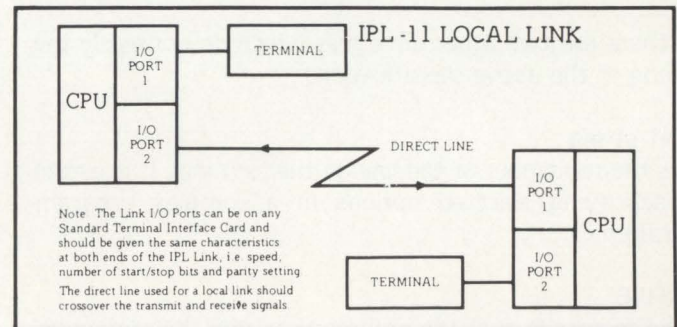
- DATA INTEGRITY -

CRC checking by software to CCITT recommendation V41. Re-transmission of bad blocks is used for error recovery.

- TRANSMISSION -
Asynchronous transmission with selectable speeds up to 9600 baud.

- OPERATING SYSTEMS -
Versions of this package are available now to run under RSX-11M, RT-11, RSTS/E, (and equivalent versions of CTS-300 and CTS-500), IAS and VAX/VMS in RSX-11M compatibility mode.

- LICENCE -
The package is normally supplied under a 5-year licence. A separate licence is required for each combination of CPU and operating system under which IPL-11 is to run. The two (5-year) licences required to link two CPU's cost \$1350.00 each, whichever pair of operating systems is specified. For larger orders a system of discounts is applied.



Note:- Xoren Computing is currently setting up a distributor network for XOREN IPL-11 in the US. For further information contact Xoren Computing Ltd direct.

Xoren Computing Ltd.
28 Maddox Street,
London W1R 9PF
England
Telephone (01) 629 5932

Specialists in:-
Minicomputer Software
Microcomputer Software
Data Communications
Interprocessor Links

INIT already exists then this attempt to specify new limits is considered an error.

Most often, this form is used in a module which is INCLUDED by a programmer almost at the beginning of his source code. If he does this, the PSECTS will be set up for him, and he does not need to bother about establishing his own PSECTS.

.PSECT INIT

causes the preprocessor to switch back into the INIT section, resuming code output where it left off when it was switched out. Thus, since BLINK normally generates code with increments of ten between line numbers, if the last line generated in the INIT section was 1050, it will recommence output at line 1060 (unless a line number appears to reset the program counter).

Lastly, the PSECT directive may be used without any arguments. This is principally convenient for INCLUDED and library modules, as it switches into the PSECT which was in force when the module text was commenced.

One particularly useful application of the PSECT directive has been in error handling. Our standard program skeletons all set up a PSECT named ERRORS, and included initialisation code which traps errors to the start of this section.

Any INCLUDED module can specify private error trapping with the sequence

.PSECT ERRORS

```
IF ERL = @LAB% AND ERR = 99%      &  
THEN RESUME @LAB2                 &
```

.PSECT

This obviates the necessity for routines to specify a private error trap, which we have found to be a fruitful source of errors in the past. Note also that, since label references are translated into line numbers by BLINK, it is legal to use them to specify an integer error line number.

Miscellaneous Directives

There are two other directives which do not easily fall into one of the above classifications.

.PRINT string

prints the remainder of the line on the terminal. It is useful for identifying selected options in a complex program generation.

.ENDFILE

is used to specify that the preprocessor skip the remainder of the file or library module it is currently processing. This is rarely used (although BLINK generates on internally when it actually encounters the end of any source file!), but can reduce processing time if an error is detected in the generation parameters.

PROGRAMMING TECHNIQUES

It is difficult to present the features of the BLINK language in a coherent way given the space limitations imposed on this paper. There are many techniques which we have found lead to increased productivity by sharing code or reducing debugging time. In particular, we feel that the major advantages of the BLINK preprocessor are:

- A standard program skeleton can be written for each type of program (file create, report, et cetera), and these skeletons can be used by programmers with insufficient experience to define a sensible program structure for themselves. Use of these skeletons allows one programmer to understand another's code much more readily.
- The standard library, with disk handling routines and echo-control terminal input functions are available by writing a single directive for each module.
- When an error is detected in a standard module, it only need be corrected once.
- Since the standard modules are effectively hidden at source level, program listings are both smaller and more comprehensible. Also, we are encouraged to treat the standard routines as 'black boxes', and use only the interface descriptions rather than reading the code.
- Because BLINK encourages a modular approach to programming, programmers tend to structure their own code in a sensible way.

Without exaggeration, we claim that programming in BLINK is more enjoyable — although this is obviously subjective opinion.

We have traded the ability to develop programs on-line for the flexibility of using a higher-level language, and we feel that the exchange is in our favour.

We can write programs which have virtual array sizes adjusted to the requirements of each customer (by setting the dimensions in set symbols), thus saving disk space.

We can generate multiple programs from the same source code, using conditional assembly and putting INCLUDED file names into set symbols, saving a maintenance load as well as keeping the number of source files down to more manageable proportions.

USER EXPERIENCE

BLINK has been used to produce several fairly complex sets of applications programs, now all in use on a timeshared PDP-11/34 in parallel with further program development.

Sales and Purchase Ledgers

This comprises about twenty programs, plus some associated file maintenance utilities. With the exception of

\$895⁰⁰

MENU THE MISSING LINK.

MENU is an applications development aid that provides the DEC RSTS software developer with a powerful process control device.

MENU easily generates menus that guide the application user into selected programs based on his security level.

MENU supplements RSTS security by allowing multi-level access capabilities

within an individual account. Separate project-level control minimizes System Manager interaction for system level security.

MENU is driven by simple text files which determine extent of program control, type and level of security, screen displays, and presentation of on-line '/HELP' information.

MENU provides a separate Run Time System to prevent unauthorized access to RSTS 'ready state' resources.

MENU installs in minutes and requires no software modifications.

MENU provides a common interface for all your users and application needs.

For information on **MENU**, please give us a call. We would be delighted to show you the missing link.



**North County
Computer Services, Inc.**
2235 Meyers Ave.
Escondido, California 92025
(714) 745-6006, Telex: 182773

DEC and RSTS are registered trademarks of Digital Equipment Corporation.

© Copyright NCCS

CIRCLE 38 ON READER CARD

**60 DAY TRIAL
FOR \$50⁰⁰**

programs concerned with producing cheques and remittance advices (unnecessary in a sales ledger), all programs are produced in two versions from a common source.

It is hard to estimate the amount of effort this has saved, but it has definitely cut the load of maintaining the system, which is in daily use by up to eight bureau clients.

Magnetic Tape Archiving System

The principal feature of this system is an index, held in virtual arrays, specifying which files were dumped to which tapes on which dates. It would obviously be wasteful to make the file big enough to cope with a thousand files when only a couple of hundred were to be archived.

A source file specifying the size of all these arrays is INCLUDED in each program, allowing a simple systems generation procedure for turnkey clients requiring the package. At the same time they are customised as to client name.

Contract Costing System

This system was the first to provide integration between the sales and purchase ledgers, and a payroll system written outside our company in straight BASIC PLUS.

As purchase, sales and payroll details are entered they are also used to update a cost ledger maintained for all the client's contracts.

This application has its own source libraries, which contain routines to access hash-indexed files. They proved to be so successful that a more general version of them is shortly to be incorporated into the systems library.

General Libraries

These libraries are used by all applications suites. When turnkey packages are required for an outside customer it is a simple matter, for example, to update the routines concerned with visual display addressing and output and regenerate the programs to the new requirement.

Applications Libraries

As an example of the versatility of these library modules, should a turnkey customer require tight formatting on customer account references, only one library module need be changed. This

update would automatically be incorporated into all programs as they were generated.

Some Problems

The major current problem is that of sharing the development machine with an interactive bureau service. BLINK is written in BLINK, and can place quite a heavy load on a PDP-11/34 with storage module drives. Fortunately, the major development effort on the ledger packages is now over, and amendments can usually be made by editing the sources during the day and submitting an overnight batch job to regenerate the suite.

We feel, nevertheless, that BLINK could be usefully rewritten in some compiled language, although no statistics are available to indicate whether the generation process is disk- or CPU-bound.

BASIC PLUS 2 is the obvious candidate, since the generation process will work just as well for that as for BASIC PLUS. String handling still imposes quite a heavy overhead in PLUS 2, however, and for this reason we feel that MACRO-11 or PASCAL might give better performance.

At the same time, it would be useful to incorporate at least some lexical and syntactic analysis of the program. This would allow the preprocessor to decide for itself which functions had no definitions in the source programs, and should therefore be sought in the libraries.

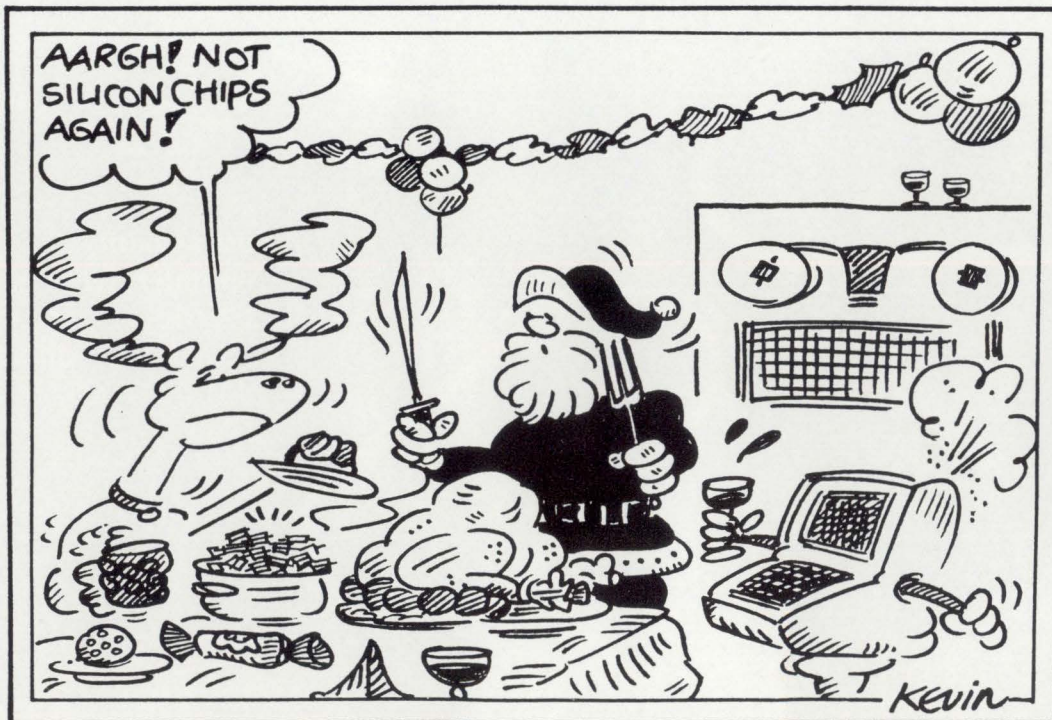
It is somewhat annoying to go through the whole process of program generation only to be informed by the BASIC run-time system of syntax errors or undefined function calls introduced by the last edit.

Eventually it may be possible to produce a true compiler for BLINK, which output object modules for processing by TKB or LINK. With the increased usability of VAX BASIC and BASIC PLUS 2, however, the major advantage of this would

be portability of applications software.

In this respect, a compiler which output code to run under RT11 would be especially useful in allowing migration of applications packages down to the LSI-11 based systems.

With proper design it should even be possible to transfer the language to completely different series of machines, as



Cartoon by Kevin Macey, Submitted by Peter Dick, Silver Programs, London

\$REORDR — Sorting Alphabetically

By George T.A. May, Special Systems Division, Software Sciences Limited, UK

RSTS professionals may be interested in this slight variation to \$REORDR which allows it additionally to sort the directory in alpha-
betic order as well as the normal date and time sorts. This can be use-
ful if you want to look for a particular filename in a directory. I have
found it helpful for program development directories where one
tends to have a mass of files and date order is not so interesting, but

names are. The following source file \$REORDR.BAC < 124 > or
whatever it is called in a particular system.

I've attached an installation process and demonstration run. You've
got to follow the installation process else RSTS Basic Plus routine
system screws you up because I've redefined function FNI%.

INSTALLATION

```
Ready
OLD [1,200]REORDR.OLD
Ready
DELETE 4030,15210
Ready
REPLACE REORDR.TMP
Ready
OLD REORDR.TMP
Ready
APPEND REORDR.APP
Ready
COMPILE
Ready
PIP [1,140]/LI
```

```
HKI010.BAK 33 < 48> 17-Jun-81
HKI010.CBL 33 < 48> 17-Jun-81
HKI010.HLP 2 < 60> 19-May-81
HKI010.LST 87 < 60> 16-Jun-81
HKI010.OLD 76C <124> 05-Jun-81
HKI010.V01 32 < 48> 17-Jun-81
IMV .CNU 2 < 60> 17-Jun-81
INV1 .CNU 2 < 60> 17-Jun-81
IV050 .BAK 156 < 60> 16-Jun-81
IV050 .CBL 156 < 60> 17-Jun-81
POL .CNU 2 < 60> 17-Jun-81
POLL .CNU 2 < 60> 17-Jun-81
PRO30 .BAK 34 < 60> 05-May-81
PRO30 .CBL 35 < 60> 17-Jun-81
PRO30 .OLD 81C <124> 05-May-81
STD01 .BAK 24 < 60> 13-May-81
STD01 .CBL 24 < 60> 17-Jun-81
STD01 .OBJ 23 < 40> 13-May-81
STD01 .SKL 1 < 60> 13-May-81
TEMP49.TMP 36 < 60> 17-Jun-81
UKBF02.DAT 163 < 60> 14-May-81
```

Total of 1076 blocks in 24 files in SY:[1,140]

DEMONSTRATION

```
Name .Ext Size Prot Date SY:[1,140]
COMMON.MAC 31 < 40> 30-Apr-80
HKI010.HLP 2 < 60> 19-May-81
DWAM .BAC 31C <124> 10-Jun-81
ERMB0.MAC 10 < 60> 12-Jun-81
HKI010.OLD 76C <124> 05-Jun-81
HKI010.LST 87 < 60> 16-Jun-81
IV050 .BAK 156 < 60> 16-Jun-81
UKBF02.DAT 163 < 60> 14-May-81
IV050 .CBL 156 < 60> 17-Jun-81
POL .CNU 2 < 60> 17-Jun-81
IMV .CNU 2 < 60> 17-Jun-81
HKI010.V01 32 < 48> 17-Jun-81
STD01 .OBJ 23 < 40> 13-May-81
STD01 .BAK 24 < 60> 13-May-81
STD01 .CBL 24 < 60> 17-Jun-81
STD01 .SKL 1 < 60> 13-May-81
PRO30 .OLD 81C <124> 05-May-81
PRO30 .BAK 34 < 60> 05-May-81
PRO30 .CBL 35 < 60> 17-Jun-81
POLL .CNU 2 < 60> 17-Jun-81
INV1 .CNU 2 < 60> 17-Jun-81
HKI010.BAK 33 < 48> 17-Jun-81
HKI010.CBL 33 < 48> 17-Jun-81
TEMP49.TMP 36 < 60> 17-Jun-81
```

Total of 1076 blocks in 24 files in SY:[1,140]

```
Ready
RUN REORDR
REORDR V7.0-08 RSTS V7.0-07 HR-AUDIT
Directory Reordering Program
Sort Directory(s) (YES/NO) <NO>? Y
Order by CREation Date/Time, ACCess Date, or ALPhabetically <CRE>? ALP
In FORWARD or REVERSE? Order<FOR>? FOR
Device and UFD Specification(s)? DR0:[1,140]
Directory _DR0:[1,140] has been Reordered
Sort Directory(s) (YES/NO) <NO>? ^C
Ready
PIP [1,140]/LI
```

```
Name .Ext Size Prot Date SY:[1,140]
COMMON.MAC 31 < 40> 30-Apr-80
DWAM .BAC 31C <124> 10-Jun-81
ERMB0.MAC 10 < 60> 12-Jun-81
```

```
Ready
BYEF
```

```
10 EXTEND
21 1 VERVED EDIT DATE REASON
1 7.0-08 15-JUN-81 ORDER ALPHABETICALLY
1 &
410 1 AX INDEX INTO NEXT AVAILABLE TARGET UFD ENTRY &
1 AOX POINTER TO FIRST BLOCK BEYOND LAST NAME ENTRY &
1 BX NUMBER OF BLOCKS IN UFD &
1 CX UFD CLUSTERSIZE &
1 CS USER ENTERED INPUT &
1 CSX = 1 IF REORDER BY ACCESS DATE &
= 0 IF REORDER BY CREATION DATE/TIME &
= 1 IF SORTING ALPHABETICALLY &
1 DS UFD FILE SPECIFICATION OF UFD CURRENTLY BEING &
PROCESSED &
1 DX DUMMY VARIABLE FOR INTERNAL FUNCTIONS &
1 DX() THE UFD TO BE REORDERED SET UP AS A WORD ARRAY &
1 FX() FREE ENTRY COUNTER FOR EACH DIRECTORY BLOCK &
1 FOX WORKING VARIABLE &
1 FIX WORKING VARIABLE &
1 IX VERSION/EDIT #'S TEXT STRING &
1 IX() ARRAY USED TO HOLD RETURN FROM FILE STRING SCAN &
OF USER ENTERED DEVICE/UFD SPECIFICATION(S)
600 1 SX( , ) ARRAY FOR SORTING NAMES BY ACCESS OR CREATION DATA &
COUNT OF NUMBER OF ATTRIBUTE, ACCOUNTING AND &
RETRIEVAL ENTRIES FOR CURRENT FILE BEING REORDERED. &
= 0X IF NO SORT BY ACCESS DATA DESIRED &
= 1X IF SORT BY ACCESS DATA DESIRED &
= FIRST SORT PARAMETER TO FNIX - REPLACES DX &
= SECOND SORT PARAMETER TO FNIX - REPLACES IX &
= THIRD SORT PARAMETER TO FNIX - NEW &
1 IX WORKING VARIABLE &
1 IX() TARGET UFD SET UP AS A WORD ARRAY &
1 TOX WORKING VARIABLE USED IN MOVING UFD CLUSTERMAPS &
1 TIX POINTER TO NAME ENTRY, USED IN CREATION/ACCESS SORT &
1 T2X POINTER TO ACCOUNTING ENTRY, USED IN CREATION/ACCESS &
SORT. &
1 WS FILE SPEC FOR UFD TARGET ARRAY, &
1 WX WORKING VARIABLE &
1 WIX WORKING VARIABLE &
1 ZX WORKING VARIABLE &
852 1 FNSAMP( ) 17000 WHEN SORTING ALPHABETICALLY, RAPS CREATES &
A FUNNY ORDER FOR THE NUMBERS (IE +VE &
NUMBERS OCCUR FOR HIGH (NOT LOW) LETTER &
VALUES. FNSAMP REORDERS THEM PROPERLY &
BY ADDING 32768 TO +VE VALUES AND SUBTRACT &
-ING 32768 FROM +VE VALUES &
899 1 FNIX(IX,DX,IX) 15200 GIVEN DIRECTORY LINK, DATE AND TIME, ADD TO &
NAME ENTRY SORT LIST ARRAY IN SORTED ORDER &
1 FNOX( ) 16000 CHECK TO SEE IF THIS IS A LARGE FILE SYSTEM &
AND IF SO, IF ANY FILES ARE OPEN ON THE &
ACCOUNT TO BE REORDERED. &
1 IS = "V7.0-08"
1 SET UP VERSION/EDIT #'S FOR HEADER
1205 1 PRINT
1 INPUT #31, "ORDER BY CREATION DATE/TIME, ACCESS DATE, OR ALPHABETICALLY <CRE>? "ICSR
1 GET USER SELECTION
1210 1 GOTO 1210 IF LEN(CS) = 0X
1 CS = LEFT(CV(FS(CS,32)),31)
1 GOTO 1220 IF CS = "CR"
1 GOTO 1205 IF CS <> "ACC" AND CS <> "ALP"
1 CSX = 1X IF CS = "ALP"
1 CSX = 1X IF CS = "ALP"
1 PRESSET TO DEFAULT VALUE, DONE IF DEFAULT INPUT, ELSE,
1 GET FIRST THREE CHARACTERS IN UPPER-CASE ONLY, DONE IF
1 "CRE", GET AGAIN IF NOT "ACC" OR "ALP", OTHERWISE,
1 SET TO ACCESS DATE SETTING
4005 1 PX = 0X
1 SX(IX,IX) = 0X
1 NULENTX = 0X
1 NULENTX = -32767X-1X IF CSX = 1X
1 SX(IX,IX) = NULENTX
```


TIPS & TECHNIQUES

A Column For The Advanced RSTS/E User

By Steven L. Edwards, Software Techniques

In this issue, we will look at a patch to add form names to the one-shot spooling sys-call, the time differences between Basic-Plus and Basic-Plus-2 (or CSPCOM) CUSPs (Commonly Used System Programs), and the memory differences between Basic-Plus and Basic-Plus-2 CUSPs.

Correction!

In the last issue, I presented a patch to set a VT100 to VT52 mode upon exiting EDT V2. The patch functions by printing the escape sequence to change to VT52 mode. Due to the printing font, the last character in that escape sequence looks like a "1" (ASCII 49), when it should be a lower case "l" (ASCII 108).

One-Shot Spooling Sys-Call Form names

The release of RSTS/E V7.0 added a new sys-call called the one-shot spooling sys-call. This call allows a user program to submit requests to the spooling system (SPOOL and BATCH). Now, programs can be written that interface to the spooling system directly, instead of chaining back and forth to QUE.

Unfortunately, the implementation of the call requires all parameters to be passed in FIRQB, instead of using XRB to point to the command string (like the .FSS call). This means that there is not enough room to specify a form name. This is a limitation to systems that either print on a variety of forms, or use the form name to allow the operator to select which request gets printed when.

There is not enough free room in the call to allow the specification of the form name, but there is a free bit in the flag word passed in the call. This patch tells QUMRUN that if the 6'th bit (value = 64) is on in the flag word to use the first six characters of the file name as the form name.

```
RUN [1,2]CPATCH
CPATCH V7.0-07 RSTS V7.0-07 Softec Dev 11/70
File to patch - QUMRUN.BAS=QUMRUN.BAS
#KB:/CS:1299
*H/2I/V<cr>
2I< tab> < tab> PROGRAM< tab> < tab>: QUMRUN.BAS< cr>
*H/10140/V<cr>
10140< tab> M$=JOBPS &< cr>
*OAI< cr>
10135< tab> IF< tab> ONE.SHOT% &< cr>
< tab> THEN< tab> IF< tab> (ASCII(MID(M$, 4%, 1%)) AND 64%) &< cr>
< tab> < tab> THEN< tab> LSET FORM$ = MID(M$, 9%, 4%) &< cr>
< tab> < tab> ! IF THIS IS A ONE-SHOT SPOOLING CALL. &< cr>
< tab> < tab> ! THEN IF THEY SET OUR FORM NAME BIT IN THE FLAG WORD. &< cr>
< tab> < tab> ! THEN JAM IN THE FILE NAME AS THE FORM NAME. &< cr>
< cr>
< esc> *V< cr>
10140< tab> M$=JOBPS &< cr>
*EX< cr>
Patch from —KB:[x.x].CMD complete.
#IZ
File to patch - IZ
```

After installing this patch, if you set the 6'th bit in the flag word (as in the example below) when you send a spooling request, the file name will be used as the form name.

```
3010 TEMP.OS = SYS(CHR$(6) + CHR$(-28) + CVT%$(0) &
+ MID(SYS(CHR$(6) + CHR$(-10) + FILE.NAME$), 5, 8) &
+ STRING$(6,0) + CVT%$(SWAP%(4 + 64))) &
! QUE THE FILE TO "LP:" USING THE FILE NAME AS THE FORM NAME. &
! AND FLAG IT TO BE DELETED WHEN DONE. &
```

Time differences between Basic-Plus and Basic-Plus-2 CUSPs

My initial goal was to demonstrate the time (clock and cpu) differences between Basic-Plus and Basic-Plus-2. I figured that the demonstration must be directed towards some practical end, so I chose 4 of the CUSPs that I felt were particularly 'piggy.' The CUSPs I chose were ANALYS+ANALY1, BACDIR, DIRECT, and REORDR. This would show us the benefits to be gained just by using the Basic-Plus-2 compiler instead of the Basic-Plus interpreter.

This was not particularly fair to Basic-Plus-2 because it was not designed to be a faster Basic-Plus. What is efficient in one language is not always efficient in another language.

So, I decided to make some minor modifications to the programs to make them more efficient for Basic-Plus 2. Since all of these programs use virtual arrays, and Basic-Plus-2 will do 'program internal' random caching of virtual array file blocks if given the memory, we open the files with a recordsize that is a multiple of 512+6 (6 bytes to keep track of which block is where). I chose a recordsize of 4144 ((512+6)*8) which meant that we were reducing the potential disk activity by a factor of 8 in exchange for a little more than 2 KW of memory.

The experiment was as follows:

1. Run each program in Basic-Plus (BAC), Basic-Plus-2 (TSK), and Basic-Plus-2 with large recordsizes (LRS) 12 times, recording the clock and cpu time consumed in each run.
 1. Run ANALYS, telling it to output to NL:.
 2. Replace BACMNT with a program that just exits, run BACKUP, telling it to backup one large account. Note that this means that the times presented below include the time spent on BACKUP and BACCON.
 3. Run DIRECT, telling it to output a /S listing of one large account to NL:.
 4. Run REORDR, telling it to sort one large account.
2. Discard the lowest, and the highest times to reduce the chance of a stray value influencing the result.
3. Calculate the mean times.

The results of the experiment are:

"I (for once) was speechless."

— Dave Mallery, March, 1981 issue of
RSTS PROFESSIONAL

To tell the truth, so are we. We knew DISKIT would amaze RSTS users, but, frankly, we were unprepared for the response. Phone calls, letters, and now the RSTS PROFESSIONAL -- all saying what we want you to know:

DISKIT is a remarkable software tool!

Listen to what else Dave has to say:

"...using DISKIT, I created 130 accounts and fully extended their centered UFDs in 3 minutes and 40 seconds (a job that used to take 4 to 8 hours.)"

"I then copied the full contents of a 300 MB RM05 equivalent to this new 'well-structured' disk in 45 minutes, optimizing clustersize and contiguity in the process..."

DISKIT IS A DISK STRUCTURING UTILITY

As Dave discovered, DISKIT's disk structuring utility, DSU, is fast. It also:

- Optimizes file clustersizes
- Places and pre-extends UFDs
- Performs transfers between unlike disks
- Saves all accounting data
- Allows manual file placement
- Provides full logging and statistics
- Includes sophisticated error handling and recovery

DISKIT IS A DIRECTORY PROGRAM

But DISKIT is more than a disk structuring utility. DISKIT's Macro-11 directory program, DIR, displays directories 12 times faster than before -- looking up files by name, extension, and date (with wildcards) at the incredible rate of 250 files/second.

And DIR is smart. It supports all standard DIRECT switches (including backwards, up to 1,000 files) with features you won't find elsewhere - like password lookup, UFD placement, and UFD size.

DIR even works as a diagnostic tool on dismounted disks, detecting bad directory structures and identifying them with comprehensive error messages.

DISKIT IS AN OPEN FILES DISPLAY PROGRAM

DISKIT's Macro-11 OPEN program displays open files by job -- with complete job and file statistics. It even has a "sleep switch", allowing you to dynamically update information at any desired interval.

DISKIT LETS YOU WRITE YOUR OWN DISK HANDLING ROUTINES

Best of all, the very same routines used in DISKIT are included, with documentation, so you can write your own disk handling routines. In minutes.

DISKIT IS THE FIRST SOFTWARE TOOL KIT FOR COMPLETE DISK MANAGEMENT

The DISKIT package provides all the tools and utilities you need to create and manage a well-structured disk. The entire DISKIT package, with extensive documentation is available now for only \$1250.

DISKIT, Dave says, "...is the 'final solution' to structured disks, eliminating all of the time and complexity and reducing the job to one of a SAVRES."

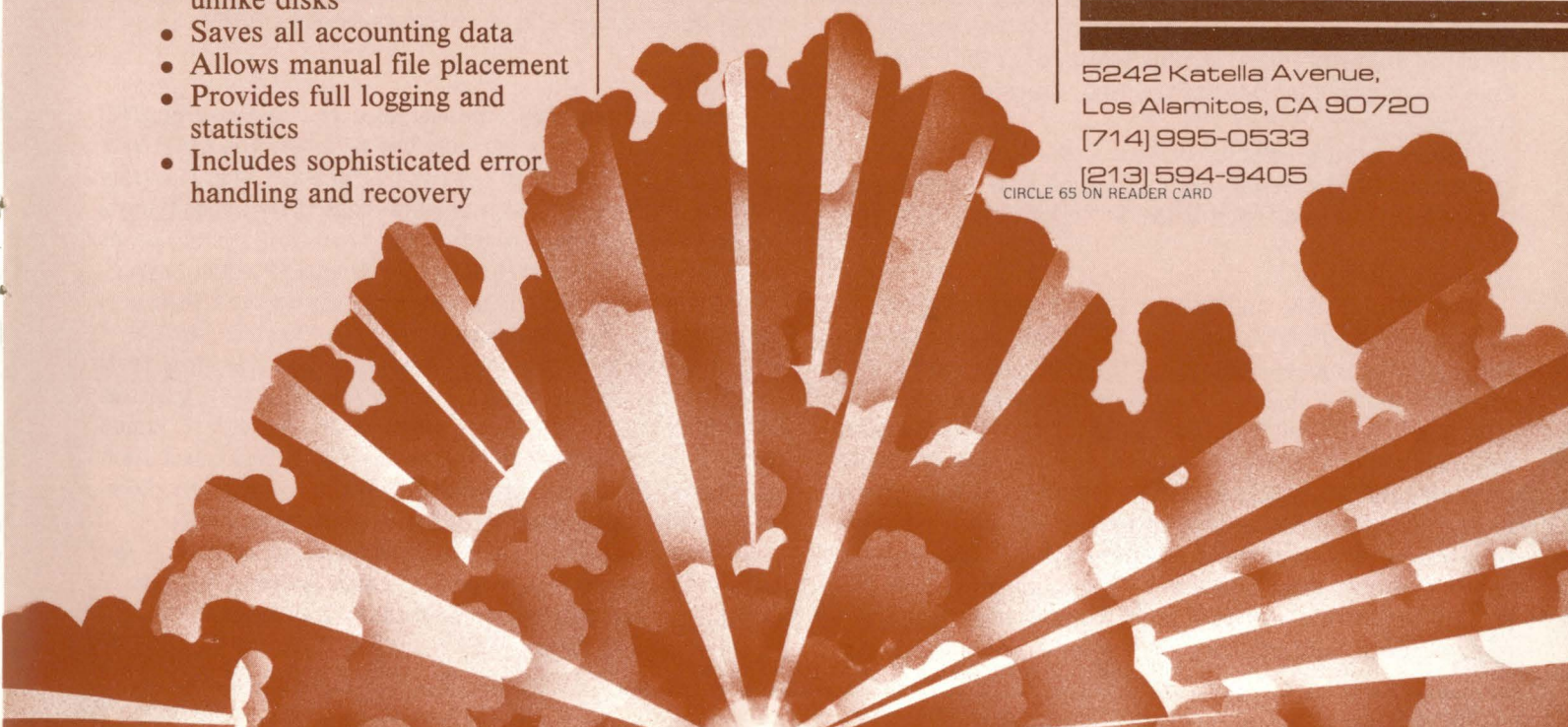
What more could we say?

Once again, we've got the answer.

Software Techniques, Inc.

5242 Katella Avenue,
Los Alamitos, CA 90720
(714) 995-0533
(213) 594-9405

CIRCLE 65 ON READER CARD



RSTS DEFECTOR

By Joel Schwartz, M.D.

I have a confession to make. I have defected from RSTS. But before you put my article down let me explain. I've played all the games available to me on the RSTS system so I went in search of new game horizons on other computers and here's what I found. There seems to be two basic types of games available. One is the Arcade type game. Space Invaders and Asteroids would be examples of this type. The other is the Adventure D and D type of which Adventure and Dungeon would be an example. The games can be found on 51/4 floppy discs and can be run on computers that are named after a fruit, a household animal, a naval officer and one that begins and ends with a vowel. (I had to disguise these in order to keep my job.) If you have any trouble figuring them out, send a self-addressed envelope to me and I will send the answers.

I want to tell you about two of the latter type games. The first is called ULTIMA (California Pacific Computer Co. by Lord British). Billed as the ultimate adventure game, in

many ways that's exactly what this game is. There is the ability to create new characters of different races (human, elf, hobbit, dwarf) and types (fighter, cleric, wizard, thief). Once formed the character can travel on land, across oceans, in the air, and eventually into space. Towns, castles, islands and dungeons are but a few of the places on the adventurers way. The game is filled with quests, challenges and monsters which test your skill and creativity. A well documented manual and separate player reference card come with each game. I have been deliberately vague, for to be more specific would only spoil the game.

The second game I enjoyed was THE PRISONER (EduWare, Conoga Park, California). Based on the popular English TV program, the object of the game is to escape from the ISLAND where you are being held prisoner. While you are trying to do this the computer is trying to get you to reveal three digit secret code number it gives you at the beginning of the game and forewarns you never to reveal. Every psychological trick in the book is used to obtain your information. If you are easily frustrated, this is not the game for you. However, if you are . . . but that would be telling.

Program	Clock	%/BAC	Cpu	%/BAC	KW(prgrm + shared)
ANALYS.BAC	333.5	100.0	301.3	100.0	32(16 + 16)
BACDIR.BAC	103.0	100.0	14.4	100.0	31(15 + 16)
DIRECT.BAC	40.9	100.0	30.1	100.0	30(14 + 16)
REORDR.BAC	176.7	100.0	95.7	100.0	30(14 + 16)
Using the Basic-Plus run-time system					
ANALYS.TSK	196.9	59.0	161.4	53.6	31(15 + 16)
BACDIR.TSK	121.9	118.4	10.0	69.4	30(14 + 16)
DIRECT.TSK	45.1	110.3	20.4	67.8	29(13 + 16)
REORDR.TSK	122.0	69.0	33.8	35.3	29(13 + 16)
Using the Basic-Plus-2 run-time system					
ANALYS.LRS	180.4	54.1	160.9	53.4	29(21 + 8)
BACDIR.LRS	24.0	23.3	6.6	45.8	30(22 + 8)
DIRECT.LRS	24.1	58.9	20.5	68.1	27(19 + 8)
REORDR.LRS	44.2	25.0	26.0	27.2	27(19 + 8)
Using the Basics resident library					

From these results we can see that a program written to be efficient in Basic-Plus may run faster or slower in Basic-Plus-2, but with a little effort, programs can run considerably faster. For our system, this minor modification reduced the clock time for BACDIR to get a directory of the whole system from over 12 minutes to just over 1 minute. This made running BACKUP during timesharing almost painless.

Based on the results of this experiment, I plan to modify all of the programs in the spooling and backup packages to use large recordsizes on virtual array file opens.

Multi-user Tasks

Multi-user tasks are tasks that have been task-built to allow the separation of read-only code from read-write code. For RSTS this means separating the read-only code into a separate resident library. This means that when a user runs a program, RSTS will load the resident library and the read-write section of the task. When a second user runs the same program, RSTS will just load the read-write section of the task. The exact procedures to create a multi-user task have been published several times in several publications, and will not be repeated here. (See Vol 3-2 p 8, June 1981.)

As a practical example of how multi-user tasks can reduce total system memory requirements, we will look at a

real-life example:

One of our clients is a heavy user of the spooling system. During the morning operator's shift, 5 print spoolers run almost constantly. All CUSPs are running in Basic-Plus-2, linked with the Basics resident library.

SPLRUN runs in 18KW, so if we multiply these numbers out we get 90KW of print spoolers competing for memory. If we build SPLRUN as a multi-user task, we get a SPLRUN that runs in 3KW, with a 15KW SPLRUN resident library. Since all 5 spoolers will share their 15KW read-only segment, they only require 30KW to run all 5 spoolers at the same time. This saves 60KW of their total system memory requirements by re-task-building a single program!

The only 'fly-in-the-ointment' is that resident libraries must be loaded at specific addresses. This is a serious limitation to systems with small amounts of physical memory. I hope DEC changes this in future releases of RSTS.

Quick Tip

On occasion, programmers need to execute a Basic[2] program starting at a non-zero line number. The usual way to accomplish this is to switch to the Basic-Plus run-time system and execute an immediate mode CHAIN specifying a non-zero line number. If you are in a run-time system other than Basic-Plus, type 'RUN programname/PO:linenumber'. If you are in the Basic-Plus run-time system type 'RUN programname <LF> linenumber.'

The reason this works is that when you execute a .RUN monitor call (which Basic[2] considers a CHAIN), you can specify a parameter word (@ FIRQB + FQNT) which Basic[2] will interpret as the line number to start execution at. This is the same location in FIRQB that receives the Position switch value after a file name string scan (.FSS).

If you have any questions or suggestions for things you would like to see in this column, please address them to:

Steven Edwards, Software Techniques,
5242 Katella, Los Alamitos, CA 90720

TIP THE



**\$14,500
THROUGH
JANUARY 31, 1982**

APPLICATIONS SCALE IN YOUR FAVOR...WITH USER-11.

USER-11 is a comprehensive applications development facility for the DEC RSTS operating environment. Dozens of integrated programs harness RSTS's power for unparalleled productivity and performance in constructing on-line and batch application systems.

PRODUCTIVITY...

A MATTER OF TIME.

More than a data management system, **USER-11** features common-function programs that permit numerous applications to be installed without writing a single line of code. Complete building blocks and interfaces are provided for those remaining applications requiring custom work.

PERFORMANCE...

SIMPLY INCREDIBLE.

USER-11 combines advanced BASIC and MACRO coding techniques with ultra-efficient file accessing mechanisms to optimize application system performance.

RELIABILITY...

A PROVEN FACT.

USER-11 is currently installed on hundreds of time-sharing systems world-wide with a reliability record that users repeatedly praise. All software is exhaustively tested and benchmarked prior to any distribution release.

SECURITY...

MORE THAN RSTS.

USER-11 incorporates a unique MENU system which flexibly and securely controls all processes. Secondary, encoded security databases are provided for each project. A special Run Time System is invoked to prevent accessing the RSTS ready state, unless the software developer desires this for the user.

STANDARDIZATION...

A BYPRODUCT.

All **USER-11** generated packages employ programming and documentation conventions which enhance compatibility, readability, and maintainability.

ADAPTABILITY...

NO PROBLEM.

USER-11 programs are dictionary and parameter driven throughout. Files can be restructured without program modifications.

DOCUMENTATION...

GOOD AND PLENTIFUL.

USER-11 features a wealth of easy-to-follow documentation. An extensive on-line "/HELP" facility is at software developer and user fingertips. All documentation is maintained and distributed on your system's compatible media.

TRAINING...ALL KINDS.

USER-11 training courses are held frequently with instructional programs to suit your need—beginner to expert.

FEATURES...ON AND ON.

USER-11 includes virtually every facility needed to quickly construct high performance management applications—nothing else is required. If you find this hard to believe or would like more information, contact us; we will furnish you with solid user proof!



**North County
Computer Services, Inc.**
2235 Meyers Ave.
Escondido, California 92025
(714) 745-6006, Telex: 182773

DEC and RSTS are registered trademarks of Digital Equipment Corporation.

© Copyright NCCS

CIRCLE 30 ON READER CARD

DON'T BUBBLE — QUICK SORT

By Neil Robertson, Management Controls Systems, Manchester, U.K.

It is a fairly common requirement to sort an array of numbers or strings. If the array is small this is often done using a "bubble" sort, which is possibly the slowest method available. The automatic stacking of parameters with Basic Plus functions enables a compact "quick" sort to be written. Such a function for sorting strings is given below, and also a table showing CPU seconds used when sorting random eight byte strings using Basic Plus on an 11/70. The string length is not important when using Basic Plus since the comparison is normally resolved in the first 3 or 4 bytes, and only the pointers are changed. With Basic + 2 the times are considerably longer, since the strings themselves are moved rather than the pointers.

Integer and Floating Point sorts require the string variables to be changed to integer or floating point variables, and the function name changed. The array must of course be dimensioned. Only the array elements in the range given by the first two parameters are sorted. The third parameter

is used to save the variable X% for use by the second recursive function call. This function should not be used to sort virtual arrays. If you must do such a thing use a bubble sort.

Purists may not like the "WHILE" syntax but it works.

Array size	100	200	300	400	800	1600
Bubble sort-CPU sec	2.3	9.9	21.2	38.3	144	621
Quick sort -CPU sec	0.5	1.0	1.6	2.2	4.7	10.1

```

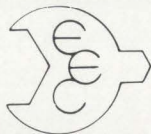
24840 DEF* FNGSORT$(Y1Z,Y2Z,XZ)
! QUICK SORT STRINGS X$(Y1Z) TO X$(Y2Z)
! XX IS A DUMMY PARAMETER -- SET=0
\
\ XX=Y1Z
\ YZ=Y2Z
\ Z=X$((Y1Z+Y2Z)/2Z)
\ WHILE XX<YZ
\   XX=XX+1Z WHILE X$(XZ)<Z$
\   YZ=YZ-1Z WHILE X$(YZ)>Z$
\   IF XX=YZ THEN X1=X$(XZ) \ X$(XZ)=X$(YZ) \ X$(YZ)=X1$
\   XX=XX+1Z \ YZ=YZ-1Z
24850 NEXT
\ X1=FNGSORT$(Y1Z,YZ,0Z) IF Y1Z<YZ
\ X1=FNGSORT$(XZ,Y2Z,0Z) IF XX<Y2Z
\ FNEED
  
```

EEC SYSTEMS

SUPERB SOFTWARE for DEC

LEX-11

The leader for integrated word and data processing in the DEC marketplace. Available for all DEC operating systems and UNIX, TSX-PLUS, for LSI-11, PDP-11 and VAX. OEM discounts.



EEC Systems

RJ-11 COBOL

Compiler. ANSI-74 compatible. Twice DEC's features for half the price. Available for RT-11, RSTS/E, and TSX-PLUS.

286 Boston Post Road
Wayland, MA 01778

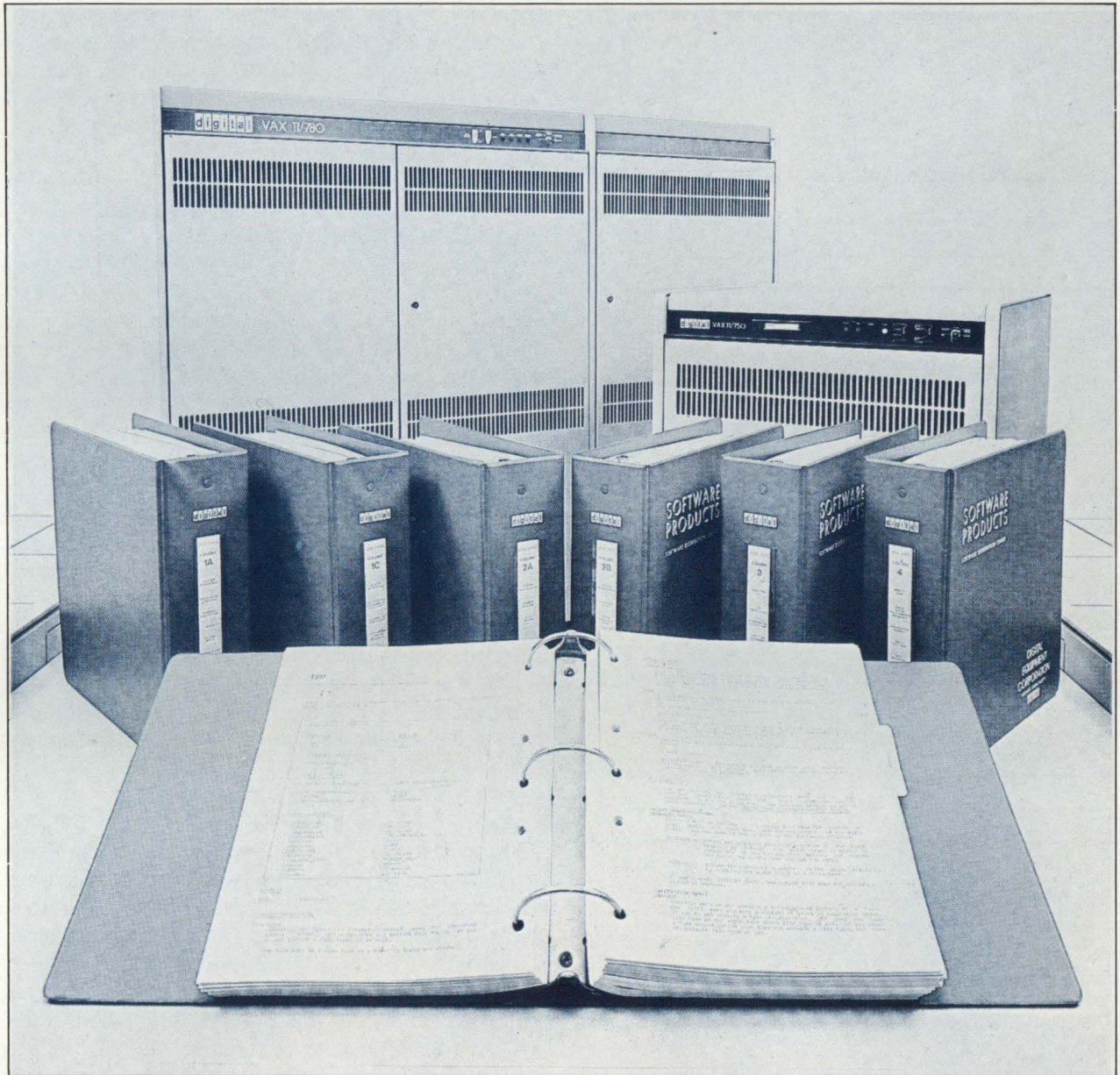
(617) 358-7781/2
(617) 443-6376

The VAX-SCENE

Number 5

(RSTS PROFESSIONAL, Vol. 3, No. 4)

December 1981



INSIDE:

- Transportable Software on RSTS/E and VAX/VMS
Using a Support Library Technique

TRANSPORTABLE SOFTWARE ON RSTS/E AND VAX/VMS USING A SUPPORT LIBRARY TECHNIQUE

By Dave Froble

Transcomm Data Systems Incorporated is an Authorized DIGITAL Computer Distributor specializing in business software packages. Our experience with both DEC and RSTS dates back to 1972, when we took delivery of one of the first PDP-11/40 RSTS systems. Since that beginning, the company has developed a full line of distribution management and financial control software packages in use by clients in North America and abroad.

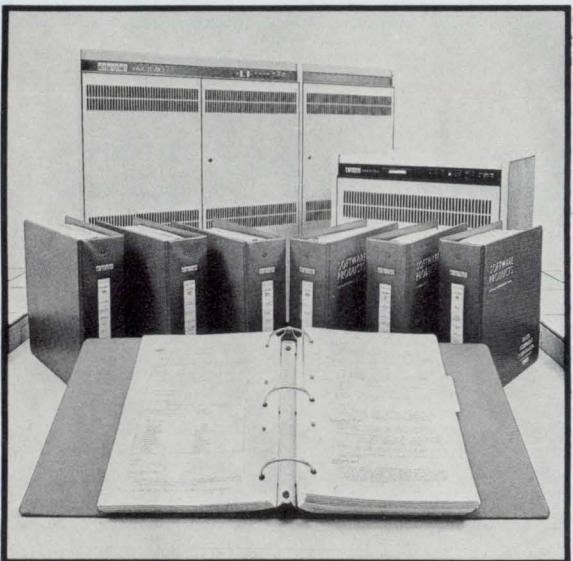
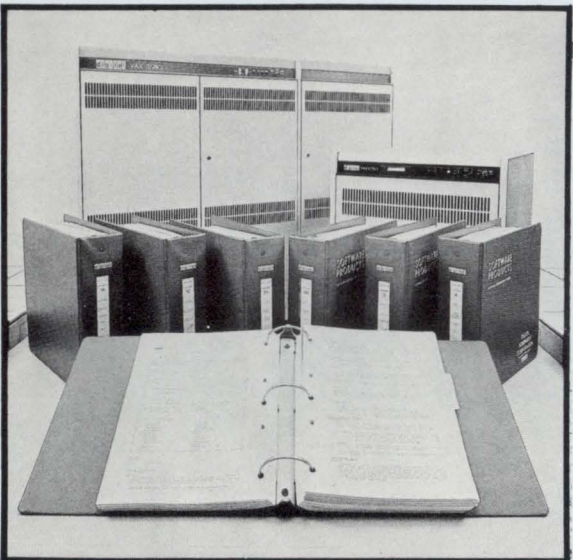
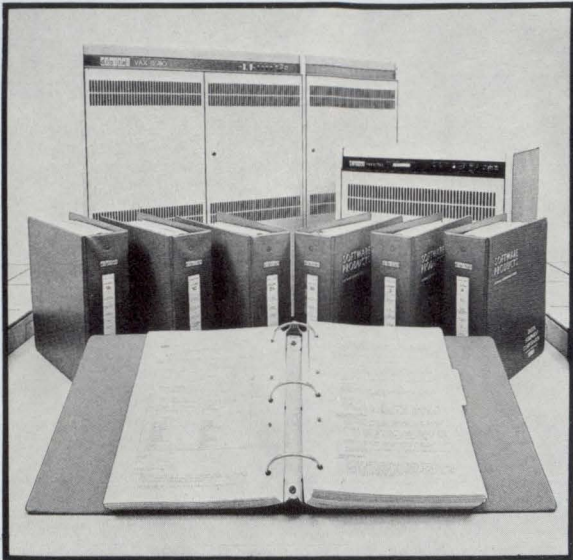
Known as TOLAS (the Transcomm On-Line Accounting System), this software was initially written in BASIC-PLUS. When BASIC-PLUS-2 (BP2) became generally available several years ago, certain critical parts of the system were converted to take advantage of the increased program size and run-time efficiencies provided by the new language.

When it became clear in 1980 that Digital would be placing increasing technical and marketing emphasis on the new 32-bit VAX line of computers, Transcomm initiated a study to evaluate how best to transport TOLAS to the new operating environment. Two alternative strategies were considered. First, we could have used a compatibility mode technique. Although the actual conversion effort would have been fairly simple, the resulting software would not fully utilize the additional capabilities of the VAX and VMS. A better solution, we felt, would be to implement our software on the VAX using VAX-11 BASIC. With this alternative, the resulting software would take full advantage of the new VAX technology.

The problems associated with maintaining two sets of software (one for the PDP-11 and one for the VAX) were also considered. It was decided to avoid these problems by producing a single set of software that would run on both machines. The solution involved creating separate support libraries of external subprograms for each machine. These libraries would provide the basis for transporting common application programs between the two types of systems.

The first step in this support library technique for the VAX conversion was to translate all the RSTS/E programs into BASIC-PLUS 2. Following this step, unique support libraries were created for each type of system. These application program support routines perform the functions that are unique either to the specific computer or to its operating system. In theory, the approach could be extended to other computer systems in the future.

The first major problem we encountered concerned disk files. Disk files on the VAX are handled in a manner that differs greatly from the RSTS/E native files. Mainly, all files on the VAX are accessed using the RMS file processing



Using Sort-11 from Basic Plus-2 on RSTS/E Version 7.0

By Ron D. Troy

According to DEC manuals, it is possible to use Sort-11 via the use of call statements or their equivalent from COBOL, Fortran, or Macro-11. This use of Sort-11 from Basic has not previously been documented by DEC, but can be done without difficulty. Using Sort-11 in your own Basic program negates the need to run \$Sort directly or with an indirect command file. It allows one to use nearly any type of file structure, to run Sort within your program (invaluable in an on-line mode), and to customize the sort process to your own needs for the greatest efficiency possible. It also allows you to chain to a sort and then have the sort program chain back when finished. One could use a customized sort program or a sophisticated sort program for all sorting with the parameters passed via core common. This chain ability is available with other sorts (ex. FSORT3) and is now available with DEC's Sort-11. The sort can also be included as part of a multiple part task. Sort may not be as fast as some other sort packages, but it may be much more convenient and reliable.

Enclosed is an operational sample program for using the sort via call statements. The sequence of the calls, the arguments passed to and from the sort modules, setting up of a single 6 byte key and manipulating it are all clearly shown and explained. The sample program is set up to sort a 30 byte RMS sequential record containing a 6 byte alphanumeric key. For test purposes a 10000 record file was set up, using any method available to speed up the sort. The sort work-area size was set as large as possible, and the number of sort work-files and buffers per work file were manipulated to achieve the minimum sort time. (On a PDP 11/44 with 1 RPO-6 and 256k words core, the minimum time achieved was 1 minute, 50 seconds.) For demonstration purposes, the example given is of a full record sort, but tag (key) sorts are also possible. For large records, a tag sort would probably be faster than a full record sort. For more information on achieving maximum efficiency, see the Sort-11 manual.

The sample shown does not show the use of multiple keys, or descending sorts. To sort by multiple keys, append to your work area from left to right the various keys, in the order of their importance (the most important first). Then flip the entire key and set the 'address' variable name to start at the beginning of the last word in the key. Note that the length of the entire key must be even, or a Sort error will occur. To cause a descending sort on any one key, use the change statement to create an array, subtract the value of A(I) from the value 255, and change the array back to a string. Then, append that key to its proper position in the complete key. The individual key will be given a descending sort. Note that all work on the key must be done in an area separate from the area where the record is stored.

The Sort Sequence

The first action for a sort is to open a null buffer and map out the various work areas, and call arguments. All arguments. All arguments used by the various call statements must be mapped out, including the integer variables. The various variables must then be set or cleared (ex. IWKSIZ% = 24000% as the sort work-area size). The input file is then opened and mapped out. The first call — RSORT, is used to initialize the sort. The call, as in other calls to sort, must be a "CALL BY REF (arg1, arg2, etc)." Passed in this first call are a variable for error codes to be returned (used in all calls), the keysize — which must be even and positive, maximum record size (variable length records can be sorted), key address (actually the last word of the key), sort work-area name, sort work-area size (as large as possible), the number of work files (> 3 and < 8 — try varying for greater sort speed), the number of buffers open for each file — also try varying this variable). An error code of 0 returned by the sort shows that it was properly initialized.

The second major action is to read the file records and pass them to Sort. The loop ends when end of file is reached. All keys are appended into one contiguous work area. The entire key is then flipped, and is now ready to be sorted. This can easily be done using the 'Change' statement, a 'For-Next' loop, and another 'Change' statement as shown in the sample program. The call statement here has as arguments the error-code integer variable, the size of the current record, (allowing for variable length records), and the name of the area containing the record. Again, an error code of '0' signifies proper operation.

The third action is the merge. All that is passed here is the error-code variable. Sort will return control when the merge is complete.

The fourth action is the return of the sorted records in order from Sort. First, the input file is closed, the output file opened and mapped out. The return loop has a call and a put to the output file. The call uses the error-code variable, the record size — returned by Sort, and the output record. If an error-code < 0 is returned, then all records have been returned, while a 0 indicates normal operation.

The fifth, and final operation, is to close out the sort. The only argument used is the error-code variable. The output file should then be closed, and the sort is then complete.

All calls should be done exactly as shown in the sample program. Sort will return its own error codes or RMS error codes if an error occurs. These error codes are listed in the PDP-11 Sort Reference Manual. The manual also describes the sort sequences and the key manipulation — but not for Basic. The individual words are not reversed as described in the manual; rather the entire key is reversed. The manual

also has information on sort efficiency, but this information may be inaccurate. An SPR has been submitted concerning the errors or deficiencies in the manual. Note that use of the above routines is not officially supported yet by DEC, although it is reasonable to assume that such support would soon be forthcoming. Also, it should be noted that 'SORTS.OBJ' and 'SIORMS.OBJ' must be present on the system and in the account that the task build is performed. The build statement is "BUILD PROGNAME, SORTS, SIORMS/SEQ" (the SEQ for the RMS sequential files used here). SORTS, and SIORMS contain all the required object modules. For further information, read Chapter 5 of the Sort manual.

The use of Sort as described above is a convenient method for easy and simple inline sorting of RMS or most other types of files. Unlike some other sorts, it is not necessary to use header records, to know how many records are being sorted, or to otherwise manipulate your file to make it acceptable to a sort, nor is it necessary to run the sort directly or through an indirect command file. You can now adapt the sort to your file, and set it up to run as efficiently as possible for your file. You can also set up a common sort program that will read nearly any type of file without problems. This is something that few, if any, other sort packages are likely to offer for RSTS/E and Basic Plus-2. It also reduces the need to buy a non-Dec sort package that might not work under future versions of RSTS/E.

```

6000 !MODULE FOR CALLING SORT-11. THIS MODULE CALLS THE
!VARIOUS MODULES REQUIRED FOR A SORT AND INCORPORATES
!THE TYPE OF LOGIC DESCRIBED IN THE PDF-11 SORT MANUAL.
!THE ADDRESSES CALLED FOR BY THE MODULES MUST BE ON
!WORD BOUNDARIES (1ST BYTE OF MAPPED AREA, 3RD, ETC.).
!THE RECORD MUST BE MANIPULATED IF THE FIELDS USED FOR
!THE SORT KEY ARE NOT CONSECUTIVE (HIGH FIELD FIRST),
!THIS CAN BE DONE BY MOVING THE VARIOUS FIELDS TO AN
!NULL DEVICE OPEN ON ANOTHER CHANNEL. ANY CHANNELS
!USED MUST BE FROM CHANNELS 1 TO 4. THE NAME OF THE
!FIELD CONTAINING ONLY THE LAST WORD OF THE TOTAL KEY
!IS PASSED IN THE CALL STATEMENTS, ALONG WITH
!THE LENGTH OF THE KEY AND THE LENGTH OF THE ENTIRE
!RECORD.
!**** FOR MORE INFORMATION, PLEASE SEE CHAPTER 5. ****
ON ERROR GOTO 19000 !SET ERROR TRAPPING
\OPEN *NL:* AS FILE 2%, RECORDSIZE 24200%,
MAP WORKS
\MAP (WORKS)
KEY.AREA#=6%,
WORK.AREA#=24000%,
IERROR%,
KEYSIZ%,
MAXREC%,
IWKSI%,
IFILE%,
IRECS%,
BIGBUF%,
IX
\MAP (WORKS)
FIL.2%=4%,
KEY.ADR%=2%
\KEY.AREA# = *
\WORK.AREA# = *
\IERROR% = 0% !CLEAR ERROR CODE FIELD
\KEYSIZ% = 6% !KEYSIZE (MUST BE EVEN)
!PUT YOUR KEYSIZE HERE
\MAXREC% = 30% !RECSIZE (MUST BE EVEN)
!PUT YOUR RECSIZE HERE
\IWKSI% = 24000% !WORK AREA SIZE IN BYTES
\IFILE% = 4% !SCRATCH FILES (>2)
\BIGBUF% = 14% !BUFFER COUNT
\OPEN *SORTST.DAT* FOR INPUT AS FILE 1%, !OPEN INPUT
ORGANIZATION SEQUENTIAL FIXED,
ACCESS READ,
CLUSTERSIZE 128%,
MODE 2304%,
MAP SRTREC
\MAP (SRTREC) !MAP OUT INPUT RECORD
SORT.REC#=30%
\MAP (SRTREC)
KEY.1%=6%,
SORT.DAT%=24%
\CALL RSORT BY REF (IERROR%, KEYSIZ%, MAXREC%, KEY.ADR%,
WORK.AREA#, IWKSI%, IFILE%, BIGBUF%)
!CALL RSORT TO INITIALIZE THE SORT
\GOTO 6900 IF IERROR% <> 0 !GOTO IF ERROR OCCURED
\IRECS% = 30% !RECSIZE SAME AS MAX
    
```

```

6020 !SECOND CALL
GET #1% !READ NEXT INPUT RECORD
!AT LINE 19000+ INCLUDE AN EOF (ERR=11) TEST WITH THE
!RESUME AT 6030 TO GO ON TO THE NEXT STAGE OF THE SORT.
!SET UP KEY ROUTINE *****
!AS EXPLAINED ABOVE IN THE INTRODUCTION, THIS MUST BE
!DONE IF THE SORT KEY FIELDS ARE NOT CONSECUTIVE AND/OR
!DO NOT START ON THE WORD BOUNDARY. THE MAJOR SORT KEY
!COMES FIRST, FOLLOWED IN ORDER BY EACH OF THE MINOR
!KEYS. ONCE THIS IS ACCOMPLISHED, THE ENTIRE KEY IS
!REVERSED IN ORDER. THE SORT THAT WILL OCCUR IS AN
!ASCENDING KEY SORT. BY CHANGING ALL CHARACTERS IN A
!KEY TO THEIR COMPLEMENT VALUE, A DESCENDING SORT FOR
!THAT KEY SHOULD TAKE PLACE. BOTH THIS SWITCH AND THE
!ENTIRE KEY REVERSAL ARE ACCOMPLISHED USING THE 'CHANGE'
!STATEMENT.
\SET KEY.AREA#=KEY.1% !+KEY.2#+KEY.3%, ETC. *****
\CHANGE KEY.AREA# TO A
\B(J%)=A(KEYSIZ%-J%+1%) FOR J% = KEYSIZ% TO 1% STEP -1%
\B(0%)=A(0%)
\CHANGE B TO KEY.AREA#
\CALL RELES BY REF (IERROR%, IRECS%, SORT.REC%) !SORT
\GOTO 6020 IF IERROR%=0% !NEXT RECORD IN
\GOTO 6910 !GOTO IF ERROR
6030 !END OF MAIN SORT LOOP
CLOSE 1% !CLOSE INPUT FILE
\CALL MERGE BY REF (IERROR%) !CALL THE MERGE
\GOTO 6930 IF IERROR% <> 0 !GOTO IF ERROR
6040 OPEN *SORTST.SRT* FOR OUTPUT AS FILE 3%, !OPEN OUTPUT
ORGANIZATION SEQUENTIAL FIXED,
ACCESS WRITE,
ALLOW NONE,
CLUSTERSIZE 128%,
FILESIZE 700%,
CONTIGUOUS,
MAP OUTREC
\MAP (OUTREC)
OUT.REC#=30%
6050 !OUTPUT WRITING
CALL RETRN BY REF (IERROR%, RRECSIZ%, OUT.REC%)
!RRECSIZ IS SIZE OF THE RETURNED RECORD
!IF IERROR%=0% THEN PUT #3%
\GOTO 6050
6055 GOTO 6060 IF IERROR% < 0 !END OF FILE IN SORT
\GOTO 6940 !ERROR
6060 !CLOSE OUT ROUTINE
CLOSE #3%
\CALL ENDS BY REF (IERROR%)
\GOTO 6950 IF IERROR% <> 0 !GOTO IF ERROR
\GOTO 32000 !CLOSE FILES AND END
6900 PRINT *INITIALIZATION *;
\GOTO 6990
6910 PRINT *READ *;
\GOTO 6990
6930 PRINT *MERGE *;
\GOTO 6990
6940 PRINT *OUTPUT *;
\GOTO 6990
6950 PRINT *CLOSE *;
\GOTO 6990
6990 PRINT *ERROR IN SORT MODULE *;IERROR%
\STOP
19000 RESUME 6030 IF ERR=11 !END OF INPUT FILE
!PRINT ERR; * AT *;ERL \ STOP \ RESUME 32000
32000 !END OF PROGRAM
CLOSE 1% FOR 1% = 1% TO 12%
32767 END
    
```

SPECIAL OFFER

**MACHINE READABLE 800 BPI DOS LABEL
OF MAJOR PROGRAMS IN THIS ISSUE**

Part of the proceeds will be going to the authors.

Send \$50 to:

M SYSTEMS, INC.

Box 361, Fort Washington, PA 19034-0361

RSTS PROFESSIONAL RSTS PROFESSIONAL RSTS PROFESSIONAL RSTS PROFESSIONAL RSTS PROFESSIONAL RSTS PROFESSIONAL RSTS

INTRODUCING

DELTA/C.M.I.
COMPUTER MANAGED INSTRUCTION

COMPUTER MANAGED INSTRUCTION, a tool for the modern educator

CMI, a powerful tool for the modern educator is now a reality. CMI is a computer program designed to provide computerized instruction, tests and remedial work to students. It adapts to each student's ability allowing them to work at their own pace. Students log on and access the information using the EXPLORE function, an easy to use, interactive program. The educator can design multiple learning modules using the flexible CREATE function and monitor each student's performance via the REPORT function. CMI was designed by university faculty and is currently being used in secondary schools, vocational schools and universities.

*For RSTS Users.

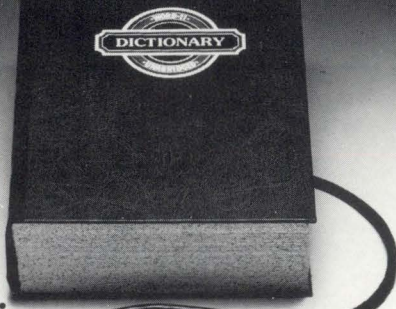
To find out how CMI can benefit your institution contact

ELEX

The Effective Learning Exchange

P.O. Box 14, Medford, New Jersey 08055
609/654-1100 CIRCLE 68 ON READER CARD

TALENTED. SHARABLE. RESPONSIVE WORD PROCESSING.



WORD-11.

WORD-11 is so sophisticated word processing—with features like list processing, automatic spelling error detection, indexing and footnoting—yet it's easy to learn and use.

It's sharable, up to sixty terminals. It's flexible. It's relatively inexpensive. And it's been operating for four years in hundreds of installations around the world.

CIRCLE 69 ON READER CARD

Available on RSTS/E, RSX and VAX/VMS versions, for DEC's family of mini-computers.

Call or write:

Corporate Office: 181 W. Orangethorpe Ave., Suite F., Placentia, CA 92670

(714) 993-4160. **New York**

Office: (212) 687-0104.

Washington, D.C. Office: (301)

657-4098.



Data Processing Design, Inc.

AUTHORIZED COMPUTER DISTRIBUTOR

Optimizing Space Allocation of Literals in Basic-Plus-2

By Brad Smith, Allied Data, Olympia, WA

The author has worked on PDP-11's for 5 years in several languages. He now specializes in the design and optimization of BASIC-PLUS-2 application systems.

When a BASIC-PLUS-2 is compiled, space is allocated for every literal and variable that occurs in the program. Being aware of the differences in how literals and variables are stored can help in fine-tuning a program.

STRINGS:

Literals and variables are stored in a similar manner, with a 1-word pointer to the data, a 1-word length counter, and the data. However, literal data are stored at compile time, while only the header (pointer and length counter) are allocated for variables. Furthermore, if the same literal occurs more than once in a program, it is still stored only once. This feature can be useful if a program contains a number of similar literals which can be modified so that they are identical. For instance, if a program uses 'ABC' in one place and 'ABCD' in another, then using 'ABCD' in both places saves 8 bytes (4-byte header, 3 bytes of data, and in this case, 1 null byte because string literals always occupy an even number of bytes).

FLOATING-POINT:

Space for both literals and variables is allocated at compile time, and floating-point literal data are stored in the same areas, and interspersed with, string literals. As with strings, a floating-point literal is stored only once no matter how many times it occurs in the program, so it may be possible to save space by reducing the number of different literals. For instance, if a program contains these two conditionals—

IF $A >= 100$, and IF $A <= 99$.

—then 4 or 8 bytes (single or double precision) can be saved by using

IF $A >= 100$, and IF $A < 100$.

INTEGERS:

Integer variables are allocated at compile time, similarly to string and floating-point numbers, and an integer variable operand is actually a pointer to the datum itself, also similarly to floating-point numbers. (A string operand points to the header, which itself contains a pointer.) However, an integer literal is stored in line at compile time: that is, the operand is the number itself. Thus, an integer literal occupies less space than a variable and undoubtedly will be processed more quickly.

In many programs, these suggestions would not save an appreciable amount of space. But with that crucial program that takes a shoehorn (or a sledgehammer) to fit in memory every time you add a few lines of code, checking literal usage may well be worthwhile.



VTEDIT.TEC

By David Spencer, Infinity Software Corporation

The following is a continuation of David's article which appeared in the previous issue of the RSTS Professional (Vol. 3, #3).

"I have provided the new VTEDIT macro in a "run"able source program form. I realize that everybody is going to want to stick in some extra feature. So, it is left as an exercise to the student to squish VTEDIT down to a smaller size. (The source program runs in 10K, and can be cut down to at least 7K, and perhaps 6.) Infinity Software Corp. makes no guarantee for the performance of VTEDIT.TEC, nor takes any responsibility for the use of VTEDIT.TEC. However, inquiries, suggestions, bugs, or whatever can be sent to: David Spencer, Infinity Software Corp., 2210 Wilshire Blvd., Suite 801, Santa Monica, Calif. 90403, phone: (213) 820-2702."

```

! *****!
! Copyright (C) 1981 by!
! Infinity Software Corporation!
! This software is solely for!
! non-commercial use and may be only!
! used and/or copied with the inclusion!
! of this notice. No title to or!
! ownership of the software is hereby!
! transferred.!
! The information in this software is!
! subject to change without notice and!
! should not be construed as a!
! commitment by Infinity Software!
! Corporation.!
! *****!
! !!VTEDIT.TEC V01.02!!
! Author: David Spencer - 19-Jun-81!
! WARNING: This VTEDIT destroys the!
! numeric registers.!
! Registers used:
! 0% Character read from KB
! 0S Accept numeric argument
! 1% Current direction
! 1S Word delimiter list
! 2% Maskeu character
! 2S Control character mask
! 3% GOLD key flag
! 3S Keypad character mask
! 4% State of exit
! 4S GOLD character mask
! 5% Command entered flag
! 5S Enter TECO command/search string!
! 6% Current character position
! 6S TECO command entered
! 7% Scratch
! 7S String to search for
! 8% Scratch
! 8S Scratch
! 9% Numeric argument
! 9S Display HELP frame
! C% Undelete direction flag
! CS Undelete Character register
! 1% Old ET value
! 1S This VTEDIT macro
! 1% Restore direction flag
! LS Undelete Line register

@^U9S
155^T @^A/[2J/ 155^T
@^A#[H Backsp Go to beginning of line
Delete Delete character
<1> Delete to start of word Gold Help Fndnxt Del L
CTRL/A Append to buffer Page Sect Append Del W
CTRL/D Delete to end of line
CTRL/Eq Execute register "q"
CTRL/F Move forward a word Command|Nxt Page Replace Uno W
CTRL/K Cut region to Q-reg
CTRL/R Move backward a word
CTRL/U Delete to start of line Advance Backup Cut Del C
CTRL/V Enter/exit "view-all"
CTRL/W Refresh screen Bottom Top Paste Uno C
CTRL/X Copy region to Q-reg
GOLD CTRL/X Append region to Q-reg Word Eol Char Enter
CTRL/Y Yank a page Chngcase| Del Eol| Specsins
GOLD CTRL/Z Close and exit TECO Line Select
GOLD Q Abort edit and exit
CTRL/\ Insert ASCII or next
GOLD ARROW Move upward a screen
GOLD ARROW v Move downward a screen
! *****!
! Load the HELP macro
! *****!

@^U9S
155^T @^A/[2J/ 155^T
@^A#[H Backsp Go to beginning of line
Delete Delete character
<1> Delete to start of word Gold Help Fndnxt Del L
CTRL/A Append to buffer Page Sect Append Del W
CTRL/D Delete to end of line
CTRL/Eq Execute register "q"
CTRL/F Move forward a word Command|Nxt Page Replace Uno W
CTRL/K Cut region to Q-reg
CTRL/R Move backward a word
CTRL/U Delete to start of line Advance Backup Cut Del C
CTRL/V Enter/exit "view-all"
CTRL/W Refresh screen Bottom Top Paste Uno C
CTRL/X Copy region to Q-reg
GOLD CTRL/X Append region to Q-reg Word Eol Char Enter
CTRL/Y Yank a page Chngcase| Del Eol| Specsins
GOLD CTRL/Z Close and exit TECO Line Select
GOLD Q Abort edit and exit
CTRL/\ Insert ASCII or next
GOLD ARROW Move upward a screen
GOLD ARROW v Move downward a screen

```

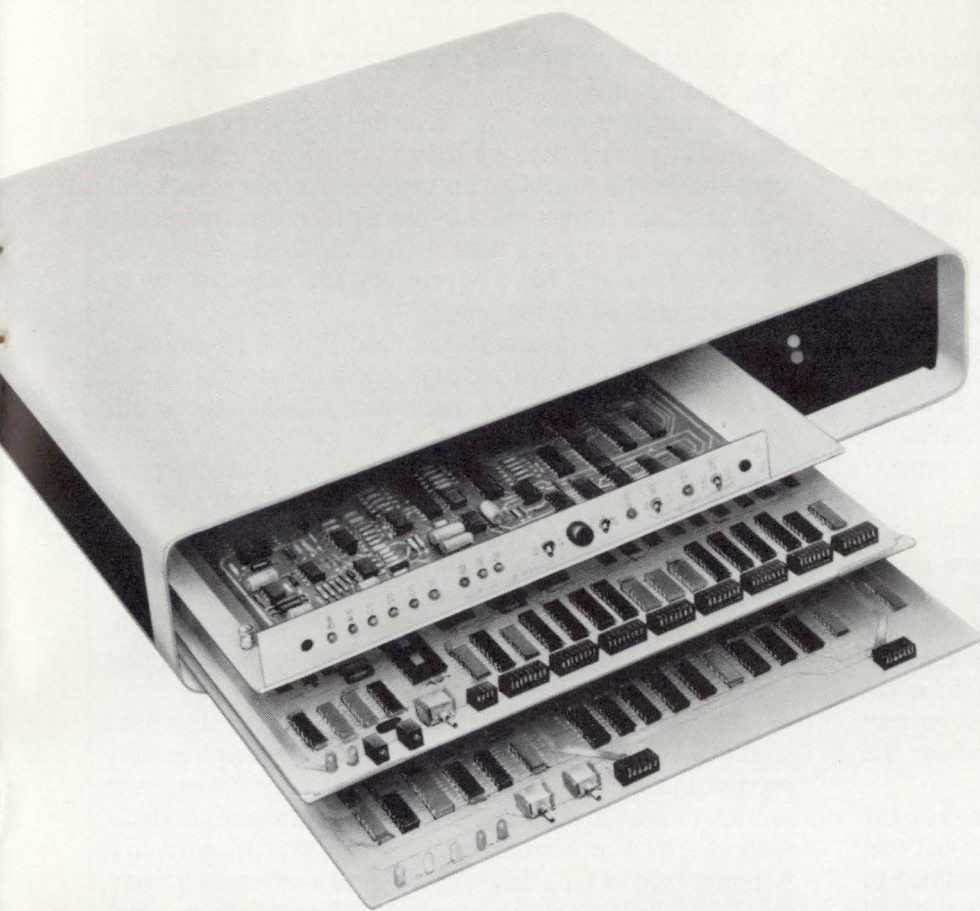
```

Hit any key to continue #
^T^
S

! *****!
! Load VTEDIT.TEC macro
! *****!

@^UIS
[C [L [W 0U6 [6 0U6 1U1 0UC 32@UC//
ETUI
2#4#B#256#512#32768ET
4,U;W
155^T @^A/< 155^T @^A/=
Z*= ETU7 ET^32ET ^T^< Q7ET M9 ^ Q7ET ^
<
32768:W&12700 1U9 0U3
!A! Q0-127*= Q9< -; ; R 0A@UC// D > 0UC F< ^
Q3*= 0Q02U2 @01^EU2C1 | 0Q04U2 @01^EU2GC1 ^
!MC! TU0 Q9< 13@1// 10@1// > F<
!C! TU0 Q0^D M0 @01A! ^ TU0 Q0Q3U2 Q3*= @01^EU2K1 | @01^EU2GK1 ^
!PK!
!PGK! -1U3 TU0 Q0-27*= @01C1 ^ Q0^D M0 ^ @01A!
!pK! Q1> Q9L F< ^
!HC! 0^Q^N -1^9 ^ | 0L ^ Q9< -L > F<
!gK! 01^< @01RC1 ^
!FC! Q9< @S/^EGL// ; @S/^N^EGL// ; R > .^*= ZJ ^ 0U6 F<
!RC! Q9< -; ; R @-S/^N^EGL// ; R @-S/^EGL// ; > 0U6 F<
!rGK!
!DC! -.2^N Q9< .U7 L -.2^N 2R ^ .U8 Q7J ^ .Q8XL K >
-.2^N 13@1// 10@1// 2R ^ ^ -1UL F<
!JC! Q9< .U7 -; ; R :@S-/^N^EGL/^S R ^ :@S-/^EGL/^U J ^ Q7..XW Q7..D >
DUW F<
!UC! 0XL 0K 0UL F<
!C! Q9< Q0@1// > F<
!AK! -Q9U9
!BK! 0U7 .U8 0L Q8- < 0A-9*= (Q7/8+1)*8U7 | ^7 ^ | ^ C >
Q9L
Q6U9 J6 Q9-Q7^N Q7U6 ^
0UB
< -.Z; 0A-13^N 0A-9*= (Q8/8+1)*8U8 | ^8 ^ | ^ C Q8-Q6; > ^
Q6-Q8^N 0A-13^N R -1^8 ^ | ^ | 16 Q8U6 F<
!sK! Q1^< @01DK1 ^
!CK! .+09-Z^< Q9C | ZL ^ 0U6 F<
!DK! -.Q9^> Q9R | J ^ 0U6 F<
!MA! F<
!RK! .U7 Q9^Q1:@S/^EQ7/^U Q7J @01ERR1 ^ F<
!SK! Q9< XL K > -1UL F<
!LK! Q9< -.Z; 0A@UC// D > -1UC F<
!mK! Q9< .U7 :@S/^EGL/^U ZJ 0; ^ :@S/^N^EGL/^S R | ZJ ^ Q7..XW Q7..D >
-1UW F<
!mK! 4W^*= .+1.4;W | @01ERR1 ^ F<
!CK! 0A-13^N -1^9 ^ | ^ Q1^Q9L -.Z^N L 2R ^ F<
!tK! 1U1 F<
!uK! -1U1 F<
!wK! 12^U8// Q9^Q1:@S/^EQ8/^U Q1^< J | ZJ ^ ^ F<
!xK! Q9^Q1^16L F<
!MGK! -.+07^< @01ERR1 ^ .U7 :Q7R :@S/^EQ7/^U Q7J @01ERR1 ^
Q9< ^SD GP .U7 :@S/^EQ7/^S > F^ | Q7J @01ERR1 ^ F<
!RGK! @U8/Search for/ 7M5^N .U7 Q1^Q9^@S/^EQ7/^U Q7J @01ERR1 ^ ^ F<
!SGK! Q9< GL QL^N ^SC ^ > F<
!LgK! Q9< GC > QC^N Q9R ^ F<
!mGK! Q9< GW QW^N ^SC ^ > F<
!ngK! 0.4;W 0U3 F<
!pGK! Q9< 13@1// 10@1// > Q9^2R F<
!qGK! 4;WU7 Q7^N -1^7 ^ | -Q7U9 Q9^> Q7J | -Q9U9 ^ 0.4;W ^
Q9< 0A^A 0A^V 0A-32@1// | 0A-32@1// ^ D | C ^ -.Z; > F<
!gK! Q9@1// F<
!tK! ZJ F<
!uGK! 0J F<
!AGK! -.+Q9U9
!BGK! 24^Q9L F<
!XC! 0U2 @01REG1
!KC!
!VK! -1U2 @01REG1
!YK!
!XG! 1U2
!REG! 0U0 4;WU7 Q7^N -1^7 ^ | ^ TU8
Q8^R @^U8//
Q2^> @:^U8/:Q/ Q8:@^U8// @:^U8/U0/ ^
@:^U8// .Q7/
Q2^> @:^U8:// ^
@:^U8/X/
Q8:@^U8//
@:^U8// -Q7^< Q7J ^ Q2^N -(Q/ Q8:@^U8// @:^U8/-Q0)D ^ 0.4;W/
M8 | @01ERR1 ^ | @01ERR1 ^ F<
!VGK!
!EVG! ^TU7 Q7^R .U8 Q3*= @1M/ | @1G/ ^ Q7@1// Q8..X8 -2D Q9< M8 >
| @01ERR1 ^ F<
!wGK! @^U8/TECO Command/ 6M5^N 27@:^U6// Q9< M6 > ^ F<
!XGK!
!YK! @ER// G* ^S*= @01ERR1 ^
^SD @EW// G*
Q2-^X^N ^S*= Q9< Y > | ^SD @01ERR1 ^
|^S^N ^SD Q9< P > | @01ERR1 ^ ^ F<
!yGK! 4;WU7 Q7^N -1^7 ^ | ^TU2 Q2^R @1G/ Q2@1// .-2..X8 -2D
.^Q7D M8 0.4;W | @01ERR1 ^ ^ F<
!VC! 3;W-3;W F<
!WC! 4.0;W 155^T @^A/=/ F<
!V^! ^ F<
!YK! @ER// G* ^S*= @01ERR1 ^
^SD @EW// G* ^S*= Q9< Y > | ^SD @01ERR1 ^ F<
!AC! @ER// G* ^S*= @01ERR1 ^ ^SD
Q9< ^E^N .U7 ZJ 12@1// Q7J ^ A > F<
!ZGC! @EW// G* ^S*= @01ERR1 ^ ^SD 1U4 @^U8/Exiting/ 0;
!CC!
!ZC! 0U4 @^U8/Returning control to TECO/ 0;
!qGK!
!qGC! -1U4 @^U8/Aborting edit/ 0;
!ERR!
!CGK!
!DGK!
!K!
!G!
!GC!
!GK! 7^T 32768W F>
!QK!
!QK! M9
>
QIET
155^T @^A/[2J/ 155^T @^A/[J/ 10^T :G8 @^A/.../ 155^T @^A//
Q4^N 155^T @^A/[?21/ Q4^> EC 13^T 10^T EX | HK EK 13^T 10^T EX ^ ^ 13^T 10^T
J6 |C |L |W
S
@EI//
MI
SS

```

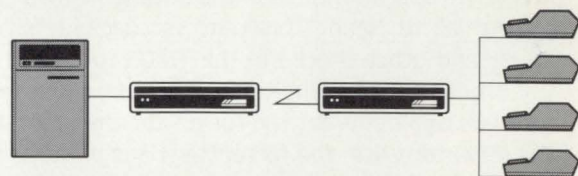
Minicomputer users:

Timeplex offers a one-stop system solution for communicating with multiple remote terminals. Economically.

Asynchronous statistical multiplexer

Synchronous statistical multiplexer

High speed modem



The Timeplex E/SERIES is a complete data concentrator system designed to economically link clusters of remote terminals to your minicomputer.

E/SERIES: Cuts communications costs. Suddenly, saving communications costs by linking several terminals to one shared telephone line becomes easy.

Unlike the competition, the Timeplex E/SERIES simplifies the challenge of point-to-point communications by incorporating three functions in a single compact unit. One system offers you a statistical multiplexer supporting 4 to 16 asynchronous channels, *plus* an optional statistical multiplexer for an additional synchronous channel, *plus* an optional integral high speed modem.

E/SERIES: Puts it all together. Putting three functionally distinct modules in one enclosure eliminates external communications units and bulky, expensive cables. And, a minicomputer interface option further reduces costs. The result: System planning and installation is extremely simple. Reliability is enhanced. Costs are dramatically reduced.

Free step-by-step Guide. This easy-to-understand booklet contains all the facts on how to remote your terminals, simply *and* economically. Just write or call Timeplex for your free copy.

For the name of the E/SERIES stocking distributor nearest you, call 201-368-0736.

Timeplex, Inc./One Communications Plaza/
Rochelle Park, N.J. 07662.

CIRCLE 71 ON READER CARD



**The technology leader
in data communications**

THE RSTS/E SYSTEM MANAGER

By Jeffrey R. Harrow, 485 Creekview Dr., Stone Mountain, GA 30083

Welcome to the anniversary issue of the RSTS Pro. I've taken a moment to look back through these issues and I'm **really** pleased at the amount of directly applicable information which I've extracted from the many articles I've read and is not available from any other source. Carl and Dave, here's one individual who appreciates what you're doing and hopes that it continues and grows.

Last issue I left you with a somewhat cryptic indication that there was a problem with using the TU77 tape drive. It appeared that the drive could write data on the tape which in reality didn't look like the data that it **thought** it had written, yet the system **did not** catch this error when the drive/formatter combination did its "implicit read after write."

Let's take a brief look at what I mean by an "implicit read after write":

The path the tape follows begins on the supply reel, goes past the Erase Head, then the Write Head, and finally past the Read Head on its way to the take-up reel.

During a "Write" operation, the Erase Head puts the tape in a "no data" condition, the Write Head writes the data onto the tape, and then the Read Head (even though we are in a **write** operation!) reads the data which was just written by the Write Head. This data is checked for validity via parity and other checks in the TM03 formatter.

Under normal conditions, if the data just **read** does not compare against what the formatter **thinks** that it just told the drive to **write**, the formatter flags an error condition to the MM Driver software (part of RSTS/E) which attempts to re-write the block of data using a number of industry accepted techniques (such as writing a Long Inter-record Gap over the "bad" section of tape and trying again).

Therefore, you should almost **never** be able to write a tape which you feel is "good" and is not later readable on, at least, the same drive.

This problem came to light when I began noticing that, during the COMPARE phase (which is another full Read of the tape) of BACKUP (you **do** use a COMPARE, don't you?) I was occasionally getting UNEQUAL COMPAREs on blocks of some files which I was able to verify had not changed during the BACKUP procedure. Additionally, I would usually see BAD BLOCK ON BACKUP VOLUME (ON COMPARE) errors at the same time (indicating that the drive could not read a block on the tape).

An investigation of the error log indicated that we were **NOT** taking any correctable "write" errors, much less UNcorrectable "write" errors while this tape was being written, yet, just minutes later, the same tape could not be read even after RSTS/E (the MM Driver) did its 16 retries! Further

tests indicated that, indeed, the data in these blocks on the tape **was not** invalid.

To make a **very** long investigation short, DEC determined that there was a problem in their technical instructions for setting the Ramp-up speed of the tape drive's capstan (and are correcting this problem). Furthermore, the software group found a "hole" in the MM Driver which they felt would account for this problem and sent a software specialist out to test the patch.

The software specialist had Field Service intentionally misadjust the tape drive and reproduced the error condition. He then installed the patch **and it didn't resolve the errors**, so he and his patch went away.

About a month later, after I had experienced the problem several more times, DEC finally indicated that they **still** felt that the patch (which is going to be published in the Software Dispatch) would cure the problem and that it didn't work when the tape drive was intentionally misadjusted because of the magnitude of the misadjustment. In any event, the patch is now installed in my operating monitor, and I'm waiting to see if it reoccurs.

I have just two reservations at this time: first, what I seem to be hearing is that the error detection/recovery process should only work if the drive isn't **too** broken; and second, I have provided DEC with several instances of documented error logs where the same thing has happened on a different type of tape drive (TE16), with a different formatter (TM02), using a different software driver (MT), on a different type of PDP-11 . . .

I'll keep you informed and still recommend that you utilize a COMPARE **whenever** possible!

On another subject, Bob Nixon from England wrote to me with a suggestion for providing system protection for the Basic Primer CMI package. With the executable program in the user's account given a protection code of <232> (Temporary Privilege) and the data file in the user's account (which that program reads) given a protection code of <63> (no Read, Write, or Delete access to the owner), the user should not be able to modify the data file (and hence change the name of the program to which the executable program will later CHAIN with Temporary Privilege).

Unfortunately, while the owner of a file **can't** do any of **these** things to a file with a protection code of <63>, he **can** RENAME the file to a protection code of <60>, and then perform whatever modifications he desires, thereby seriously breaching the system's security.

Bob, thanks for your input and I would like to hear from anyone who is interested in these topics. — See you next issue.

REPRINT POLICY

All content in this publication is copyrighted material.

All reprints must be purchased from M Systems, Inc. No other reprints are authorized.

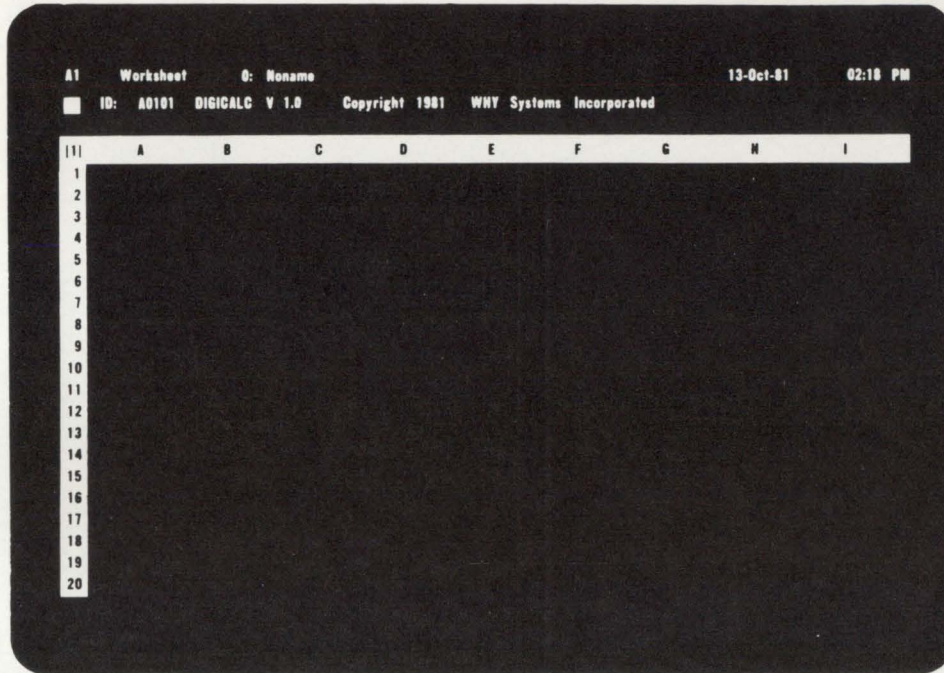
All reprints shall contain both a cover and a subscription blank.

Price quotations available on request.

WHY SYSTEMS

announces . . .

DIGICALC™



THE ELECTRONIC SPREADSHEET FOR DEC COMPUTER SYSTEMS

APPLICATIONS

Financial statements
Business forecasting
Resource management
Investment analysis
Job costing
Performance analysis
Cash Flow analysis
Error analysis

EASY TO USE

Built-in automatic training procedure
Instant HELP available at the terminal
User-friendly / interactive

SAVES MONEY

Reduces demand on information system personnel
Eliminates long hours of "what if" calculations
Report and form printing costs reduced or eliminated
Low, one-time investment

PROVIDES INSTANT REPORTS

Choice of formats
Working copies or board-room quality

- AUTOMATIC CALCULATION
- EXTENSIVE MATH FUNCTIONS
 - ALGEBRAIC
 - LOGICAL
 - FINANCIAL
 - SCIENTIFIC
 - USER DEFINED FUNCTIONS
- TEN KEY NUMERIC DATA ENTRY
- EXTENSIVE **HELP** AT TERMINAL
- INCLUDES SELF TEACHING MODE
- WORKSHEET CONSOLIDATION
- VARIETY OF "BOARDROOM" QUALITY REPORTS
- SAVES AND RECALLS WORKSHEETS

ON RSTS/E, RSX-11M, VMS
SOON ON RT-11, TOPS 20
VT-100 OR MOST EMULATORS

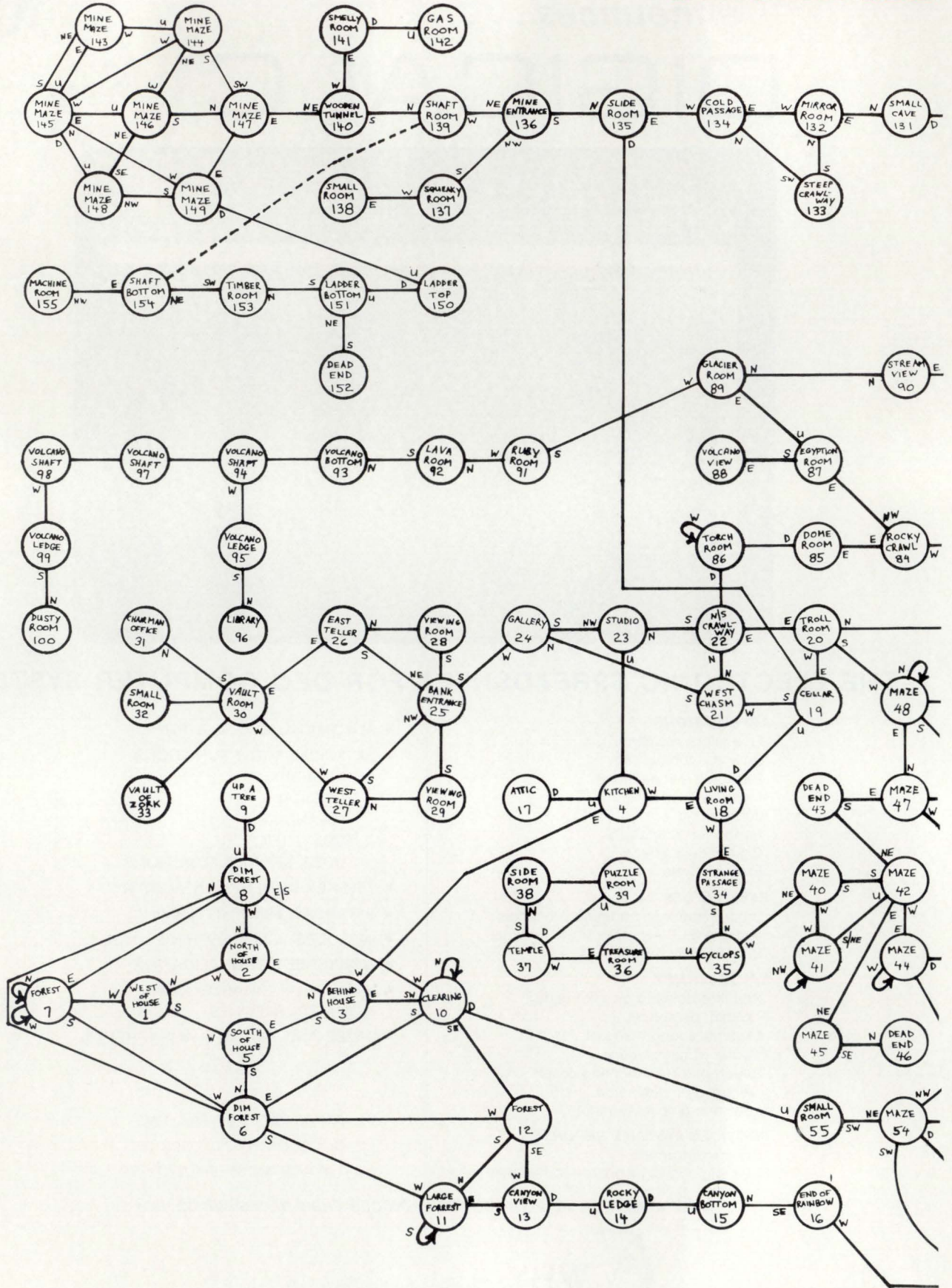
NOW AVAILABLE AT INTRODUCTORY PRICES THRU DECEMBER 31, 1981

**WHY
SYSTEMS
INCORPORATED**

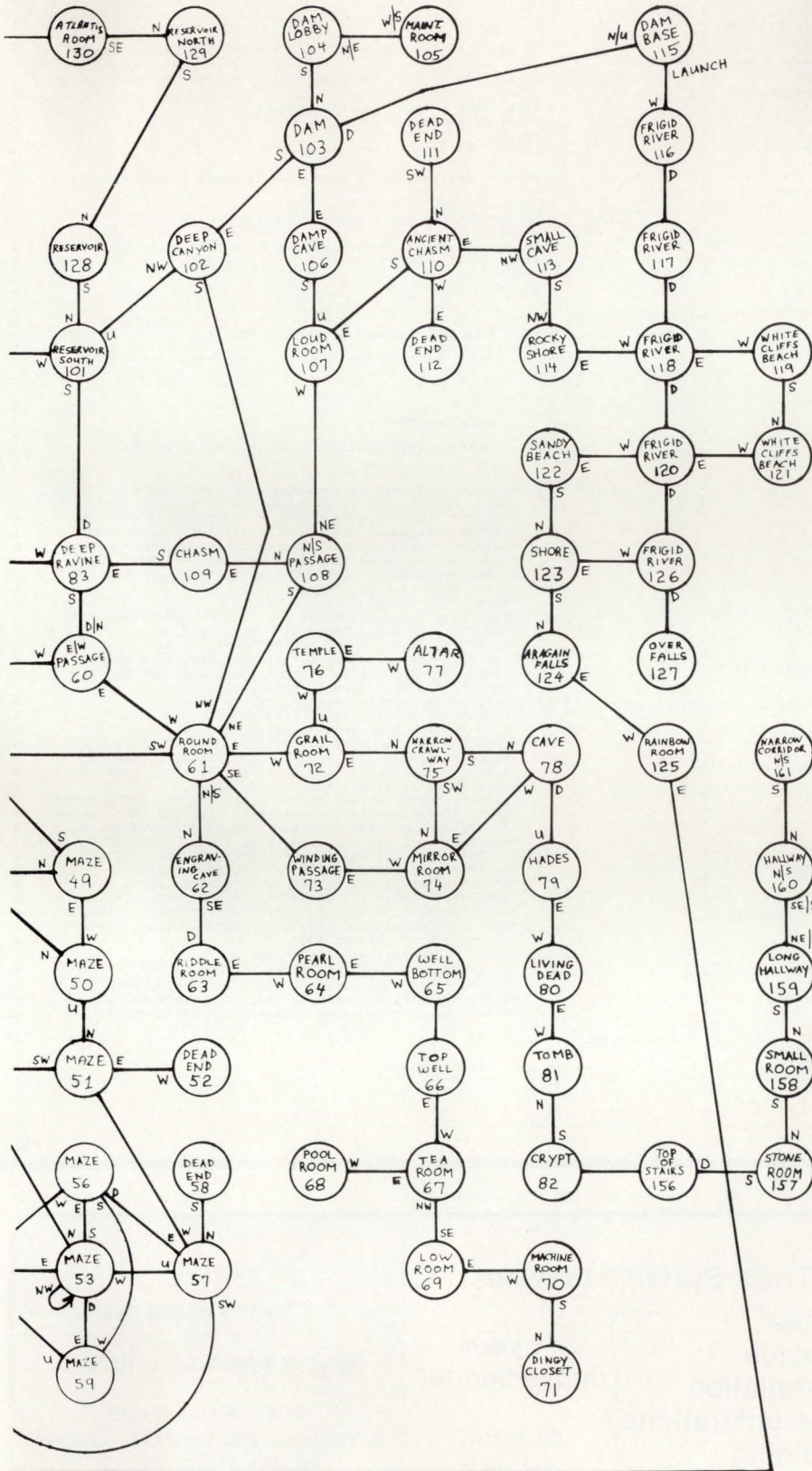
17130 Avondale Way, N.E.
Redmond, Washington 98052
(206) 881-2331

CALL TODAY FOR A DIAL-UP DEMONSTRATION

DUNGEON MAP

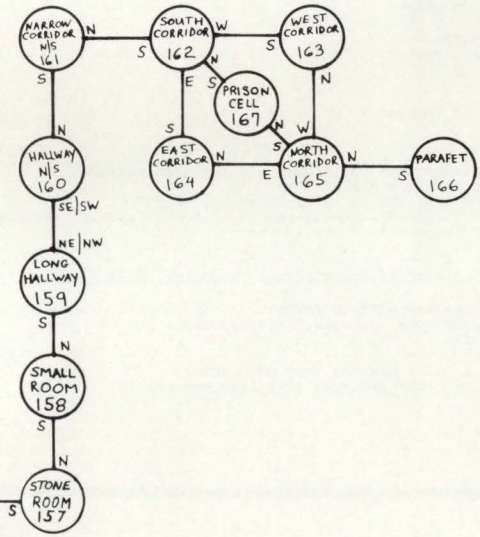


By Brian Lomasky, 528C Shennecossett Road, Groton, CT 06340

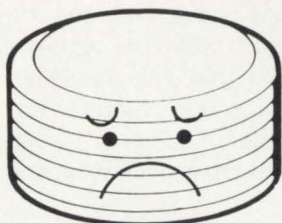


MM	MM	MM	MM	MM	MM	MM	MM
MM	HH	SS			SS		MM
MM	SS		MM		LL	SS	MM
MM					MM		MM
MM	SS	LL			SS	GG	SS
MM					SS		MM
MM	MM		MM				MM
MM	MM	MM	MM	MM	MM	MM	MM

PUZZLE ROOM
 MM=MARBLE WALL
 SS=SANDSTONE WALL
 LL=LADDER
 DD=STEEL DOOR
 GG=GOLD CARD
 HH=HOLE IN CEILING

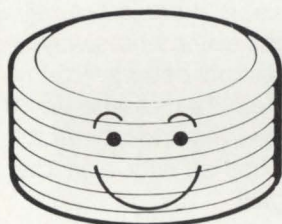


OPTIMIZE your RSTS/E* disks with DSKBLD.



FILING PROBLEMS YOU MAY BE PUTTING UP WITH.

- Fragmented UFD's
- Excessive arm movement.
- Poor file clustersizes.
- Excessive FIP overhead.
- Scattered files.
- Disk bound response problems.



DSKBLD HAS THE SOLUTIONS.

- Contiguous UFD's.
- Minimized FIP overhead.
- Organized files.
- Reduced arm movement.
- Optimum clustersizes.
- Lower disk access overheads.

Overtime for disk management is greatly reduced or eliminated. Disks get rebuilt more often because it's painless.

DSKBLD is an extremely fast RSTS/E disk-to-disk utility which:

- Supports mixed disk types.
- Untangles directory links.
- Creates contiguous UFD's to proper size.
- Creates accounts in sorted order.
- Locates files adjacent to owner's UFD.
- Optimizes file clustersizes.
- Saves accounting data.
- Is easy to use and fully documented.
- Provides logging and statistics.
- Is inexpensive.

INTRODUCTORY PRICE \$500.

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

MANUS
SERVICES CORPORATION

AUTHORIZED
digital
COMPUTER DISTRIBUTOR

HOME OFFICE CONTACT:
Gary Vowels
1700 Westlake Avenue N.
Lake Union Building
Seattle, Washington 98109
206-285-3260

PORTLAND OFFICE:
Don Shafer
9498 S.W. Barbur Blvd.
Portland, Oregon 97219
503-245-1261

Send me more information on the DSKBLD RSTS/E Disk.

Name _____
 Title _____
 Company _____
 Address _____
 City _____
 State _____ Zip _____
 Telephone _____
 (Area) (Number) (Ext.)

SYSTEM MANAGEMENT FOR RSTS/E

By Gary W. Miller, System Software Manager, Garden Way Manufacturing

NOTE: Due to the tremendous amount of code, all programs referenced here are available on the distribution tape.

For the last eight years I have been working on RSTS/E systems, in that time the need for additional programs for system managers has become apparent. Since I have been a system manager for the last four years I feel that I am fairly well qualified to break down those need into areas where the data is available from RSTS/E.

What I have done is to create a group of programs to supplement or replace some of DEC's CUSPS. With these programs I am able to charge out the system time to users on a consistent basis as well as charge for disk blocks. I am also able to charge for paper usage on either a block or page basis. The following paragraphs will explain each program or modification and how they enable me to give management a 'reasonable' figure to use for charge backs. This software was not intended to give a one-hundred percent figure because the system cost and development time would be unacceptable.

The first program in the package is the ACCGWM.BAS program, which is a replacement for DEC's REACT.BAS. This program allows the system manager to add, delete, and change information for an account on the system. The program keeps the information on the account in a data file for use by other parts of the billing package. In addition to the information that the DEC program requires this program needs an account description and client code be entered. The accounts are stored using logical names instead of physical device names to allow for packs being placed in different drives.

The program BILLER.BAS gathers the system time and number of disk blocks in each account that is in the data file created by ACCGWM.BAS. The program reads thru the data file to get the device and ppn of the account and then uses the read and reset accounting sys call (Prog Man 7-89) to read the accounting information from the system. This information is accumulated in the data file after the system data is zeroed. The program then goes thru the account accumulating disk blocks returned by the disk wild card directory lookup sys call (Prog Man 7-100). The total disks blocks in the account is added to the data file and the sample count in the file is incremented as well. On our systems we run this program every night to allow us to get a 'reasonable' average disk block number.

The MONTH.BAS program reads the data from the disk file created by ACCGWM.BAS and prints it out in either ppn or client code order. This program will also allow the user to zero out the data at the end of a billing period. The user can also request that the program print the passwords on the report if that is required.

The PASWRD.BAS program allows the user to change his/her password at any time without the need to contact the system manager. This also means that the responsibility for account security is now on the user, where it should be. This ability would shake up some shops because there

maybe times when someone in Data Processing may need to get into the user account and getting the password could be a problem. In a later section this problem will be addressed as to my solution to the problem.

The next section of this article will deal with modification to the DEC CUSPS that I have made to control use of the system as well as measure what the system is doing.

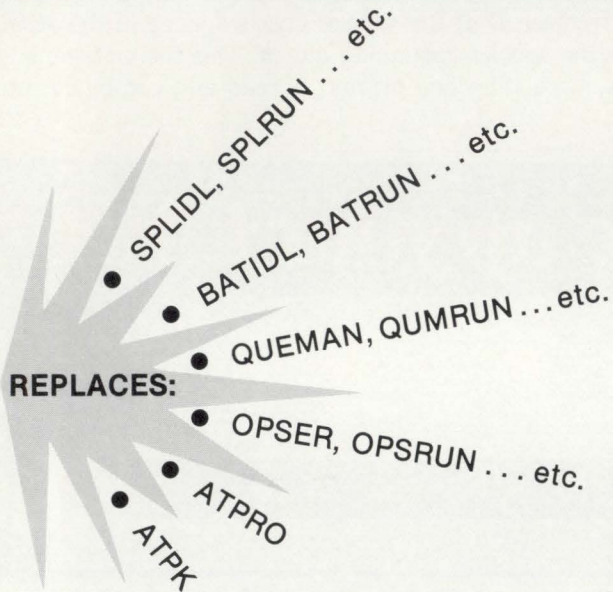
The first modification that was made was to the LOGIN.BAS program to add special features for ease of system use. There are a number of features that were added so I will just list them with text to explain what they are and why they were made.

1. The first change to the program was to force the user to see the \$NOTICE.TXT file, if there is one, all the time. This was done to prevent users who always use '/' on login from missing important notices. I think a better way would have been to print a message that there was a notice on file and maybe even print the first line of the message so if the user had already read the message they need not read it again.
2. The next change was to allow the programming department the ability to get into user accounts without knowing the password. On our system all of our programmers are privileged users, even though this is not the usual case the ability to move into other account is a requirement of most programming departments. What was done was to create three super passwords with certain abilities to each. There is one that will let you into any non-privileged account, one to let you into any privileged account other than those connected with system management, and the last one will let you into any account on the system. Now I realize once you have let a programmer into a privileged account they could write a program to read passwords from the system, but in a reasonable shop this may be something you could live with. With this patch just for non-privileged accounts you could then live with users controlling their own passwords.
3. The next change was build a crude auto-baud feature into the program for dial up lines. This only affects logged out users on dial-ups. When the line first comes active the line is set up to come up at 1200 baud with the '/RING' option of TTYSET since the fastest dial-up here is that speed. The keyboard is then opened binary mode and the user must type CTRL/C, the code then checks the buffer for a CHAR\$(3%) and if found then the speed of the line is correct, otherwise the line speed is changed to 300 baud and the user will have to type CTRL/C again. The same check is made and if the speed is not correct then the line speed is set to 110 baud and then the program continues at the logged out entry. This patch means that the user will have to type CTRL/C at most two times to set the line speed since I only support the three speeds.

VERSION 2.1 NOW AVAILABLE:

QUE-11 — V2.1

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 \$995 single cpu license
- Trial Version \$100

For more information contact:

On Track Systems, Inc.
P.O Box 245
Ambler, PA 19002-0245
Phone: 215/542-7133

master file by the program SPLPUF.BAS. How this master file is used is up to the system manager. The recommended procedure would be have all queues be done with a job name describing what department the report is for and what form the report is being printed on. In our case we use the first three characters of the job name for the department and the last three for the form being used. This allows the master file to be sorted by job name/file name and then a report printed breaking on change in file name then breaking on change in form type and then breaking on department. This report can then be broken apart and then sent to the respective departments to show their usage.

The last program that will discuss is the program we use for our weekly backup disk to disk. Our policy is to use SAVRES on Tuesday, Wednesday, and Thursday nights to backup the system. All of these backups are done in IMAGE mode disk to disk. Since the directories on RSTS/E system seem to get very disorganized in a short period of time, the ability to rebuild the directories on a regular basis is beneficial. The program is set up to run on systems with at least three drives of similar capacity. The operator is prompted for the disk being backed up and the drive to write to. The destination drive is then locked and an attempt is made to dismount the pack in the drive, any failures are reported to the operator with appropriate corrective measures. To insure that the wrong pack is not mounted the program requires the the pack to be written to has the same pack ID as the source pack. When the correct pack is mounted then the program opens a PK and starts a dialog to copy the source disk to the output disk. The dialog starts by running DSKINT.BAS to clear the pack of information and then using UTILTY to mount the pack. After the pack is mounted the program MAKMFD.BAS is run to create a MFD on the output pack with the accounts in ascending order. The copy is then done by using PIP.SAV running at 28K to speed the transfer of data. When the copy is complete the pack is dismounted and the operator is requested to put the original pack back in the output drive. This procedure is not the best method for backup but the UFD's seem to be in good shape after the procedure. With this program we can copy four 300 mb disk drives in just under three hours.

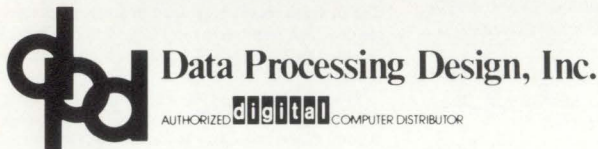
These changes have been made and the programs written to satisfy the demands of management, but they also serve the D.P. department and the user. I hope that this kind of information is of interest to you the reader. ♥



DON'T BACKUP.

If your system backup puts everything else on the back burner, you need SAVER for RSTS/E.™ It's the only disc to tape backup that doesn't monopolize your PDP-11™ system resources.

SAVER restores files easily, and creates optimally-structured discs. And it's supported by Data Processing Design, developers of WORD-11 and other fine products for DEC's family of mini-computers. Inexpensive at \$850.00. Call or write.



Corporate office: 181 W. Orangethorpe, Suite F,
Placentia, CA 92670 (714) 993-4160
New York Office: (212) 687-0104
Washington, D.C. Office: (301) 657-4098

CIRCLE 74 ON READER CARD

BENCHMARK DIBOL VS. BASIC+2

PDP/11-RSTS/E Environment

Analysis Conducted by Frank Metcalf, ICS Computing, Ltd., UK

Benchmark Overview

The following benchmark results show advantages for each language (BP2 and DIBOL). However, the more important advantages appear to be in favor of BP2.

Two of the more important considerations in a RSTS/E environment are disc accessing and CPU usage. In each of these areas, BP2 was clearly the most efficient. Also, job mix is of major importance in a RSTS/E environment. BP2 is clearly the best performing language.

The above points should be key considerations in evaluating languages and operating systems in any environment where development and production are on-going.

The general feeling, prior to the comparison, was that BP2 was a better language for the RSTS/E environment, which the benchmark has proven.

The attached should provide sufficient detail to support the findings.

Comparison of DIBOL vs. BASIC+2

System compared: Supplier masterfile maintenance 1300 records on VENMAS.

1. COMPARISON OF DISC ACCESSING

The figures represent the disc accessing counts for the data and index files only. Directory, DEVICE.DDF and overlay accesses are ignored.

Each activity was performed 5 times.

Activity		Reads		Writes		Total Accesses
		Accesses	Blocks	Accesses	Blocks	
Search fails	BP2	12	22	--	--	12
	DIBOL	157	157	2	2	159
Inserts	BP2	28	54	5	10	33
	DIBOL	240	240	18	18	258
Amends	BP2	17	32	5	10	22
	DIBOL	192	192	8	8	200
Deletes	BP2	17	32	5	10	22
	DIBOL	202	202	18	18	220

For the DIBOL systems, no count was taken of the accesses required to sort the index to include recent inserts or to remove deleted records from the data file.

The amount of disc usage has an important effect on machine performance. Thus, the above results point very much to BP2.

2. COMPARISON OF CPU USAGE

Each activity was performed 10 times.

Activity	BP2	DIBOL
Inserts	6.6	21.0
Amends	6.4	10.5
Deletes	6.0	7.0

For the DIBOL system, no count was taken of the CPU time required to sort inserted records into the index or to remove deleted records from the file.

This also points to BP2 and, as with 1, is an important factor on the 11/70's.

3. COMPARISON OF TSK FILE SIZES

WHEN YOU NEED
DEC...

TERMINALS
• VT-100 • VT-103
• LA34 • LA120
INTERFACES
MODEMS
PERIPHERALS
LSI/11 MODULES
PDP 11/03
PDP 11/23
SYSTEMS
Standard & Custom
Software Packages
WORD PROCESSING

UNITRONIX
CORPORATION

(201) 231-9400
197 Meister Ave.
Somerville, NJ 08876
TELEX: 833184

CIRCLE 25 ON READER CARD

Basic Plus 2 systems:

VENMNT. TSK	161	(176)
DIBOL systems:		
VENMNT. TSK	25	(32)
VENPRT. TSK	23	(32)
ORGVEN. TSK	22	(32)
VENCNT. TSK	7	(16)
SRTVID. TSK	27	(32)
111	—	—
Total	104	(144)

The figures, in brackets, represent the total blocks allocated, allowing for a disc clustersize of 16.

The file size above for BP2 will not decrease significantly for even very small programs. However, only 1 file is required for a module, whereas, 5 programs make up a DIBOL module. Disc space on 11/70 is not a problem, although the size of a directory could be. There is probably little advantage either way. Also this has a less important impact on performance than either section 1 or 2.

4. COMPARISON OF FILE SIZES

RMS uses more space to hold indexes and allow for inserts (depends on packing density).

RMS packs numeric fields more efficiently (in binary).

In the systems used for comparison, these 2 factors balanced to give equal file sizes.

However, the number of files in RMS system would be smaller since indexes and data files are incorporated in 1 file, unlike DIBOL.

5. COMPARISON OF TASK BUILD TIMES

Basic Plus 2:

1 min., 2.9 secs.	CPU time
18 mins.	Elapsed time
to build VENMNT.TSK	

DIBOL:

14 secs.	CPU time
3 mins.	Elapsed time
to build VENMNT.TSK	

6. MISCELLANEOUS

DIBOL programs are scheduled and run for each character entered. This produces (1) a heavy system load and (2) delays in echoing characters entered, which is annoying to a user.

BP2 is no more difficult to use than DIBOL, although it could take a little longer to learn.

BP2/V1.6 and V7.0 of RSTS/E have improved Task (TSK) file sizes, compilation time and overall performance. Benchmark times were not run against BP2/V1.6 and RSTS/E V7.0.

7. CONCLUSION

BP2 is clearly more suitable to a RSTS/E machine than DIBOL. The figures in numbers 1 and 2 indicate considerably more BP2 jobs could be run on the 11/70 than DIBOL. ♥

Stocking genuine DEC computer spares

Spare Parts for:

DEC PDP Computer Family
DEC Line Printers
DEC Video Screens
LSI-11
Others

Call Radgo. We handle genuine DEC parts manufactured by DIGITAL EQUIPMENT CORPORATION. We sell at factory prices with FACTORY WARRANTY. Radgo is stocked and staffed to help you with your requirements for most general purpose K & M series and many COMPUTER SPARE MODULES and COMPONENTS. We also stock a large assortment of COMPUTER SUPPLIES.

Radgo Sales Co.

To order or for free catalog call:
1-800-543-1986. Ohio customers phone
1-513-752-6880.

3988 McMANN RD., CINCINNATI, OH 45245
CIRCLE 75 ON READER CARD

Hardware Accessories For DEC Equipment Users

C-XX Overtemperature Protection System

Standard DEC PDP 11, VAX, and System 10-20 machines are NOT adequately protected from equipment damage due to high machine room temperatures. This unit provides aural warning signal and total system power shutdown with two customer adjustable temperature limits and approved interface to standard DEC AC power control system.

B-11 CPU Speedup for PDP 11/45 and 70

This timing generator modification allows 11/45 or 11/70 CPU instruction throughput rate to be increased by up to 15%.

Both products are attractively priced, install in a few minutes with no special tools or skills, and are allowed by DEC Field Service in contract systems. Call or write for further information.



Nassau Systems
P.O. Box 19329
Cincinnati, Ohio 45219
(513) 231-1283

DEC, VAX, and PDP are trademarks of Digital Equipment Corp.

OVER THE I/O PAGE

(or, 'A Trip Through Magic Kingdom . . .')

By Bob "MACRO MAN" Meyer

The following article describes in detail a method of accessing the I/O page of the PDP-11 while running under RSTS from a user-mode Macro program. Also we'll cover some possible uses for this type of programming, as well as some potential dangers.

This project came into existence a few years ago when a good friend of mine (he wishes to remain nameless, so we'll call him by his alias, 'Farquart') and myself decided how nice it would be if we could 'peek' into XBUF. We realized very soon that this was not at all feasible using existing techniques. "But, Farquart," I exclaimed, "RSTS certainly has access to the XBUF area, why can't we?". In other words, what is the difference between the Monitor accessing memory, and the user program accessing memory?

The answer: **Memory Mapping.**

In order to look around the system's memory at will, the user program must have the ability to re-map itself as needed; And in order to re-map one's own job image, we must have access to the Memory Management APR's. The following examples will demonstrate how this can be done on any RSTS machine. I would like to recommend, however, that you read this article thoroughly before trying any of this, and if you're not intimately familiar with the PDP-11 Memory Management hardware, do your experimenting with a stand-alone system.

Building the Library

The first step is to build a resident library (reslibs must be genned in, of course) with a length of 4K. This library will then be placed at the very top of the system's memory, and with a gentle 'shove', will land directly over the I/O page. The program below will do just fine:

```
.TITLE  USR07
.IDENT  /1.0/
USR07: .BLKW  4096.          ;BLOCK OUT 4K WORDS
.END
```

to assemble:

```
MAC
MAC>USR07 = USR07
MAC>1Z
```

now to link up the reslib:

```
TKB
TKB>USR07//HD = USR07
TKB>/
ENTER OPTIONS:
TKB>PAR = USR07:160000:20000  ;STARTING ADDR & LENGTH OF LIBRARY
TKB>STACK = 0                ;DON'T ALLOCATE ANY STACK SPACE
TKB>//
```

and finally, we create the library image file:

```
RUN $MAKSIL
[maksil's header]
Resident Library name? USR07
Task-build Resident Library input file <USR07.TSK>? <CR>
Include symbol table (yes/no) <Yes>? NO
Resident Library output file <USR07.LIB>? <CR>
USR07 build in 4 K-words, 0 symbols in the directory
USR07.TSK renamed to USR07.TSK<40>
```

We now have a USR07.LIB file in the current account. To put up this library we type:

```
UT ADD LIBRARY [p,pn]USR07>40> /ADDR:xxx/STAY/RW
```

where:

[p,pn]is the current account, and xxx is the amount of memory on your system minus four so if your system has 512K Words, specify '/ADDR:508'.

Finally, we convince RSTS to let our library live over the physical I/O page of the system. First we log into [1,1] and run ODT:

```
RUN SODT
[odt's banner]
File <Memory>? <LF>
```

Type the letter 'C' to get a listing of all monitor tables, watching near the bottom for 'RTSLST'.

```
JOBCNT 072564
RTSLST 056306
```

(note: these values are very system dependant)

The list of Resident Libraries, 'LIBLST' starts just after 'RTSLST'. So in order to access the first entry in the RESLIB list on this system, we would look at location 56310:

```
*56310/ 006540
```

However, we're interested in the last library in the list (the one we just added:USR07), so we must traverse the library links as follows (in this case, we have two other libraries before USR07):

```
*56310/ 006540 @
006540/ 005200 @
005200/ 015600 @
015600/ 000000
```

We're now at the start of the memory control sub-block for the third library on the system. (see common.mac for more details.) To verify this, we can check the name of the library:

```
015600/ 000000 <LF>
015602/ 103112 %USR          ;use '%' to get rad50 string
015604/ 140512 %07
```

If you get the library name in the second two locations of the control block, you're in the right place. If not, back up and start over.

Now that we've found the entry for our library, we can change it's physical start address so that the Monitor 'thinks' it exists over the I/O page:

```
15620/ 056400 <i/o page address> <CR>
```

where <i/o page address> is 177600 for the 11/70 & 11/44, and 7600 for all others.

Accessing the Library

A simple way of trying out this new Resident Library is to link RSXODT with it, and nose around a bit: (you should be somewhat familiar with ODT at this point) First

HAVE WE GOT A MESSAGE FOR YOU!

SPECIAL
DELIVERY!

INTECOM brings to RSTS users an extremely versatile message system. Much more than 'electronic mail,' **INTECOM** is an application package designed expressly for controlled user communications, correspondence management, and item scheduling.

INTECOM can be used for short, paperless office memos; for lengthy, printed reports; or for any text file exchanges. **INTECOM** interfaces with the editor of your choice for complete text formatting capabilities.

A built-in calendar function permits you to date messages for automatic future transmission to any valid **INTECOM** user -- including yourself. Incoming messages may be placed on hold for specific periods of time. Communications on hold disappear temporarily from your 'in-basket' and re-appear on the scheduled reactivation date! **INTECOM** even notifies the user's designated terminal when incoming messages arrive. **INTECOM's** unique message labelling feature permits you to store your own key identifiers for message retrieval. You can identify messages by keyword, sender, receiver, date, time, or subject.

INtelligent TExt COMmunicator Specifications

- ✓ user name logicals
- ✓ user/account validation
- ✓ menu operation with full HELP text
- ✓ IN, OUT, and HOLD transaction databases for each user
- ✓ message reply routing
- ✓ text file *or* text record manipulation
- ✓ hardcopy queueing (message selectable)
- ✓ message labelling for keyword retrieval (incoming and outgoing)
- ✓ full message journalling (user selectable)
- ✓ message waiting light for VT100 (user selectable)
- ✓ bell notification of message receipt (user selectable)
- ✓ incoming message forwarding with supplementary text
- ✓ message sending to multiple users
- ✓ message serialization, date and time marking
- ✓ dated message sending (relative days or absolute date)
- ✓ dated message suspending (relative days or absolute date)
- ✓ multiple account operation for each user
- ✓ text editor interface from menu control (user selectable)
- ✓ message subject scanning with selective read
- ✓ incoming, outgoing, and dated message reporting

INTECOM's features can be applied to every application environment. Contact us for the impressive details. **INTECOM** is being offered at an introductory price \$2880. Demonstration plans are available.

INtelligent TExt COMmunicator ...

another **QUALITY** software product from:



**North County
Computer Services, Inc.**

2235 Meyers Ave.,
Escondido, California 92025
(714) 745-6006, Telex: 182773

RSTS/E MONITOR INTERNALS

Part 4

By Mike Mayfield, Northwest Digital Software, Box 2-743, Newport, WA 99156

This is the last article in my series on RSTS/E monitor tables. By this time you should be getting a better understanding of how RSTS works inside and where to find the information you need for specialized systems programs and system tuning. Now all of the monitor's secrets are just a PEEK away.

This last article covers the data structures used by send receive and CCL commands. These data structures are the RIB and PMB for send receive and the CCL block for CCL commands.

5.0 SEND RECEIVE

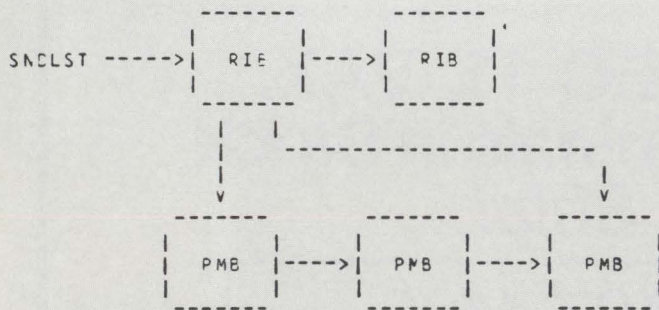
The send/receive capability of RSTS allows information to be sent between programs. The data structures required to provide this capability are the receiver ID block (RIB) and the pending message block (PMB).

Each program that has declared itself to be a receiver is allocated a receiver ID block. This ID block is pointed to by the job's second JDB.

The receiver ID blocks on the system are linked together in a list of message receivers. When a message is sent this list is searched for the specified receiver.

Each message that is waiting to be received has a pending message block (PMB) associated with it. The PMBs for all of a particular receiver's pending messages are kept in a linked list pointed to by the word S.MLST in the receiver's RIB. The list is kept in first-in first-out order.

The following figure shows the relationship between the RIBs and the PMBs:



For more information on the contents of the receiver ID block and the pending message blocks see the "RSTS/E Programming Manual" and "Network Programming in BASIC-PLUS and BASIC-PLUS-2".

5.1 RIB — RECEIVER ID BLOCK

When a job declares itself a receiver (using the .MESAG monitor directive or SYS(6 + 22) in BASIC-PLUS) a receiver ID block (RIB) is allocated. The receiver ID block contains all the information necessary to allow receipt of intra-CPU and inter-CPU messages. The RIB is pointed to by the word J2MPTR in its owner's JDB2.

In addition, all the RIBs in the system are linked together in a list. The first element in this list is pointed to by the location SNDLST. Since the receiver name ERRLOG is always present in the system, SNDLST will never be 0.

The format of a receiver ID block is as follows:

	Link to next RIB	0	S.LINK
	Receiver ID (6 bytes of ASCII)	2	S.RCID
		4	
		6	
S.OBJT 9	Object type Job number *2	8	S.JBNC
	Unused Access flags	10	S.ACCS
	Buffer maximum	12	S.BMAX
S.MCNT 15	Pending count Message maximum	14	S.MMAX
	Pointer to first PMB, if any	16	S.MLST
	Pointer to last PMB, if any	18	
S.LCNT	Link count Max link count	20	S.LMAX
	Pointer to first logical link block	22	S.LLST
	Pointer to last logical link block	24	
		26	
	Reserved for network use	28	
		30	

Offset	Symbol	Description
0	S.LINK	This word contains the address of the next RIB in the linked list. If this is the last RIB in the list this word will contain a 0.
2	S.RCID	These six bytes contain the receiver name as six bytes of ASCII. If the receiver name is less than six bytes in length it will be filled with trailing zero bytes.
8	S.JBNO	This byte contains the job number times two of the job to which this RIB belongs.
9	S.OBJT	This byte contains the object type for use in DECNET messages.
10	S.ACCS	This byte contains the access controls for this receiver (see 5.1.1).
11		Unused.
12	S.BMAX	This word contains the buffer usage restrictions specified at the time the receiver was declared.
14	S.MMAX	This byte contains the restriction on the number of pending messages specified at the time the receiver was declared.

Structured disks for all!

Announcing **REACT2.TSK**

- a 'REACT' replacement that locates & extends UFD's
- all standard 'REACT' functions (Delete, Standard, Enter)
- user specified location and length for new UFD.
- high speed — this product uses software developed and licensed by Software Techniques, Inc.
- distributed as an RSX or BP2 task on 9 track 800/1600 tape

**Introductory Price: \$200.⁰⁰
Single CPU License**

Nationwide Data Dialog

70 James Way
Southampton, PA 18966
(215) 364-2800

Call For Quick Service

CIRCLE 7 ON READER CARD

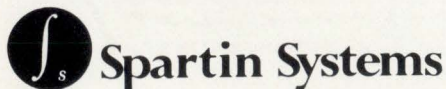
DEC/RSTS

Rapidly growing Houston, Texas
OEM seeking RSTS Programmer/
Analysts

- Suburban location
- Large data center with five 11/70's
- Data base applications development
- National timesharing services and OEM sales

For quick consideration call
collect or send resume to:

Murray Stinson
14925-A Memorial Drive, Dept. 389
Houston, Tx 77079
(713) 496-0771



CIRCLE 79 ON READER CARD

Offset	Symbol	Description
0	P\$LINK	This word contains the address of the next PMB in the linked list. If this is the last PMB in the list it will contain a 0.
2	P\$BUFA	This word contains the "contorted" address of the buffer containing the large message data. If the low order five bits of the address are zero the pointer is to a small buffer. Otherwise, the address has been rotated left seven bits and points into the extended buffer area, XBUF. If this word is 0 no large message data was sent with this message.
4	P\$TYPE	This byte specifies the type of message described by the PMB (see 5.2.1).
5	P\$SNDR	This byte contains the job number times 2 of the message sender.
6	P\$SPPN	This word contains the PPN of the message sender.
8		This word is unused.
10	P\$BREM	This word specifies the number of bytes remaining in the message buffer. It can be less than the original length of the message if the message was too long to fit in the user's buffer on a receive request and truncation was not requested. See 5.2.2 for information on the buffer format.
12	P\$PARM	These ten words contain the user specified parameters for small message send/receive and the network parameters for DECNET.

5.2.1 P\$TYPE — Message Type

If DECNET is not used, the message type (P\$TYPE) will always be -1 to signify local sender message received. However, if DECNET is used several other message types are possible, as follows:

Type	Description
-1	Local sender message received.
-2	Connect initiate received.
-3	Connect confirm received.
-4	Connect reject received.
-5	Network sender message received.
-6	Interrupt received.
-7	Link service received.
-8	Disconnect received.
-9	Link abort received.

5.2.2 Buffer Format

Each large message is stored in a buffer of up to 516 bytes using either a block of small buffers or a buffer from XBUF. The actual data of the message is preceded by a two word header, as follows:

Offset	Symbol	Description
0		This word contains the buffer size, in bytes.

2 This word contains the offset from the beginning of the buffer (including this header) to the first byte to be returned for the message data. It is initially set to 4 and is updated if a receive request overflows the user's buffer and truncation is not requested.

4 This word starts the message data area. It can be up to 512 bytes in length.

6.0 CCL — CONCISE COMMAND LANGUAGE BLOCK

Each concise command language (CCL) definition is stored in a CCL block. These blocks are allocated and deallocated dynamically from the small buffer pool.

The CCL blocks are kept in a linked list in the order in which they were defined. The first element in the list is pointed to by the location CCLLST.

The format of a CCL block is as follows:

Link to next CCL block	0
Pointer to CCL text within CCL block	2
Pointer to first optional character	4
Associated program's PPN	6
Filename of program to run (RAD50)	8
Filename extension (RAD50)	10
CCL command text (terminated by 377(octal))	12
Associated program's device name	24
Unit real flag Unit number	25
Pointer to first optional character	28
Parameter words, line number for RLV	30

Offset	Symbol	Description
0		This word contains the address of the next CCL block in the linked list. This word contains a 0 for the last CCL block.
2		This word is a pointer to the first character in the CCL command text string.
4		This word is a pointer to the first optional character in the CCL command text string.
6		This word contains the PPN of the program to run in response to this CCL command.
8		These two words contain the file name (in RAD50) of the program to run in response to this CCL command.
12		This word contains the file extension (in RAD50) of the program to run in

IT'S 2:28 AM

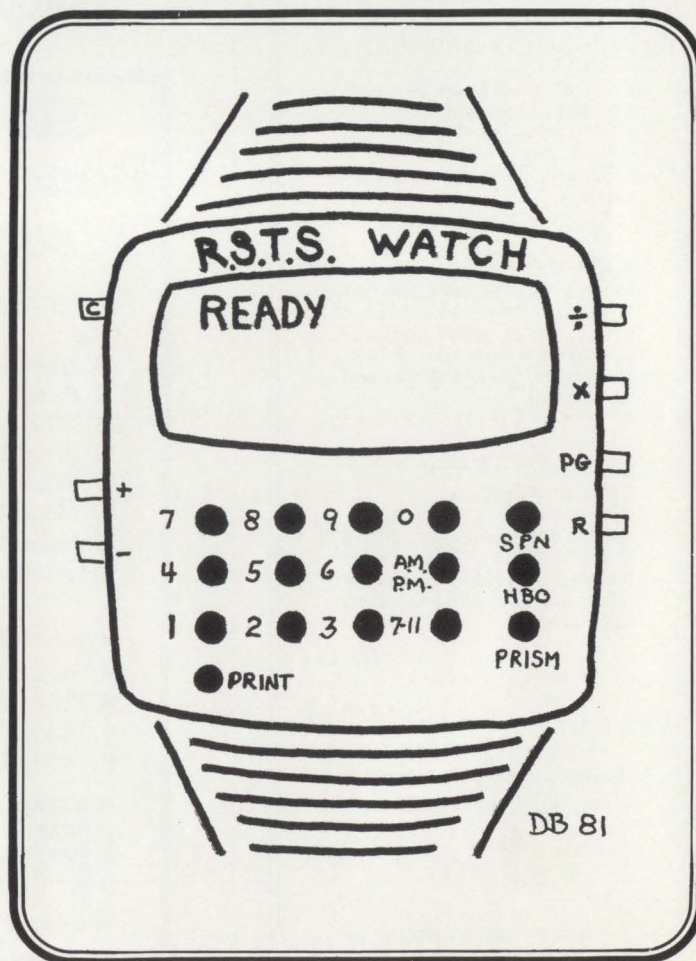
Some kid with a MODEM just figured out that you have 1,100; DEMO.

LOTS OF LUCK!

LOCK-11

SPD on Page 49

CIRCLE 80 ON READER CARD



THE RSTS EXTENSION

ADDING TERMINAL LINKS TO YOUR RSTS SYSTEM

By Kevin Paul Herbert

Copyright © 1981 by

Software Techniques, Inc., 5242 Katella Avenue, Suite #101, Los Alamitos, Calif. 90720

This is the first in a series of articles on adding new features to RSTS/E. Many RSTS/E users "wish" for enhancements to the operating system, either at DECUS or by sending DEC SPR forms, and often, these changes are incorporated into future versions of RSTS. More often, however, these wishes, despite their usefulness, are considered to be not worth the effort to develop and support, and are therefore not implemented. In this series of articles, I plan on providing patches to RSTS/E V7.0 that will make your RSTS system much more useful. Note that use of any of these patches will void your software support warranty. In addition, note that although every attempt has been made to insure that these patches do indeed function, no guarantee is made that these patches actually do work. With this in mind, welcome to The RSTS Extension.

Many system managers desire the capability to spy on other users (monitor all their terminal activity without being detected) or be able to communicate with other terminal users without the need to run a TALK program (so that the terminal can be used for both the communication of user messages and the demonstration of program runs and development.) In addition, many users would like additional functionality from keyboard "control" keys.

In this article, I will describe the addition of several terminal service enhancements. Five new "control" keys have been added to the terminal service. First, Control/E is a programmable function key. It is possible to program up to 32 characters to be entered to the system with the depression of the Control/E key. This function can be used to enter common responses (such as YES or NO), or to execute a common system command (such as SYSTAT). Continuing, Control/F can be used to list all of your job's open files. Although the Control/F code is listed in a previous version of the RSTS Professional, the original version conflicts with changes made to support the new features described in this article. To optionally add Control/F support and the changes detailed in this article requires a copy of TTOPNF.MAC written by Steve Davis. TTDVR.TEC is not needed, and will fail (creating an unusable monitor) if an attempt is made to use it with these changes. Further, Control/W can be used to erase the last word typed. This key is a compromise between Rubout, which erases the last character, and Control/U, which erases the last line. Control/X will cancel all un-processed type-ahead. This key can be used as a "panic" key to abort any unwanted type-ahead but not abort the current running program (as a Control/C would do). Finally, Control/Y allows a privileged user to detach his current running job.

The format for the call to program the Control/E key is as follows:

```
PRINT RECORD 8192%, STRING.TO.PROGRAM$:
```

Where STRING.TO.PROGRAM\$ is a 1 to 32 character string. All control characters except for <LF> and <CR> are ignored. Note that because of an idiosyncrasy in Basic-Plus, the trailing semi-colon is required. Also note that it is valid to program Control/E for other terminals that you own by including a channel number in the write.

In addition to the new control keys, terminal links have been added to the monitor. With terminal links, it is possible to spy and talk to other users without the overhead of a communication program. In addition, both users are free to work normally while a spy or talk link is initiated. When talking, both users see each others output, and when spying, the spying user sees the spied-upon user's output, but the spied-upon user is unaware of this operation. The capability to TALK is non-privileged (if the receiving user allows users to TALK to him) and the SPY capability is privileged.

I have supplied a utility to control the terminal link capability (TLU). It is also possible for user programs to use the terminal link calls in their own programs. The following text describes the basic mechanics of "terminal links", and the use of the new utility calls.

To add these new features to the terminal driver required adding additional data storage to the monitor. This was accomplished through modifications to TTDINT.MAC. It was necessary to add storage both on a system-wide basis and a per-terminal basis. Each terminal has a device data block (DDB). The DDB contains all of the information needed by the monitor and terminal driver to control a terminal interface on a per-terminal basis. In order to enhance the terminal service, the terminal DDB has been expanded by three words (I confess, this does reduce the maximum number of small buffers possible at a rate of one small buffer per 5.33 terminals). The first of these three new words is called TTFUNK. This word contains either a zero, indicating that Control/E is not in use, or points to a small buffer containing the function key text associated with Control/E. The second word is called TTLINK. This either contains a zero, indicating that the terminal is not linked, or points to a small buffer containing internally coded pointers and flags, and 26 keyboard numbers (it is possible to link to up to 26 users). This small buffer (the link table) is shared by all members of a link group ("conference" links use no more small buffer resources than do "one-on-one" links.) The third word added to the monitor is TTLNKS. This word contains the "talk status" (defined in the next paragraph), internal flags, and a user scratch pad byte (defined later). The modification to add storage for the terminal driver on a system-wide basis consisted of adding one word (LNKLST) to the monitor. This word contains the root of the linked list of all the link tables currently in use. Although the primary

purpose of LNKLST was for debugging during development, it would be possible, by studying the link table layout in LINKS.MAC, to write a program to give a link status report. LNKLST is always located at the address TTYHCT + 2 in the monitor. The value for TTYHCT is returned by the UU.TB2 (Get Monitor Tables Part II) UUU (SYS call). If there is interest in this application, I will consider writing such a utility in a future article.

Every terminal has an associated "talk status". This "talk status" determines how the monitor handles the create a link call. The currently defined talk states are OFF, QUERY, and ON. If a user's talk status is OFF, the monitor will return an error to any user attempting to create a link (unless privileged). The user with the talk status of OFF receives no notification of the attempt, and is completely unaffected. If a user is in the QUERY state, the monitor will return an error (identical to the OFF state), although the user in QUERY state will be notified that an attempt to link was made. If a user is in the ON state, the monitor will allow any user to create a link, and will display a notification message upon the creation of the link (unless inhibited by a privileged user).

The general format of a terminal link call is:

RESULT% = SPEC%(FUNCTION%, ARGUMENT% + SWAP%(FLAGS%), 0%, 2%)

There are four functions. They are link to a user, break all links, set talk status, and return talk status. Function code 4096% is the create a link call. The argument is the keyboard number to create a link to. In addition, privileged

users can select four flags in the link creation. They are: bit 0 to inhibit the monitor from announcing the creation of the link, bit 1 to ignore the destination user's link status, bit 2 to spy on a user, and bit 3 to not notify (query) a user if the destination's talk status is set to query. Any of these bits can be used in combination, although there are illogical combinations (for example, spy implies ignoring talk status and

no notification, and ignoring talk status means that a user will never be queried).

Function code 4097% is the break all links call. This call takes no arguments or flags, and removes a user from any links he might have. Function code 4098% is used to set talk status. The argument is the talk status to set and is a bit oriented field. If bit 0 is set, a user's talk status is "on" and any user may link to this user. If bit 1 is set, a user's talk status is "query", and this user can not be linked to, but will be notified when any other user makes an attempt to create a link. If bit 2 is set (the bit is privileged), this terminal's talk status will "stay" constant across jobs and can only be changed by a privileged user. If the "stay" bit is not set, a terminal will be set to the system default talk status when the

Now available for
PDP-11 and VAX-11.

How to count your chickens before they hatch.

Surprises can be expensive. Even good news can cost money if your company is not prepared for it.

With financial modeling you can avoid surprises and plan calmly for whatever the future has in store.

FINAR is the latest financial analysis and reporting system. It will help you plan:

- Budgets
- Cash flow
- Capital investment
- Project evaluation
- Forecasts
- Consolidation

All you need is a DEC PDP-11 with RSTS or a VAX-11, and FINAR—the Financial Analysis and Reporting Language.

If you'd like to know how to count your chickens before they hatch, call or write:

Finar Systems Limited
6000 E. Evans, Suite 2-300
Denver, CO 80222 • (303) 758-7561

New York • (212) 222-2784
Chicago • (312) 876-1081
Houston • (713) 960-0848
San Francisco • (415) 956-1178
Toronto • (416) 245-8473

FINAR

CIRCLE 51 ON READER CARD

job is created. Bits 3 through 5 are reserved for future use, and bits 6 and 7 are reserved for internal housekeeping flags (they can never be set and will always be read as zero). Any bits can be set in combination, but the ON bit overrides the QUERY bit. Note that if neither ON nor QUERY are set, the user's talk status is OFF, and any attempts to link to the user will be denied, giving the caller an error message (unless overridden by a privileged user). The flags consist of an

eight-bit number that is never used by the monitor and can be used to communicate a terminal type value to application programs. Finally, function code 4099% is used to return talk status. There are no arguments or parameters, and the result is the talk status as set by the set talk status call. For further information on these calls, see the TLU program.

Adding these new features to your RSTS monitor is a fairly simple procedure. The most difficult step will be doing the new SYSGEN to build a monitor with the enhanced version of the terminal service. For detailed information on system generation, see The System Generation Manual. The complete procedure for adding terminal links is detailed below.

Step 1 — Copy all necessary files from your RSTS/E distribution.

If your RSTS/E SYSGEN distribution is on tape, mount the tape write-locked on a magtape unit. If your distribution is on disk, mount the disk write-locked. Log in to your privileged account (the account where you would like to keep the patched version of the terminal service). For disk distributions, it will be necessary to logically mount the pack (with the mount command). The pack ID will be either SYSGNG (for RSTS/E V7.0-07) or SYSGNH (for RSTS/E V7.0-08). Run PIP and execute the following commands (note that this example assumes that your distribution is located on MTO:; if this is not true, change all references to MTO: to the appropriate device).

```
RUN [1,2]PIP
*SY: = MTO:[1,2]TTDVR.MAC
*SY: = MTO:[1,2]TTDINT.MAC
*SY: = MTO:[1,2]TBL.MAC
*SY: = MTO:[1,2]COMMON.MAC
*SY: = MTO:[1,2]KERNEL.MAC
*SY: = MTO:[1,2]KBDEF.MAC
*SY: = MTO:[1,2]CHECK.MAC
*SY: = MTO:[1,2].OBJ
*SY:[1,2] = MTO:[1,2]ERR.STB
*SY:[1,2]/PR:NOWARN = MTO:[1,2]SYSGEN.SAV
*SY:[1,2]/PR:NOWARN = MTO:[1,2]SYSBAT.SAV
*IZ (Control/Z to Exit)
```

Step 2 — Run the SYSGEN program

SYSGEN will ask questions dealing with your system's hardware configuration. For information on SYSGEN, see the System Generation Manual. Do not run SYSBAT when finished with SYSGEN, you DO have special requirements.

```
RUN [1,2]SYSGEN
.. Answer all questions to create a monitor ...
```

Step 3 — Install the patches

```
RUN [1,2]CPATCH CPATCH V7.0-07 RSTS V7.0-07 Softec Dev 11/70
File to patch - TTDVR.MAC = TTDVR.MAC
#TTDVR/CS:20188
... Patch will be installed ...
#IZ (Control/Z to return to previous question)
File to patch -TTDINT.MAC = TTDINT.MAC
#TTDINT/CS:18991
... Patch will be installed ...
#IZ (Control/Z to return to previous question)
File to patch - SYSGEN.CTL = SYSGEN.CTL
SYSGEN/CS:20746
#IZ (Control/Z to return to previous question)
File to patch - IZ (Control/Z to exit)
```

The next step is necessary only if you want Control/F support in your monitor. The code necessary for Control/F is not included in this article, although a command file for building it in is. Note that the Control/F patch file (TTDVR.TEC) will not work properly with these changes, and will cause unpredictable results. TTDVR.TEC is not necessary for Control/F installation, although TTOPNF.MAC is. It is assumed that a copy of TTOPNF.MAC can be found in the current account. If this is not true, copy TTOPNF.MAC to the current account. It will be needed during the SYSBAT procedure. DO NOT FOLLOW THE INSTALLATION INSTRUCTIONS THAT WERE SUPPLIED WITH CONTROL/F! ALL THAT IS NECESSARY FOR THIS CONTROL/F INSTALLATION IS THAT A COPY OF TTOPNF.MAC BE LOCATED ON THE CURRENT ACCOUNT. THE REST IS AUTOMATIC!

Step 4 (Optional) — Add Control/F support

```
RUN [1,2]CPATCH
CPATCH V7.0-07 RSTS V7.0-07 Softec Dev 11/70
File to patch - SYSGEN.CTL = SYSGEN.CTL
#SYSGCF/CS:36018
#IZ (Control/Z to return to previous question)
File to patch - IZ (Control/Z to exit)
```

The next step is to run the SYSBAT program to create a monitor. Follow the procedure in the System Generation Manual for use of the SYSBAT program. After running SYSBAT, shut down your system and install the new monitor. The new system should run normally, and many new features should be available. At this point, you may install any of the new feature patches using ONLPAT (the one for Control/F and non-privileged users WILL work) and build the TLU program (if desired). Note that the TLU program should be stored with a protection code of < 104 >. The procedure for building TLU is as follows:

```
RUN [1,2]CSPCOM
CSP>TLU.OBJ/OBJ = TLU.B2S
CSP>IZ (Control/Z to exit)
```

```
RUN [1,2]TKB
TKB>@TLU
```

```
RUN [1,2]PIP
*[1,2]< 104 > = TLU.TSK
*IZ (Control/Z to exit)
```

```
12-Nov-81 16:37 [1,13] TL0303.001 Page 1
```

```
! TL0303.001 - Make Control/Y function for non-privileged users
File to patch?
Module name? TER
Base address? ..PRCY
Offset address? 0
Base Offset Old New?
?????? 000000 001405 ? 240
?????? 000002 ?????? ? ^C
```

```
12-Nov-81 16:37 [1,13] TL0303.002 Page 1
```

```
! TL0303.002 - Set system wide default talk status
! Replace the "n" below with the desired default talk status.
! Note that this patch will not effect terminals with the
! "stay" characteristic set.
File to patch?
Module name? TER
Base address? ..DFTK
Offset address? 0
Base Offset Old New?
?????? 000000 000001 ? n.
?????? 000002 ?????? ? ^C
```

```
12-Nov-81 16:38 [1,13] TL0303.003 Page 1
```

```
! TL0303.003 - Set privileged talk states.
! Replace the "n" with the sum of the privileged talk states.
! If it is desired to make this call privileged, replace the
! "n" with a -1.
File to patch?
Module name? TER
Base address? ..KSPPR
Offset address? 0
Base Offset Old New?
?????? 000000 000004 ? n.
?????? 000002 ?????? ? ^C
```

12-Nov-81 16:37 [1,13] TL0303.004 Page 1

```

! TL0303.004 - Change a terminal's default talk status
! Note: For this to work, the STAY bit must be set. This implies
! that non-privileged users will be unable to change the talk status
! Change the "n" below to a keyboard number, and the "m" below
! to the desired talk status.
File to patch?
Module name? RSTS
Base address? KBDDDB+(n.*DDS.KB)
Offset address? TTLNKS
Base Offset Old New?
????? ???? 000001 ? m.14
????? ???? ?????? ? ^C

```

```

20001 &
!
! Start or MAIN &
!
! &
!
2010 GOTO 32767 IF E0% &
PRINT "TLU"; &
GET #0% &
MOVE FROM #0%, CMD$=RECOUNT &
CMD$=EDIT$(CMD$,-1%) &
! If this was a CCL entry, exit now. &
! Prompt for input. &
! Read a line from the terminal &
! Get it into CMD$. &
! Trim off any garbage. &

```

12-Nov-81 16:38 [1,13] TLU.B2S Page 1

```

11 Title: T L U &
!
! Version: V7.0-02 &
!
! Edit date: 20-Oct-81 &
!
! Written by: Kevin Paul Herbert &
!
! Date: 14-Oct-81 &
!
! Package: Terminal Links &
!
! Description: Terminal Link Utility Program &
111 Copyright (C) 1981 &
Software Techniques &
Los Alamitos, CA 90720 &
!
! This software is provided free of charge to readers of the RSTS &
! Professional and may be copied only with the inclusion of the &
! above copyright notice. This software, or any other copies &
! thereof, may not be provided or otherwise made available to any &
! other person except for non-commercial use and to one who agrees &
! to these license terms. Title to and ownership of the software &
! shall at all times remain in Software Techniques. &
!
! The information in this document is subject to change without &
! notice and should not be construed as a commitment by Software &
! Techniques. &
!
! This software is un-released and Software Techniques has no &
! commitment to support it at this time, unless stated elsewhere in &
! writing. &

```

```

2020 PRINT IF CCPOS(0%) &
GOTO 2010 UNLESS LEN(CMD$) &
RESTORE &
UNTIL 0% &
READ CMD.CHK$,SIG%,FLAGS%,VECTOR% &
IF (SIG% AND NOT.PRIV%)>=0% &
THEN SIG%=SIG% IF SIG%<0% &
MATCH%=FNMATCH$(CMD$,CMD.CHK$,SIG%) &
IF MATCH% &
THEN CMD$=RIGHT(CMD$,MATCH%) &
ON VECTOR% GOTO 3000,3100,3200,2020,3300,3400,3500,3600 &
! Return the carriage if needed. &
! Take input again if a null string. &
! Restore the command list. &
! Loop forever &
! Get the command, minimum match, flags, and vector. &
! If it is ok to do this command, &
! Negate the sig. chars if necessary. &
! Check for a match. &
! If we got a match, &
! Save only the argument to the command. &
! Go to the handler for that command. &
2030 NEXT &
! Finish the loop. &
2040 PRINT "Unknown command - "+CMD$ &
PRINT "Type 'HELP' if you need it." UNLESS E0% &
GOTO 2010 &
! Report an error. &
! Tell the user about HELP unless CCL entry. &
! Go back for more. &
3000 ! This handles the TALK, SPY, and LINK commands. &
! The TALK command is used to create a link. &
! The SPY command (privileged) is used to spy. &
! The LINK command (privileged) is used to TALK ignoring &
! the terminal's link status. &
! Privileges have been verified. &
! Flags% contains the appropriate bits to set. &
! CMD$ contains the keyboard number. &
KB%=VAL(CMD$) &
IF KB%>127% OR KB%<0% &
THEN PRINT "?invalid keyboard number" &
GOTO 2010 &
ELSE IF LEN(CMD$)=0% &
THEN PRINT "Keyboard number: "; &
GET #0% &
MOVE FROM #0%, CMD$=RECOUNT &
CMD$=EDIT$(CMD$,-1%) &
IF LEN(CMD$) &
THEN GOTO 3000 &
ELSE PRINT "?Keyboard number must be specified" &
GOTO 2010 &
ELSE I%=SPEC$(.CREATE.LINK%,KB%+FLAGS%,0%,2%) &
GOTO 2010 &
! Get the keyboard number. &
! If it is illegal, &
! give an error. &
! Go back for more. &
! If it is legal, &
! If it is a null string, &
! Ask for it. &
! Take the input. &
! Move it into a string. &
! Clean it out. &
! If it is null, &
! Give an error, &
! If not, &
! Try again to parse it, &
! If it is not null, &
! Try to create the link. &
! Go back for more (error trap reports errors.) &

```

```

201 &
! Modification History &
!
! Ver/Edit Date Reason (Who) &
! -----
211 V7.0-01 14-Oct-81 Initial conception. &
V7.0-02 20-Oct-81 Fixed STAY problem. (KPH) &
Updated for LINKS V2.1-04 &

```

```

3100 ! This is the BREAK command. &
! It is used to break all of a user's links (or SPYs). &
I%=SPEC$(.BREAK.LINK%,0%,0%,2%) &
GOTO 2010 &
! Break all links. &
! Go back for more. &
3200 ! This is the STATUS command. &
! It will list the user's current talk status and display &
! the terminal type byte if it is non-zero. &
STS%=SPEC$(.GET.LINK%,0%,0%,2%) &
PRINT "Talk status is "; &
IF STS% AND .TALK.ON% &
THEN PRINT "On." &
ELSE IF STS% AND .TALK.QUERY% &
THEN PRINT "Query." &
ELSE PRINT "Off." &
! Get the talk status. &
! Print Talk status is... &
! If on, &
! Say "On." &
! If not on, &
! If query, &
! Say "Query." &
! If not query, &
! Say "Off." &

```

```

1001 &
! Program Description &
!
! This program allows a user to create and break terminal links. &
! In addition, it allows the privileged user to spy on other users. &

```

```

3300 ! This is the HELP command. &
! We will print out how to use this program. &
PRINT "Commands are:" + CR + LF + &
HT + "TALK n" + HT + "Talk to keyboard N" + CR + LF + &
HT + "BREAK" + HT + "Break all links" + CR + LF + &

```

```

4001 &
! Variables and Arrays &
!
5001 &
! Compile time variables &
!
501 .Define .Version$ = "V7.0-02" &
.Define .Create.Link% = 4096% &
.Define .Ignore.Status% = 512% &
.Define .Spy.Call% = 1024% &
.Define .Break.Link% = 4097% &
.Define .Set.Link% = 4098% &
.Define .Get.Link% = 4099% &
.Define .Talk.On% = 1% &
.Define .Talk.Query% = 2% &
.Define .Talk.Stay% = 4% &
! Program Version. &
! Create a link call &

```

```

7501 &
! Functions &
!
! Name Line # Description &
! ----
FNMATCH% 14000 Check for string matches. &

```

```

3400 ! This is the STATUS command. &
! It will list the user's current talk status and display &
! the terminal type byte if it is non-zero. &
STS%=SPEC$(.GET.LINK%,0%,0%,2%) &
PRINT "Talk status is "; &
IF STS% AND .TALK.ON% &
THEN PRINT "On." &
ELSE IF STS% AND .TALK.QUERY% &
THEN PRINT "Query." &
ELSE PRINT "Off." &
! Get the talk status. &
! Print Talk status is... &
! If on, &
! Say "On." &
! If not on, &
! If query, &
! Say "Query." &
! If not query, &
! Say "Off." &

```

```

9001 &
! Dimension Declaration &
!
! 901-929 local dimension declarations &
! 930-949 library dimension declarations &
! 950-979 MAP statements &

```

```

950 MAP (KEYWRD) KEYWRD% &
! Define the keyword. &
! For this to work, the following line must be added &
! into the TKB command file: &
! VSECT=KEYWRD:400:2 &

```

```

9991 &
! Start of Initialization &
!

```

```

3500 ! This is the HELP command. &
! We will print out how to use this program. &
PRINT "Commands are:" + CR + LF + &
HT + "TALK n" + HT + "Talk to keyboard N" + CR + LF + &
HT + "BREAK" + HT + "Break all links" + CR + LF + &

```

```

1000 Onerror Goto 19000 &
! Set standard error trap. &
1010 Print "Tlu " + .Version$ + " Software Techniques" &
+ CR + LF + "Terminal Link Utility" + CR + LF &
Unless E0% &
! Print standard header on 'Run' entry. &

```

```

1030 ! Derive various variables. &
NOT.PRIV%=(KEYWRD% AND 1024%)=0% &
I$=SYS(CHRS(6%)+CHRS(-21%)) &
GOTO 2020 IF E0% &
! Set up the non-priv flag (-1 for non-privileged users.) &
! Get rid of privileges (we shouldn't have them anyway.) &
! Skip the first-time prompt if CCL entry. &

```


RSTS PROFESSIONAL RSTS PROFESSIONAL RSTS PROFESSIONAL RSTS PROFESSIONAL RSTS PROFESSIONAL RSTS PROFESSIONAL RSTS PROFESSIONAL RSTS PROFESSIONAL RSTS PROFESSIONAL RSTS PROFESSIONAL RSTS PROFESSIONAL RSTS PROFESSIONAL RSTS

```

BIT #TT2741,DDFLAG(R1) ;;2741 TYPE TERMINAL? (2-BIT=0 IF SO)
BH1 90$ ;;DO <BS> CORRECTION (C=2=0 => RUBBED 2741)
.ENDC ;KB2741

80$: RETURN ;;ELSE JUST EXIT

90$: JMPX TTCKBS ;;GO AND DO BACKSPACE CORRECTION
100$: JMPX CHOUTE ;;GO AND OUTPUT A CHARACTER

UNORG

.ENDC ;INC$LK
.IF NE INC$CH

.SBTTL HOOK FOR OUR NEW INPUT FUNCTION KEYS

;+
; INHOOK - HOOK FOR THE NEW INPUT FUNCTION KEYS
;
; (SP) HAS THE FUNCTION INDEX
;
; 0 = CONTROL/E
; 2 = CONTROL/F
; 4 = CONTROL/W
; 6 = CONTROL/X
; 10 = CONTROL/Y
;-

INHOOK::MOV (SP),R3 ;;R3 HAS THE FUNCTION CODE (DON'T POP STACK!)
.IF NE EKOCTL

BEQ 10$ ;;FUNCTION 0 WORKS WITH ECHO CONTROL
.ASSUME IN.CE EQ 0
CMP R3,#IN.CW ;;WAS THE KEY A CONTROL/W
BEQ 10$ ;;YES, THAT WORKS WITH ECHO CONTROL ALSO.
CALLX CHKINC ;;SEE IF INPUT FUNCTIONS ARE ALLOWED
BR 20$ ;;AND JOIN UP WITH COMMON CODE
CALLX CHKNEK ;;CHECK TO SEE IF THIS FUNCTION IS ALLOWED

.IFF ;EKOCTL

CALLX CHKINC ;;SEE IF INPUT FUNCTIONS ARE ALLOWED

.ENDC ;EKOCTL

20$: MOV @#KISAR6,(SP) ;;SAVE THE CURRENT PAR6 MAPPING (RE-USE STACK)
MOV @#KISAR5,@#KISAR6 ;;DUPLICATE OUR PAR5 VALUE
ADD #200,@#KISAR6 ;;NOW MAKE IT 4K HIGHER
CALL @TISDSP(R3) ;;OFF TO THE ROUTINE
MOV (SP)+,@#KISAR6 ;;RESTORE THE KISAR6 MAPPING
JMPX TTIN04 ;;AND GO TO THE "NORMAL" EXIT

;+
; INCDMP - DUMP AN INPUT CHARACTER AND FIX PAR6
;
; THE STACK CONTAINS A RETURN TO THE NORMAL COMPLETION
; ROUTINE, AND THE SAVED PAR6 VALUE. THIS ROUTINE WILL
; POP THE STACK, RESTORE PAR6, AND JUMP TO "TTINST", WHICH
; WILL MAKE THE CHARACTER A "NORMAL" CHARACTER.
;-

INCDMP::TST (SP)+ ;;DUMP THE NORMAL RETURN ADDRESS
MOV (SP)+,@#KISAR6 ;;RESTORE PAR6
JMPX TTINST ;;AND PROCESS THE CHARACTER

.ENDC ;INC$CH
.IF NE INC$CE

.SBTTL ^E HANDLER

;+
; TI$CE- ^E HANDLER
;-

TMPORG TTMAPX ;;THIS GOES IN PAR6

TI$CE:: BIT #TTTECO|TTTECS,TTMODE(R1) ;;TECO/TECO SCOPE MODE?
BNE TTNRML ;;MAKE THIS A NORMAL CHARACTER
BIT #TTDDT,DDFLAG(R1) ;;DDT MODE?
BNE TTNRML ;;YES, MAKE THIS NORMAL
MOV TTPUNK(R1),R4 ;;GET THE FUNCTION KEY POINTER
BEQ 20$ ;;NONE, IGNORE THIS CONTROL/E
TSTB TTESCC(R1) ;;ARE WE IN AN ESCAPE SEQUENCE
BNE 20$ ;;YES, ANOTHER REASON TO IGNORE THIS
10$: MOV (R4)+,R2 ;;GET A CHARACTER
BEQ 20$ ;;NO MORE LEFT
MOV R4,-(SP) ;;SAVE BUFFER POINTER
CALLX TTIN09 ;;"FORCE" THE CHARACTER
MOV (SP)+,R4 ;;RESTORE BUFFER POINTER
BIT #<40-1>,R4 ;;ANY MORE TO DO?
BNE 10$ ;;YES, GO AND GET IT
20$: RETURN ;;WE'RE DONE...

GLOBAL <TTFUNK>
.ENDC ;INC$CE

.IF NE INC$CH

TTNRML::JMP INCDMP ;;MAKE THE INPUT CHARACTER NORMAL

UNORG
.ENDC ;INC$CH
.IF NE INC$CY

.SBTTL CONTROL/Y HANDLER (DETACH JOB)

;+
; TI$CY - DETACH JOB
;-

TMPORG TTMAPX

.ENABL LSB

TI$CY:: MOV B DJBNO(R1),R4 ;;GET OWNING JOB # *2
BIC #C'63.*2>,R4 ;;AND ENSURE A VALID JOB # *2
BEQ 20$ ;;NO JOB, NO DETACH
BIT #DDCONS,DCNT(R1) ;;IS THIS THE CONSOLE?
BEQ 20$ ;;NO, DON'T DETACH
MOV JOBTL(R4),R4 ;;GET JOB DATA DETACH POINTER
BEQ 20$ ;;NO JOB, NO DETACH
BIT #JFNOPR,JDPLG(R4) ;;IS THIS LOGGED IN?
BNE 20$ ;;NOPE, DON'T ACT FOOLISH
BIT #JFPRIV,JDPLG(R4) ;;PRIVILEGED USER?
    
```

```

BEQ 20$ ;;NOPE, IGNORE THIS
..PRCY == -.2 ; **PATCH** MAKE NOP TO MAKE CTRL/Y NON-PRIV'D
10$: CALLX ASCOUT,R5,DETMMSG ;;OUTPUT THE MESSAGE
CALLX DETJOB ;;DETACH THE JOB
RETURN ;;AND THAT'S IT

GLOBAL <JOBTL>

.DSABL LSB

UNORG
.ENDC ;INC$CY
.IF NE INC$CX

.SBTTL ^X HANDLER

;+
; TI$CX - CANCEL ALL TYPE-AHEAD
;-

TMPORG TTMAPX

TI$CX:: CALLX ASCOUE,R5,TTICXC ;;OUTPUT ^X<CR><LF>
CALLRX CLRTTI ;;AND CANCEL TYPE AHEAD

UNORG

.ENDC ;INC$CX
.IF NE INC$CY

.SBTTL ^W HANDLER

;+
; TI$CW - ERASE A WORD
;-

TMPORG TTMAPX

TI$CW:: BIT #TAPE,DDFLAG(R1) ;;TAPE MODE?
BNE 10$ ;;YES, IGNORE THIS
TSTB TTESCC(R1) ;;IN AN ESCAPE SEQUENCE?
BNE 10$ ;;YES, IGNORE THIS ^W
BIT #TTDDT,DDFLAG(R1) ;;DDT MODE?
BNE TTNRML ;;YES, MAKE THIS NORMAL.
BITB #TTTECO,TTMODE(R1) ;;TECO MODE?
BNE TTNRML ;;YES, MAKE NORMAL
CALLX TTGET ;;ERASE FIRST CHARACTER
BCS 10$ ;;NOTHING THERE
BIT #NOECHO,DDFLAG(R1) ;;SHOULD WE PRINT ANYTHING?
BNE 20$ ;;NO WAY
MOV #20$,-(SP) ;;PUSH RETURN ADDRESS FOR LINKS

.IF NE INC$LK

CALL EXELNK ;;EXECUTE THE FOLLOWING FOR LINKS

.ENDC ;INC$LK

BIT #TTSCOP,TTCHAR(R1) ;;SCOPE TERMINAL?
BNE 10$ ;;YES, DON'T OUTPUT AN UNDERLINE
CALLX RSXCHR,R5,'_ ;;OUTPUT AN UNDERLINE
RETURN ;;AND LOOP FOR LINKS AND CONTINUE
20$: CMP R2,#11 ;;WAS THE DELETED CHARACTER A TAB?
BEQ 60$ ;;YES, SPECIAL CHECKING NEEDED
CMP R2,#40 ;;WAS THE CHARACTER A SPACE
BEQ 60$ ;;THAT NEEDS SPECIAL CHECKING ALSO
BIS #RUBOUT,DDFLAG(R1) ;;INDICATE THAT WE ARE DELETING CHARACTERS
BIT #NOECHO,DDFLAG(R1) ;;SHOULD WE OUTPUT ANYTHING?
BNE 50$ ;;NO, JUST DELETE MORE
CMP R2,#40 ;;WAS DELETED CHARACTER PRINTABLE?
BHS 40$ ;;YES, ONE DELETE WILL DO
BIT #TTUPAR,TTCHAR(R1) ;;WAS UPARROW SET?
BEQ 50$ ;;NO, NO DELETES NEEDED
CALL 90$ ;;DELETE ONE CHARACTER
CALL 90$ ;;AND ANOTHER
50$: CALLX TTGET ;;GET ANOTHER CHARACTER
BCC 20$ ;;PROCESS IT IF PRESENT
BR 70$ ;;ELSE CLEAN UP SHOP
BIT #RUBOUT,DDFLAG(R1) ;;WERE WE DELETING CHARACTERS?
BEQ 30$ ;;NO, GO BACK FOR MORE
CALLX STOREX ;;REPLACE DELETED CHARACTER
70$: BIC #RUBOUT,DDFLAG(R1) ;;RESET RUBOUT MODE
RETURN ;;AND WE'RE DONE

90$:

.IF EQ EKOCTL

MOV #40,R2 ;;USE <SPACE> FOR PAINT

.IFF

MOV B EKOPNT(R1),R2 ;;GET PAINT CHARACTER
BIC #C'<177>,R2 ;;TRIM PARITY

GLOBAL <EKOPNT>

.ENDC ;EKOCTL

100$:
    
```

the IBM Solution

HASBOX™

SOFTWARE RESULTS CORPORATION

the CDC Solution

UT200BOX™

RSTS·VAX·RSX!

DATA COMMUNICATIONS

- INSTALLATIONS THROUGHOUT NORTH AMERICA AND EUROPE
- FRONT END PROCESSOR-BASED FOR LOW OVERHEAD
- FULL LINE UTILIZATION FOR HI-THRUPUT (UP TO 19.2 KB)
- USER INTERFACE DESIGNED FOR RELIABILITY AND SIMPLICITY

614/421/2094

COLUMBUS, OH 43212 USA

TWX 810 482 1631

CIRCLE 44 ON READER CARD


```

130$: MOV  TLINK(R2),R0 ;;ARE BOTH TERMINALS LINKED?
      BEQ  150$          ;;NO, REAL CRAZINESS DOESN'T HAPPEN
      MOVB (R4),R5      ;;GET USER COUNT FROM OTHER USERS LINK TABLE
      MOV  R4,R2        ;;COPY OTHER USER'S LINK TABLE POINTER
      TST  (R2)+        ;;ADVANCE TO FIRST ENTRY IN TABLE
      ADD  R5,R2        ;;POINT R2 PAST LAST ENTRY IN TABLE
      ADD  (R0),R5      ;;R5 NOW HAS TOTAL NUMBER OF LINKED TERMINALS
      CMBP R5,#26.      ;;MORE THAN 26. TERMINALS?
      BHI  160$          ;;YES, OH LORD!
      MOVB R5,(R4)      ;;UPDATE SIZE OF NEW LINK TABLE
      MOVB (R0),R5      ;;GET OLD LENGTH
      MOV  R0,(SP)      ;;SAVE POINTER
      TST  (R0)+        ;;AND POINT TO START
140$: MOVB (R0),(R2)+   ;;COPY ENTRY
      MOVB (R0)+,R1     ;;AND PICK IT UP
      ASL  R1           ;;MAKING IT KB * 2
      MOV  DEV.KB(R1),R1 ;;R1 -> DDB
      MOV  R4,TLINK(R1) ;;POINT DDB TO TABLE
      SOB  R5,140$      ;;AND LOOP
      MOV  (SP)+,R4     ;;GET POINTER
      CALL UNLINK       ;;UNLINK THIS BUFFER AND RETURN IT (PAR6)
      BR   180$        ;;END OF THIS MADNESS!
150$: CMBP (R4),#26.   ;;ARE WE FULL?
      BNE  170$        ;;NO, LET HIM ENTER THE TABLE
160$: ERROR NOROOM    ;;ERROR - LINK TABLE FULL
170$: MOV  R4,TLINK(R2) ;;PUT POINTER IN DDB
      MOVB (R4),R0     ;;GET SIZE
      INC  (R4)+       ;;POINT TO FIRST AND SET SIZE
      ADD  R0,R4       ;;POINT AFTER LAST
      MOVB DDUNT(R2),(R4) ;;AND STORE THIS
180$: MOV  (SP)+,R1     ;;RESTORE DDB POINTER
      BIT  (#CRL.SP,(R3) ;;DOES HE WANT TO SPY?
      BEQ  190$        ;;NO, DON'T PLAY WITH THE DDB
      BIS  #LS.SPY,TLINKS(R1) ;;THIS TERMINAL IS NOW SPYING
      BR   220$        ;;SKIP ANNOUNCEMENT

GLOBAL <DEV.KB,TLINK,TLINKS>
190$: BIT  #CRL.NW,(R3) ;;SHOULD WE NOTIFY?
      BNE  220$        ;;NOPE
      MOVB DDUNT(R1),R5 ;;GET KEYBOARD NUMBER
      MOV  R5,R0       ;;ALSO INTO R0
      ASL  R0         ;;MAKE IT TIMES 2
      CALLX ASCOUT,R5,ANLNK ;;PRINT SUCCESS ANNOUNCEMENT
      CALL 210$       ;;PRINT IDENTIFIER
      CALLX ASCOUT,R5,CRLF.0 ;;AND DO A <CR/LF>
      CALLRX CHKLN   ;;STARTUP LINE
210$: CALLX ASCOUT,R5,ANLNK1 ;;OUTPUT "KB"
      MOV  R5,R3       ;;GET KEYBOARD NUMBER
      CALL NUMOUT    ;;AND OUTPUT IT
      CALLX ASCOUT,R5,ANLNK2 ;;OUTPUT "*"
      MOV  JOBJD2,R2   ;;GET SECOND JOB DATA BLOCK
      MOV  J2PPN(R2),R2 ;;NOW GET THE PPN
      MOV  R2,R3       ;;AND SAVE IN R3
      SWAB R3         ;;GET THE PROJECT CODE
      BIC  #C<377>,R3 ;;AND TRASH THE PROGRAMMER
      CALL NUMOUT    ;;OUTPUT IT
      CALLX RSXCHR,R5,054 ;;OUTPUT A "*",
      MOV  R2,R3       ;;GET PPN AGAIN
      BIC  #C<377>,R3 ;;THIS TIME GETTING PROGRAMMER
      CALL NUMOUT    ;;OUTPUT IT
      CALLX RSXCHR,R5,['] ;;OUTPUT A "]"
220$: RETURN        ;;NOW WE'RE DONE

GLOBAL <CRLF.0>
UNORG
;+
; BRAS#C - SPECIAL FUNCTION TO BREAK A LINK
;-
BRKSPC::CALL CHKDET     ;;CHECK IF THE USER IS ATTACHED
          SPLC  5        ;;UP TO LEVEL 5
          CALLX MAPPED,R5,BRKLNK ;;BREAK THE LINK
          RETURN        ;;AND OFF

;+
; BRKLNK - BREAK A TERMINAL'S LINKS
; UNLINK - UNLINK A LINK TABLE AND RETURN IT TO THE MONITOR'S POOL
;
; R1 -> DDB
; R4 -> LINK TABLE (FOR UNLINK)
;
; CALL MAPPED,R5,BRKLNK
; -OR-
; CALL MAPPED,R5,UNLINK
;
; RETURN WITH ALL REGISTERS PRESERVED
; -OR-
; R5 = RANDOM (FOR UNLINK)
;-

TMPORG TMAPX

.ENABL LSB

BRKLNK::REGSCR
MOV  TLINK(R1),R4 ;;SAVE ALL REGISTERS
BEQ  60$          ;;PICK UP LINK TABLE POINTER
BIC  #LS.LNK!LS.SPY,TLINKS(R1) ;;NONE, SO DO NOTHING
CLR  TLINK(R1)    ;;CLEAR LINK POINTER
MOVB (R4),R2     ;;GET LINK COUNT
MOV  R4,R3       ;;COPY POINTER
TST  (R3)+       ;;AND POINT TO START
10$: CMBP (R3)+,DDUNT(R1) ;;IS THIS OUR ENTRY?
      BNE  10$     ;;NO, LOOP
      ADD  R4,R2   ;;R2 -> LAST ENTRY - 1
      MOVB 1(R2),-(R3) ;;UPDATE LINK TABLE
      DECB (R4)    ;;AND USER COUNT
      CMBP (R4),#1 ;;ONLY ONE USER LEFT?
      BNE  20$    ;;NO
      MOVB 2(R4),R0 ;;GET UNIT NUMBER
      ASL  R0     ;;MAKE THIS UNIT NUMBER * 2
      MOV  DEV.KB(R0),R1 ;;PICK UP DDB ADDRESS
      BIC  #LS.LNK!LS.SPY,TLINKS(R1) ;;CLEAR FLAGS
      CLR  TLINK(R1) ;;AND REMOVE POINTER
      BR   UNLINK  ;;DUMP BUFFER AND EXIT
20$: MOVB (R4),R2   ;;GET LINK COUNT
      TSTB (R3)+   ;;POINT TO END + 1
30$: MOVB -(R3),R0  ;;GET KEYBOARD NUMBER
      ASL  R0     ;;MAKE IT KEYBOARD * 2
      MOV  DEV.KB(R0),R1 ;;R1 -> DDB
      BIT  #LS.SPY,TLINKS(R1) ;;SPYING TERMINAL?
      BEQ  60$    ;;NO, LINK CAN STAY
      SOB  R2,30$ ;;LOOP FOR MORE
      MOVB (R4),R2 ;;RE-GET LINK COUNT
40$: MOVB (R3)+,R0  ;;GET KEYBOARD NUMBER
      ASL  R0     ;;MAKE IT * 2
      MOV  DEV.KB(R0),R1 ;;PICK UP DDB ADDRESS
      BIC  #LS.SPY!LS.LNK,TLINKS(R1) ;;CLEAR FLAGS
      CLR  TLINK(R1) ;;AND REMOVE POINTER
      SOB  R2,40$  ;;LOOP FOR ALL ENTRIES
    
```

**DEC
RSTS/E USERS**

From one of the pioneers in commercial data processing using RSTS. Off the shelf software ready for immediate delivery. Completely interactive. Extensively documented. Fully supported. Ideal for OEM's, service bureaus or end users. Cost effective solutions including:

- ACCOUNTS PAYABLE
- GENERAL LEDGER
- FINANCIAL REPORTING
- ACCOUNTS RECEIVABLE
- PAYROLL
- FIXED ASSETS

For complete details, contact us at:

PlyCOM services, inc.
P.O. Box 160
Plymouth, IN 46563
(219) 935-5121

CIRCLE 42 ON READER CARD

RSTS RESCUE SQUAD

We salvage all kinds of disasters:

- unreadable disks
- immediate response
- on-site
- custom recovery
- ruined UFDs and MFDs repaired
- telephone DIAL-UP
- software tools
- 90% success to date
- more than 1 GB rescued to data

Brought to you by
On-Track Systems, Inc.
and a well known (and read)
Disk Directory expert.
CALL 24 HOURS
215—542-7133

DEC

WHY PAY MORE???

SAVE WITH DEC COMPATIBLE PRODUCTS

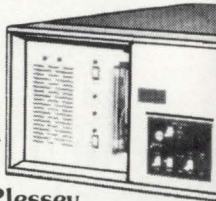
8-64 LINE DH11 ON ONE HEX CARD
1/2 PRICE - TWICE THE PERFORMANCE



EMULEX

LSI-11/23

MSV11, DLV11
28 MB Winch
10 MB Tape



Plessey

DISK SUBSYSTEMS

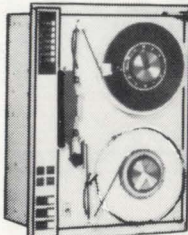
10-675 MB
LSI-11,
PDP-11, VAX



CONTROL DATA

MAG TAPE SUBSYSTEMS

45-125 IPS
LSI-11,
PDP-11, VAX



KENNEDY

Also Available: DL11, DD11, MS11, DZ11, MA20



CALIFORNIA
COMPUTER
GROUP

Toll Free
800-854-7488

In California
714-966-1661

TELEX: 183519 CCG CSMA

3303 Harbor Blvd., #K-11, Costa Mesa, CA 92626

CIRCLE 53 ON READER CARD

TSXplus AN ALTERNATIVE TO RSTS

By Ed Judge, Northampton, MA 01060

I am writing this little piece to acquaint the RSTS user with another timesharing operating system that runs on DEC computers, TSX and TSXplus. Both are from S&H Computer, in Nashville, Tennessee. They are quite popular among smaller (11-03,23,34) system owners.

S&H Computers was founded in 1974 by Harry R. Sanders, who is still president. He was a professor of Electrical Engineering at Vanderbilt University, teaching Digital Circuits and related electronics. Phil Sherrod, one of the principal writers of the system, was helping develop an operating system and later a FORTRAN compiler for a Xerox Sigma 7 computer. They got together and TSX was written.

TSX, the first available system in this line, was available around 1976. It executed with the RT11 monitor as a program that controlled who was in core and how long he stayed there. It used only 64K of memory and swapped workspaces to a file on disk storage. With hard disks, performance with 2 or 3 terminals was acceptable. It cost \$1200 and had a spooler.

When memory above 64K was available, an XM option was made available that allowed the upper memory portion to be used as a swapping area, and when it was filled, it would swap to disc. Later, a device handler (VM.SYS — Virtual Memory) was created that treated the extra memory like a disk (and is still around in public domain). The swap file was placed there, speeding up response time significantly. These were the hot setups among TSX users until TSX-Plus became available in August 1980.

TSX-Plus is like RSTS, and in most instances, runs as fast or faster than RSTS, but costs only \$2000. It maps user jobs into extended memory and executes them there. It can use all 256K of memory available with the 18 bit address capability of the 11-23 (or whatever) in a very efficient manner. A 22 bit system will be available later when the 22 bit QBUS becomes available from DEC. The small overhead it introduces and its simple but efficient time slice algorithm are the secret of its fast response time. It has a protected environment for each user space, spooling, spooling backup, forms support for the spooler, dynamic alteration of constants in the memory swapping algorithm that allows the privileged operator to optimize the system to the current workload, logon and system usage accounting. It has a debugger that can be loaded into the system without the need of linking it to the program in question. Interterminal message facilities allow notes to be passed, printed on the terminal at log-on time. Both of these systems show the single user an RT11 environment having a max of 56KB workspace, with certain enhancements mentioned below. They are a few restrictions, as certain commands from the single-user RT11 environment would be unacceptable in a multiuser system.

Terminals can be locked to certain jobs or devices, so that any nonprivileged users cannot peruse the system. Records can be locked to one job or shared. Many different terminals are supported, and protocols for many are available, which set pagesize, rubout mode, clear screen if appropriate. Xon-Xoff, etc. Multiplexer lines as well as standard serial and parallel lines are supported, along with modems and timeout. More than 15 lines can be connected, the writers claim, although the most I have seen is 6 terminals and 3 printers on an 11-23 with 256KB (with very good response times, I might add). TT buffers, data caches, and the performance monitor are stored in extended memory to preserve useable workspace memory. A user can have a full 56KB of workspace without any system code taking up space. A full-featured BASIC can be loaded and will still leave over 40KB of useable memory. If this space is not needed, the memory limit can be lowered to permit more users in core.

August, 1981

VERSION 2.0 OF ROSS/V

Lexington, MA — Version 2.0 of ROSS/V, a software package which provides a RSTS/E operating system environment on DEC's VAX-11 computers under VMS, is now available for use. Among the many new features of version 2.0 are:

(i) enhancements to ROSS/V's terminal-handling capabilities, including support for RSTS/E's binary input mode, "echo control" mode, multi-terminal I/O, and input and output escape sequence handling.

(ii) support for reading and writing DOS and ANSI file structured magtapes, as well as for specification of tape density and parity.

(iii) support for a number of RSTS/E "bys calls" not previously handled by ROSS/V, including assigning and deassigning of devices, zeroing a magtape, and setting terminal characteristics.

(iv) support for RSTS/E pseudo-keyboards.

ROSS/V, which is written in VAX-11 MACRO, interfaces to programs running under it, in the VAX's PDP-11 compatibility mode, at the RSTS/E monitor call level. Therefore, it is capable of running RSTS/E user programs without regard to the language in which they are written. BASIC-PLUS, BASIC-PLUS-2, DIBOL, FORTRAN IV, and MACRO-11 are among the RSTS/E languages which have been successfully used under ROSS/V.

ROSS/V version 2.0 is priced at \$10,000 for a single-CPU perpetual license, with a one-year warranty including software support and any update releases of ROSS/V during that period. ROSS/V is available from: (Eastern U.S.) Evans Griffiths & Hart, Inc., 55 Waltham St., Lexington, Mass. 02173, (617) 861-0670; (Central U.S.) Interactive Information Systems, Inc., 10 Knollcrest Dr., Cincinnati, Ohio 45237, (513) 761-0132; (Western U.S.) OnLine Data Processing, Inc., N. 637 Hamilton, Spokane, Wash. 99202, (509) 484-3400.

May, 1981

SSI INTRODUCES SYNCH/ASYNCH SERIAL COMMUNICATIONS INTERFACE

Fort Lauderdale, Fla. — Southern Systems, Inc. (SSI), FL-based computer printer firm, has announced development of a completely programmable Serial Communications Interface capable of either synchronous or asynchronous printer-computer operations.

The microprocessor-controlled interface was developed by engineers in the company's Clearwater manufacturing plant.

Southern Systems' new interface, termed the SI-9076, functions with the company's 200 lpm matrix printer, its B-Series printers ranging from 300 to 900 lpm and with the new line of QT band printers that the company is presently introducing.

Programmed for synchronous operation, the SI-9076 enables Southern Systems printers to function with IBM 2780/3780, Burroughs Poll and Select, Univac 1004, Honeywell VIP 7700 and other industry standard protocols.

"Since our newest interface is completely programmable, it can handle many protocols, as well as special purpose tasks," said James W. Rule, vice president/marketing. "In general, the serial communications interface allows one to implement a 'smart' printer on a serial communications network. In some cases the interface can replace communications processors."

The SI-9076 operates full or half duplex or as a null modem hookup. The synchronous serial data rate may be from DC to 64K baud. The asynchronous rate is from 50 baud to 19.2K baud. Modem control signals are provided. Provisions also are made allowing these signals to be "forced true" with switches, said Rule.

SSI's serial communications interface can perform standard or special code conversions (ASCII, EBCDIC, BAUDOT, IBM SELECTRIC and others). Sufficient programming area is provided so that a complete "Lookup Table"

can be implemented and any character converted to any other character.

The interface is useful in systems that require data verification before printing. Special block checks, cyclical redundancy characters (CRC), or error-correcting techniques can be programmed. Parity checking is standard, but may be disabled.

The SI-9076 allows one to implement additional features in the printer that are not normally present. Features incorporated as standards within the 9076 include foldover, space compression, vertical TAB, Serial DAVFU and Wraparound.

In addition, the SI-9076 microprocessor can read the standard set up switches (USART Setup and Baud Rate). In the test mode, a printout of the board set-up is performed and other readable switches are provided for address, terminal number, etc. This allows any user to verify the exact configuration of his communications printer.

SI-9076 contains 5K bytes of RAM for data buffering, although all 5K does not have to be installed. This memory may be partitioned as a single buffer or multiple buffers. It is normally programmable with dual buffers so that a record is being printed while a record is received. By using Double Buffers, maximum printer speed is attained at all times.

Up to 2K bytes of RAM are supported for programming the interface board. This is generally more than required to handle specific tasks and also provides code conversion area, "canned messages," etc.

A summary of features provided by the SI-9076: Z80A Microprocessor, 5K Bytes of RAM, 2K Bytes of EPROM, Double Buffering, Crystal Controlled Baud Rate Generator, Synch/Asynch USART, Concatenated Rs232 Input/Output for line sharing, Current Loop Input, Modem Control Signals, 32 Readable Switches, Printer Output Buffer.

Specifications summary: Z80A Microprocessor System; 2K Bytes of program area; 5K Bytes of buffer storage; Synch/Asynch USART; Synchronous baud rate - DC to 64K baud, the external clock must be supplied by the modem (or comm. system); Asynchronous baud rate - switch selectable from 50 baud to 19.2K baud; Parity - switch selectable for even, odd, or no parity; stop bits - switch selectable for 1, 1½, or 2 stop bits; Self test/Reset switch - a SPDT momentary contact switch is provided to reset all electronics (A self test feature is incorporated within the 9076 board); Concatenated RS232 Input/Output - separate RS232 input and output connectors are provided (multiple terminals can share a common communications line); Current Loop Interface - a 20MA current loop interface is provided (60MA with resistor changes). Either the XMITER or receiver can be made active or passive (active provides a current source).

September, 1981

SATURN SYSTEMS RELEASES VERSION 4.0 OF WP SATURN

Minneapolis, MN — Saturn Systems, Inc. has just released version 4.0 of WP Saturn, a Word and List Processing package that runs on Digital Equipment Corporations PDP-11 and VAX series of computers.

The enhancements include:

1. Formatting in the Editor to allow viewing of centering, tabs and the margins without leaving the editor.

2. A Master Menu to simplify document selection and editing.

3. Bold Face printing.

4. Footers.

5. In addition to automatic decimal numbering of paragraphs (i.e., 3.7, 3.7.1, ...), the system now has standard outline format as an option (i.e., Roman Numerals, Capital letters, Numbers, ...).

For more information contact Saturn Systems, Inc., 6875 Washington Ave. So., Suite

218, Minneapolis, MN 55435, (800) 328-6145 or (612) 944-2452.

December, 1981

PRINTER SHARING DEVICE

Palo Alto, CA — The Model 9 printer sharing device from Ross Systems, Inc., is a free standing unit which allows one serial printer to be shared by up to nine computers, either locally or at a remote site. It polls each computer using X-on and X-off. It will work with any printer which sends X-off when its buffer is full and X-on when its buffer is empty. Additionally, it will detect printer offline through the data terminal ready signal.

The Model 9 carries a 180 day warranty and is available within 30 days after receipt of order.

Ross Systems, Inc., 1900 Embarcadero Road, Suite #208, Palo Alto, CA 94303, (415) 856-1100.

**ASYNCHRONOUS PRINTER SELECTOR**

- Efficiently connects multiple mini computers to a single line printer either locally or at a remote site.
- Polls up to 9 computers.*
- Communicates with computers using X-on and X-off.
- Works with any printer which sends X-off when its buffer is full and X-on when its buffer is ready to accept data.
- Detects printer offline through pin 20 (dtr).
- 180 day parts and labor warranty.
- Supports speeds from 300 to 9600 bits per second.
- Delivery 30 days after receipt of order.

Call or Write: John Benedict, Ross Systems, Inc., 1900 Embarcadero Rd., Palo Alto, CA 94303, (415) 856-1100

*More ports available by individual quote

December, 1981

COMPUTER MANAGED INSTRUCTION, a Tool for the Modern Educator

Medford, NJ — CMI, a powerful tool for the modern educator is now a reality. CMI is a computer program designed to provide computerized instruction, tests and remedial work to students. It adapts to each student's ability allowing them to work at their own pace. Students log on and access the information using the EXPLORE function, an easy to use, interactive program. The educator can design multiple learning modules using the flexible CREATE function and monitor each student's performance via the REPORT function. CMI was designed by university faculty and is currently being used in secondary school, vocational school and universities.

To find out how CMI can benefit your institution contact: ELEX (The Effective Learning Exchange), P.O. Box 14, Medford, New Jersey 08055, (609) 654-1100.

October 1981

SOFTWARE PACKAGE TRANSFERS FILES BETWEEN TWO DEC MINICOMPUTERS Without Using Communications Interfaces
London, England — Suitable for LSI-11's and VAX-11's as well as PDP-11's, including processors with different operating systems and different storage media.

A low-cost and easy-to-implement means of transferring files (data and/or programs) between two Digital Equipment Corp. PDP-11, LSI-11 or VAX-11 computers, is making its debut in the U.S. Called XOREN IPL-11, it is a

software package which enables file-transfers to be carried out asynchronously between any two machines from the three families, either on the same site or in different geographical locations.

Developed by Koren Computing Ltd, of London, England, IPL-11 enables a two-way communications system to be set up for less than \$3,000 (excluding line costs) and put into operation as soon as the program has been installed on each machine.

Transfers can take place in both directions simultaneously at rates up to 9600 bits/second, over a direct line or telephone line; and can be initiated manually or, in certain circumstances, under the control of a user's applications program.

The package carries out and monitors the entire transfer process. It performs CRC error-correction to CCITT Recommendation V41 and, when it detects errors, re-transmits the block or blocks in which the errors were found.

A key feature of the package is that it eliminates the need to buy dedicated communications interfaces. Transfers are via a terminal I/O port on each computer. Thus, for inter-processor communications between two computers on the same site, each machine requires only a standard DEC DL-11 or equivalent single-channel terminal interface card, or a single port on a multi-channel interface card such as a DZ-11 or DH-11.

For communications between computers on different sites, each processor's interface card requires only the addition of the necessary modem.

IPL-11 provides a number of operational advantages. It is media-independent, i.e., transfers data irrespective of the data storage devices in use on each system. It eliminates the need to transport disks and other media when transferring data on or off site, and it allows data to be quickly stored at a remote site for security.

The package also allows multi-system access to expensive peripherals such as large disk systems; rapid updating of data and/or programs on computers at remote sites; communications between word-processing and information-processing systems, and instant distribution of information in electronic mail applications.

OPERATING SYSTEM INDEPENDENT

Another attractive feature of IPL-11 is that it is largely operating-system independent. Versions are available for most major DEC operating systems, including RSX-11M, RT-11, VAX-VMS (RSX-11 compatibility mode), IAS and, most recently, RSTS/E. Each version can communicate with any other.

The version for RSX-11M supports indirect command files and has a Remote Activation facility which enables a user at one computer to transfer files to or from another without an operator being present at the other machine. An optional file-conversion utility, FLC-11, provides a one-pass conversion for applications where the two operating systems use different file formats.

A further option, IPLLIB, enables transfers under the RSX-11M version, to be controlled by users applications programs. The package is supplied under a 5-year licence. A separate licence is required for each combination of cpu and operating system under which IPL-11 is to run. The two licences required to link two operating systems cost \$1350 each whichever pair of operating systems is specified. For larger orders a system of discounts is applied.

Koren is currently setting up a network of distributors in the US and Canada to market the product to OEM's and end-users.

Koren Computing Ltd is an independent systems/software company formed in 1974. It has developed real-time computer systems and software for several large organisations in the UK, Europe and North America, including the British Post Office, I.T.T. and the British Columbia Telephone Company and has developed a number of communications-oriented software

products.

For more information contact: Mr. John Jarvis, Koren Computing Limited, 28 Maddox Street, London W1R 9PF, England. Telephone: LONDON (01) 629 5932.

September, 1981

WHY SYSTEMS INTRODUCES DIGICALC
Redmond, Washington — Our development staff, directed by Wayne Yarnall, has designed an electronic spreadsheet program to run on Digital Equipment Corporation computer systems.

DIGICALC is perfect for doing budgets, financial modeling, pro forma financial statements, investment and resource management, and job costing. The strong success of Personal Software's VisiCalc shows a clear need for this type of program in commercial environments. DIGICALC meets this need and contains the enhancements that you would expect from an advanced program running on a more powerful computer.

DIGICALC displays a tabular worksheet on the screen of a VT-100 terminal. The user writes equations, values, and textual data onto the worksheet by way of the keyboard and auxiliary keypad. The program calculates the equations entered and recalculates each time related data is added or changed. Worksheets can be saved, recalled, and consolidated, and reports can be printed as needed.

DIGICALC is designed to be very user-friendly with extensive HELP available on the terminal. An inexperienced operator can sit down at the terminal and learn to use DIGICALC in less than an hour.

As a DEC system user, you will find DIGICALC useful at every executive work station in your organization. Give us a call for an on-site demonstration to evaluate DIGICALC for your business needs. WHY Systems Inc., 20915 NE 77th St., Redmond, Wash. 98052, phone: (206) 881-2331.

November, 1981

NEW RELEASE 3.0 "RABBIT-3", JOB ACCOUNTING AND PERFORMANCE MONITOR FOR DEC RSTS/E Users

West Palm Beach, FL — RAXCO Inc. announces the immediate availability of release 3.0 of RABBIT-3, Job Accounting and Performance Monitoring Software System for Digital Equipment Corporation RSTS/E operating system users.

The purpose of RABBIT-3 is to monitor system activities and create data that can be utilized by performance analysis (RABBIT-2) and/or billing (RABBIT-1) programs. RABBIT-3 output may also be utilized by Datatrieve, as well as Fortran, Cobol and Basic users.

RABBIT-3 is a stand-alone system written in PDP macro that runs in 6-7 K memory system degradation is minimal.

RABBIT-3 requires no sysgen and contains an auto-load parameter.

RABBIT-3 creates records for the following resource types:

- Job Records . . . contain systems resources utilized by job.
- CPU Statistics Record . . . contain periodic statistics in percentages of CPU utilization.
- Disk Space Records . . . periodic records containing the disk space available.
- Disk Catalog Record . . . all information by filename, for all files on system including public and private disks.
- Disk Statistics Records . . . contain disk utilization information for each disk.

RABBIT-3 basic price is \$2495 plus options. Rental is \$99 per month plus options.

RABBIT Systems are in world-wide use on VAX-VMS and PDP-11 RSTS computers. For more information contact: RAXCO, Inc., 3336 N. Flagler Dr., West Palm Beach, FL 33407. Phone: (305) 842-2115.

LIST OF ADVERTISERS

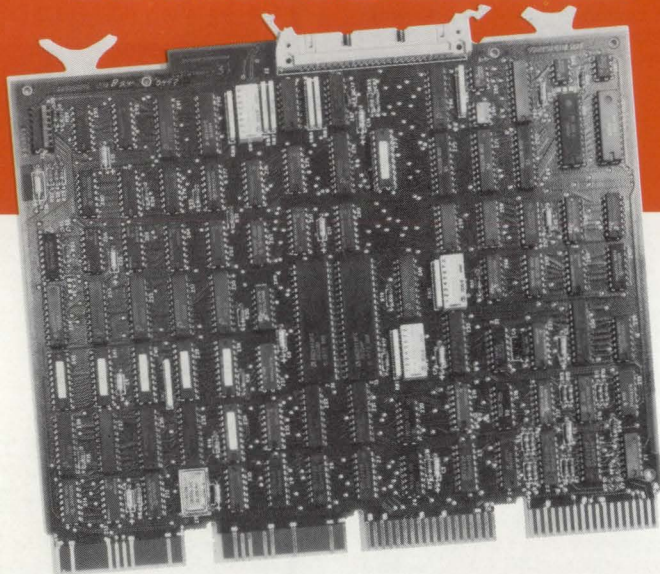
Aardvark Software Inc.	p. 23
Able Computer	I.B. Cover
Ancor	pp. 2, 107
American Used Computer	p. 87
C.D. Smith & Associates, Inc.	p. 111
California Computer Group	p. 106
Casher Associates Inc.	p. 89
Clyde Digital Systems	p. 11
ComDesign, Inc.	p. 25
Computer & Terminal Exchange	pp. 30-31
Data Processign Design, Inc.	pp. 68, 82, B. Cover
Datamedia, Corp	p. 5
Dataram Corp.	p. 112
Datasouth Computer Corp.	p. 79
Dataware, Inc.	p. 19
Digital Communications Assoc., Inc.	p. 33
Direct Inc.	p. 69
Distributed Logic Corp	p. 41
EEC Systems	p. 58
Effective Learning Exchange	p. 68
Emulex Corp	pp. 20-21
Enterprise Technology Corp.	pp. 7, 19
Evans Griffiths & Hart, Inc.	pp. 51, 63
Finar	p. 94
Hamilton Rentals	p. 13
Hardcopy	p. 55
Infinity Software Corp	p. 9
Instor Corporation	p. 105
Macro-Man	p. 86
Manus Services Corp.	p. 77
McHugh, Freeman & Associates, Inc.	p. 92
Nassau Systems	p. 83
National Peripherals, Inc.	p. 35
National Public Radio	p. 104
Nationwide Data Dialog	pp. 49, 90, 91
NCA Corp.	p. 92
North County Computer Services, Inc.	pp. 47, 56-57, 85
On-Track Systems, Inc.	pp. 76, 80
Oregon Software	pp. 26-27
Plycom Services, Inc.	p. 103
Radgo Sales Company	p. 83
Raxco Inc.	p. 15
Real Time Products	p. 17
Resource-11	p. 86
Ross Systems, Inc.	I.F. Cover
Software Results, Inc.	p. 101
Software Techniques, Inc.	p. 53
Southern Systems, Inc.	p. 1
Spartin Systems	p. 90
System Industries	p. 37
T.F. Hudgins & Associates, Inc.	p. 36
Telecom Computer Systems, Inc.	p. 39
Theta Business Systems	p. 43
Timeplex, Inc.	p. 71
Tridacor Systems, Inc.	p. 97
Unitronix Corp	p. 82
WHY Systems, Inc.	p. 73
Xoren Computing, Ltd.	p. 45

**DEC
SYSTEMS
&
COMPONENTS**

**C.D. SMITH &
ASSOCIATES, INC.**
12605 E. Freeway,
Suite 318
Houston, TX 77015
(713) 468-2384

CIRCLE 54 ON READER CARD

DEC[®]-COMPATIBLE PERIPHERAL CONTROLLERS



Dataram Corporation offers the industry's widest range of DEC-compatible peripheral controllers — from comparatively simple NRZI tape controllers to complex 300 MB storage module drive (SMD) controllers.

An impressive array of state-of-the-art controllers, all built around high-speed bipolar microprocessors. All software compatible with the host LSI-11[®], PDP[®]-11, or VAX[®] minicomputer...and all available now.

And Dataram's controllers are designed to save you money, and, more importantly, space — our controllers typically occupy half the space required for the comparable controller from DEC. Doing it with a level of performance that makes any member of this family worth looking at.

The chart shows our current family of peripheral controllers, growing every day. If you don't see the controller you need, we're probably working on it right now. Call us and discuss your requirements.

**DATARAM
CORPORATION**

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

Canada: Ahearn & Soper Ltd., 416-245-4848 • Denmark: Technitron ApS, 02 96 98 22 • Finland: Systek OY, (80) 73 72 33 • France: YREL, (03) 956 81 42 • Hungary/Poland/Romania: Unitronex Corporation, WARSAW 39 6218 • Italy: ESE s.r.l., 02/607 3626 • Netherlands: Technitron b.v., (020) 45 87 55 • Sweden: M. Stenhardt AB, (08) 739 00 50 • Switzerland: ADCOMP AG, 01/730 48 48 • United Kingdom: Sintrom Ellinor Ltd., (0734) 85464 • West Germany: O.E.M.-Elektronik GmbH, 07 11-79 80 47 • Yugoslavia: Institut "Jozef Stefan", 263-261 • Australia/New Zealand: Anderson Digital Equipment, (03) 544-3444 • India: Infosystems Private Limited, 79281 • Israel: Minix Computers & Systems Ltd., 03-298783 • Japan: Matsushita Electric Trading Co., Ltd., 06 (282) 5111 • Taiwan: Rabbit Associates, Ltd., 7219573-5 • Hong Kong: Automated Systems (HK) Ltd., 5-630256-9 • Malaysia: Automated Systems (M) Sdn Bhd., 773777 • Indonesia: P. T. Daya ASL, 584306 • Singapore: Automated Systems (PTE) Ltd., 2354133

CONTROLLER	DESCRIPTION	COMPATIBILITY
C03	Cartridge disk controller	RK05
C33	Cartridge disk controller	RK05
T03	NRZI mag tape controller	TM11/TU10
T04/N	NRZI mag tape controller	TM11/TU10
T04/D	Dual density mag tape controller	TM11/TU10
T34/N	NRZI mag tape controller	TM11/TU10
T34/D	Dual density mag tape controller	TM11/TU10
T36	Dual density mag tape controller	TM11/TU10
S03/A	80MB/300MB SMD controller	RM02/RM05
S03/A1	160MB SMD controller	RM02
S03/B	80MB/300MB SMD controller	RK07
S03/C	200MB/300MB SMD controller	RP06
S03/D	96MB CMD controller	RK06
S33/A	80 MB/300 MB SMD controller	RM02/RM05
S33/A1	80 MB/160 MB SMD controller	RM02
S33/B	80 MB/300 MB SMD controller	RK07
S33/C	200 MB/300 MB SMD controller	RP06
S33/D	96 MB CMD controller	RK06

Products printed in red are LSI-11 Bus compatible.
Products printed in black are UNIBUS[®] compatible for PDP-11 and/or VAX minicomputers.

DEC, LSI-11, PDP, UNIBUS and VAX are registered trademarks of Digital Equipment Corporation.

ABLE VaxDZ clears up your data traffic jams.

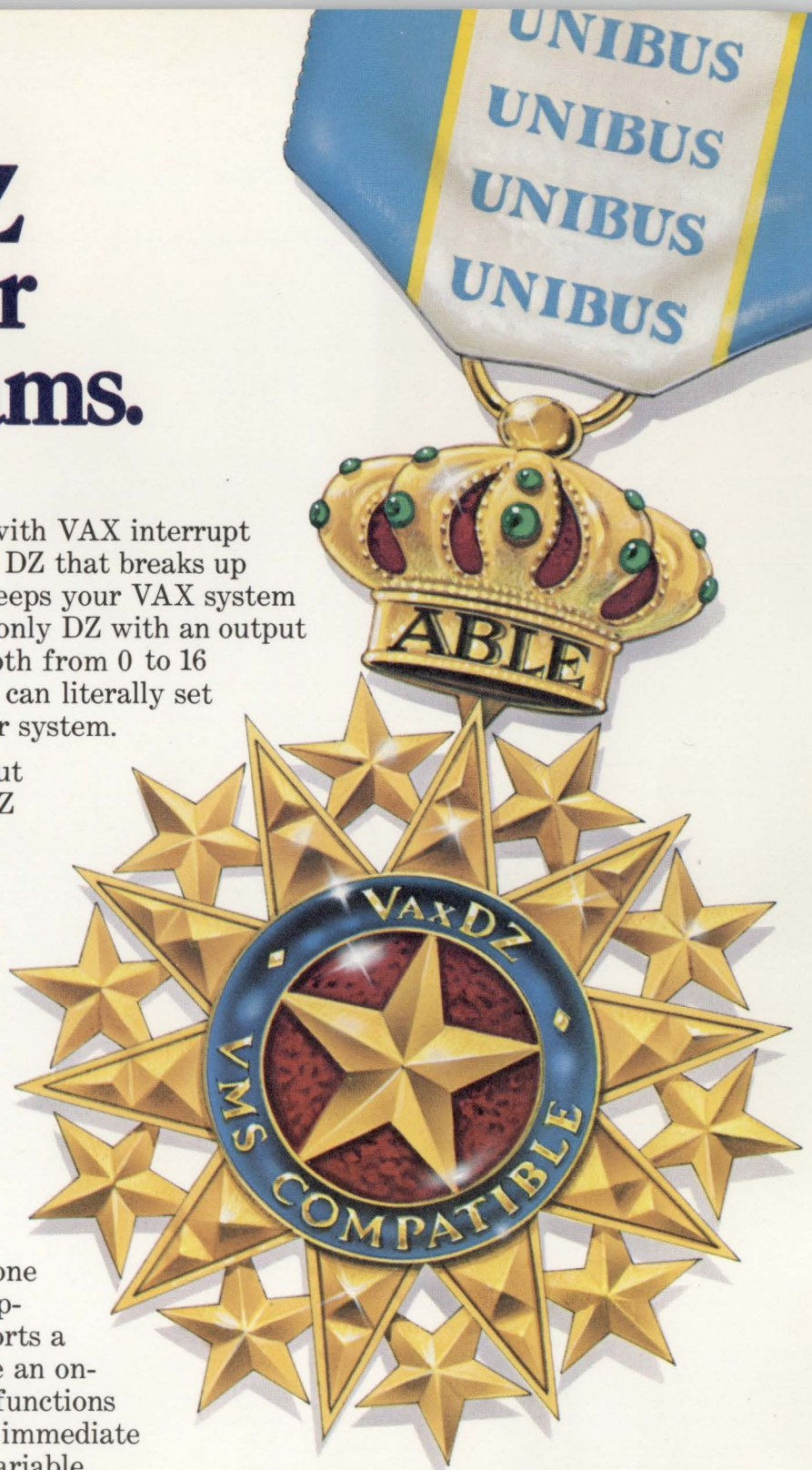
You've been hurting over the problem with VAX interrupt capacity for some time. Now we have a DZ that breaks up the terminal handling bottleneck and keeps your VAX system in the fast lane. It's ABLE VaxDZ, the only DZ with an output buffer which lets you select any silo depth from 0 to 16 characters. With this novel feature, you can literally set the optimum performance level for your system.

That alone should clear up the traffic, but there's more! We've given ABLE VaxDZ an intelligent input silo two times as big as the standard DZ buffer. Both big and smart means doubling the VAX input data-handling capacity in some systems or providing dramatic improvement in every system all the way up to the maximum line configuration. We've even included a "data throttle" which allows any external device to control the clear-to-send (CTS) line and optimize its own data rate.

VaxDZ puts sixteen lines with modem control on a single hex-width board at one unit load and includes a panel which supports EIA only (an optional panel supports a mix of EIA/CL). Other features include an on-board LED display for pinpointing malfunctions automatically, an on-board self-test for immediate verification of system integrity and a variable PROM set for proprietary OEM applications.

Now, here is the best part. ABLE VaxDZ will match or beat DH performance in VAX systems without the addition of foreign software.

You don't have to be a hero to deserve an ABLE VaxDZ medal. Just be smart enough to use our new multiplexer. Write for details. We'll include information on the ABLE line of UNIBUS-compatible products, as well as the MAGNUM™ Series of computer systems.



ABLE

the computer experts

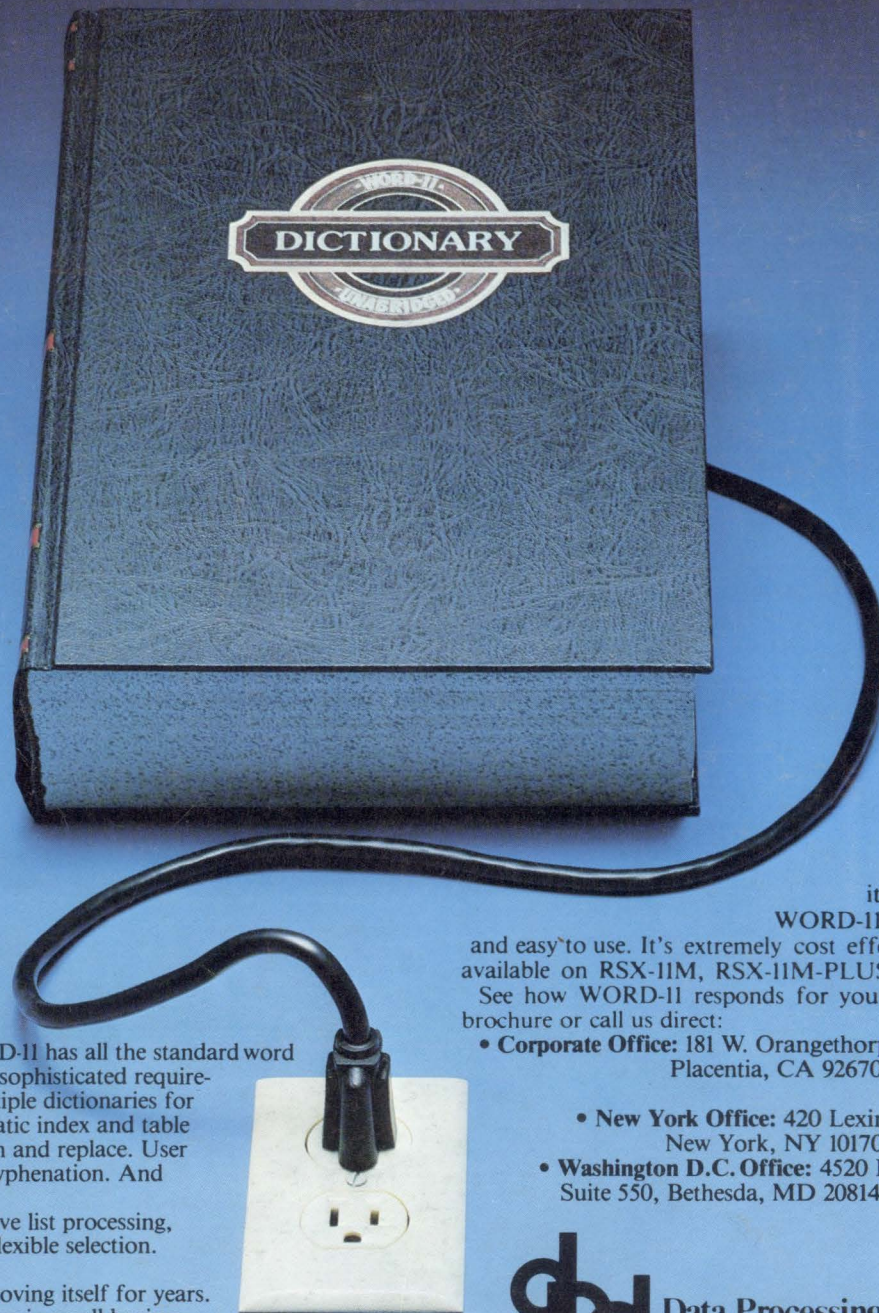
ABLE COMPUTER, 1732 Reynolds Avenue,
Irvine, California 92714. (714) 979-7030.
TWX 910-595-1729 ACT IRIN.

ABLE COMPUTER-UK, 74/76 Northbrook Street,
Newbury, Berkshire, England RG13 1AE.
(0635) 32125. TELEX 848507 HJULPHG.

ABLE COMPUTER-GERMANY, Forsthausstrasse 1,
8013 Haar (Near Munich), West Germany.
089/463080, 463089.

VAX and UNIBUS are trademarks of Digital Equipment Corporation.

Responsive Word Processing. Take Our Word For It.



WORD-11.

WORD-11 is a complete word processing system. It's responsive. It's powerful. And it's sharable on up to fifty terminals while running concurrently with data processing.

WORD-11 is talented, too. Designed to work on Digital's family of mini-computers, WORD-11 has all the standard word processing functions. For more sophisticated requirements, WORD-11 provides multiple dictionaries for spelling error detection. Automatic index and table of contents creation. Text search and replace. User defined keys. User-controlled hyphenation. And automatic footnoting.

Included with comprehensive list processing, WORD-11 offers fast sorting. Flexible selection. And extensive math functions.

And WORD-11 has been proving itself for years. You'll find successful installations in small businesses, Fortune 500 companies, in universities and in banks—wherever Digital computers are in place.

Yet despite its sophistication, WORD-11 is easy to learn and easy to use. It's extremely cost effective. And it's available on RSX-11M, RSX-11M-PLUS, and RSTS/E. See how WORD-11 responds for you. Write for our brochure or call us direct:

- **Corporate Office:** 181 W. Orangethorpe Ave., Suite F, Placentia, CA 92670, (714) 993-4160, Telex 182-278.
- **New York Office:** 420 Lexington, Suite 647, New York, NY 10170, (212) 687-0104.
- **Washington D.C. Office:** 4520 East-West Hwy., Suite 550, Bethesda, MD 20814, (301) 657-4098.

dpd Data Processing Design, Inc.
AUTHORIZED **digital** COMPUTER DISTRIBUTOR

Overseas Distributors:

Management Information Services PTY. LTD.
Melbourne, Australia

Jenson, LTD.
Bristol, England

Network Computer Services PTY. LTD.
Sydney, Australia

Systeme, LTD.
Leeds, England

On-Line Computing PTY. LTD.
Subiaco, W. Australia