INSIDE:

- Logging Into An Account Without LOGIN
- DECWORD/DP
- IMPRS
- EDT Hints & Kinks
- How To Use Build
- Enable Compatibility with NON-DEC Peripherals
- More Notes on Literals and Strings in Basic-Plus 2
- The VAX-SCENE
  Learning VAX MACRO For Fun and Profit
  Replacing RSTS SYS Calls With VAX/FMS System Services
- The Basics of Networking and Digital Communication for the System Manager
- The Ultimate Push/Pull Macros
- Polysoft Application Language
- EXTRACT
- Tips & Techniques
- The RSTS/E Environment
- RPTMAN — Report Manager
- More . . .
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."
Adding a printer?  
Add-on the printer company instead.  
Southern Systems.

You need maximum performance and minimum downtime from your printer. So, don’t add-on just a machine. Add-on the unmatched printer expertise of Southern Systems, recognized by users as the leader in printer systems, from technological innovation through installation to the long-term service you need.

In more than 4,000 installations, Southern Systems has proven it’s the best source for printer systems from 200 to 2,000 lpm. Printer systems that are guaranteed compatible with your computer. Printers that are serviced by SSI specialists nationwide. Printers that work for you at less cost than any other printer source can offer.

Add-on a Southern Systems printer system and you add-on Southern Systems. Just ask our customers.

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

Tell me about your printers from:  
☐ 200–300 lpm  
☐ 600–900 lpm  
☐ 1000 plus lpm  

My computer system is______________  
Name______________________________  
Title______________________________  
Company___________________________  
Address____________________________  
City________________State________Zip________  
Telephone__________________________  

CIRCLE 2 ON READER CARD
**SOFTWARE**

**Link Up Whenever You Like.** When you run the program CONTRL at your terminal you may elect to capture and link up with any keyboard on the system. CONTRL performs the link requested immediately. There is no interruption to the user’s task. It makes no difference what the user may be doing. The user may be in mid-keystroke or logged off the system. The user's keyboard may even be turned off.

**Link Is Invisible to the User.** The linking process is invisible to the user. Except on a heavily loaded system the user will not notice so much as a hesitation from one keystroke to the next when the link up takes place. In fact, an inspection of job status will appear normal to the user.

**Do Remote User Training.** When a new procedure or application is put onto your system, CONTRL may be used to do remote training. The user logs onto the system and then calls you by telephone. You run CONTRL at your terminal. While speaking to the user you link up with the user’s keyboard. Now you walk the user through the new procedures while you watch at your screen. Each user keystroke together with the system's responses is presented to your terminal.

**Interact for Remote User Support.** With CONTRL you may interact with the user. Anything you do at your keyboard after linking with the user is as though you did it at the user’s keyboard. When a user calls you with a question or concern about his job you may link up and give assistance directly from your keyboard.

**Provide Remote Demonstrations.** If you need to demonstrate an application to a remote group, CONTRL will solve the problem.

**CONTRL**

<table>
<thead>
<tr>
<th>LINK YOUR TERMINAL TO ANY KEYBOARD ON THE SYSTEM FOR:</th>
</tr>
</thead>
<tbody>
<tr>
<td>USER TRAINING AND SUPPORT</td>
</tr>
<tr>
<td>DYNAMIC SECURITY</td>
</tr>
<tr>
<td>REMOTE DEMONSTRATIONS</td>
</tr>
</tbody>
</table>

Don't pay the travel costs to get your team together with their team to see some programs run. Consider what many are now doing with CONTRL. The application review team gets together at their own site. They gather around a terminal that is logged into your system. Then they call you on the telephone. Most often they will use a speaker phone. As you exercise the application at your terminal they see everything at their remote screen. If you wish, they may be instructed to interact with the application themselves. This serves to convey the dynamic nature of your demonstration, while involving your listeners.

**Inspect User Activity.** CONTRL allows you to inspect a user’s activity on the system. It is often necessary for management to observe training effectiveness among their clerical personnel. With CONTRL a clerk’s grasp of an application can be observed unintrusively.

**Do Dynamic System Security.** The inappropriate, unwise, and covert use of your system can be monitored. Experience with CONTRL in this area indicates that knowledge of its existence on the system and its potential for invisible use on selected keyboards is an effective threat to covert users.

For inappropriate and unwise use of the system, CONTRL gives management a means for taking specific corrective action.

**Keep a Log File of the Activity.** This is well worth noting. A complete log file of the user activity is kept by CONTRL. Every keystroke entered at either your keyboard or the user’s keyboard goes to the log file together with every response from the system. The session in its entirety is captured. The keystrokes are underlined to distinguish the user from the system when the log file is played back.

**Release Link Whenever You Like.** The link can be released immediately and at any time. Releasing CONTRL has no effect whatever on the user’s job. The user may be in mid-keystroke or logged off. The user’s terminal may even be turned off when the keyboard is released.

**Some of the CONTRL Options.** You may get a log file or elect to turn it off. You may disable the user’s keyboard or prevent output from the user’s job from going back to the user’s screen. CONTRL gets its name from being in control of the linked keyboard. Everything that moves between the user’s keyboard and the system goes through CONTRL.

**How to Get More Information.** Call Janet at (617) 275-6642, or write: Clyde Digital Systems, Inc., P.O. Box 348, Bedford, MA 01730.
Contents

LOGGING INTO AN ACCOUNT WITHOUT LOGIN ........................................ 8
Patrick Holmay and Robert Schilloeller
Want to move quickly and easily from one account to another?
How about a program you can chain to. Here it is.

DECWORD/DP or How I Tried to Get a Free VT100 Advanced Video Option ... 11
Joe Doyle
Free Advanced Video? Not this time.

IMPRS, A Productivity Relational Data Base Language .......................... 12
Jacob F. Ruf
More programmer productivity software, to help us do our jobs.

EDT HINTS & KINKS ........................................................................ 24
David Spencer
The users and David will make EDT great.

HOW TO USE BUILD ...................................................................... 32
Richard W. Hill
A manual that was missing from the documentation.

ENABLE COMPATIBILITY WITH NON-DEC PERIPHERALS .................. 46
Ken Fleming
Frustrations lead to great satisfactions. Hard work and perseverance pay off.

MORE NOTES ON LITERALS AND STRINGS IN BASIC-PLUS 2 .............. 47
Brad Smith
Knowing about what your compiler does will help your programs.

THE VAX-SCENE
Learning VAX MACRO for Fun & Profit ........................................... 52
Bob "MACRO MAN" Meyer
Is "Macro Man" moving to VAX? Or is there truth to the rumor that VAX Macro
will soon be available on RSTS
Replacing RSTS SYS Calls With VAX/VMS Systems Services ............ 54
Bob Stanely
More "How To" Handle RSTS -> VAX conversions.

THE BASICS OF NETWORKING AND DIGITAL COMMUNICATION FOR
THE SYSTEM MANAGER .................................................................. 58
Michael H. Koplitz
A much misunderstood topic is clearly explained. Mike has a real talent for teaching.

THE ULTIMATE PUSH/PULL MACROS ........................................... 62
Ruby Bazelmans
A veteran author augments your Macro programming.

POLYSOFT APPLICATION LANGUAGE ............................................ 66
Stanley Zuk
How you can program without a program.

TIPS AND TECHNIQUES .................................................................. 72
Steven Edwards
How to make your BP2 program into an RTS!

THE RSTS/E ENVIRONMENT ......................................................... 74
Michael H. Koplitz
RSTS internals by our east coast internist.

EXTRACT.BAS ............................................................................. 85
Stephen Munyan
A selective APPEND utility.

RPGED.TEC ..................................................................................... 87
Austin Kinsella
A TECO program for RPG users.

RPTMAN — REPORT MANAGER ..................................................... 90
Jonathan M. Prigot
Organize and print files with this utility.

Coming . . .

- RSTS Security
- A DEC of Cards
- Word Processing
- Another Dungeon Map (in living color)
- Disc Inversion Map
- TYPE.RTS
- VT100 Printer Port
- More Networking
- More VAX Macro
- More RSTS News
- More . . .

From the Editors ................................................................. 4
Letters to the RSTS Pro ......................................................... 6
Dear RSTS Man ......................................................................... 36
Tips and Techniques ............................................................. 72
Classifieds ............................................................................... 93
News Releases .......................................................................... 94
List of Advertisers ................................................................... 94

The RSTS Professional Magazine, April 1, 1982,
Vol. 4, No. 2. Published bi-monthly. Single copy price $10.00, $35.00 per year.
Send all correspondence and change of address to: Box 361, Ft. Washington, Pa 19034-0361. Copyright © 1982 by M Systems, Inc. No part of this publication
may be reproduced in any form without written permission from the publisher.
From the editors...  

**IS IT REALLY FAIR?**

Carl Marbach

The late Chief Justice Earl Warren of the United States Supreme court would sometimes stop an eloquent argument from an attorney at the bar with a wave of his hand, "Yes, yes, yes", he would say, "but is it really fair?" Sometimes all the rational arguments in the world don't suffice when people are at stake. My question to DEC is, "why are you leaving your loyal PDP-11 people alone at the top? What do I do now that the 11/70 is going to be gone?" And they reply, "Go VAX young man".

I attended a meeting recently with a major OEM and DEC people discussing the subject of what do they sell now that they won't be able to get 11/70s after next summer. The OEM hunched up his shoulders, grunmmpffed a few times, took a deep breath and said, "now let me get this straight: I can expect 80% of the performance, 120% of the price, a lower discount schedule, and I must make a major software conversion to VAX-11 BASIC." He was comparing 11/70s to a VAX 11/780. It is also true that he runs more than one 11/70 with 120+ terminals, and doesn't think that 96 terminals is unusual for his software on the "70". The VAX.SPD only admits to 96 terminals and with DZ's we know it will be character bound.

Memory is getting very cheap. Computers have more bang for the buck. Packaging has improved and downtime is less and less frequent. Programmers make more money than ever and are harder to find. Software is now a larger investment on most machines than the machine itself. All this argues for an environment where programmer time and effort are spent in the most productive way: Building co-trees and overlays to fit into the 16 bit addressing space is inefficient use of their expensive time. 32 bits solve lots of problems. The operating system can be more complex and do more for the user. Languages can be compiled faster and optimized more into native mode rather than the 16 bit "threaded code". Programmer space limitations becomes a thing of the past. All in all 32 bits beat the devil out of 16 bits.

Despite all these facts, there are many people and small businesses out in PDP-land with large investments in hardware and software. It really is a "bet your business" move when you marry a computer. OEMs, software houses and bureaus are now required to shoulder a heavy burden in moving to the 32 bit VAX. Wouldn't it have been better if we had been given a good migration tool rather than the 'it's easy to convert to VAX BASIC and RMS' panacea that has been handed us. Shouldn't we be treated to a product that fills the needs of an established user base, instead of being forced into a world we don't need or may not want. Are we really to accept 80% of the performance, 120% of the price and less attractive terms? I know, 32 bits and VAX makes a lot of business sense to DEC; and 32 bits is the wave of the future. Yes, Yes, Yes; but is it really fair?

**SOFTWARE SOUP**

Dave Mallery

Recently, I wrote an editorial praising two software packages that I had bought and especially liked. Overnight, I was quoted in ads in every computer journal under the sun. I never realized the power of my "speechlessness"!

I have recently experienced another "fallout" from that editorial. It seems that every nascent software house in the world wants me to have a free home demonstration of their product in hopes that I will also recommend their product.

Obviously, this is not what I had in mind.

There are two reasons why I can't be everyone's beta test. First, I am very busy. Second, I am also an author and seller of software (under another hat).

So, once and for all, a policy on software editorializing in the RSTS Pro:

1) We will review anything and everything.
2) We will review only what we have bought.
3) We don't want any free home demonstrations.

Another major area of concern. We publish a LOT of programs. We even make the sources available on magtape. We are a little magazine, not a three billion dollar computer company.

ALL PROGRAMS PUBLISHED IN THE RSTS PROFESSIONAL ARE WARRANTED TO PERFORM NO USEFUL FUNCTION. THEY ARE GUARANTEED TO CONTAIN BUGS THEY ARE DESIGNED TO GET YOU OFF YOUR BUTT AND THINKING. THEY ARE INTENDED TO EDUCATE AND ENTERTAIN. THEY ARE PUBLISHED ON THE PREMISE THAT IT IS BETTER TO SPREAD PEOPLE'S BEST EFFORTS AROUND EVEN IF THERE IS AN OCCASIONAL PROBLEM. IF YOU USE THEM, MAKE THEM YOUR OWN, AND YOU WILL NOT GO WRONG. IF YOU WANT CLASS 'A' SOFTWARE SUPPORT, GO PAY DEC SIXTY FIVE DOLLARS AN HOUR.

I hate to sound so clear, but I have had it with complaints.

See y'all in Atlanta!
DMG/NET provides RSTS/E users with easy access to packet-switched (X.25) networks. It permits two-way file transfer and interactive dialogue with other RSTS/E systems and locally initiated communication with non-RSTS/E systems. From a RSTS/E host to other RSTS/E computers, to other DEC computers, even to non-DEC computers...communication is quick, simple and extremely inexpensive.

**EASY TO USE**

Even a non-technical person can access the entire communication network quickly and easily with DMG/NET. There is no need to memorize a long series of network codes, numbers and procedures. Instead, a user accesses a remote database with only the familiar local sign-on and a short, easily remembered identification code...DMG/NET does the rest.

What's more, DMG/NET allows any locally connected terminal to access any specified remote database, eliminating the need for separate terminals, complicated switch boxes or terminal setting changes.

To find out how DMG/NET can meet your RSTS/E networking needs, contact Digital Management Group Ltd.

DMG/NET

RSTS/E and DEC are registered trademarks of Digital Equipment Corporation
DMG/NET is a registered trademark of Digital Management Group Ltd.
LETTERS to the RSTS Pro...

Send letters to: Letters to the RSTS Pro, P.O. Box 361, Ft. Washington, PA 19034-0361.

Nice magazine! Keep up the good work. I'd like to comment on the "Top-down" article on p.8 in the Dec. 1981 issue. I make a living rewriting and reworking systems that are designed by people who think there is just ONE way to properly implement. I've rewritten beautifully structured code and achieved speed improvements of 20 times. Were I to utilize ALL of the structure rules that I learned in college, I would be cranking out code that ran slow, overlayed too much, and was bigger than necessary. Structured design is a tool; but, with all tools, if it isn't used wisely, it can't be used to create a masterpiece.

Steve Roy, Diversified Consulting Co.
Bloomfield, CT 06002-0284

If you clowns think I'm going to pay a 75% rate increase, you're crazy. N.D. Harris, IN

Look closer, Dale. Our rate has gone DOWN. We increased the number of publications per year not the cost per issue. Old rate: $25. for 4 ($6.25 each); New rate: $35. for 6 (5.83 each). That's a 7% DECREASE. Com'on back!

I recently got myself a subscription to your wonderful magazine. Each new copy gives me a few hours of pleasant reading and contains a lot of valuable information.

The December '81 issue however, contained at least two errors. One of them could be serious the other one makes me wonder where the author of the article did his writing.

On p. 81 of David Leffen's article, if it is true that memory has to lie physically between the CPU and the Able Cache/434. This is exactly where the memory should not be.

The Able CACHE memory's have to be installed BETWEEN the CPU and the MEMORY it has to cache. Normally the 434 will replace the jumper between the CPU-backplane and the next backplane. All memory can then be installed in the second backplane. The CPU-backplane can be used for simple I/O devices like DL-11's, p.t. equipment and the like. DMA devices should follow the memory's.

Ten years of experience with PDP 11's gives me the following preference:

First: CPU plus attached boards
Then: Simple I/O devices (non DMA)
If applicable: External cache memory's (like the Able 434)
Always: All memory
Followed by: Fast DMA devices
And last of all: Slow DMA devices

Keep the bus as short as possible. A good system will even run without any problem with a DC bus-load of more than 20.

On page 16, the right hand column 8 lines from the bottom, Michael Schwartz states that a data encryption utility should be available for use in BATH. I wonder why. Is he afraid of spying submarine's circling underneath the water surface in his tub? Does he store confidential information disguised as tooth paste or do we need plumbers for maintenance on these utilities? On reflection the above opens complete new uncovered grounds for research.

Keep up the good work!
Jan Willem Brier, Datcare B.V.
3700 AA Zeist, The Netherlands

Dear Dave and Carl,
Thank you for your kindness in dedicating your February, 1982 issue to me. Besides expressing my heartfelt appreciation for your act of friendship, I must tell you that I want to share this recognition with the "RSTS Team", to whom the credit must belong.

I speak not only of legendary superstars such as Mark Bramhall and Anton Chernoof. To name but a few, Jim Miller, Jim "Woody" Woodriddle, Nancy Covitz, Joe Mulvev, Mark Goodrich, Rich Witek, Steve Morris, Bill Joyce, Jim Condict, Andy Riels and Bill Sconce are among the scores of competent professionals that I have worked beside over the years. And I speak not only of the software engineers (more familiarly known as "developers"), but also of the writers, who are the unseen hero(ines), as well as Software Support specialists such as Martin Minov, my outspoken "conscience."

Equally important, it was my pleasure to meet hundreds of users, talked to and listened to. Some of them, symposia coordinators, SIG chairmen and other leaders, I grew to know better than others as we worked together, but the collective experience of interacting with the entire community of the RSTS SIG has been most educational to me. I thank them all.

Lastly, I would like to reciprocate by wishing you and your readers continued success with RSTS, as I am sure you will have. And whatever I am working on, RSTS will always be close to my heart. Yours sincerely, Simon Szeto

Many thanks to you and Paul R. Laba for his fine article on FIP's Alignment Algorithm (Sept. '81). This undocumented design feature has bitten me several times. Please note: If you have a disk giving "Bad Directory for Device" errors which the various CLEAN's won't fix, there is a good chance that the FIP Alignment Algorithm is the problem. It does seem like DEC could at least report these alignment problems as errors in the CLEAN's.

Chris Rapp, System Manager
CSULA Computer Center
P.S. I wear large, if you're still giving away T-shirts.

I have just finished reading the "Benchmark DIBOL vs BASIC + 2" in your December issue and feel that a few comments are warranted.

1. The article is hardly a comparison between DIBOL and Basic + 2. It really compares IMSAM file handling and DIBOL Iam file handling. It is very much a case of apples and prunes, is it not?
On the one hand we will have RSTS written in macro with who knows how many thousands of man days behind it and on the other a very simple single key IMSAM structure written mostly in a high level language.

2. There would appear to be no justification for the comparison, given that DIBOLR is available which also utilizes the IMSAM file structures. Now that would be a comparison worth taking note of. I wonder what the I/O would be like when BASIC + 2 starts reading on its RMS overlays (especially for storing new records) while the DIBOLR program can do it all from its task image.

When I open the pages of RSTS Professional, I expect to read balanced, unbiased articles that have been filtered through a panel of competent referees. If you wish to print largely irrelevant comparisons hiding behind the term "Benchmark", then perhaps your publication should be called the RSTS Amateur.

I must confess to being a DIBOL fan ever since COS350, however, I think my comments are reasonable.

I have just attempted to implement CALLER-BAS (v.3, 64, p.76) with a fair degree of success. However, the printed version contains some drawbacks.

1. The "Help" feature precludes use of the DEC HELP package.
2. The CCL parser expands commands to their... continued on page 71.
There really is a difference . . . .

THE

DOPTER
DIFFERENCE

DOPTER is an easy to use RSTS/E disk copying program which produces an optimized disk structure providing greater system throughput and improved response time.

Here's Why
With only two short commands, DOPTER automatically performs the following:

- Properly places and pre-extends the MFD. Yes, the first cluster of the MFD must start at device cluster zero, but DOPTER only writes in the first block of the first cluster. All of the rest of the MFD entries are in the interior of the volume where they belong.
- Places and contiguously pre-extends the UFD's. The UFD's, as well as the MFD, are only extended as much as is necessary to contain their current information plus some room for expansion.
- Places the UFD's with the most activity toward the front of the MFD for quickest access.
- With its proprietary techniques, produces better optimized MFD/ UFD's which remain that way longer than does REORDR or any other process.
- Deletes unused file attributes from source, task, and object library files saving UFD and cache accesses.
- Places the most used files at the front of the UFD's for quicker access.
- Places the most used files in the center of the active files. Places "unused" files separate from the active files leaving the active files more compactly placed.
- Performs all steps starting with disk initialization and ending by "hooking" the output volume and installing the current SIL without operator intervention.

Furthermore
DOPTER optimizes file cluster size and makes files contiguous where appropriate.
DOPTER preserves previous bad block information on the output volume so that patterns need not be re-run.
DOPTER prevents accidentally copying an older DOPTER'd volume onto a newer one.
DOPTER preserves all accounting data.
DOPTER allows all volume, UFD, MFD, and file defaults to be manually overridden.

For More Information
If you would like more information on the DOPTER DIFFERENCE, mail the coupon, circle the Reader Card number, or call us. We'll send you a free copy of the DOPTER User Manual and License Agreement.
Single CPU price: $2,500.

Please Send More Information To:
Name _____________________________
Company _____________________________
Address _____________________________
City ___________________ Zip ________
Telephone _____________________________

RSTS/E is a registered trademark of Digital Equipment Corporation

System Performance House, Inc.
5522 Loch More Court • Dublin, Ohio 43017 • 614-265-7788

CIRCLE 106 ON READER CARD
Logging Into An Account Without LOGIN

By Patrick Holmey & Robert Schilmoeller, Computation Laboratory, St. John's University

"JUMP" provides RSTS users the ability to cross from one account to another without using the "LOGIN" program. It was mainly designed to minimize the frustrations and headaches of having passwords for every privileged account, reduce the number of times one has to look up a password for any account whether it be privileged or non-privileged, and to be able to log into those accounts that may have an "*" for a password.

This program was written in BASIC-PLUS for RSTS/E Version 7. It was developed to run as a stand-alone, CCL or Chain Entry program (Line 30000 should be specified for CCL entry; line 30999 has been designated for Chain Entry).

The user can log into a specific account using the following three methods. The first method is by entering the project-programmer number separated by a comma. The second method is by entering the project-programmer number separated by a slash. Finally, the third method is by entering a wildcard for a specific account (the entered wildcard is checked with those present in WILDCARDS in the program).

Once the user has typed the specific account number he/she would like to jump to, all temporary files created with that user's job number in the original account will be deleted. All system accounting data is updated. If the user enters a comma to separate the project-programmer number, those jobs detached under the specified account, if any, will be displayed. If there are any detached jobs, the user will be prompted for the job number to attach. The total number of users logged into the new account, if any, will also be displayed.

This program has been running smoothly for the last year. It has saved a tremendous amount of time logging into a specific account. Should there be anyone out in RSTS-land who would be interested in obtaining this program, you can contact us. (This software is being provided for nothing, therefore, we do not feel obligated to maintain it.)
For more than a decade, SI has provided reliable disk storage alternatives to DEC and Data General minicomputer users in business, education and government.

20,000 Installations. SI has given thousands of DEC and DG users just what they wanted. Reliability. Flexibility. Broad choice. Fast delivery. And savings in both money and floor space.

A Worldwide Service Network. SI maintains fully staffed and equipped service centers throughout the U.S. and Europe to ensure prompt response and complete repair capability.

The reliable alternative for over a decade.
Structured disks for all!
Announcing
REACT2.TSK

- a 'REACT' replacement that locates & extends UF'Ds
- 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

SPSS® makes data analysis simple for DEC
PDP-11 users! Now PDP-11 users can enjoy all the
benefits that have made SPSS® the world's largest
selling Data Analysis System. It's easy to use and
learn, thanks to its response to English language
commands and comprehensive documentation.
It's also sophisticated, giving researchers and
business managers alike a full range of capabili-
ties for statistical analysis and report generation.
For full information, call or write SPSS® today:
Sue Phelan, SPSS®, Inc., 444 N. Michigan Avenue,
Chicago, IL 60611, 312/329-2400.

© Copyright 1981 SPSS, Inc.

SPSS-11 runs on DEC LSI-11 through PDP-11/70. Compatible with DEC Systems RSTS,
RT-11, RSX-11M, IAS/Bell Labs UNIX/S&H Computer Systems TSX.

CIRCLE 123 ON READER CARD
It started the day the cardboard keyboard arrived. You know; you all received one. Certainly not your typical four color glossy folder-in-an-envelope promo. Rather, a full size, full color, cardboard keyboard announcing DECWORD, a new DEC software product. It seemed that DEC had made an arrangement with Data Processing Design to market a version of their product, WORD-11. By simply mailing in a postcard, you would receive a version of the Computer Based Training Program, a VT100 Advanced Video Option, and a genuine, authentic, brand new VT100 keyboard. Now I have a confession to make. First, we already have WORD-11 on our system. We have no need for DECWORD. Second, we have a vast mixture of terminal types already set up with WORD-11, DPD supplied keycaps or stickers. Third, all our VT100s already have the advanced video option. But, who can resist a freebee? After all, Word-11 does not have a computer aided instruction system. Also, who wouldn’t like to have a spare keyboard available. (An industrial site I worked at. I did. The call-logging person had never heard of DECWORD or the AVO, so I called DEC. The employee who answered the phone had never heard of DECWORD. . . . continued on page 89.

The next step was to load the training tape. No problem, PIP!!

The direction sheet is clear. No CCL’s to eat up those small buffers. One logical. I ran the product on my VT100 with the original AVO. It worked excellently. It also worked on a DT-80 and a VT52. The Hazeltine 1552 had some functions, but not all. I think it will work to help train out new personnel in WORD-11. Unfortunately, the program does change the terminal characteristics without restoring them. Namely, CTRL/R and ESC SEQ. It appears DEC has accepted this type of inconsistent terminal hocus-pocus as a standard.

Now for the Advanced Video Board. I didn’t need it, but I wanted to try it. The included installation book is excellent, with the exception of the page describing the AVO switches (page 14). First of all, there are two sets of switches. The manual describes how to set one set but not the other. The manual describes setting the “E19 switch pack”. There was no E19 switch pack on my board, only E18. The manual describes the switches as being ON or OFF. Mine were OPEN. (We know from DZ experience that OPEN can mean both OFF and ON). My switch pack was upside down compared to the switch pack in the picture or were the numbers on backward? No matter. I tried what I thought was the correct setting. And another. And another . . . Then I cut my hand with the screwdriver.

After many more attempts, I had experienced the ultimate high in visual display fantasies. However, no advanced video. The manual (EK-100CK-IN-001) clearly states,

**IF YOU NEED HELP:**

* Call Digital Field Service

I did. The call-logging person had never heard of DECWORD or the promo package. However, DEC would be glad to send over one of those Field Service Technicians at $76.20 per hour. When I re-explained the problem, she gave me the phone number of the terminal group. I called. After three conversations, I was passed to a terminal engineer. So far, no one had ever heard of DECWORD. The engineer sounded friendly and wanted to help. He did suggest I call Field Service. He also admitted that he had heard of a new AVO board with switches, but had never met one in person. He assured me that E18 meant E19, and that OPEN meant OFF except on DZ's. He took the manual number and told me if he could find some additional information he would call back. I tried his ideas. (I had tried them all before — honest.) The board still failed. All that was left was the DEC SALES OFFICE. I dialed the number on the cover letter. The employee who answered the phone had never heard of DECWORD...
IMPRS
A Productivity Relational Data Base Language
By Jacob F. Ruf, Ruf Corporation, Olathe, Kansas

ABSTRACT

IMPRS (Information Management Processing Reporting System) is a software language product of Ruf Corporation.

IMPRS provides the capability of developing complete application software from five to ten times faster than would be possible using conventional programming approaches. This productivity is accomplished through the modular, operation-oriented structure of IMPRS and its powerful relational data base language. The IMPRS user is commanding his file management processing and reporting in terms of "What needs to be accomplished?" as opposed to the detail level of "How?" required by conventional data base managers.

The flexibility and efficiency of IMPRS with its imbedded high-speed sort, versatile record subset selection, and interactive report generation make it the most powerful relational data base programming language currently operating on hardware in this size and price range. The execution speed of IMPRS programs is two to three times faster than that of COBOL or BASIC-PLUS.

IMPRS has been in constant use on Ruf Corporation public time-sharing DEC 11 systems and many other computer systems in the Kansas City area since March, 1977. Application packages (over 2,500 programs) are available supporting practically all information management fields. The basic IMPRS package consists of over 500 Fortran subroutines linked into over thirty (.SAV) programs complete with documentation.

IMPRS is currently supported on DEC 11 systems running under RT11, RSX, and RSTS/E operating systems.

INTRODUCTION

A systems analyst must first understand the philosophy under which a business operates before he can successfully develop an information system to serve that business. This is necessary because the information system which he develops must operate under the same philosophy.

Successful businessmen are goal-directed. Their decision making is performed with the use of rational processes. They measure the attainment of their goals through quantifiable evaluations.

To effectively place this philosophy into operation in the management of our business we continually ask the questions outlined in Figure 1.

In order to answer those questions, we are continually drawing upon information. With the advent of the computer, the information system has become more structured in serving management's needs. Today's information systems consist of people, data, hardware, software, and education or procedures.

The value of the computerized information system is illustrated in Figure 2. The purpose of an information system is to (1) collect and manage data, (2) process data into information and (3) to disseminate the information to people for their use in answering the questions of Figure 1 (above).

FUNCTIONAL USE OF DATA

INFORMATION DELIVERY SYSTEM

DISSEMINATION PROCESS

INFORMATION SYSTEM HUMAN SYSTEM

FIGURE 2.

DATA MANAGEMENT — THE KEY

The one component of the information system that "flows" and involves all the other components is data. Suc-
WHEN SHELL OIL WANTED A FINANCIAL PLANNING PACKAGE FOR THEIR DEC INSTALLATIONS WORLDWIDE, THEY CHOSE FPS80.

WHY FPS80?

IT'S A WELL PROVEN PRODUCT, RUNNING ON A WIDE RANGE OF MACHINES, WITH BUILT IN SOPHISTICATED FINANCIAL FUNCTIONS AND IS USED EXTENSIVELY VIA TIMESHARING.

IS FPS80 FOR YOU?

IF YOU RUN RSTS/E, RT11, RSX11-M OR VMS AND YOU'RE LOOKING FOR THE BEST IN INTERACTIVE FINANCIAL PLANNING WE ARE YOUR FIRST CHOICE.

FOR INFORMATION, WRITE TO
D HOLROYD
RTZ COMPUTER SERVICES
103 JERMYN STREET
LONDON SW1Y 6EB
OR 'PHONE 01-920 4163
successful system engineering has proven the need to single out the categories of data and the functional use of this data in the operation of the business. See Figure 2A for an example.

James Martin (1) has been referred to as the "computer industry's most widely-read author and best-attended lecturer". He states that "by the end of the 1970's, much of the computing in industry and government will relate to the data bases which have been painfully constructed piece by piece and management effectiveness will relate to the quality of their organization's data sources and the versatility with which they can be used".

![Diagram: Summary Data, Detailed Data, Data Bank, Functional Data Bank]

**SOARING SOFTWARE COST**

Throughout the last decade, program development productivity on a national basis has not increased. Recent studies by Ferrentino (2) show that a programmer produces about ten lines of program source code per day, as was the standard ten years ago.

Today, the cost of software equals 70% of the total cost of operating a computerized information system. Hardware cost is equivalent to only 30% of that total. By the end of this decade it is estimated that the ratio will be 90% software compared to 10% hardware.

**LOW U.S. PRODUCTIVITY**

The United States has an annual productivity growth rate of about zero compared to more than 11 percent for the Japanese. The time lag between the development of a new technology and putting it to use averages seven years in the United States. In contrast, the technology transfer time for West Germany is five years and that for Japan is only three years.

It is a fact that over 50% of all American technology is originated in small businesses. However, large financial resources are required to adequately implement technology. Due to the lack of capital in small business and the status quo approaches of large business, the United States has fallen drastically behind in our productivity. The low productivity growth is a contributing factor to the increasing inflation in this country.

**PAST SOFTWARE QUANTUM LEAPS**

Computer language productivity has developed in a series of leaps throughout the history of the computer. Figure 3 illustrates these leaps in productivity. Language technology advances have generally been accompanied by a seven to one increase in productivity. It usually takes seven times longer to develop a program in assembly language as it does in COBOL or Fortran.

![Diagram: Application Software Productivity, assembly language, machine language, COBOL, FORTRAN, BASIC, RPG-II, IMPRS]
According to James Martin (5), "Until recently, 90% of commercial programming was done in COBOL. We cannot go on using COBOL," he said, "because COBOL requires too many scarce programming resources. PL/1 faces the same fate." Martin believes we need a fourth generation of programming languages to begin to solve the "colossal problem" of programming productivity. Recent language developments do not solve this problem. It appears that we are overdue for a quantum leap in software productivity.

PAST "BRIGHT" APPROACHES

During the 1970's, formalized software was developed to act as the "file secretary" and to augment the host language. This software supported the programmer with the retrieval and filing of data upon his programmed request. These systems were called Data Base Management Systems (DBMS).

With the stress on centralized control of data, it appeared that the DBMS of the 70's offered the best solution to information processing. But, because of the complexities and lack of flexibility of the hierarchical tree and complex approaches used in the early DBMS, disappointments and failures have been widespread. As with many innovations, users sometimes got wrapped up in the methodology of the system and lost track of the desired goal. These "LBJ's" of the New Frontier in the 70's had taken over. These "Lightning Bug Jocks" were so concerned about "How it works" that they lost track of providing what was needed. They were bright boys, but it was all in their rear end. I am reminded of a saying of my Chemical Engineer Thesis Professor at the University of Kansas, the late Dr. Fred Kurata. He was teaching the thermodynamic law on work where work is equal to the product of Pressure times Volume, i.e. \( W = PV \). He made the observation to the class that "We can have all the P in the world, but if we don't have any V, we will not have any work". As we look back on the history of the DBMS, in many cases it seems that the mountain toiled and brought forth the mouse.

"The structured techniques of the 1970's are completely inadequate," says James Martin (6). "We need new types of methodology, with end user involvement, user-driven computing." It appears that we have overdone the methodology of the hierarchical DBMS. With the stress on centralized control of data, it appeared that the DBMS of the 70's offered the best solution to information processing. But, because of the complexities and lack of flexibility of the hierarchical tree and complex approaches used in the early DBMS, disappointments and failures have been widespread. As with many innovations, users sometimes got wrapped up in the methodology of the system and lost track of the desired goal. These "LBJ's" of the New Frontier in the 70's had taken over. These "Lightning Bug Jocks" were so concerned about "How it works" that they lost track of providing what was needed. They were bright boys, but it was all in their rear end. I am reminded of a saying of my Chemical Engineer Thesis Professor at the University of Kansas, the late Dr. Fred Kurata. He was teaching the thermodynamic law on work where work is equal to the product of Pressure times Volume, i.e. \( W = PV \). He made the observation to the class that "We can have all the P in the world, but if we don't have any V, we will not have any work". As we look back on the history of the DBMS, in many cases it seems that the mountain toiled and brought forth the mouse. "The structured techniques of the 1970's are completely inadequate," says James Martin (6). "We need new types of methodology, with end user involvement, user-driven computing." It appears that we have overdone the methodology of the hierarchical DBMS. Data processing organizational goals have become increasingly unrelated to the goals and objectives of the institution they serve. Those data processing people who have not experienced the problems of the early prototype DBMS should feel fortunate that there are much more efficient approaches available today.

IMPRS PRODUCTIVITY

Information Management Processing Reporting System (IMPRS) is a relational data base software product developed by Ruf Corporation which gives that quantum leap in software productivity. Since the development of IMPRS in 1976, we have experienced and documented software development productivity factors of 5 to 30 times faster with IMPRS. In a recent comparison of 54 payroll and accounts payable programs, we were able to develop at the rate of 300 equivalent COBOL lines per day compared to the national average of 10 per day. This is 30 times faster with IMPRS.
We have developed complete business systems in COBOL and BASIC-PLUS and executed them for several years on our time-sharing systems. We then redeveloped the systems in IMPRS. We are not only seeing the quantum increase in program development performance with IMPRS, but are experiencing execution speeds of two to three times faster than the equivalent COBOL or BASIC-PLUS programs. These observations were based on reliable studies with the use of our time-sharing computer resource accounting software over a four year period.

The imbedded high-speed sort, versatile record subset selection, and interactive report generation make IMPRS a powerful data base programming language. The productivity is accomplished through the modular, operation-oriented structure of IMPRS, with its highly flexible relational data base.

The key to IMPRS is the interactive nature of its processing and reporting. The IMPRS user is commanding the file management in terms of "What results are required?" as opposed to the question "How must we get the result?" used in conventional application software. This very important difference translates into a more versatile, flexible and faster system.

**IMPRS — A RELATIONAL DATA BASE**

James Martin (1) supports the relational approach over the hard-linked hierarchical methods, stating, "There is a simple and more elegant method — the use of relational data bases." He defines a Relational Data Base as "a data base made up of relations. (Flat file, two-dimensional array of data elements). Its data base management system has the capability to recombine the data elements to form different relations thus giving great flexibility in the usage of data".

He continues, "Throughout the history of engineering a principle seems to emerge: great engineering is simple engineering. Ideas which become too cumbersome and inflexible tend to be replaced with newer, conceptually cleaner ideas."

It is possible to avoid the tangled webs that build up in tree and plex structures by a technique called normalization. Normalization is the subdivision of a complex data base into its lowest common denominator of relationally linked flat files. This technique was originally designed and advocated by E. F. Codd (3). Codd sets relational data bases apart from hierarchical and other popular designs by its ability to automatically navigate to data.

The advantages of the relational data base approach outlined in Table 1 below have been clearly substantiated over five years experience with Ruf Corporation's IMPRS.

**IMPRS COMPONENTS**

Figure 4 portrays the more visible components of IMPRS, which are Relational Data Base Management Systems (RDBMS), Data Manipulation Language (DML), Query Language Interface (QLI), and Utilities.

Figure 5 more accurately illustrates the components of IMPRS and their interrelation.

---

**TABLE 1**

<table>
<thead>
<tr>
<th>Relational Data Base Advantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) Simple and clear</td>
</tr>
<tr>
<td>Supports structured modular approaches</td>
</tr>
<tr>
<td>(2) Transparent to change</td>
</tr>
<tr>
<td>Data independent</td>
</tr>
<tr>
<td>(3) Flexible access and relatability</td>
</tr>
<tr>
<td>Geographically independent links</td>
</tr>
<tr>
<td>Logical linkages</td>
</tr>
<tr>
<td>(4) Ease of reporting</td>
</tr>
<tr>
<td>(5) High level DML support</td>
</tr>
<tr>
<td>(6) Minimum data redundance</td>
</tr>
<tr>
<td>(7) Completeness</td>
</tr>
<tr>
<td>Ease of use and implementation</td>
</tr>
<tr>
<td>Flexibility</td>
</tr>
<tr>
<td>Precision</td>
</tr>
<tr>
<td>Security</td>
</tr>
</tbody>
</table>
The RDBMS, Data Description Language (DDL), and Data Dictionary (DD) serve as the system level components and are imbedded throughout the operational components which are the Data Manipulation Language (DML), Query Language Interface (QLI) and Utilities. The heart of the IMPRS operational components is the DML and QLI.

With the DML we have written a complete interactive order entry program in four hours. This program built the Order Master and Item files, checked on customer credit and part availability, relieved inventory and priced the order. This was completed with the use of 150 DML program steps.

The QLI provides on-the-spot new management reporting, utilizing menu directed approaches, guiding the novice step by step to the desired reports. Utilizing the Boolean record selection, the user has almost unlimited subset selectivity in his reporting.

**IMPRS FEATURES**

IMPRS is written in a high-level ANSI language - Fortran IV - and consists of over 500 linked subroutines ... complete with documentation. We have found Fortran a most appropriate language for developing systems of this nature. IMPRS is a portable system, providing the capability to operate on a multitude of different computers as it rigidly adheres to ANSI programming standards.

IMPRS has been in constant use on Ruf Corporation's public time-sharing systems, and other systems, since 1976. It has proven to be an effective system in actual information management situations ranging from private business to medical research labs; education to manufacturing information systems.

IMPRS is currently supported on Digital Equipment Corporation PDP 11 systems running under RT11, RSX, and RSTS/E operating systems.

The main features of IMPRS are listed in Table 2 below.

---

**TABLE 2. IMPRS Features**

**Information Management (RDBMS)**
- Full Relational Linkage (Interactive/Batch)
- Interactive or Batch DB Management
- File Maintenance Utilities
- Full Cursor Control
- High-Level Structured DML

**Information Processing (DML)**
- Interactive Multi-File Processing
- Multi-User Simultaneous File Access
- Inter-Job Communication
- Imbedded Sorts
- Structured Data Processing Language

**Information Reporting (QLI)**
- Interactive Multi-File Report Generator with Boolean Record Selection
- Full Report Writer & Report Generation
- Multi-Contingency Frequency Reporting
- Data Quality Consistency Reporting
- High-Level Structured Reporting Language

**Information System (RDBMS)**
- High-Level Structured Programming Language
- Security - System to Item Levels
- Multiple Key Access to Files
- Variable Length Records and Fields
- Interfaces to All Languages
- Data Dictionary
- Full Interactive Programming Support
- Automatic System Documentation
- Automatic System Flowcharting
- Full Debugging Aids
- Data Base Disk Usage Management Aids
- Interface to Computer Utilization Accounting
- Interface to High-Level Simulation Language
- Full Cross Reference Programming Aid
IMPRS EXAMPLE

Figure 6 illustrates a daily work-in-process and labor performance system which was developed in a four hour period using IMPRS. A similar system developed previously using COBOL took two man-months. The IMPRS system consisted of a DML program (DAYWK1.TRN), utilizing three files (EMP, STD, and DAYWK), and a QLI interactively reporting labor performance. The DML program interactively received daily labor statistics from the keyboard and extended them using the employee master and job standard files. It then calculated the labor performance and wrote the record into the DAYWK.DAT file for future reporting.

The relational linkage of the employee master and job standard master files to the WIP master is demonstrated in Figure 7. The linkage is performed without the use of hard disk pointers.

| TABLE 3. IMPRS DBPAR Parameter File Listing 15-Nov-81 19:00 |
| Parameter file: EMP.PAR | 2 Blocks |
| Data file: EMP.DAT | 2 Blocks |
| Key file: EMP.KEY | 1 Block |
| Description: EMPLOYEE MASTER |
| Control item number: 0 Maximum number of records: 20 |
| Number of headings: 0 Number of bytes per record: 30 |

| TABLE 4. IMPRS DBPAR Parameter File Listing 15-Nov-81 18:59 |
| Parameter file: STD.PAR | 2 Blocks |
| Data file: STD.DAT | 2 Blocks |
| Key file: STD.KEY | 1 Block |
| Description: STANDARD RATE MASTER |
| Control item number: 0 Maximum number of records: 25 |
| Number of headings: 0 Number of bytes per record: 28 |

The listing of the data description language (DDL) for the three files is included as Tables 3, 4, and 5. These tables contain all the data attributes that the data base required for this system.

The listing of the DML program (Table 6) consists of fourteen commands and six calculations. The command structure consists of a sequence name, alpha command, and four parameters. The parameters make reference to file numbers as described in the beginning of the DML, item numbers described in the DDL (Tables 3-5), DML sequence names, and other required attributes. The calculations are directed in terms of file numbers, item numbers, and the standard arithmetic operators (+,-,*,/). The function of each command is listed in the command description column.
EMULEX IS MORE THAN ABLE TO COMMUNICATE WITH DEC.

We're also able to save you plenty. For instance, you get DH11 performance for a DZ11 price. Four new space-saving single-board communications multiplexers. And an increase in VAX-11 terminal handling capacity by up to 50%. Maintained nationwide by Control Data. Microprocessor-based architecture and common hardware deliver faster, more flexible line-handling. Self-test on power-up. Full software transparency. And Emulex reliability standards.


CS11/H (PDP-11) $7560 for 48 lines* CS11/Y (VAX-11) $7884 for 48 lines*
CS11/U (VAX-11) $7884 for 48 lines*
CS21/H (PDP-11) $2520 for 16 lines
CS21/U (VAX-11) $2520 for 16 lines
CS21/Z $2550*
CS21/Z (VAX-11) $2944 for 16 lines*
CS21/Z (PDP-11) $2520 for 16 lines*

Up to 64 DH11 channels from one board.
DH11-compatible MUX lets you mix RS-232 & current loop interfaces in 8-line groups.
Built-in DM11-compatible modem control. DMA output eliminates host interrupts. Self-test capabilities. Transparent to PDP-11 software. Emulex own software on VAX.

Higher DV11 performance, lower price.
DV11-compatible multiplexer. Mixes 8-lines synchronous & asynchronous on PDP-11s. Ideal for Bisync & DECNET. 8-32 lines per controller. DMA input & output. Software transparent under DECNET. Compact package offering higher line-handling speeds & improved throughput.

Replace DEC DZ11/E and save.
Perfect if you don't need DH11 performance. Software-transparent to all DEC operating systems. Easy PROM change enables quick upgrade to DH-11 performance. Saves one slot per 16 lines.

New economical DH11-type multiplexer.
Lowest cost, high-performance communications MUX. Priced way less than DEC's DZ11, with DMA to boot. 16 RS-232 lines per board, modem control included. Can use H317 distribution panel. Transparent to PDP-11 software. Emulex software on VAX.

*Price each in 100 quantities. All Emulex disk, tape, and communications products can be combined to reach quantity price breaks.

CIRCLE 58 ON READER CARD
TABLE 5.
DATA DESCRIPTION LANGUAGE FOR VIP DATA BASE

<table>
<thead>
<tr>
<th>IMPRS</th>
<th>DBPAR Parameter File Listings 09-Nov-81 17:35</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter file: DAYWK.PAR</td>
<td>3 Blocks</td>
</tr>
<tr>
<td>Data file:</td>
<td>DAYWK.DAT</td>
</tr>
<tr>
<td>Key file:</td>
<td>DAYWK.KEY</td>
</tr>
<tr>
<td>Description:</td>
<td>DAILY WORK HRS</td>
</tr>
<tr>
<td>Control item number:</td>
<td>8</td>
</tr>
<tr>
<td>Maximum number of records:</td>
<td>50</td>
</tr>
<tr>
<td>Number of headings:</td>
<td>0</td>
</tr>
<tr>
<td>Number of bytes per record:</td>
<td>96</td>
</tr>
</tbody>
</table>

Print Format

(1#H; 2(/I))

For demonstration purposes, I have selected the interactive reporter from the battery of IMPRS reporting facilities. Table 7 illustrates the ease of reporting and selecting break totals. The user can select sub-sets of data for reporting required for labor performance which can be accomplished from the DAYWK.DAT file.

SYSTEM DESIGN TESTING WITH IMPRS

Because of its relational orientation, IMPRS users are not required to define all data and potential uses that will be required before they can efficiently design their system. The past inflexibility of the hard-linked tree and plex approaches have forced costly unnecessary planning which seldom gets implemented in their rigid top-down design. The cost of changing the hierarchical software is so great that it forces the system planner to cover all possibilities, disallowing the use of the more productive modified top-down design explained by Samid (4).

In my 22 years of business system design and development of large information systems, I have never experienced where it was possible to define all logical relations and usages that would be required of the system throughout its life. The most productive system design and development...
Consider the products and prices below:

You’ll soon see that Emulex is as unbeatable on price as we are in quality and performance. Included are two new Winchester disk controllers with remarkably high performance and reliability (MTBF over 70,000 hours).

All Emulex Q-bus tape and disk controllers have the same microprocessor architecture and all the key features of our PDP-11 and VAX-11 products, including error correction, microdiagnostics, and software transparency. Price our performance. Write or call: Emulex Corporation, 2001 Deere Ave., Santa Ana, CA 92705; (714) 557-7580; TWX 910-595-2521.

Outside Calif.: (800) 854-7112.

For immediate off-the-shelf delivery, call our national distributor: First Computer Corporation, 645 Blackhawk Dr., Westmont, IL 60559; (312) 920-1050.


Put big SMD drives on your LSI-11.

Links Q-bus with 1-2 SMD-type drives. Software transparent & media compatible with DEC RM02, RM05, RP06. Features 3-sector data buffer, 32-bit ECC, up to half a billion bytes capacity. Over 1500 units in service!

Low cost for smaller-sized disks.

Single quad-board interfaces LSI-11s to 8" & 14" SMD hard disk drives. Same great SC01-level performance in most applications. Software transparent. Full 32-bit ECC, self-test, 512-word bootstrap, real-time clock control, and bus terminators. Mix and match drives on one controller. 72,000 hours MTBF!

New! ANSI interfacing for 8" Winchesters.

Supports up to 8 drives per single quad-board controller. Fits into any single LSI-11 back plane quad slot. Same design, performance, and high reliability as the SC02.

Q-bus embedded dual-density tape controller.

Handles all open-reel half-inch tapes—800/1600 bpi, operating at 12.5-75 ips. Compatible with DEC’s TU10/TM11. Daisy-chain up to 4 drives. Firmware includes a self-test and extended diagnostics. Fully embedded.

The genuine alternative
TABLE 7.

IRPT

- - I M P R S - -
by Ruf Corporation

Enter Manager parameter file name: DAYWK

Data file is DAYWK.DAT
Description is DAILY WORK HRS
Key file is DAYWK.KEY
# Records= 10
# Deletes= 0
Control total= 20.00000

Function codes are:
<0> Print this menu
<1> Sort keys <3> Chg. key, sort & print
<2> Get & print <4> End
Function= ? 1

Sorting . . . Please wait
Function= ? 2

Enter <1> for report defaults
Top-of-file available <1>/yes 1
Enter item numbers to include in report
Item no. 7
Item no. 9
Item no. 8
Enter <1> for item total 1
Item no. 13
Enter <1> for item total 1
Item no. 14
Enter <1> for item total 1
Item no. 1
Enter number of breaks 1
Item no. 1
Enter <1> for page change after break
Enter <1> for totals only
Enter <1> to print all 1

---

RUF, JACOB
ASSEMBLING 45.00 5.00 11.1
DRILLING 41.00 -1.00 -2.4
FILING 50.00 5.00 10.0
136.00 9.00

WOODLE, FORREST
CUTTING 63.00 -9.00 -14.3
DRILLING 49.50 -13.50 -27.3
FILING 55.80 -6.30 -11.3
168.30 -26.80

WOODLE, RUTH
DRILLING 36.00 -20.00 -55.6
FILING 40.00 -13.60 -34.0
RASPING 24.00 4.80 20.0
SNEEPING 52.00 -12.00 -23.1
152.00 -40.80
456.30 -60.60

The low cost of developing application software with IMPRS provides the ability to effectively utilize the top-down loop-back method. With IMPRS, the analyst can perform a coarse requirement definition and a coarse system design, then proceed to the development stage to test out certain design assumptions. He can therefore locate potential system problems, loop back, and retune his system design resulting in considerable overall savings.

This guided trial and error approach is not new to us. Engineers throughout the ages have developed scale models of their planned constructions to test the various design assumptions. It would definitely have paid off if this method had been used in the design of the catwalk of the Kansas City Hyatt Regency Hotel.
SUMMARY

The data base management systems of the 80's must be designed to interact with the real world. They must be flexible, providing for logical access and reporting, ease of future linkage and change. These qualities are inherent in IMPRS, a Relational Data Base Management System (RDBMS).

Because of these qualities, IMPRS users are able to isolate sub-areas and develop them with the assurance of future linkage and minimum modification.

Martin (6) says, "For the end user to program, data must be organized and represented in simple fashion. Software engineering, the process and logic of data, must be ‘minimized’ while information engineering, the organizing of data, must be ‘maximized’".

Relational data base technology linked with data manipulation and query language capability will have a significant effect on the characteristics of our organizations. Ultimately, the availability of on-line information at much reduced cost will alter the role of management. The RDBMS and high-level languages will bring the power of the computer closer to the user. Computer hardware design will accommodate the techniques of the relational data base along with higher level languages such as that available in IMPRS.

At Ruf Corporation we have a motto which states, "We work smarter and our software works harder for you". We have accomplished this with IMPRS, a relational data base management system for the 80's.

REFERENCES

XOREN IPL-11...

...THE SOFTWARE PACKAGE WHICH TRANSfers FILES...
...BETWEEN DEC PDP-11's LSI-11's & VAX-11's...
...EVEN WHEN THEY HAVE DIFFERENT OPERATING SYSTEMS

- INTERFACE HARDWARE -
XOREN IPL-11 links the two CPU's together. 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. Recovers from errors by re-transmitting only the blocks affected.

- TRANSMISSION -
Asynchronous transmission with selectable speeds up to 9600 baud. Any type of file including binary program files may be transferred in either direction (Note RMS files should first be converted to sequential files).

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

EXISTING INSTALLATIONS — Over 100 copies of XOREN IPL-11 are currently installed.

XOREN IPL-11 available from:- US: EEC Systems
286 Boston Post Road
Wayland
MA 01778, USA
Tel (617) 358 7782

UK & Europe: Xoren Computing Ltd
28 Maddox Street
London W1R 9PF
England
Tel (01) 629 5932

Note:- Xoren is currently setting up a network of distributors in the US. For further information contact Xoren Computing direct.
EDT HINTS & KINKS

By David Spencer, Infinity Software Corp., 2210 Wilshire Blvd., Suite 801, Santa Monica, CA 90403

1.0 INTRODUCTION

Last issue I discussed an EDT initializer file. That ini­
itializer allows EDT to perform buffer manipulation and
input/output. This article is dedicated to making the most of EDT.

2.0 EDT'S INTERNAL TABLE

Each possible editing keystroke has a unique number
for it in an internal EDT table (figure 1). EDT allows access to
these keystrokes by both mnemonics (such as “GOLD CONT
Z”) and the internal number. There are some obscure
keystrokes that are definable only by their internal number,
and some can be defined, but cannot used at all!

Besides being of general interest, knowledge of the
numbering scheme provides us with some useful functions.
First, we now know the limits to key definitions. No key
that is not listed in the table may be defined for editing.
Those keys which cannot be defined with a mnemonic but
only with the internal number can be made of use.

Another useful by-product of a list of the internal
numbers is a compressed initializer file (figure 2). Although
it is more difficult to read than the initializer file from the
previous issue, EDT processes it faster. The increase in speed
isn’t overwhelming, only ten to fifteen percent. But a sav­
ings can be made. There are those willing to accept a little
unreadability for a quicker editing session start.

3.0 INTERESTING SPECIAL FUNCTION DEFINITIONS

There are some very specialized things you can do with
defined keys. Here is a list of some that I have come across.

1. Macro block comment.

This command will ask some information and
create a comment block for a macro routine. It is in­
voked by typing “GOLD :”. You will then be asked
for the routine name and a short description. These
will be combined with a comment block that be in­
serted into the buffer.

Insert the following text into the initializer file in
the macro definition area.

```
+ MACRO BLOCK COMMENT
M_B_C

SBTTL ~ / ~ / ~
:+
: ~ / ~
: DESCRIPTION:
: ~ / ~
: CALLING SEQUENCE:
: CALL ~ / ~
```

2. Redefine <cr> to insert <sp>&<cr>.

Basic Plus Two programs require ampersands at
the end of each line. Everybody forgets to put them
on all the time. The cost for missing ampersands is
usually an extra program compile.

The following key definitions allow an “amper­
sand” mode. Typing “GOLD &” will cause EDT to in­
sert a space, ampersand, carriage-return for each
carriage-return typed. Typing “GOLD <cr>” will ex­
fit ampersand mode.

To add this command to EDT, insert the following
text into the initializer file at the key definition area.

```
DEF K GOLD & AS " EXT DEF K CONT MAS "I & I Z I M ."
```

3. Change lines for dial-up, VT100’s with AVO

“GOLD CONT L” toggles the screen between
twenty-two lines on the screen and twelve lines.
This command is very nice for use over 1200 baud
lines, and with VT100’s without AVO in 132 column
mode.

To add this command, insert the following lines
into the initializer file at the macro definition area.

```
+ SCREEN LINES MACROS

M_LINES_12
```

April 1982
The most complete line of Interface Products for PDP-11, LSI-11 and VAX Computers with features you can't get anywhere else

For DEC users...
MDB makes the difference!

You can have it all! Along with the benefits of your DEC computer, you can get interface products that significantly extend the capability, flexibility and economy of DEC systems. MDB makes more DEC system boards than any independent manufacturer—some with plain vanilla compatibility; others with plain incredible performance boosting features. For example—

- Line printer controllers for every major line printer, with complete self-testing capability and optional RS-422 "long lines".
- Peripheral device controllers for card readers, X-Y plotters, electrostatic printer/plotters and paper tape readers/punches.
- DZ11 compatible multiplexors for LSI-11 and PDP-11 users; some combine RS-232 and current loop or RS-422 with a single board.
- High speed synchronous serial interfaces; one DUP-11 compatible model has all bit and byte protocols even for LSI plus X.25 capability.
- Inter processor links between Unibus computers or between Unibus and Q-bus.
- System modules including DMA modules with RS-422, general purpose parallel and digital I/O interfaces, an IEEE 488 bus controller and a programmable real time clock.
- PROM memory modules, some with an on-board PROM programmer.
- General purpose interface and bus foundation modules.
- LSI-11 based subsystems and systems with capabilities like TU-58 cartridge storage and memory management.
- LSI-11 system boxes with 22-bit addressing and switching power supplies.

In addition to the DEC compatible products, MDB also manufactures comparable interfaces for Data General, Perkin-Elmer, Intel and IBM Series/1 computers. All MDB products are available under GSA contract #GS-00C-02851.

Give your DEC system all the benefits. Discover the difference MDB interface products can make.

*Trademark Digital Equipment Corp.

MDB SYSTEMS INC.
1995 N. Batavia Street
Orange, California 92665
714-998-6900
TWX: 910-539-1339

Circle 115 for referenced product. For complete information, circle 116 for PDP, 117 for LSI, 118 for DG, 119 for P-E, 120 for Intel, 121 for IBM.
4. Define keystrokes to insert words
A very handy timesaver for typists. A whole series of keys could be defined to insert words into the text buffer. I have here a simple definition for "CONT N" to insert the word "the". This is really a poor key to use for this purpose, but it demonstrates how easily this can be done.

Enter this line or something like it in the key definition area.
DEF K CONT N AS " EXT lthe I".

Some of these commands (like ampersand mode, word inserts) could be incorporated into the word delimiter macros. In other words, one could have some keystrokes defined to do useful things for programmers in programming mode, and other definitions for word processing mode.

4.0 HINTS
There are some things, when known, prevent waste and generally improve productivity.

1. Avoid journals, use "/RO"
If you are looking at a file exclusively in an inspection mode, use the "/RO" switch. This switch will prevent any accidental changes to files, and not clutter up accounts with journal files.

2. Understand the capabilities of "/RECOVER"
Even though I just said not to create journal files, don't ignore them altogether. The "/RECOVER" command is very useful to restore work obliterated by system crashes, etc. The journal file is automatically retained with the initializer "GOLD Q" quit command. Review the EDT manual and try recovering an edit or two. It's really fun to watch EDT work at warp speed reproducing your edit session.

3. Learn how to define keys
Whenever a key is defined, try to remember to enclose it in parens and terminate it with a period. This will allow the command to be executed with a repetition count. Even though this hint is in the EDT manual, most people don’t bother to read it.

4. Use "DEFINE KEY" for pseudo learn mode
Even though EDT doesn't have a "learn" mode, some things can be done with defining keys. For example, if you must transpose the sixth and seventh character of a number of lines, a key can be defined to do that. It's true that this isn't much, but it's the most we've got for now.

5. Use lots of buffers and pull in files.
Buffers are cheap, so there is no reason not to use plenty. Pull in as many files as you need. Borrowing old code is a real time-saver.

6. Install Steven Edwards' EDT patches.
Everything that I've seen Steve do has turned out to be quite useful. I recommend anything that he publishes.

5.0 KINKS
As with any new product, EDT does have a few problems. I have to admit that most of my complaints are about the screen window handler.

I don't know what the EDT development group has in store in the future, but I hope that they too have noticed these problems and are doing something about them.

1. "GOLD ." refreshes whole screen, not selected region.
Regardless of the region size, EDT will refresh the entire screen on this command. Needless to say, this is just a little wasteful. According to the manual the "GOLD ." command is supposed to cancel the selected region. It shouldn't be too hard to refresh only the inverted area on the screen.

2. Cursor flash on end-of-line operations
EDT has an annoying habit of temporarily moving the cursor to the beginning of the line on any operation on the end of the line. Try it at a low baud rate. Type some text into the buffer, and go to the end of the line. Delete some characters, one at a time. On each delete, EDT will remove the character, move the start of the line, and then move back to the end of the line.

3. Unnecessarily refreshes inside tabs
This problem shows up at low baud rates and can be best demonstrated this way. Get into EDT, type a character, a tab, and another character. Next, go back to the start of the line and type another character. Notice that EDT will refresh the entire line all the way to the end. In reality, only the line up to the tab requires a refresh. This is a simple but annoying problem.

4. "XON" determines terminal type
When EDT is instructed to operate in character mode (usually by the initializer file), it looks at the system terminal characteristics. After determining the that the terminal is a scope, EDT checks the value of "XON". EDT assumes the terminal is a VTS2 if set "NO XON". And, of course, if "XON" is set, the terminal must be a VT100.
CP/M® personal computing option for the VT100. Option owners can choose from a large library of compatible software. Exciting news!

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

The New VT101. Economical. Same high quality as the VT100.

The New VT125. Contains graphics package. Package can be retro-fitted.

The New VT131. Advanced video, printer port, block mode.

Digital chooses MTI for Authorized Terminal Distributor.

That's why MTI has all these new DEC terminals for you.

A lot of terminal dealers wanted to be DEC Authorized Terminal Distributors, but few were chosen. We're proud that MTI was one of the few. DEC knows MTI is a solid, growing company. We've been in business over 13 years. We've always been responsive to the latest developments in the industry and changing needs of our customers.

As an Authorized Terminal Distributor, MTI will always have the latest DEC terminals in stock, ready for delivery. And whether you are buying, renting or leasing, our prices are hard to beat. Plus, we have the expertise to match your needs to the right piece of equipment. We know what we're doing and we're proud DEC knows it.

MTI is your one source for all the terminals, peripherals, systems, applications expertise and service you'll ever need. See why DEC chose us for one of its few Authorized Terminal Distributors. Call us.

New York: 516/482-3500, 212/895-7177, 518/449-5959
Outside N.Y.S.: 800/645-8016
New Jersey: 201/227-5552
Ohio: 216/464-6688

mti

Applications Specialists & Distributors, New York, New Jersey and Ohio.

CIRCLE 107 ON READER CARD
The problem here is that EDT does not interrogate the terminal to determine its true type. Thus, unless your VT100’s are set with the “XON” characteristic, they are going to be used as VTS2’s. This also works the other way around. Therefore, it is VERY important to review those old “$TTY.CMD” files and make sure the your scopes are set to the proper types.

5. Make the window handler smarter
The routine that updates the screen (window handler) does wonderful things with a VT100 terminal. However, it could be made a little smarter.

6. Allow EDT to terminate commands with any keypad key

6.0 WISH LIST
Here is a list of some things that could be very useful for EDT. Mostly these are things that we had in VEDIT or KED, but didn’t make the transition to EDT.

1. Learn mode
Both VEDIT and KED had the ability to “learn” editing keystrokes for use in later editing. This would cut down on the need for defining keys, and eliminate confusion of having to remember all the EDT non-keypad commands.

2. Default file extensions
It would be nice to be able to define a default file extension either with a patch or a command in the initializer file.

3. View-All mode
A TECO-like “memory” of last file edited would be very useful. Many times I have been frustrated wondering where the spaces and tabs are.

4. Standard initializer file for entire system
Even though it’s a little wasteful, I liked the idea of looking on the user account first for an initializer, and if not found, using the standard one from the system library account.

5. Make the window handler smarter
The routine that updates the screen (window handler) does wonderful things with a VT100 terminal. However, it could be made a little smarter.

6. Allow EDT to terminate commands with any keypad key

RSTS/E System Managers!

Having trouble controlling usage of your computer system? Is there panic when passwords are changed? Have you ever bothered to change your passwords at all? System security is a serious problem. Announcing PASMAN, the intelligent solution.

PASMAN is a REACT account management substitute. Features include:

• Performs the common REACT functions account add and delete. Easier to use than REACT, and more "forgiving".
• New features never available before. Mass change of passwords for a single account, group, or the whole system! You supply the new password, or let PASMAN make one up.
• Identify users by a logical grouping, such as ACCOUNTING, SYSTEM, or WORD-PROCESSING. Plus, each account has its own description.
• Reports of what accounts belong to who. Prevent panic when passwords are changed. Identify user growth patterns, be able to plan ahead for upgrades.

In short, PASMAN will help you to get back control of your system.

PASMAN is available for only $350 (going up to $500 on June 1st!). For more information on PASMAN or our other products like DUMPIT, our file dump utility, or ENTRY, our callable data entry subroutine, please call or write:

INFINITY SOFTWARE CORPORATION
2210 Wilshire Blvd.
Suite 801
Santa Monica, California 90403
(213) 820-2702

CIRCLE 87 ON READER CARD

KED does this. Why not EDT?

7. Ability to edit search strings/comands
It's real nice to see what the last search string was before searching again. The same for commands.

8. TECO-like “memory” of last file edited
Some people liked this feature. others can do without it. Is it of use? I would at least like it to be present so I could patch it out if I didn’t like it.

9. Macro local symbol re-ordering
Once again,
BRAEGEN MPD HAS THE DISK STORAGE SUBSYSTEM YOU NEED

INTRODUCING THE DSL-SERIES OF DISK STORAGE SUBSYSTEMS

Braegen’s Minicomputer Peripherals Division is committed to providing the systems and peripherals DEC-users need.

BENEFITS

- Compatible with *DEC *PDP-11 and *VAX Massbus systems.
- Provides *RM03/RM05 Emulation with enhancements.
- Supports mixed drive sizes.
- Supports DEC Dual Port Diagnostics.
- Supports Overlap Seeks.
- Fast delivery.
- Local service and quantity discounts available.

For more information on the DSL-Series of mass storage subsystems or any of our full line of DEC-compatible computer systems, disk and tape subsystems, printers, memories, multiplexers and more, please call or write:

“Be sure to ask about our field service force in over 50 cities nationwide.”

*Trademarks of Digital Equipment Corporation

Marketing Dept:

BRAEGEN MPD
3320 East La Palma Avenue
Anaheim, California 92806
Telephone (714) 520-9200

CIRCLE 95 ON READER CARD
The following list shows the internal EDT key number, standard key number editing definition, and keystroke.

<table>
<thead>
<tr>
<th>EDT Key</th>
<th>Definition</th>
<th>Keystroke</th>
</tr>
</thead>
<tbody>
<tr>
<td>65535 D-</td>
<td>DELETE</td>
<td>0(0)</td>
</tr>
<tr>
<td>1</td>
<td>L</td>
<td>1(0)</td>
</tr>
<tr>
<td>2</td>
<td>W</td>
<td>2(0)</td>
</tr>
<tr>
<td>3</td>
<td>E</td>
<td>3(0)</td>
</tr>
<tr>
<td>4</td>
<td>C</td>
<td>4(0)</td>
</tr>
<tr>
<td>5</td>
<td>A</td>
<td>5(0)</td>
</tr>
<tr>
<td>6</td>
<td>V</td>
<td>6(0)</td>
</tr>
<tr>
<td>7</td>
<td>PASTE</td>
<td>7(0)</td>
</tr>
<tr>
<td>8</td>
<td>(DEL)</td>
<td>8(0)</td>
</tr>
<tr>
<td>9</td>
<td>APPEND</td>
<td>9(0)</td>
</tr>
<tr>
<td>10</td>
<td>HELP</td>
<td>PF+()</td>
</tr>
<tr>
<td>11</td>
<td>**</td>
<td>11(0)</td>
</tr>
<tr>
<td>12</td>
<td>-V</td>
<td>12(0)</td>
</tr>
<tr>
<td>13</td>
<td>-V</td>
<td>13(0)</td>
</tr>
<tr>
<td>14</td>
<td>C</td>
<td>14(0)</td>
</tr>
<tr>
<td>15</td>
<td>C</td>
<td>15(0)</td>
</tr>
<tr>
<td>16</td>
<td>SEL</td>
<td>16(0)</td>
</tr>
<tr>
<td>17</td>
<td>D+</td>
<td>PF+()</td>
</tr>
<tr>
<td>18</td>
<td>DEN</td>
<td>-()</td>
</tr>
<tr>
<td>19</td>
<td>D-C</td>
<td>-()</td>
</tr>
<tr>
<td>20</td>
<td>No definition</td>
<td>20(0)</td>
</tr>
<tr>
<td>21</td>
<td>1</td>
<td>M-Cl</td>
</tr>
<tr>
<td>22</td>
<td>GDICOB</td>
<td>GDICOB(0)</td>
</tr>
<tr>
<td>23</td>
<td>D+EL</td>
<td>D+EL(0)</td>
</tr>
<tr>
<td>24</td>
<td>ARC</td>
<td>ARC(0)</td>
</tr>
<tr>
<td>25</td>
<td>ER</td>
<td>ER(0)</td>
</tr>
<tr>
<td>26</td>
<td>BR</td>
<td>BR(0)</td>
</tr>
<tr>
<td>27</td>
<td>PSTE</td>
<td>PSTE(0)</td>
</tr>
<tr>
<td>28</td>
<td>EXT1/Commands.</td>
<td>EXT1/Commands.()</td>
</tr>
<tr>
<td>29</td>
<td>FILLER</td>
<td>FILLER(0)</td>
</tr>
<tr>
<td>30</td>
<td>CURSOR-DELETE PASTE</td>
<td>CURSOR-DELETE PASTE(0)</td>
</tr>
<tr>
<td>31</td>
<td>HELP</td>
<td>HELP()</td>
</tr>
<tr>
<td>32</td>
<td>*FSearch for: <em>i.</em></td>
<td>*FSearch for: <em>i.</em>()</td>
</tr>
<tr>
<td>33</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>34</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>35</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>36</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>37</td>
<td>SHL</td>
<td>SHL(0)</td>
</tr>
<tr>
<td>38</td>
<td>SHR</td>
<td>SHR(0)</td>
</tr>
<tr>
<td>39</td>
<td>RESET</td>
<td>RESET(0)</td>
</tr>
<tr>
<td>40</td>
<td>UNDF</td>
<td>UNDF(0)</td>
</tr>
<tr>
<td>41</td>
<td>UNDF</td>
<td>UNDF(0)</td>
</tr>
<tr>
<td>42</td>
<td>GOLDL</td>
<td>GOLDL(0)</td>
</tr>
<tr>
<td>43</td>
<td>CURSOR-DELETE PASTE.</td>
<td>CURSOR-DELETE PASTE.()</td>
</tr>
<tr>
<td>44</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>45</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>46</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>47</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>48</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>49</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>50</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>51</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>52</td>
<td>BL</td>
<td>BL(0)</td>
</tr>
<tr>
<td>53</td>
<td>TAB</td>
<td>TAB(0)</td>
</tr>
<tr>
<td>54</td>
<td>DBN</td>
<td>DBN(0)</td>
</tr>
<tr>
<td>55</td>
<td>DEP</td>
<td>DEP(0)</td>
</tr>
<tr>
<td>56</td>
<td>M</td>
<td>M(0)</td>
</tr>
<tr>
<td>57</td>
<td>N</td>
<td>N(0)</td>
</tr>
<tr>
<td>58</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>59</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>60</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>61</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>62</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>63</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>64</td>
<td>TADSR</td>
<td>TADSR(0)</td>
</tr>
<tr>
<td>65</td>
<td>DEL</td>
<td>DEL(0)</td>
</tr>
<tr>
<td>66</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>67</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>68</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>69</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>70</td>
<td>EX</td>
<td>EX(0)</td>
</tr>
<tr>
<td>71</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>72</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>73</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>74</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>75</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>76</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>77</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>78</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>79</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>80</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>81</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>82</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>83</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>84</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>85</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>86</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>87</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>88</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>89</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>90</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>91</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>92</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>93</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>94</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>95</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>96</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>97</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
<tr>
<td>98</td>
<td>No definition</td>
<td>No definition()</td>
</tr>
</tbody>
</table>

* This is a keypad key.
* This key can be defined by using the internal EDT number but cannot be used by either case conversion constraints or key sequence interpretations.
* This key is a valid for editing, but can be defined using the internal EDT key number only.

**FIGURE 1. Default EDT Key Assignments**
The following initializer file creates an editing environment identical to that made by the initializer file in the last issue. This initializer uses the internal key number instead of the mnemonic to define keystrokes. EDT will start a little faster when using a compressed file.

DEF M DELIM_PROG
F=DELIM_PROG
I
DEF K 75 AS "EXT DELIM_WP.
 standoff"
\end{verbatim}

My conclusion about EDT is: use it! It may be slightly flawed, but it's a lot faster than VTEDIT. It's a young product that will only become better.

At the Los Angeles DECUS meeting I had the chance to talk with the EDT people. They insured me that future releases of EDT would correct some of the problems I mentioned. Unfortunately, when we might see any future releases I couldn't find out.

I solicit any additional ideas, comments, and corrections. As space and volume permits, I will gladly share them with readers in future columns. Correspondence can be sent to:

INFINITY SOFTWARE CORPORATION
2210 Wilshire Blvd
Suite 801
Santa Monica, California 90403
(213) 820-2702
1.0 INTRODUCTION

While designing the installation procedure for our new A/P System, I discovered a serious lack of documentation regarding the BUILD program supplied with RSTS. This article is an attempt to correct this problem.

BUILD is designed to perform three basic functions:
- Read an input control file.
- Process the contents of that file.
- Produce an appropriate command file for ATPK execution.

The command file contains all of the commands necessary to build and/or patch a system. BUILD generates this command file by combining the responses to prompts with commands present in the control file. BUILD stores the responses to the prompts as values for various BUILD and user defined variables. I will refer to these variables as symbols or substitution symbols to avoid the confusion between the variables in the BUILD program and these special control file variables. As each symbol is encountered in the control file, it is replaced with the associated replacement value. The means of defining and identifying symbols will be discussed in more detail later.

2.0 BUILD COMMANDS

Seven commands are recognized by BUILD. Each command is prefixed with a dollar sign "$", and must be at least four characters long (including the dollar sign), with the exception of $BOOT which must be five characters long.

The BUILD commands are:
- $BOOT
- $BREAK
- $DOPAT
- $END
- $FORCE
- $PATCH
- $PROMPT

The first six commands are all used by BUILD for patching purposes. The last command, $PROMPT, is used for everything else. Due to the flexibility of the $PROMPT command, we will look at it before dealing with the patching commands.

3.0 SUBSTITUTION SYMBOLS

Substitution symbols in BUILD Control files are composed of:
- A tilde "~"
- Symbol name (1 to 6 characters long)
- A colon ":"
DILOG offers the widest range of single board DEC emulating disc and magnetic tape controllers for LSI-11, 11/2, 11/23, PDP-11 and VAX-11 compatibility.

DILOG/DEC controllers:

**DISC**
- Winchester—5¼", 8" and 14"
- SMD—Storage Module
- CMD—Cartridge Class
- Lark—Cartridge Class
- 2315/5440 Cartridge Class
- Flexible Disc

**MAGNETIC TAPE**
- ½" reel-to-reel NRZI/PE—Streaming or conventional
- ¼" 3M Cartridge

A wide range of emulations are offered. All fully supported by DEC operating systems.

If you're interfacing any popular drive to any DEC-11 computer, contact DILOG for the largest selection of DEC-11 compatible disc and magnetic tape controllers! Over 20 available now. Several more to be announced soon.

**CORPORATE HEADQUARTERS**
12800 Garden Grove Blvd. • Garden Grove, Calif. 92643
Phone: (714) 534-8950 • Telex: 681 399 DILOG GGVE

**EASTERN REGIONAL SALES OFFICE**
64-A White Street • Red Bank, New Jersey 07701
Phone: (201) 530-0044

**EUROPEAN SALES/SERVICE OFFICE**
12 Temple Square • Aylesbury, Buckinghamshire • England
Phone: 44-296-34319 or 34310 • Telex 837 038 DILOGI G

*DISTRIBUTED LOGIC CORP.*

*Trademark Digital Equipment Corp.*

CIRCLE 15 ON READER CARD
in the following manner:
1. All symbols are replaced by their respective replacement values.
2. Indirect command files will be opened and the next command retrieved.
3. BUILD commands which are the first thing on a line are processed.
4. The commands to the BASIC editor, OLD, APPEND, and COMPIL, will be processed to generate commands for the appropriate language processor, BASIC-PLUS, BASIC-PLUS-2, or CSPCOM.
5. Any other commands or text, will be placed as they are found in the Command file.

5.0 RUNNING BUILD
To run BUILD, log into a privileged account and enter:
RUN [1.2]BUILD
BUILD will print its header and prompt the user for five responses, along with two optional prompts.

(a) System Build < No > ?
This prompt determines what kind of action BUILD will take when processing the control file. If the response is any text starting with "Y", then BUILD will not issue the "Control file is ?" prompt. The input control file will be "BUILD.CMD", found on the input device in account [1,2], used with system generation.
If "NO" or < LF> is entered, then the control file will be specified by the user at the "Control File is ?" prompt.

(b) Source Input Device <SY: > ?
This prompt determines what device BUILD is to read the control file from. In addition this is where the input files for the installation procedure are copied from.
The format for this entry is:
LogFile = Device/Switch
The logfile specification is optional. If present it will be passed to ATPK for use as the logfile. The default logfile name is "SY:BLDnnn.LOG", where "nnn" is the current job number.
If "NO" or < LF> is entered, then the control file will be specified by the user at the "Control File is ?" prompt.

(c) Library Output Device <SY: > ?
This prompt determines on what device the library account resides. This device must be a disk. The default is "SY:"

(d) Target System Device <SYO: > ?
This prompt determines on what disk device the target system will be built. The default is "SYO:"

(e) Library Account <[1,2]> ?
This prompt determines on which account the library utilities (such as PIP, UTILITY, CSPCOM, TKB, etc.) are located. Any utilities installed by the BUILD procedure will be output to this account. BUILD will optionally create the account if it does not exist.

(f) Control File is ?
This prompt determines the main control file to be used by BUILD. This prompt is not issued if this is a system build (see prompt (a).) Unless otherwise specified, BUILD assumes this file is on the input device, in account [1,2], and has a file type of .CTL. There is no default for this filename.

(g) Additional Control File is < None > ?
This prompt determines if any additional control files are to be processed at this time. If you wish to perform several installations at one time and the control files are all on the same device, then specify the name of the next control file. Otherwise just press < CR> or < LF> to start the actual build.

6.0 PRE-DEFINED SUBSTITUTION SYMBOLS
The following substitution symbols are predefined by BUILD.
- INPDEV — Input device.
- INPUT — "INPDEV" + Input account.
- LIBDEV — Library output device.
- SYSDEV — Target system device.
- LIBACC — Library account.
- SYSTEM — "LIBDEV" + "LIBACC"
- RUNLIB — "SYSTEM" unless ATPK was not found there or we are doing a system build, in which case it is "SY:[1,2]"
- SYSACC — "LIBACC" unless BUILD was chained to by PBUILD, in which case "SYSACC" is defined by PBUILD.
- SYSDSK — "SYSDEV" + "SYSACC"
- MTMODE — .. /M0:2 " if the input device is magtape. (No rewind on a file search) Otherwise it will be null.
- LB — Current location of "LB:", set by the "$PROMPT LB" or the "$PROMPT ALB" commands.
- PATLOC — Location of the patch files, set by a chain from PBUILD or by the "$PROMPT PATCH" command.
- SAVDEV — Location of the saved patched sources, set by a chain from PBUILD or by the "$PROMPT PATCH" command. Defaults to "SYSTEM" if the specified location was not on disk.
- RTS.NM — RTS to compile programs against. This is
set by the "$PROMPT RTS" command.
- DEXT: — Default file type, set by the "$PROMPT RTS" command.
- CSPCOM — Hold result of “Use CUSP compiler” prompt (YES/NO). This is set by the "$PROMPT RTS" command.
- OLB — Object library name for compiles, set by the "$PROMPT RTS" command.

7.0 $PROMPT
The $PROMPT command will cause BUILD to print a prompt and receive input pertaining to that prompt. This command will also assign a replacement value to a symbol. Both of these functions are determined by the parameters used in the command.

The $PROMPT command has two basic formats. The first format is:

$PROMPT xxx
Where “xxx” is one of the pre-defined modifiers listed below:
- I — Comment entry internal to the Control file.
- ALB — Prompt for the location of LB:
- LB — Prompt for the location of LB:
- PATCH — Prompt for patching information
- RTS — Prompt for default RTS information

The other format is used for other special prompts, messages, and substitution symbol assignments. This format is:

$PROMPT String-1, String-2, Integer-1, String-3, String-4
Where:
- String-1 is the prompt to print
- String-2 is the default to print
- Integer-1 is a bit encoded flag word
- String-3 is the default file specification
- String-4 is the symbol

7.1 $PROMPT — Format 1
I will first discuss the five modifiers for the $PROMPT command, to see how they are used and what substitution symbols they define.

7.1.1 $PROMPT I — INTERNAL BUILD COMMENT.
This command is used for internal comment entries within the BUILD control files. Any line starting with the "$PROMPT I" command will be ignored. This command is used to make comment entries which will not be included in the ATPK command file.

7.1.2 $PROMPT ALB — PROMPT FOR THE LOCATION OF "LB:"
This command prompts for the location of the logical "LB:". The result of this command is shown below.

Locate logical 'LB:' on <SY:1.0>?

The default response to this prompt is the current location of "LB:" or "SY:1.1" if the logical was not defined. The response will be checked for validity. It must be a device and/or PPN specification only. If the specified account does not exist, BUILD will ask if you want to create it.

If the Library output disk and the Target System disk are both part of the public disk structure then the logical

RSTS/E SOFTWARE PACKAGES

- **KDSS**, a multi-terminal key-to-disk data entry system. (Also available for RSX-11M.)

- **TAM**, a multi-terminal screen-handling facility for transaction-processing applications. (Also available for RSX-11M.)

- **FSORT3**, a very fast sort. Directly sorts RSTS/E files containing up to 16 million keys or records. Up to 70 times as fast as the RSTS-11 Sort package in CPU time.

- **SELECT**, a convenient, very quick package for extracting records that meet user-specified selection criteria.

- **BSC/DV**, a device driver for the DEC DV11 synchronous multiplexer that handles most bisynchronous protocols.

- **COLINK**, a package that links two RSTS/E systems together using DMC11s. Supports file transfers, virtual terminals, and across-the-link task communication.

- **DIALUP**, a package that uses an asynchronous terminal line to link a local RSTS/E system to a remote computer system. Supports file transfers, virtual terminals, and dial-out through a DN11.

(The performance-critical portions of the first five packages are implemented in assembly language for efficiency.)

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

CIRCLE 29 ON READER CARD
DEAR RSTS MAN:

I have an 11/44 system which unfortunately came with dual TU58 drives (DECTAPE-II tape cartridges). I have yet to find a use for these little wastefuls of money except for running diagnostics as they are bootable. If I could put the RSTS initialization code on one, I could boot-up a TU58 (I even have the bootstrap ROM chip) to the Option: level to re-initial a pack, do a SAVRES, or whatever. They seem to hold about 500 blocks or so, they should be large enough.

My question is: How can I put a bootable copy of INIT.SYS on a DECTAPE-II cartridge? I've tried to HOOK.SAV it without success.

Econ. Oppor. Assn., Inc.
Brookville, PA

Dear Bret: As far as we know, there is no supported (or otherwise) way to do what you want. You can write on it with FIT, but that's the RT11 file structure, and HOOK doesn't know about it. If you could de-assemble the boot block...

DEAR RSTS MAN:

I just received my first VT131 and it doesn't work right. My Terminal Vendor told me that the VT131 was a VT100 with advanced video and a printer port that could do block mode but didn't have to. He reported that they had decided to carry these rather than the VT102 which doesn't do block mode. They thought that the very small price differential between the VT102 and VT131 wouldn't matter.

Why doesn't my software work properly with the VT131? VT Dear VT: DEC has done it again! As you know DEC VT terminals return a special escape sequence when queried by escape sequence from a program. The sequence can be used to tell terminals apart (i.e., VT52's, VT50's, VT100's, etc.). Although this terminal will respond to most commands like a VT100 with advanced video, the escape sequence it returns upon a query is different!

A VT100 returns ESC[?1n where n tells you about the terminal attributes (Advanced video, printer port). The VT131 returns ESC[?7C which is quite different from the VT100. Because the 1 is missing from the VT131 you will have to re-program your code to recognize the VT131. A shame indeed that DEC didn't use some compatible way to tell us it is a VT131. Worse, some of my code gives bad and unexpected errors when it encounters a VT131. ILLEGAL NUMBER AT LINE xxx. Maybe the VT102 is a better buy.

"LB:" will be removed and added with the new specification otherwise a local device assignment for "LB:" will be made in the ATPK command file.

Once the "$PROMPT ALB" command has been executed, additional occurrences of this command will be ignored.

This command will define the substitution symbol "LB" with the value of the new logical "LB:".

7.1.3 $PROMPT LB — PROMPT FOR THE LOCATION OF "LB:".

This command prompts for the location of "LB:". It works the same way as does the "$PROMPT ALB" with one exception, if either the Library output disk or the Target system disk is a private disk then this command will be ignored.

Once the "$PROMPT LB" command has been executed, additional occurrences of this command will be ignored.

7.1.4 $PROMPT PATCH — PROMPT FOR PATCHING INFORMATION.

This command is used to determine if patching is to be done during the installation, and if so where the patch files are located.

The "$PROMPT PATCH" command will cause the following prompts to be issued.

Function (Build/Patch, Patch, Build) < Build/Patch >?
Patch file input location < SY:[200,200] >?
Save patched sources <No >?
Write patched sources to < SY:[200,200] >?

7.1.4.1 Function (Build/Patch, Patch, Build)

< Build/Patch >?

If the user enters "BUILD", then the remainder of the prompts are skipped. During the processing of the rest of the BUILD control file, the $DOPAT command will be ignored, but the other patching commands will be executed as normal.

If the user enters "PATCH", then only patching will be done. During the processing of the rest of the BUILD control file everything but "$PROMPT", "$DOPAT", and "$BREAK" commands will be ignored.

If the default response is taken then both patching and the build will be done.

7.1.4.2 Patch file input location < SY:[200,200] >?

This prompt determines the location of the patch command files. The specified location must be a disk device (use [1.2]PATCPY to copy the files from tape to disk first). The default location is "SY:[200,200]".

7.1.4.3 Save patched sources < No >?

This prompt determines if you desire to save the patched BASIC-PLUS sources. The only sources which will be saved are the BASIC-PLUS sources which are patched with CPATCH using the $PATCH command.

7.1.4.4 Write patched sources to < SY:[200,200] >?

If you desire to save these sources then this prompt is issued to determine where the patched sources will be written to. The default response is "SY:[200,200]". This location does not have to be a disk device.
You can get more from your RSTS system with MAS-M.

MAS-M is the application software system from Martin Marietta Data Systems that can help you do more with your DEC hardware. That's because MAS-M is the on-line software system that gives you much more than you'd expect from packaged software.

More Flexibility.
MAS-M's modular design lets you choose from 10 different application systems:
- Accounts Receivable
- Accounts Payable
- General Ledger
- Order Processing
- Invoicing
- Inventory Control
- Inventory Accounting
- Bill of Materials
- Material Requirements Planning
- Purchasing

You can implement just the modules you need to satisfy your demands. And no matter which combination you choose, the MAS-M system is always fully integrated.

MAS-M's flexible design also makes it easy to install, and simple for your users to operate. And, since MAS-M is written in BASIC-PLUS-2, and based on the RMS-11 data management system, the software is fully compatible with your current RSTS/E operating system and DEC software.

More Control.
You can count on MAS-M for more comprehensive data accuracy and security, too.

MAS-M’s powerful transaction processing MONITOR gives you maximum control over your data—from start to finish. User passwords and menu selections are checked against user security profiles. Data entry validation is also standardized in the MAS-M MONITOR, so any invalid data can be corrected before it reaches your application program.

More Productivity.
MONITOR is also an important tool in developing new applications. You can use MONITOR to create input screens and validation rules on-line. And, MONITOR can help you improve programmer productivity by providing a standard framework for input of code that minimizes the difficulties of user interface and terminal characteristics.

More Support.
You can count on Martin Marietta Data Systems for system development and implementation, comprehensive training, and clear, concise documentation. We can also provide an extensive Maintenance Service to support your MAS-M system.

What it all adds up to is a packaged software system that can give you everything you need to get your jobs done. And more. Write or phone us today, and we'll tell you more about how the MAS-M software system can work for you.
The "$PROMPT PATCH" command defines the following substitution symbols:

- **PATLOC** — Location of the patch command files.
- **SAVDEV** — Location of the saved sources. If they are not being saved to a disk device then this becomes the same as the substitution symbol "SYSTEM".

This command will be ignored after being used once or when BUILD has been chained to from PBUILD.

### 7.1.5 $PROMPT RTS — PROMPT FOR RTS INFORMATION

This prompt determines under what run-time system ATPK is to start up the controlled job performing the build. The prompt from this command is:

```
Run-Time System < RSX > ?
```

The default response is the current system default run-time system. An error will be printed and the prompt issued again if any of the following are true:

- The specified run-time system was not found in "SY:[0,1]" with a file type of "RTS".
- The specified run-time system has not been added as a keyboard monitor.
- The default file type for executable programs under this run-time system is not " .TSK" or " .BAC".
- If the selected run-time system was "RSX" or "BP2COM" then a second prompt is issued to determine whether to use the CSPCOM compiler or not. This prompt is:

```
Use the CUSP compiler 'CSPCOM' <Yes> ?
```

If the selected run-time system was "RSX" then the default for this prompt is "YES", otherwise the default is "NO".

If the run-time system has not been installed, BUILD will ask if you want it installed. Respond with "YES" to have BUILD install it.

Upon completion of the $PROMPT RTS command, the following symbols are defined:

- **RTS.NM** — The specified run-time system name.
- **DEXT** — The default RTS file type (".TSK" or ".BAC")
- **CSPCOM** — "YES" or "NO" depending if CSPCOM will be used or not.
- **OLB** — Either "CSPCOM" if CSPCOM is being used, or the RTS name.

Once the "$PROMPT RTS" command has been executed additional occurrences of the command are ignored.

### 7.2 $PROMPT — Format 2

The second format of the $PROMPT command allows the assignment of user defined substitution symbols for use in processing in the remainder of the control file. I will discuss the various arguments for the second format of the $PROMPT command. Then at some common uses of this command. The format is:

```
$PROMPT String-1, String-2, Integer-1, String-3, String-4
```

Where:

- **String-1** is the prompt to print
- **String-2** is the default to print
- **Integer-1** is a bit encoded flag word
- **String-3** is the default file specification
- **String-4** is the symbol

#### 7.2.1 STRING-1 — THE PROMPT TO PRINT

This argument is a character string which is printed as the pseudo prompt. The characters are all left just as they were found in the control file. Leading and trailing spaces and tabs however are removed.

#### 7.2.2 STRING-2 — THE DEFAULT VALUE TO PRINT

This string is printed as the default response in the prompt. This string will be enclosed in angle brackets (" < ", " > "). It remains unaltered by BUILD and is only used as the printed default. No default is printed if this string is null.

#### 7.2.3 INTEGER-1 — BIT ENCODED FLAG WORD

This integer tells BUILD how to process the $PROMPT command. With the various bits of this word set and/or cleared, BUILD can force the input to be a file specification or part of one, a specific response, or just an informational prompt. If this argument is not a valid integer BUILD will abort.

The bit values are:

- **0 (1) — Lookup filename**
  - If this bit is set, then BUILD will verify that the file or account currently exists. Otherwise, the file or account will not be looked up. If the file was not found, BUILD will re-prompt for a correct filename.
- **1 (2) — Allow wildcards**
  - If this bit is set, then wildcards are allowed within the file specification (but not the PPN.) If this bit has been cleared, then no wildcards are allowed.
- **2 (4) — Allow/Disallow device specification**
  - If this bit is set, then a device name is allowed within the entered file specification. Otherwise no device specification is allowed.
- **3 (8) — Allow/Disallow PPN**
  - If this bit is set, then an account number is allowed within the file specification. Otherwise no account number is allowed.
- **4 (16) — Expand null Device/PPN to SY: and current account**
  - If this bit is set, then if the user does not enter a device specification, it will be expanded to SY: . If an account is not specified, then it will be expanded to include the current account.
- **5 (32) — Disallow/Allow filename**
  - If this bit is set, then a filename will not be permitted. Only a device and/or PPN will be allowed depending on the status of bits 2 and 3. Otherwise, a filename will be allowed.
- **6 (64) — Check input against values**
  - If this bit is set, then the input will be checked against the values following the $PROMPT command line. The values must be in the following format:

```

```
7.2.4 STRING-3 — DEFAULT FILE SPECIFICATION

This field is the default file specification. Any portion of an entered file specification which are missing will be taken from this specification.

7.2.5 STRING-4 — THE SUBSTITUTION SYMBOL

This is the definition for a symbol to be replaced by the value entered at the prompt. If the substitution symbol has already been defined by either a $PROMPT command or by BUILD, it will be replaced by the new definition.

7.3 Using $PROMPT

As you can see the $PROMPT command is very versatile with many uses. We will look at some of the uses for this command, concentrating on the second format. The various uses for the $PROMPT revolve around the bit values of the integer flag word.

(a) Print an informational prompt

Sometimes it is desirable to print a heading before the actual prompts start or to give additional information prior to issuing a prompt. To do this use a value of 4096 for the flag word.

$PROMPT "Software Techniques...", Replacement-Value-

This will cause the following message to be printed.

"Software Techniques"

As you will note, the default to print and the default file specification are missing. This should always be done to avoid the possibility of the command being processed incorrectly. All $PROMPT commands (format 2) must have a substitution symbol specified. The best thing to do is to use the same symbol (JUNQUE in the above example) for all of your informational prompts. If you use a symbol...
which has already been defined, the replacement value of that symbol will be discarded and a null string will take its place.

(b) Require a device name only

$PROMPT Enter a device name.SY:.37,SY:.DEVICE
This will appear as:

Enter a device name < SY: > ?

If you don’t want the default to be printed, then use a flag word of \(-32731 \ (1 + 4 + 32 + (-32768)).

(c) Enter an existing filename

$PROMPT Enter a filename.JUNQUE.IT.17.JUNQUE.IT.FILE
This will appear as:

Enter a filename < JUNQUE.IT > ?

The response to this prompt will cause BUILD to lookup the file entered or JUNQUE.IT if nothing was entered, in the current account on SY:. If the file was not found BUILD will print an error message and re-prompt you.

(d) Enter a specific response

Require the user to enter a specific response to the prompt such as YES or NO.

$PROMPT Really continue.No .8256 .. YES.NO
2
YES = YES
NO = NO
This will appear as:

Really continue < No > ?

If an invalid response is given, BUILD will display the correct responses and re-prompt as is shown below:

Valid options are:

YES
NO
Really continue < No > ?

(e) Enter a replacement file type

Use BUILD to do a replacement on a file type as input by the user.

$PROMPT Enter the file type.B2S.64 .. B2S.EXT
2
B2S = .B2S
BAS = .BAS
This command when executed will assign the file type which was entered, along with the preceding dot to the symbol “EXT”. The symbol “〜EXT” may then be used in the command file to cause the replacement of all occurrences of "〜EXT:" with the desired file type. For example:

Enter the file type < B2S > ? BAS

Then the command:

PIP JUNQUE.”EXT” = MTO:JUNQUE.”EXT:
Will appear as:

PIP JUNQUE.BAS = MTO:JUNQUE.BAS

(f) Logic manipulation in BUILD control files

Through the responses to $PROMPT pseudo prompts, substitution symbols can be used as logi­
cals with true/false values. The definition for logic
The actual abort is done with the following command:

$PROMPT "Aborting",YES,-8192..ABORT
The substitution symbol must be "ABORT" and the replacement value must be "YES" for an abort to take place. In this example we used a dummy prompt and forced the "YES" response.

8.0 PATCHING FROM BUILD

The following commands are used with the patching function of BUILD. Patching, however will not be done unless the "Patch" or "Build/patch" function was selected with the "$SPROMPT PATCH" command.

These commands are:
• $BOOT
• $BREAK
• $DOPAT
• $SEND
• $FORCE
• $PATCH

8.1 $BOOT n

This command will cause BUILD to be reloaded by ATPK after all of the commands up to the $BOOT have been processed. The $BOOT command will only be processed if the value "n" is greater than the current value of BT.LEV% which is defined at line 1000 of BUILD. Currently this value is zero, so $BOOT 1 will cause BUILD to be booted.

The following actions take place in response to the $BOOT command.
1. The input control file and the output command file are closed.
2. BUILD chains to ATPK.
3. ATPK processes the command file and chains back to BUILD.
4. BUILD will read through the control file ignoring everything except $DOPAT and $SPROMPT commands until the $BOOT command is reached. The "$SPROMPT PATCH" command is ignored after a boot.
5. BUILD processes the remainder of the control file normally.

8.2 $BREAK

This command is the terminator for a $DOPAT command control block. See section 8.3 for information on the $DOPAT command. If this command is found anywhere but the end of a $DOPAT command control block it is treated as a no-op.

8.3 $DOPAT @Filename

The $DOPAT command will cause the input control file to be changed to the patch command file specified by "Filename". The default file type on the patch command file is ".CMD". The contents of this patch command file are patch commands using the BUILD commands $SEND, $FORCE, $PATCH, and $SPROMPT. See the sections on these commands for more information on their use.

The $DOPAT command will only be used if the "$SPROMPT PATCH" command was processed and the user choose to "Patch" or "Build/patch". If the $DOPAT command is executed, all commands after it are ignored until a $BREAK command is found. If the $DOPAT is not executed then the commands following it are used and the $BREAK is treated as a no-op.

8.4 $SEND Filename

This command is used to terminate the $FORCE and $PATCH command control blocks. In addition to terminating these control blocks, when terminating the $PATCH command, it will cause the program just patched to be compiled. The "Filename" is the name of the new compiled program. The "Filename" defaults to the file specified with the $PATCH command. There is an optional switch for this command to specify that no compiling is to be done. This switch is "/NC".

8.5 $FORCE

The $FORCE command is used to place special patching commands in the ATPK command file. This command starts a control block which contains the special patching commands. The control block is terminated with either the $SEND or the $PATCH commands.

An example of the use of this command would be when a BASIC program is to be patched through the $PATCH command but requires several modules to be appended prior to patching. You would place the necessary append commands in this control block along with a command to save the program to a temporary file for patching.

The $FORCE can also be used to do patching with ONLPAT. All that would have to be done in this case is to place the commands necessary to invoke ONLPAT in the control block.

The $FORCE command causes some special processing to be done on the commands within the control block.
1. All normal BUILD substitutions are performed.
2. The BUILD commands $SEND, $PATCH, and $SPROMPT are properly executed.
3. The following special substitutions are performed:
   • I: is replaced by the input device and the input account (INPUT).
   • S: is replaced by the system device and the input account (SYSDSK).
   • L: is replaced by the library device and the input account (SYSTEM).
   • O: is replaced by the location of the patched sources (SAVDEV).
These special substitutions are only done when the substitution string is immediately preceded by one of the following:
   Double quote mark, single quote mark, left bracket, left parenthesis, semi-colon, comma, equal sign, space, or a horizontal tab.
4. The text with all substitutions made is placed in the ATPK command file.

8.6 $PATCH Filename

The $PATCH command is used to set up the commands for patching with CPATCH. The $PATCH command starts a command control block which is terminated with either the $END or the $FORCE commands. The filename specified is the name of the BASIC program to be patched. This filename will have a default file type of ".BAS".
We will look at the result of a sample SPATCH command control block. The run-time system is BASIC, the patch files are in SY:[200,201], and the sources will be saved in SY:[200,201]. The $PATCH command control block is:

```
$PATCH JUNQUE
JUNQUE.PAT
$END JUNQUE
```

The above commands will produce:

```
RUN SY:[1,2]PATCH
SY:[200,201]JUNQUE.BAS = SY:[1,2]JUNQUE.BAS
SY:[200,200]JUNQUE.PAT
12
12
SCALE 0
OLD SY:[200,201]JUNQUE.BAS
```

The filename specified in the $PATCH command must exist, if it does not then BUILD will abort with an error.

### 9.0 BUILD INDIRECT COMMAND FILES

BUILD will allow references to indirect command files. These indirect command files may be nested 15 deep.

To denote an indirect command file reference, use the commercial at sign (@) before the command file name. This command must be the only thing on the line. For example:

```
@JUNQUE.CMD
```

The default file type for the indirect command file is "CMD".

To place indirect command references for other programs or utilities such as PIP place an underscore (_) before the "@". ATPK will discard the underscore character if it is the first character on the line. For example:

```
RUN _@ APBLD1.CMD
@APBLD2.CMD
@APBLD3.CMD
12
```

This will cause the following to be sent to the PK by ATPK:

```
RUN SY:[1,2]PIP.SAV
@APBLD1.CMD
@APBLD2.CMD
@APBLD3.CMD
12
```

### 10.0 ATPK COMMAND RECOGNIZED BY BUILD

The only ATPK command which is recognized and processed by BUILD is the "$DETACH" command. When BUILD detects this command it sets up the ATPK command line with the "/DET" switch. This is the same as is done for the "/DETACH" command on the input device prompt. All additional occurrences of the command will be ignored. After detecting the "$DETACH" command, BUILD will not prompt for an additional control file when the end of the current control file is reached.

### 11.0 OLD, APPEND, and COMPILE commands

The commands to the BASIC editor, OLD, APPEND, and COMPILE will be treated differently depending on the responses to the "$SPROMPT RTS" prompt. The best way to describe the actions taken by BUILD when encountering these commands is to show the result of a simple command when each of the various options are selected.
1982 DECUS AUSTRALIA SYMPOSIUM

ONE DAY PRE-SYMPOSIUM SEMINAR

"HOW TO GET THE MOST OUT OF YOUR RSTS SYSTEM"

to be given by

Carl Marbach and Dave Mallery

of the RSTS Professional Magazine

VENUE: The Melbourne Hilton
DATE: Thursday 22nd July, 1982
TIME: 10:00 am to 7:00 pm
REG. FEE: $115.00 (including am/pm coffee & lunch)

Registration details: Contact Decus Australia, P.O. Box 384, Chatswood, NSW 2067, Australia, tel. 02.412.5252 or the Decus Secretary in New Zealand at P.O. Box 17-039, Greenlane, Auckland 5, tel. 591.289.

Registrations to be made before 15th June, 1982.
(c) Using the BP2COM RTS with CSPCOM.

RUN SY:(1.2)BP2COM.TSK
JUNQUE.OBJ/OBJ = JUNQUE.JUNK.APP
!Z

RUN SY:(1.2)TKB.TSK
JUNQUE.TSK/FP = JUNQUE.OBJ/SY:(1,1)BP2COM.OLB/LB
/
HISEG = BP2COM
UNITS = 12
ASG = SY:5:6:7:8:9:10:11:12
/
RUN SY:(1.2)PI.P.SAV
JUNQUE.OBJ/DE:NO
!Z

(d) Using the RSX RTS without CSPCOM.

SCALE 0
OLD JUNQUE
APPEND JUNK.APP
COMPILE JUNQUE.OBJ/CHA/UN/NODEB/OBJ
RUN SY:(1.2)TKB.TSK
JUNQUE.TSK/FP = JUNQUE.OBJ/SY:(1,1)RSX.OLB/LB
/
UNITS = 12
ASG = SY:5:6:7:8:9:10:11:12
/
RUN SY:(1.2)PI.P.SAV
JUNQUE.OBJ/DE:NO
!Z

If you will note, these commands will fail because the RSX emulator does not know how to deal with the commands SCALE, OLD, APPEND, and COMPILE. Therefore you should always specify CSPCOM when building under RSX or a similar run-time system.

(e) Using the RSX RTS with CSPCOM.

RUN SY:(1.2)CSPCOM.TSK
JUNQUE.OBJ/OBJ = JUNQUE.JUNK.APP
!Z

RUN SY:(1.2)TKB.TSK
JUNQUE.TSK/FP = JUNQUE.OBJ/SY:(1,1)CSPCOM.OLB/LB
/
UNITS = 12
ASG = SY:5:6:7:8:9:10:11:12
/
RUN SY:(1.2)PI.P.SAV
JUNQUE.OBJ/DE:NO
!Z

Amcor Computer Corporation is one of the oldest mini-computer software vendors in the world. We have over a decade of experience in developing software solutions for a multitude of industries.

The people of Amcor are committed to achieving their goal of software excellence. Responsiveness to the requirements of the industries served by Amcor's New Generation Software Products is one of the essential elements that assures our continued success as a leading producer of software. Amcor offers a full range of Accounting and Business Control application software products, as well as a comprehensive Data Base Management System. If you would like more information on how you can join our ever-growing list of satisfied customers, just clip the coupon below and drop it in the mail, or give us a call. We will be happy to send you free information immediately.

---

Amcor Computer Corporation is one of the oldest mini-computer software vendors in the world. We have over a decade of experience in developing software solutions for a multitude of industries.

The people of Amcor are committed to achieving their goal of software excellence. Responsiveness to the requirements of the industries served by Amcor's New Generation Software Products is one of the essential elements that assures our continued success as a leading producer of software. Amcor offers a full range of Accounting and Business Control application software products, as well as a comprehensive Data Base Management System. If you would like more information on how you can join our ever-growing list of satisfied customers, just clip the coupon below and drop it in the mail, or give us a call. We will be happy to send you free information immediately.

---

Amcor Computer Corporation is one of the oldest mini-computer software vendors in the world. We have over a decade of experience in developing software solutions for a multitude of industries.

The people of Amcor are committed to achieving their goal of software excellence. Responsiveness to the requirements of the industries served by Amcor's New Generation Software Products is one of the essential elements that assures our continued success as a leading producer of software. Amcor offers a full range of Accounting and Business Control application software products, as well as a comprehensive Data Base Management System. If you would like more information on how you can join our ever-growing list of satisfied customers, just clip the coupon below and drop it in the mail, or give us a call. We will be happy to send you free information immediately.

---

Amcor Computer Corporation is one of the oldest mini-computer software vendors in the world. We have over a decade of experience in developing software solutions for a multitude of industries.

The people of Amcor are committed to achieving their goal of software excellence. Responsiveness to the requirements of the industries served by Amcor's New Generation Software Products is one of the essential elements that assures our continued success as a leading producer of software. Amcor offers a full range of Accounting and Business Control application software products, as well as a comprehensive Data Base Management System. If you would like more information on how you can join our ever-growing list of satisfied customers, just clip the coupon below and drop it in the mail, or give us a call. We will be happy to send you free information immediately.

---

Amcor Computer Corporation is one of the oldest mini-computer software vendors in the world. We have over a decade of experience in developing software solutions for a multitude of industries.

The people of Amcor are committed to achieving their goal of software excellence. Responsiveness to the requirements of the industries served by Amcor's New Generation Software Products is one of the essential elements that assures our continued success as a leading producer of software. Amcor offers a full range of Accounting and Business Control application software products, as well as a comprehensive Data Base Management System. If you would like more information on how you can join our ever-growing list of satisfied customers, just clip the coupon below and drop it in the mail, or give us a call. We will be happy to send you free information immediately.

---

Amcor Computer Corporation is one of the oldest mini-computer software vendors in the world. We have over a decade of experience in developing software solutions for a multitude of industries.

The people of Amcor are committed to achieving their goal of software excellence. Responsiveness to the requirements of the industries served by Amcor's New Generation Software Products is one of the essential elements that assures our continued success as a leading producer of software. Amcor offers a full range of Accounting and Business Control application software products, as well as a comprehensive Data Base Management System. If you would like more information on how you can join our ever-growing list of satisfied customers, just clip the coupon below and drop it in the mail, or give us a call. We will be happy to send you free information immediately.

---

Amcor Computer Corporation is one of the oldest mini-computer software vendors in the world. We have over a decade of experience in developing software solutions for a multitude of industries.

The people of Amcor are committed to achieving their goal of software excellence. Responsiveness to the requirements of the industries served by Amcor's New Generation Software Products is one of the essential elements that assures our continued success as a leading producer of software. Amcor offers a full range of Accounting and Business Control application software products, as well as a comprehensive Data Base Management System. If you would like more information on how you can join our ever-growing list of satisfied customers, just clip the coupon below and drop it in the mail, or give us a call. We will be happy to send you free information immediately.

---

Amcor Computer Corporation is one of the oldest mini-computer software vendors in the world. We have over a decade of experience in developing software solutions for a multitude of industries.

The people of Amcor are committed to achieving their goal of software excellence. Responsiveness to the requirements of the industries served by Amcor's New Generation Software Products is one of the essential elements that assures our continued success as a leading producer of software. Amcor offers a full range of Accounting and Business Control application software products, as well as a comprehensive Data Base Management System. If you would like more information on how you can join our ever-growing list of satisfied customers, just clip the coupon below and drop it in the mail, or give us a call. We will be happy to send you free information immediately.
ENABLE COMPATIBILITY WITH NON-DEC PERIPHERALS

By Ken Fleming, Multi-List/McGraw-Hill

In August of 1981, we installed the Able ENABLE “Memory Expander” and one megabyte of Mostec memory on an 11/45 with System Industries’ RM05 look-alike drives (S.I. 9400 controller with CDC 9766 drives). We decided to take this approach because (a) we already owned the 11/45, and (b) we are in the process of switching to VAX 11/780’s, so we did not wish to buy another PDP 11/70. The 11/45 is a very fast machine, but is limited in memory. We reasoned that with enough memory the problem of job swapping could be reduced to acceptable proportions.

We, the steering committee par excellence, had sold management on the vast improvement in terminal response that users would see (due to less job swapping) when we expanded from 256K DEC memory to 1 megabyte Mostec. The morning after installing Enable, we were forced to report that everything went well, but because of an as yet undefined “Glitch”, we were still operating at our original 256K with the Enable installed!

Defining that “Glitch” became the challenge of the day — for too many days. The Enable device ran with RPO4’s on an RH11 controller with no problem. However, when we substituted the S.I. drives for the RPO4’s, we could not get past the memory map section of INIT.SYS (no message-system hung). Further investigation revealed that by not using the software patch that turns on the extended memory mapping, the Enable device worked fine with the S.I. drives.

System Industries’ only answer was that the problem must be in the “other” device. Able’s response was immediate. Les Wellington asked if he could come to our site and try to fix the problem for us. The next night Les, Joe Burdec, and Wayne Needer arrived armed with scopes, logic analyzers, revised boards, soldering guns, and spare parts galore. They worked all night with Bob Kelly (our in-house electronics wizard) and myself to try and fix the problem. Unfortunately we still had not defined the problem by morning.

The next day I called System Industries again, this time to request an S.I. 9400 controller for Able to test with their Enable device. The response from System Industries was far from adequate. Les Wellington was also pursuing getting an S.I. 9400 controller on a loan basis. Two weeks went by with no response from S.I.

Finally S.I. agreed to send their best technician (not an engineer) to our site to check things out. Up to this time the only person at S.I. who appeared the least bit interested in our problem was Dick Mann. When the technician could not define the problem, we were forced to start calling higher S.I. management in an attempt to get some action. Able was doing everything they could without the S.I. 9400 controller. In fact, Les had discovered that their device would work with various third party controllers. S.I. seemed to be the only problem.

Finally, after applying constant pressure on S.I., Les Wellington was invited to Sunnyvale to work on S.I. equipment at S.I.’s expense. This was an excellent idea and Les agreed at once. However, by now it had taken a month to get S.I. to escalate beyond a “Gee, that’s too bad” attitude.

As perseverance and curiosity are our long suit at Multi-List/McGraw-Hill, this author had finally prevailed and solved the impasse in the following manner.

The Enable device may be installed with up to four megabytes of memory, but it cannot address more than 256K bytes without a patch to INIT.SYS and the SIL. The Enable worked just fine on the 256K; but as soon as we patched INIT and the SIL, we could not bring up the system. This would immediately make one suspect the software patches. Joe Burdec assured me that it was not the patches, citing the fact that they were the same patches installed on every other system, and the only problems that they had encountered had been with S.I. equipment. This satisfied me for awhile, but I am responsible for (among other things) Sysgens, installation of new software, and patching.

One of the things that I had done recently was install a special INIT.SYS from S.I. to allow the CDC 9766’s to run as large RM03’s. This puzzled me — so I did some investigation. By comparing the INIT.SYS V7.0-07 and S.I.’s INIT, I discovered significant differences in DSK, ROOT, COPY, and BOOT. I talked to Dick Mann at S.I. and he assured me that there should be no conflict with the Able software patch because they should be different areas in the code.

By now, weeks had elapsed and I was more and more inclined to look toward software. I compared INIT V7.0-08, INIT V7.0-07, and S.I.’s INIT. The differences between INIT V7.0-08 and INIT V7.0-07 were insignificant. However, the differences between the two standard DEC INIT’s and S.I.’s INIT were numerous. Then I noticed S.I.’s INIT always asked for cluster size. Somewhere I had read about this being a bug in a very old INIT.SYS.

By now I was convinced that the problem was a conflict between Software Technique patches and S.I.’s patched INIT. So late one night I changed the S.I. drives from RM03 emulation to the RM05 emulation, mounted the new pack with DEC RM05 software with the Software Technique patches, and, lo and behold, everything worked. RSTS recognized all of our megabyte of memory. We have been running now for three months with no problems with the Enable device or S.I. drives.

When I inquired of the S.I. field tech the reason we were running in RM03 emulation, I was told it was because that was the way he was trained to do it. No one at S.I. could tell me the reason for this. The overall impression from dealing with S.I. was lack of field support training, both in software and hardware.

On a more positive note, since resolving this one major “Glitch”, we have had no problems with either the S.I. drives or the Enable.

For all you hardware types, the Enable fits in an SPC hex slot. All DMA devices should be in front of the Enable board and the memory goes behind. This means that the Enable will normally be the last device on the bus. One item of interest is that you don’t use a bus terminator with the Enable. Be sure you make this clear to your field tech service to avoid grief. Bob Kelly actually put a sign in the expansion box.

Provisions are made for you to piggy-back your present 18 bit address memory behind the Enable and 22 bit addressable memory; however, a separate SPC backplane is required. ABLE says you can go up to 4 meg, but we only have 1¼ megabyte; 1 meg of Mostek 8015 memory and ¼ meg of DEC MS 11-LD.

The S.I. interface also goes in one HEX slot; however, if you buy the 9400 controller instead of the 6100 single board, you will need some rack space. The most important benefit of the 9400 over the 6100 is the dual porting option which, with S.I. switch panel will allow up to four CPU to address up to 32 disk drives. The reader should take great care in deciding which CPU can write to which disk drive, since the disk map on disk and the disk map in memory won’t match on all the CPUs at the same time. This feature could be of great value to a shop for backup purposes.
BAC into RTS / BAC into MAC / BAC into BAS

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

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

RSTS/E, RTI, Macro-11 and Basic-Plus are trademarks of Digital Equipment Corporation.
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 timesharing 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
VERSION 2.1 NOW AVAILABLE:

QUE-11 — V2.1
ONE JOB SPOOLER
FOR RSTS/E CONTROLS
ALL SPOOLING

• SPLD1, SPLRUN . . . etc.
• BATD1, BATRUN . . . etc.
• QUEMAN, QUMRUN . . . etc.
• OPSER, OPSRUN . . . etc.
• ATPK

REPLACES:

One use of this feature is to make a program more readable — instead of
1% AND 53%
a programmer can write
1% AND (1% + 4% + 16% + 32%)
or 1% AND (1% OR 4% OR 16% OR 32%).

Lest you hope for too much from the compiler, I should point out that the SPACES, STRINGS, ASCII, LEN, LEFTS, RIGHTS, MIDS and SEGS functions with literal arguments are evaluated at run time.

A couple of further notes on strings:

In order to examine the first character of a string, it is slightly faster (at least, on an 11/70 with FPP — this could vary in a different environment) and occupies the same space to say

CHR$(ASCII(A1))

instead of

LEFT$(ASCII(A1)).

When comparing the first character of a string to a literal, it occupies less space (7 words instead of 8) to say

IF ASCII(A$) = ASCII(“A”) instead of

IF LEFT$(A, 1) = “A”.

It is still better (only 5 words of instructions, and 6 bytes less data due to the elimination of the string literal) to say

IF ASCII(A$) = 65%.

It may appear that use of these techniques could not effect significant results, but when you need another 5 miles per hour out of a program that appears to be at its speed limit, a few seemingly minor changes can add up to a noticeable improvement.

IT'S 3:15 PM
MONDAY

Tired of writing depreciation journals in 3,5;GL, your third assistant bookkeeper just discovered the joys of 4,0;PAY.

He’s on his way from the bank to the airport.

LOTS OF LUCK!

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

For more information contact:

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

One job controls all spooling
One job controls all spooling
新娘 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

IT’S 3:15 PM
MONDAY

Tired of writing depreciation journals in 3,5;GL, your third assistant bookkeeper just discovered the joys of 4,0;PAY.

He’s on his way from the bank to the airport.

LOTS OF LUCK!

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

For more information contact:

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

One job controls all spooling
One job controls all spooling
新娘 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

IT’S 3:15 PM
MONDAY

Tired of writing depreciation journals in 3,5;GL, your third assistant bookkeeper just discovered the joys of 4,0;PAY.

He’s on his way from the bank to the airport.

LOTS OF LUCK!

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

For more information contact:

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

One job controls all spooling
One job controls all spooling
新娘 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

IT’S 3:15 PM
MONDAY

Tired of writing depreciation journals in 3,5;GL, your third assistant bookkeeper just discovered the joys of 4,0;PAY.

He’s on his way from the bank to the airport.

LOTS OF LUCK!

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

For more information contact:

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

One job controls all spooling
One job controls all spooling
新娘 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

IT’S 3:15 PM
MONDAY

Tired of writing depreciation journals in 3,5;GL, your third assistant bookkeeper just discovered the joys of 4,0;PAY.

He’s on his way from the bank to the airport.

LOTS OF LUCK!

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

For more information contact:

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

One job controls all spooling
One job controls all spooling
新娘 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

IT’S 3:15 PM
MONDAY

Tired of writing depreciation journals in 3,5;GL, your third assistant bookkeeper just discovered the joys of 4,0;PAY.

He’s on his way from the bank to the airport.

LOTS OF LUCK!

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

For more information contact:
INSIDE:

☐ Learning VAX Macro for Fun & Profit

☐ Replacing RSTS SYS Calls with VAX/VMS System Services
LEARNING VAX MACRO FOR FUN & PROFIT

By Bob "MACRO MAN" Meyer

I had been doing some RSTS macro consulting for a small firm in New England (IE Systems, Newmarket, NH.). The project seemed to go quite well, and a few phone calls later I was asked if I'd like to get involved in a VAX project. 'A VAX project? Me?' I asked. 'Well, I'm willing to learn' I told them. That combined with a reduction in price landed my first VAX gig. It's been going on for almost a month now, with most of my time spent learning the ways of DCL, the assembler & linker, the instruction set, and monitor calls. The project so far has been rather interesting, for a guy that knows a fair amount of RSTS, and has done some Macro work under RSX-11M, so I thought I'd share some of the adventure of converting a RSTS Macro program over to VAX land. In the next few issues I'll touch on some of the basic I/O calls to VMS, later pointing out some of the more interesting ones.

Please remember, I'm not a VAX-man (yet). These articles are for the purpose of showing others how to do some simple things under VMS. Please forgive any errors found; I'll try to be as accurate as possible.

Of all the things that impressed me most, I must first stand and RAVE about the Help command. The help system is so elaborate, that in most cases where a question arose, about ANY area of the VAX, I could usually get some direction, if not the complete answer, by using the help command.

Well done, DEC.

Assembling and linking the small test programs I was using was quite fast unless you tried to use RMS. Small programs that assemble in around 13 seconds would jump to about 1:20 if you-know-who was called in... too bad.

The command file processor is also outrageous; it's an interpreter in itself, and lends a very helpful hand with a minimum of effort to learn the basics of it's use.

Next we'll talk about some of the simple I/O calls.

The basic I/O interface (at least from MY point of view) is VERY similar to that of RSX-11M. A channel is assigned to a device or file, and I/O requests are Queued to that channel. As in 11M, control can be returned to the user program as soon as the request is queued, and the program interrupted when it completes, making for some pretty clever programming if desired. However, being quite new at all this, I opted to take the more conventional route, and wait for my I/O to finish before doing anything else. The following directive can be used for most I/O needs:

$QIOW_S CHAN=TTCHAN,FUNC=#10S_WRITEVBLK,-
P1=BUF,P2=SIZ

(note that parameters to macros can be passed in any order)
Where:

$QIOW_S$ is the call 'Que I/O request & wait for completion'.

CHAN is the channel # to do the I/O to as returned by the $ASSIGN$ directive (that's next).

FUNC is the function we're interested in; in this case 'write virtual block'.

P1 parameter 1 is the address of the buffer to be written.

P2 is the length of the buffer.

This call will Que the I/O request, and return control to the user program when the I/O is complete.

Before this call can be executed successfully, the $ASSIGN$ directive must be issued to connect a path to the current terminal:

$ASSIGN_S$ DEVNAM = TT,CHAN = TTCHAN

Where TT is the text descriptor of the device to be opened (see example), and TTCHAN is a word which will hold the channel number returned by $ASSIGN$.

A few other things in the example which may need explaining are:

The .ASCIID directive puts the specified bytes in memory (like .ASCII in Macro-11), but precedes the data with a string descriptor consisting of the string length, some descriptor information, and a position-independent pointer to the string. This is required by the $ASSIGN$ call to access the user terminal.

The .ENTRY directive. This is an assembler directive which sets up the entry point for the program, and a mask to save specified registers on program startup.

The instruction:

BLBS r0, <label>

can be used to watch for errors after executing monitor calls. This is because VMS places a STATUS code in RO after each call. The instruction reads 'branch if low bit set (in RO) to <label>'. The branch will occur if the previous call completed successfully. If not, the RETurn instruction will be executed, bouncing control back to VMS, who, upon seeing a bad STATUS code, will print the respective error message on the user terminal.

If all is well and we're at the end of the program, the RETURN instruction will act as an EXIT directive, and just return control to VMS.

The sample program should work if keyed in. If time permits, I'll try to show some more calls next issue.

Bye for now.

.title bob
.ident /1.0 /
.tt: .asciid /TTBS:/ ;my terminal number
.ttchan: .word ;place for tt channel as returned by $ASSIGN$

msg: .asciid /Hello from VAX-lanol/ ;length of the message

.entry bob, ;entry point of program

ASSIGN_S devnam=tt,chan=ttchan ;assign a channel to the current terminal:

BLBS r0,105 ;assign worked

RETI ;assign failed; return to monitor

L0: $QIOW_S chan=ttchan,-

func=IOO_writevblk,-

P1=msg, P2=msglen ;write the message to the terminal:

RETI ;return to the monitor

.end bob

RSTS/E ON VAX
ROSS/V
(RSTS/E Operating System Simulator for VAX)

ROSS/V is a software package, written in VAX-11 MACRO, which provides a RSTS/E monitor environment for programs running in PDP-11 compatibility mode on DEC's VAX-11.

ROSS/V supports:

- The BASIC-PLUS interactive environment.
- Concurrent use of multiple run-time systems.
- Update mode (multi-user read/write access to shared files.)
- CCL (Concise Command Language) commands.
- An extensive subset of RSTS/E monitor calls.

ROSS/V runs under VMS and interfaces to programs and run-time systems at the RSTS/E monitor call level. ROSS/V makes it possible for DEC PDP-11 RSTS/E users to move many of their applications directly to the VAX with little or no modification and to continue program development on the VAX in the uniquely hospitable RSTS/E environment. Most BASIC-PLUS programs will run under an unmodified BASIC-PLUS run-time system.

ROSS/V is available from:

(Eastern U.S.)
Evans Griffihls & Hart, Inc.
55 Waltham Street
Lexington, Massachusetts 02173
(617) 861-0670

(Central U.S.)
Interactive Information Systems, Inc.
10 Knollcrest Drive
Cincinnati, Ohio 45237
(513) 761-0132
CIRCLE 67 ON READER CARD

(Western U.S.)
Online Data Processing, Inc.
N. 637 Hamilton
Spokane, Washington 99202
(509) 464-3400

RSTS, PDP-11, VAX-11, and DEC are trademarks of Digital Equipment Corporation.
REPLACING RSTS SYS CALLS WITH VAX/VMS SYSTEM SERVICES

A Few VMS Conversion Notes

By Bob Stanley, Computer Methods Corporation

INTRODUCTION

"So, you’re thinking of converting from RSTS to a VAX? Well, I’ve heard the VAX is a nice machine; big, powerful, fast. But what about all of those RSTS dependent features that I’ve heard the VAX can’t emulate?";

"How about things like direct CRT cursor addressing? Or echo control mode? Or programmable wildcard directory lookups? The VAX just can’t handle those types of business application features that RSTS performs so well."

Does that conversation sound familiar? Have those types of questions and concerns turned you off to the VAX? Well, to the surprise and delight of many, there are solutions to these problems. This article takes a first hand look at how to make your brand new 32-bit supermini look just like RSTS. By the way, the rumor that the original title of this article was "Turning Unbearable Pain Into Extra Income" is just not true!

The conversion factors described below are from an actual RSTS to VAX conversion done for a client of Computer Methods Corporation that is currently running a 50 job RSTS system that tracks and manipulates export orders. As is typical of most installations, many programs were written that take advantage of RSTS dependent features and are, therefore, not easily convertible. Several external functions were written and placed in an object library that provided the programmers with substitute methods of performing these RSTS dependent functions. The basic building blocks for all of the functions that I will be discussing are the VMS system services.

SYSTEM SERVICES

System services are the VAX version of RSTS sys calls. While sys calls are cryptic, unwieldy, difficult to understand and even more difficult to use, system services are all of this and more! Actually, system services are more straightforward and easier to use because they follow the standard VMS calling procedures. They are invoked similar to a user defined function (E% = SYS$ASSIGN), they take a list of parameters, and they return a status code as their value.

VMS maintains a very long list of internal integer status codes that can be referenced within a program via the EXTERNAL INTEGER CONSTANT statement. These codes range from VAX BASIC error codes (BAS$__CANFINIL meaning can’t find file or account) to RMS status codes (RMS$__FNF meaning file not found) to system service status codes (SS$__NOPRIV meaning insufficient privilege). Any system service return status can be tested against these status codes (IF E% = SS$__NOPRIV in the above example) to test for expected errors or a normal successful status (SS$__NORMAL).

A program that is going to call a system service must first declare the system service and any external constants (status codes) via the EXTERNAL statement. Example 1 is an example program that calls the system service SYS$BRDCST which broadcasts a message to a specified terminal. This and all of the other system services are described in detail in the SYSTEM SERVICES REFERENCE MANUAL.

Example 1

ECHO CONTROL

VMS does not handle opening a terminal in mode 8 (echo control mode). This mode is used to define specific fields (with specific lengths) that should be input from and displayed at specific positions on the terminal screen.

A typical application of this type would be the need to perform a data entry function via a predefined input screen format or to display control information while allowing an operator to move about the screen and enter selected fields of data.

While VMS does not perform echo control mode in the same fashion as RSTS, it does allow a program to do direct QIO’s to any physical device including the keyboard. A special form of a QIO called ‘read with prompt’ enables a program to effectively perform controlled field input.

TERMINAL QIO’S

The first step in performing QIO’s to any device is to assign that device to a specific channel (this is different from opening a file on a channel). This is done via the system service SYSSASSIGN. Example 2 shows an external integer function that accepts a keyboard specification (TT on the VAX rather than KB:) and returns both an assigned channel number and a terminal type (VTS2, VT100, etc.). A user supplied external function TERM__TYPE is called to provide the terminal type (this uses the system service SYS$GETDEV).

Once a channel has been assigned to the keyboard, the system service SYSSQIOW can be used to perform I/O to the terminal. A QIOW is an I/O with a wait for the device to respond. Several different functions can be performed via
SY$QIOW. The one we are interested in is read with
prompt. This is specified by passing IO$_READPROMPT
(another external constant) to the system service.

*****************************************************************************
10 FUNCTION INTEGER TERM_ASSIGN (TERMINAL.ID$, &
   TERMINAL.CHANNEL$, &
   TERMINAL.TYPE$) &
20 EXTERNAL INTEGER FUNCTION SYS$ASSIGN, &
   TERM_TYPE,
   SY$ASSIGN (TERMINAL.ID$,TERMINAL.CHANNEL$) &
   TERM_TYPE (TERMINAL.ID$,TERMINAL.TYPE$) &
32767 TERM_ASSIGN = EA &
    FUNCTIONEND &
*****************************************************************************

Example 3 shows an example of a QIOW using the
IO$_READPROMPT function. The argument list allows you
to specify the channel number assigned to the terminal, the
function to be performed, a string to receive the data read,
the length of that string, a string that determines which
characters should be terminators (we'll talk about that in a
minute), a prompt string, and the length of that prompt
string. Other parameters are allowed and can be found in
the SYSTEM SERVICES REFERENCE MANUAL and the I/O
USERS GUIDE.

*****************************************************************************
10 | SAMPLE QIOW WITH READPROMPT &
   | EXTERNAL INTEGER FUNCTION SY$QIOW &
   | EXTERNAL INTEGER CONSTANT SS$NORMAL, &
   | IO$READPROMPT &
   | READ_FUNCTION = IO$READPROMPT &
   | INPUT.BUF.LEN = 50% &
   | READ.PROMPT$ = 'Enter field - ' &
   | PROMPT.LEN = LEN(READ.PROMPT$) &
20 EA = SY$QIOW (TERM.CHANNEL$, | FROM SY$ASSIGN &
   | READ_FUNCTION$ BY VALUE,,,,, &
   | INPUT.FIELDS BY REF, &
   | INPUT.BUF.LEN$ BY VALUE,, &
   | TERMINATOR.MASK BY REF, &
   | READ.PROMPT$ BY REF, &
   | PROMPT.LEN$ BY VALUE) &
   | PRINT 'ERROR IN SYSTEM SERVICE' &
   | IF EA < SS$NORMAL &
32767 END &
*****************************************************************************

Example 3

By passing these parameters to the system service, you
can define a field of a specific length to be input from the
terminal. The field will be returned to the program either
when a terminator is typed or when the field is full. By
combining this with a function to position the cursor at aspecific
location (by printing the proper escape sequences just as under
RSTS), controlled, esthetic data entry can be performed.

TERMINATORS

A program using the SY$QIOW system service can specify
its own set of terminator characters. This is done by
turning on any of the low order 32 bits of a 64 bit quad-
word. Each of the bits 0—31 represent the ascii characters
0—31 (bit 3 is control C, 26 is control Z, etc.). Thus any of
the ascii characters 0—31 can be specified as a terminator
by turning on the appropriate bit.
The easiest way to do this is to start with a 32 bit
longword set to zero and "OR" it with the proper power of 2
to turn that bit on. Thus, if A% is a long word with a value
of zero, A% = A% OR 2**26 would cause a control Z to
become a terminator. A% = A% OR 2**1% FOR I% = 0% TO 31% would cause all of the ascii characters 0—31 to
become terminators.

As mentioned above, the low order 32 bits of a 64 bit
quadword need to be set to determine the terminators. The
simplest way to set up the quadword would be to define a
map as follows:

MAP (TERM_MASK) LONG TERMINATOR.MASK. LOW. ORDER. BITS
This defines two successive long words (a quadword). The
variable LOW. ORDER. BITS can then be used as A% was in
the above examples to set up the terminators.

The example program at the end of this article shows
an external function that can be called to execute controlled
terminal I/O. It was originally designed to facilitate the con-
version of data entry screen formats but can be used by any
application that needs to control the input of data by an
operator.

WILDCARDS

A second function that does not lend itself easily to
VMS is in-program wildcard directory lookups. Several con-
version applications needed to send individual messages to a
receiving program containing the names of each file in a
specified directory. RSTS handled this problem via the
wildcard directory sys call. VMS has no simple system ser-
vice that will return file names given a wildcard specifi-
cation. In fact, the VMS documentation's only reference to this
function is in the back of the RMS REFERENCE GUIDE
(chapter 13).

RMS does provide two system services that, with some
considerable effort, perform wildcard lookups. (This RMS-32
facility, unlike its RSTS counterpart, allows wildcard
characters in the directory specification as well as the file
name.) In order to do this, however, one needs to under-
stand and manipulate internal RMS file information struc-
tures; namely the FAB and NAM blocks.

FAB AND NAM BLOCKS

The FAB is an internal block of data that describes a
particular file. The fields of the FAB contain information
about the file such as the name of the file, the file's
organization, its record format, space allocation, etc. The
RMS REFERENCE GUIDE describes the FAB and gives a list
of all the fields contained in the FAB. A map or common
area can then be set up to define the fields of a FAB in your
program.

The first word of caution, which is very important if
you attempt to use the FAB block, is that the table in the
manual that describes each field in the FAB is in alphabetical
order. If you set up a map with the fields in the order listed
in the table, your program will provide some interesting but
highly inaccurate results. The second word of caution is that
a field, right in the middle of the FAB block, is not listed in
the table!

This little bit of information was discovered by expand-
ing the FAB MACRO definition and looking at the offsets
(listed in the table) and field lengths. Example 4 shows how
to obtain the MACRO expansion listing.
The expanded MACRO listing contains information about the internal variables used by the MACRO. The FAB and NAM block offsets and the lengths of each of the fields appears on the first two pages. If reading an expanded MACRO listing is not your cup of tea, the examples below show how to incorporate the FAB and NAM blocks into your program.

Example 5 offers a Vax Basic callable function which performs wildcard directory lookups. It shows a map for the FAB block and the NAM block (described below) that have the fields in the proper order and with the proper lengths.

The NAM block contains supplementary information about a file such as device and directory information, expanded file name strings and wild card character context. Again, the manual does not provide enough information to accurately set up a NAM block map. Example 5 contains the complete NAM block layout.

**SYSSPARSE**

The first step in performing a wildcard directory lookup is the SYSSPARSE system service (described only in the RMS REFERENCE GUIDE). This service takes information provided in the FAB block, parses it, and allocates fields in the NAM block to store the wild card character context for subsequent searches. This service need only be called once in the case of iterative directory lookups.

In order to use SYSSPARSE, certain fields in the FAB and NAM blocks must be initialized. The external function in Example 5 performs the SYSSPARSE system service in the function FNSET.UP%. The FAB$C_BID and NAM$C_BID external integers must be provided to identify the FAB and NAM blocks. Also, the length of the FAB and NAM blocks must be placed in the FAB.BLN and NAM.BLN fields. The external constants FAB$C_BLN and NAM$C_BLN can be used for this purpose.

The remainder of the fields that need to be initialized can be found in the RMS REFERENCE GUIDE, chapter 13, pages 13-4 and 13-5.

**SYSSSEARCH**

Once the wildcard specification has been parsed, the directory specified in the NAM block can be searched via the SYSSSEARCH system service. SYSSSEARCH will return one file name at a time and can be called iteratively until the status code RMS$_NORMAL is returned. The service maintains its own internal wildcard count (in NAM.WCC) so that it never gets lost in the middle of the directory.

The sample external function in Example 5 shows the
This function provides any program written in a VAX native mode language with capabilities similar to RSTS cursor and echo control modes. It allows the program to specify cursor positioning, prompting text, field length, etc. and return to the program the operator entered string. Its original purpose was to provide input capabilities similar to the Dupont sponsored VAX data entry package. It can, however, be used by any program that needs to retrieve data from the operator in a controlled, aesthetic fashion.

The function parameters are:

- PROMPT.TEXTS: Any prompting string that should appear with the input field
- PAINT.CHARACTERS: The character that will define the data position prior to data entry
- DEFAULT.ANSWERS: A string that should be returned in place of a blank field
- PROMPT.COLUMN: Starting screen column for the prompting text
- PROMPT.LINE: Starting screen line for the prompting text
- INPUT.COLUMN: Starting screen column for the input area
- INPUT.LINE: Starting screen line for the input area
- FIELD.LENGTH: Predefined length of the input field. Either a terminator or a full field will end data entry for this field
- TERMINAL.CHANNEL: The keyboard ("TT") from which data entry can be expected.
- TERMINAL.TYPES: The type of terminal to be used (VT52, FT1, etc)
- INPUT.DATAS: The operator entered data being returned to the calling program.

FUNCTION INTEGER PHO_KEYIN PROMPT.TEXTS, 
EXTERNAL INTEGER FUNCTION SYSSOIL, 
EXTERNAL INTEGER INPUT.BUFF.LEN, 
EXTERNAL STRING INPUT.FIELD$ 
EXTERNAL FUNCTION IOS_READPROMPT OR IOSH_TRMNOECHO 
EXTERNAL STRING INPUT.COLUMNS, INPUT.LINE, INPUT.FIELDS, STRING$ (FIELD.LENGTH) 
EXTERNAL CTLZS, CHR$(126), INSTR(U, INPUT.FIELDS, ESC$) 
EXTERNAL INPUT.FIELDS': INPUT.FIELD$ + ' ' 
EXTERNAL PROMPT.POSS + READ.PROMPTS 
EXTERNAL INPUT.LIN$, INPUT.POS$ 
EXTERNAL INSTR(U, INPUT.FIELDS, ESC$) 
EXTERNAL CTLCS, SPACE$, STRING$ (LENGTH) 
EXTERNAL PROMPT.LINE, PROMPT.COLUMN, INPUT.COLUMN, INPUT.LINE, TERMINAL.CHANNEL, TERMINAL.TYPES, DEFAULT.ANSWERS, PROMPT.TEXTS, PAINT.CHARACTERS, CLEAR.SCREEN, KEYPUNCH.MODE, INPUT.DATAS 
EXTERNAL INPUT.FIELDS: INPUT.FIELD$, INPUT.COLUMN, INPUT.LINE, TERMINAL.TYPES, DEFAULT.ANSWERS, PROMPT.TEXTS, PAINT.CHARACTERS, CLEAR.SCREEN, KEYPUNCH.MODE, INPUT.DATAS
THE BASICS OF NETWORKING AND DIGITAL COMMUNICATION FOR THE SYSTEM MANAGER

By Michael H. Koplitz

Digital communication is used in all aspects of computing, from the asynchronous terminal to synchronous communication between CPUs. Networking involves the use of digital communication between several devices and CPUs. The objective of this article is to acquaint the RSTS/E System Manager to the methods and terminology of digital communication.

BASIC ELEMENTS OF COMMUNICATION

1. Message — a sequence of characters used to convey information or data.
2. Transmission — the act of sending a message between the sender and receiver.
3. Sender (transmitter) — a device which has a message to communicate.
4. Receiver — a device capable of receiving or accepting a message.
5. Medium (of transmission) — the way of getting the message from the sender to the receiver.
6. Noise — anything that interferes with the process of communication.
7. Efficiency — effective use of the communication channel.

TYPES OF TRANSMISSION

Parallel transmission — the medium of parallel transmission consists of one wire for each bit in a character plus an additional wire for a clock or strobe signal. The clock or strobe tells the receiver to read the character which is on the other wires. This type of transmission is good for high speed data transmission.

Serial transmission — the medium of serial transmission consists of a pair of wires, one wire to transmit data and one wire to act as a common signal ground. Bits are transmitted serially, one after the other. Most serial transmissions can be sent over telephone lines by using a modem. A modem is a device which converts a binary (digital) signal into an analog signal by modulation at the transmitter's end. The modem at the receiver's end demodulates the analog signal into a binary signal.

20 mA transmission — a technique used to transmit binary data along serial lines. This method transmits the binary data by turning a 20 mA (milli-amp) current on and off. The flow of current indicates a "1" bit and a "0" bit is indicated by stoping the flow of current. 20 mA transmissions can not use modems.

EIA transmission — a second technique used to transmit binary data along serial lines. This method transmits data by reversing the polarity of the voltage on a dc serial line. A positive voltage on the line communicates a "0" bit and a negative voltage communicates a "1" bit.

Voltage varying systems are more susceptible to noise. The EIA system is based on standards prepared by the Electronics Industry Association and includes the definition of modem control signals. Most modems manufactured in the United States are compatible with the EIA standard RS-232C.

CCITT transmission — a third technique used to transmit binary data along serial lines. CCITT is a voltage varying system based on standards prepared by the International Consultative Committee on Telephony under the auspices of the United Nations.

MODES OF TRANSMISSION

Simplex — communication can only occur in one direction on the wire pair.

Half-duplex (HDX) — communication can occur in either direction on the wire pair but only in one direction at a time.

Full-Duplex (FDX) — communication can occur in either direction on the wire pair at the same time.

ASYNCRONOUS SERIAL TRANSMISSION

In asynchronous serial transmission, the sender transmits a character whenever a character is ready to be transmitted. Sometimes this type of transmission is called "Start/Stop" transmission. This is because a start bit is transmitted first, then the character, followed by a stop bit(s).

A line is said to be idle when no characters are being communicated. As soon as the receiver senses the start bit, the receiver starts a clock which measures bit times. The receiver then samples the next eight bits and places them into a register for transfer to memory. The next bit(s) is the stop bit, which must be a "1" bit. A stream of stop bits will indicate that the line is idle. Whenever a "0" (start) bit comes down the line the receiver would then start the clock.

This is not a very efficient way to communicate because at least two out of every ten bits serve as start and stop bits, which do not communicate data.

SYNCHRONOUS SERIAL TRANSMISSION

In synchronous communication and entire block of characters is sent at a time. Special synchronous characters are sent before and after each block to coordinate or synchronize both the sender and the receiver. There is not any need for start and stop bits since the entire block of characters is synchronized. Therefore the synchronous technique uses the line more efficiently than the asynchronous serial transmission.

SYNCHRONOUS PROTOCOL

Every protocol has the following functions: controlling data transfers, error checking and recovery, information coding, information transparency, line utilization, syn-
chronization, communications facility transparency, and bootstrapping.

Controlling data transfers — there are three elements involved, formatting, control information and "handshaking" procedures. Control data and error checking information are contained in one block:

```
header or text field
control field
trailer or error checking field
```

The header contains addressing, block sequencing, control flags and acknowledgement information. The addressing is used for determining the destination of the data. Block sequencing ensures that no transmission is lost or duplicated. The control flags are used to indicate whether the transmission is data, control-only, first, intermediate or last block of the message. Control messages are used to determine who transmitted and who received the data. It is also used for the receiver to acknowledge the receipt of data and whether it is a good or bad transmission. This procedure of acknowledgement is referred to as handshaking.

Error checking and recovery — assures correct reception of data. Check bits are transmitted with the message which are used in verifying that the data transmitted is correct. The check bits are commonly called block check characters (BCC) which make up the trailer field of the transmission block.

Methods to check errors:

Vertical Redundancy Checking (VRC) — parity checking is done on each character as the data is received. The parity can be even or odd.

Longitudinal Redundancy Checking (LRC) — uses exclusive OR logic to check the entire block for errors after the block is received. After each transmission the receiving station normally replies with a positive (ACK) acknowledgement or with a negative (NAK) acknowledgement.

Cyclic Redundancy Checking (CRC) — also checks the block after the entire block has been received. This method uses polynomial division of the data stream by a CRC polynomial to check for errors. This is a very complex method to check for errors.

Information Coding

ASCII (American Standard Code for Information Interchange) — this code was introduced by the USA Standards Institute. It is a seven bit-plus-parity code. There are several codes in the scheme which have been set aside for communication control. The parity can be either even or odd.

Data Interchange Code — is a variation of the ASCII code, some of the printing characters of the ASCII code have been replaced by non-printing control characters. The parity must be odd.

Other types of code are Extended Binary Coded Decimal Interchange Code (EBCDIC), Baudot Code (a five bit code used on the old teleprinters), Four of Eight Code, IBM Punch Card Hollerith Code, Binary Coded Decimal Code (BCD) and the six bit Transcode.
Software Product Description

Product Name: LOCK-11 Version 2.2

Description:
Lock-11 is a security superstructure built upon the standard RSTS password structure that provides the following extensions:

- Absolute control of system access by keyboard. Manager may limit any keyboard to certain accounts or groups of accounts and control time as well as day of week access.
- Password knowledge is no longer carte blanche system access. System detects unauthorized use of passwords. Privileged passwords don't work on non-privileged keyboards. Non-privileged passwords work only on specified keyboards.
- Real time system surveillance. Manager specifies a list of alarm keyboards which log all infractions and probes as they happen. Oper is not required.
- Auto-login (with or without password) and chain with specified core common contents by KB.
- Manager may establish special priority/burst settings by KB. Manager may establish default output protection code, assignment and up to three specific user logicals for each KB. Default RTS is also selectable. All assignments are made at log-in.
- Manager specifies a list of console keyboards from which security file editor may operate.
- Manager may define a KB-specific access-denied message.
- Manager may specify number of retries before access-denied and number of access-denied messages before line disable. Hangup on access denied is optional. All above may be specified on a per-kb basis.
- A macro DYNPRI program is included which performs the following functions:
  - Users may be dispatched into ten separate priority queues, separately tunable on-line. Each queue has ten levels. Queues are selectable by KB.
  - Program detects hibernating jobs and announces the fact on ALARM keyboards. Privileged jobs hibernating cause extra loud and long alarms.
  - The program produces almost no load in operation and runs in 5K words.
  - Program will hold up to fourteen files open for performance purposes.

Minimum Hardware/Software Required:
Any valid RSTS/E system running Version 7.0 or later. Any version of RSX emulation is needed.

Support: See License Agreement
Installation: User Installed
Ordering Information:
Available on 9 track 800 or 1600 BPI tape. Multiple CPU discount schedule:
- First license
- Second thru Third license
- Fourth thru Twentieth license
Licensed users desiring source code for internal use only must execute a separate Program Sources License Agreement. Sources are available at ten times the initial license fee.

License Fee:
Single CPU license: $950.00. Annual maintenance at 12% of current list price.

Contact:
Dave Mallery
Nationwide Data Dialog
215—364-2800

Information Transparency — It is important that a protocol be able to send binary data, floating point numbers, packed data, and machine language computer programs in the same format. Different methods are used to accomplish this and each different protocol does it in a different way.

Line Utilization — is the attempt to make protocol utilize the communication channel to its fullest.

Synchronization — when transmitting synchronous data the sender and receiver must be synchronized for proper reception of the transmission. To get the receiver in phase a unique group of bits called a synchronization sequence precedes the transmission. The synchronization sequence should be such that the data stream can not reproduce it.

Communication facility transparency — the idea is to have any protocol run on any facility. This would be ideal but as of yet has not been done.

Bootstrapping — ability of down line loading a computer.
WHY SYSTEMS has developed the "big machine" software package for Executives, Accountants and Professionals who need their numbers fast and accurate.

DIGICALC™, designed exclusively for DEC equipment, with an amazing range of applications and one of the finest built-in automatic training procedures on the market. Prepare everything from budgets to multi-year forecasts, change one value and DIGICALC will re-compute all of your results and give you a hardcopy too. The tabular worksheet display on the wide screen can be as simple or complex as you wish and has the capability to call on numerous mathematical functions. For an amazing display of DIGICALC's ease of operation and virtually limitless capabilities call WHY SYSTEMS. Runs on RSTS/E, VMS, and Tops-10/20.

CALL TODAY (206) 881-2331 FOR FREE BROCHURE AND DIAL-UP DEMONSTRATION
A WORD ABOUT THE AUTHOR...

Rudy Bazelmans is a Software Analyst at Sykes Datatronics Inc., where he designs and codes Language Processors.

THE ULTIMATE PUSH/PULL MACROS

By Rudy Bazelmans, Sykes Datatronics, Inc.

ABSTRACT

In Assembly Language Programming it is very common to utilize the stack for temporarily storing groups of variables. This paper presents a set of macros for easily manipulating the stack on a PDP-11. Some of the richness and power of the MACRO-11 assembly language is also demonstrated.

INTRODUCTION

When manipulating the stack in Macro-11 there are a number of inconveniences:

1. The instruction to push and pull items from the stack is awkward to write and a nuisance to remember.
   MOV VALUE,-(SP) :PUSH
   MOV (SP)+ ,VALUE :PULL

2. Only one item may be placed on the stack in each line of source code.

3. If you push a byte onto the stack you must remember to pull a byte off, otherwise you will pull a word off and you may unintentionally change a memory location.

4. After you have pushed values on the stack, you must remember to pull them back off in the reverse order.

5. Before exiting a subroutine you must remember which items are still on the stack so you can take them off.

An approach to solving these problems is through the use of macros. To my knowledge, macros have been used to solve items 1, 2, and 3 above. I am not aware of an existing solution to items 4 and 5.

The following is a group of macros which I have written to solve all five items most notably items 4 and 5. The explanation of how these macros work is broken into two parts. The first part will center around the concept of solving a subset of the problems mentioned above. The second part will describe the complete solution, which includes more features and error checking than the first part.

For those of you who are interested in using a set of macros with the above properties and are not concerned about the details of how they work, you can simply use the macros in figures 3B and 4B. All the information required to use these macros is included in figures 3A and 4A.

THE CONCEPT

The easiest way to simulate the action of a stack is through the use of another stack. That is my basic approach to solving these problems.

The first set of macros is shown in Figure 1A and 1B. You should take a moment and read the description included with them. These macros (along with the examples in figure 2A) are quite limited, but they do implement the basic idea of assembly time stacks.

There is a stack pointer in these macros called PSHCT$ which begins at zero and keeps a count of the number of items PUSHed on the stack. Remember, stacks are LIFOs, the Last item In is the First item Out. The initialization of this counter is in the user's program at line 2 of figure 2A. The counter is incremented whenever a new item is placed on the stack (line 32 of figure 1A). The counter is decremented again when the macro for that item is expanded (line 46 of figure 1A).

In order to place an item on the stack, you must first call the PUSH macro. Each argument in the group of arguments to PUSH is isolated one at a time (line 30 of figure 1A). Each argument is then moved onto the stack (line 31) and PSHCT$ is incremented to show that another value has been placed on the stack (line 32).

Lines 33-38 is where the items PUSHed are remembered for the PULL macro. PSHFL$ is used to indicate if the current argument is the first argument to the PUSH macro. If it is the first argument, PSHFL$ = 0 (line 29 of figure 1A). If it is not the first argument, PSHFL$ = 1. The setting of PSHFL$ is important to the PUSH$ macro and its significance will be discussed below.

There are three parameters passed to the PUSH$ macro: the name of the current argument being pushed on the stack, the ASCII equivalent of PSHCT$, and (if the current argument is not the first argument to PUSH) the ASCII equivalent of PSHCT$-1.

The PUSH$ macro (lines 42-50) defines a macro (lines 46-48) of the name PShname$ where name is the current value of PSHCT$. The macro definition consists of three lines. The first line restores the value of the argument from the stack (line 45). PSHCT$ is decremented in the second line in order to indicate a change in the nesting level. In the third line, a check is made to see if the argument which was passed to PUSH$ is the first argument to the PUSH macro. If it is the first argument, then we have restored all the arguments in the group. Remember that when we restore the values from the stack we have to do it in the reverse order of the way we stored them on the stack. If the current argument to PUSH$ is not the first argument to the PUSH macro, then we should call the macro that is necessary to restore the next argument of the group (line 47).

At this point, we are only defining a macro to restore the arguments from the stack, we are not actually restoring them. The actual restoration will occur when the PULL macro calls the macro which we just defined. If the user
calls the PUSH macro again this entire process will be repeated.

The next step is the retrieval of the data which was placed on the stack. The user calls the macro PULL which will pull the last group of values from the stack. This is extremely simple. First the current value is converted to its ASCII equivalent (line 28 of figure 18) in name and the macro PSHname$ is called (line 29). Remember that PSHname$ has the value of the last argument placed on the stack.

A sample execution with the expanded code is shown in figure 2A. Figure 2B shows the state of the symbol table at line 5 of the sample execution.

The manipulation of words and bytes is also supported in this improved set of macros. The default argument size is one word but if a byte is to be PUSHed, an apostrophe (') should be placed before the byte argument in the parameter list of the PUSH or PULL (line 3 of figure 5). The code to do this checking is in lines 9 and 30 of figure 3B and in line 13 of figure 4B. The unusual construct in lines 10 and 31 of figure 3B and line 14 of figure 4B is executed whenever the

FIGURE 1A. Macros can be used to simulate assembly time stacks. This macro pushes items onto the stack.

THE ULTIMATE PUSH/PULL MACROS

The second group of macros (figures 3 and 4) show the PUSH and PULL macros in their final state. This version is more flexible than the first version and does more error checking. Read figures 3A and 4A for a complete description of how these macros should be used. Some of the differences between the first and second set of macros are described below:

There is more error checking being done in this new version. Line 2 of figure 3B shows a check to see that the user specified at least one argument in the call to the PUSH macro. Lines 5-7 show a check for a stack overflow. The limit is 63, this should be more than adequate. If you are PUSHing more than 63 levels deep on the stack, you're doing something wrong. Lines 3 and 4 of figure 4B show a check for an empty stack. If you want to PULL a value and you have never PUSHed anything on the stack, you should be specifying a destination on the PULL.

FIGURE 1B. This simple macro pulls items off the assembly time stack.

FIGURE 2A. This shows a sample execution of the simplified macros.

FIGURE 2B. The state of the macros and variables at line 5 of figure 2A are shown above.
user specifies a byte argument. The byte arguments have a preceding apostrophe which must be removed before the instruction is generated. This is accomplished by using the universal unary operator (') with an apostrophe appended to the end of the argument. As a result, the argument (less the leading apostrophe) is passed to the .IRP loop.

The final macros contain several .MEXIT statements lines 16 and 45 of figure 3B and line 20 of figure 4B. These exist because the .IRP loop which surrounds them only needs to be executed once to check for a leading apostrophe on each argument.

Figure 5 shows an enhanced sample execution for the final version of the PUSH/PULL macros.

![FIGURE 3B. Here is the definition of the PUSH macro.](image)

![FIGURE 4A. This is the documentation for the PULL macro.](image)
MACRO PULL ARGS

IF B <ARGS>
  IF EQ PSHCTS
    ERROR "CAN'T PULL FROM AN EMPTY STACK"
    END
  ENDC
ENDIF

IRP NAME \PUSHCTS
PUSH NAME'
ENDD

ENDC

IRP ARG <ARCS>
IRP CHAR <ARC>
IF ION <', 'CHAR>
  MOVV VAR, -(SP)
ENDIF

HOVVAR, -(SP)
ENDD

HEX IT
ENDD

HEX IT
ENDD

ENDM PULL

FIGURE 4B
This is the definition of the PULL macro.

FIGURE 4B. This is the definition of the PULL macro.

.SBTTL EXAMPLES
.PUSH <ABC, 'DEF, 'H3>
  MOV BC, -(SP)
  MOV (SP), B

.PUSH <JHL, 'RS3>
  MOV JHL, -(SP)
  MOV (SP), JHL

.PULL RH
  MOV (SP), RH
  MOV (SP)+, RH

FIGURE 5. This is a full example of the PUSH and PULL macros in action.

COMMENTS
I have optimized the macros in figures 3 and 4 as much as possible in an effort to decrease memory requirements for frequent users, increase speed and decrease complexity.

There are three important things to be gained from this paper.

1. The constructs used in these macros are complicated, but by understanding each of them, you should be able to design much more powerful macros.

2. The overall concept of using macros and variables to create assembly time stacks is useful in many applications, especially when writing structured macros.

3. You can use the macros in figures 3 and 4 in your own shop. The use of these macros is fully documented and can be used as is. It should save your programmers time and help reduce errors.

REFERENCES
POLYSOFT APPLICATION LANGUAGE

By S. Zuk (Non-DECUS Member), Polyfibron Division, W. R. Grace and Company, Lexington, Massachusetts
Presented to DECUS Fall 1981 Conference — Los Angeles

1.0 INTRODUCTION
The Polysoft Application Language (PAL) was designed as an interface language between the polysoft data bases [1] and various business applications. PAL was developed as a user oriented language which allows development applications software without having an in-depth understanding of hardware and system software. A comprehensive report program [2] (Report Manager) was created to complement PAL.

2.0 LANGUAGE STRUCTURE
Each line of PAL source code consists of a label or line number, a function command and a series of parameters (variable slots or data base element numbers) required to execute a line of code.

PAL source code must be run through a compiler like Basic+2 program before the application program can access data base. The PAL compiler will examine the source code for such items as correct function commands, data base element verification, missing loop logic and missing or incorrect parameter declaration. The compiled object code is stored in a virtual array format and is executed through the polysoft data base management system. Upon completion of the compiler an error listing is generated. Correction of errors and resubmission to the compiler must be done before the program is executed.

The PAL application programs functions are designed to handle record I/O with all the features and techniques of BASIC+2 without the need for dealing with files at the bit and byte level. The end product is an executable program allowing for on-line interactive data base manipulation or batch processing.

3.0 LANGUAGE COMMANDS
The PAL commands fall into three main categories:
3.0.1 Commands designed to handle record I/O.
3.0.2 Commands designed to handle applications required logic to manipulate the records and elements within the data base.
3.0.3 Special feature commands

3.1 Record I/O Functions
The I/O function commands allow the user to add, change, delete and inquire on data base files, records or elements.

File Close [CLEAR]
Allows for selective closing of files over and above the data base managers dynamic housekeeping routines.

The code would appear as follows:
A. LABELXX,CLEAR | R750  !  
OR
B. LABELYY,CLEAR |  !  !
A. Will close a specific file [R750]
B. Will close all files open at that time

Find Record [FR]
This command will allow the user to select a record from a specific file using the keys specifications for the file. It will request from the system a screen containing promptable keys for searching. Once the command is executed it will save the record number of the record requested to be used by other functions.

The code would appear as follows:
LABEL11,FR | R750,SLOT1 | SCR:SCREEN.TXT | LABEL99 FIND HEADER
LABEL11 — Line Number
FR — Command
R750 — File to Search
SLOT1 — Storage Area for Record Number Found
SCR:SCREEN.TXT — User Defined Interface Screen—See Below
LABEL99 — Step to Go to Incase of Abort
FIND HEADER — Comment

Highest Record [HR]
This allows the user to find the last and highest record number issued by the system software. This can be used for controlled record access.

The code would appear as follows:
LABELXX,HR | R750,SLOT2 | IGET REC NUM
HR — Command
R750 — File to Access
SLOT2 — Storage Area for Highest Record Number

Input File [IF]
The [IF] command will retrieve from the data base a logical group of elements from a record. Retrieval is based on the record number within a specific file. This record number is totally transparent to the user.

The code would appear as follows:
LABELCC,IF | R750,SLOT3,C700 | IABORTEE | 
IF — Command
R750 — File to Access
SLOT3 — Holding Area Containing Record Number to Access
C700 — Group within the Record
IABORTEE — Line to Go To in Case of Incorrect Group Selection
NOTE: Associated with this file/group is an overlay screen that is used to display to the user record data.

Move Data [MOVE]
This instruction is like a data transfer statement in any other language. It allows the user to move elements from one file to another or move specific values to files or to other parts of the program without altering data.
The code would appear as follows:

A. LABEL11,MOVE FS002.1.R750,SLOT10,FS002.1.R850,SLOT20
B. LABEL22,MOVE I.SLOTS...,SLOT6
C. LABEL33,MOVE FS002.1.R750,SLOT10...,SLOT6
D. LABEL44,MOVE I.SLOTS,F5002.1.R850,SLOT20

A. This type of code will transfer the information contained in element [FS002.1.R750] with the record number [SLOT10] and place it in element [F5002.1.R850] with a record number index [SLOT20].

B. The slot to slot movement is passing data from [SLOTS] into [SLOTS].

C. Moves a value from data base field to a storage area [SLOTS].

D. Moves a value from a storage area [SLOTS] to a data base field.

Operation Mode [OM]

The function [OM] will allow the user program to handle data maintenance requirements as needed. It allows the user to add, delete, reinstate and inquire on records or groups within the record. Changes can be made at the record, group and field levels.

The code would appear as follows:

LABEL 11,OM
LABEL 12,OM
LABEL 13,OM
LABEL 14,OM
LABEL 15,OM
LABEL 16,OM
LABEL 17,OM FS002.1.R750 I.CHANGE FIELD

The above statements indicate the function called [OM] and the operation to be performed i.e., ADD.

Record Issue [RA]

The record issue function allows the user to get the next available record number for the file being operated upon.

The code would appear as follows:

LABEL22,RA ISLOT25,R750
RA — Command
SLOT25 — Storage Area of Record Number for Later Referal
R750 — The File to Get the Next Record Number

Record Search [SL]

This command gives the application user the ability to search other data base files within the system using information that is supplied by either data from other files or from user response.

The code would appear as follows:

LABELV.SL ISLOT2,FS002.1.R750,SLOT20,FR7505 I.T
SL — Command
FS002.1.R750 — Key Field to Search
SLOT20 — Value User to Search FS002.1.R750
FR7505 — Second Key to Search
T — Constant Value used to Search Second Key

Save Record Number [SR]

Gives the user the ability to save record numbers for future reference within the application program. This instruction usually follows the [FR] command.

The code would appear as follows:

LABELQQ.SR ISLOT200
SR — Command
SLOT200 — Storage Area for a Record Number

3.2 Logic Functions

Arithmetic Operations [A]

Allows a user to perform arithmetic operations on paired storage areas, fields and/or constant values. It allows for the four basic math functions plus the concatenation of string data.

The code would appear as follows:

LABELTT.A I.SLOT2,SL033.SLOT4.2.5
A — Command
+ — Type of Operation to be performed [also .-. .-.]
SLOT2 — Value 1
SLOT3 — Value 2
SLOT 4 — Sum of Value 1 and Value 2
2 — Number of Decimals to be Retained in the Result
5 — Number of Leading Zeros to be Inserted in the Result

IF/THEN Type Compares [C]

This command is like the IF/THEN conditional of statement used in Basic. It allows for the testing of conditions within the application to determine alternate courses of action based on parameters of the test.

The code would appear as follows:

LABELPP.C I.SLOT1 INORETAANXX.4
C — Command
= — Type of Comparison to be done
SLOT1 — Value to Compare
NO — Constant Value
LABELXX — Branch Destination if Compare is True
5 — Indicates Comparison is to be done on a String of Data

Extraction Functions [EXTRACT]

Extract function performs the same type of operations as the instruct, left and right functions in Basic. See attached documentation for more detail.

GOTO [GO]

This command allows the user to branch to specific parts of the application program unconditionally as in Basic.
The code would appear as follows:

```
LABEL23,GO  I  LABEL99  I
GO — Command
LABEL99 — Branch to Destination
```

Input Prompts [IS]

This function will suspend the application and require the user to input a response that may be used to redirect the program flow or supply a variable to be used as a search value within the current application. This prompt will appear at the bottom of the screen and is independent of any overlay screens.

The code would appear as follows:

```
LABEL77,IS  I  SLT30,10,1  I  ENTER Y or N
IS — Command
SLT30 — Storage Area for User Input
10 — Ten Second Response Timer
1 — Maximum Number Characters to be Input
ENTER Y or N — Prompt Message to Appear on Screen Before Entry
LABEL77 — Return Step
```

4.0 TESTING AND DEBUGGING

The testing and debugging of a PAL generated program uses the same techniques that one would use in debugging any other language. For example, a debugging display feature is an integral part of the language.

5.0 SUMMARY

Users who are familiar with basic programming conventions, and who have a sound knowledge of the particular data base file structure can adapt to the language quickly and easily. As in the design and programming of business type applications a good understanding of the business environment is also essential to the systems analyst. This form of language can translate design into functional applications in a short period of time.

Attached are PAL functions that make up the language, along with a copy of a designed application interfacing with the polysoft data base using several of the language features.

6.0 BIBLIOGRAPHY


DEC PDP-11 & LSI-11 Users

Until now, adding a magnetic tape subsystem to Digital Equipment's PDP-11 and LSI-11 series computers meant costly or custom-built interfaces and integrating components from a multiplicity of vendors. It didn't always work.

Now, with IDT's series 1050 magnetic tape and series 3000 cartridge tape subsystems for DEC computers, adding a cost-effective, high-performance data storage and retrieval system to PDP-11s and LSI-11s is a simple, uncompromising procedure.

Innovative Data Technology's tape subsystems are designed with compatibility in mind. Our 1/2" tape drives use industry-standard interfaces and require only one CPU slot for the controller card; or, if your card cage is full, IDT will put your components from a multiplicity of vendors. It didn't always work.

Innovative Data Technology's tape subsystems are designed with compatibility in mind. Our 1/2" tape drives use industry-standard interfaces and require only one CPU slot for the controller card; or, if your card cage is full, IDT will put your components from a multiplicity of vendors. It didn't always work.

Innovative Data Technology's tape subsystems are designed with compatibility in mind. Our 1/2" tape drives use industry-standard interfaces and require only one CPU slot for the controller card; or, if your card cage is full, IDT will put your components from a multiplicity of vendors. It didn't always work.

Innovative Data Technology's tape subsystems are designed with compatibility in mind. Our 1/2" tape drives use industry-standard interfaces and require only one CPU slot for the controller card; or, if your card cage is full, IDT will put your components from a multiplicity of vendors. It didn't always work.

Innovative Data Technology's tape subsystems are designed with compatibility in mind. Our 1/2" tape drives use industry-standard interfaces and require only one CPU slot for the controller card; or, if your card cage is full, IDT will put your components from a multiplicity of vendors. It didn't always work.
full length, e.g. SY/S is passed to $SYSSTAT and SYSTAT/S. Some programs, such as $SWITCH and CALLER itself, make no allowance for abbreviations.

3. There is no equivalent of the PRIV specifica-

There are no equivalent of the PRIV specifica-

The running of an extra program is entered. The running of an extra program is executed program are run each time a CALL com-

Thank you for a fascinating and very useful publi-

I follow the "MACRO MA" articles through your columns how he has managed to achieve this.

Thank you for a fascinating and very useful pub-

Lastly (and best-ly, for us RSX types), the SEDMSG output string editor and .TPARS command line passes are both present (in LBSYSLIB.

As a matter of 

Keep up the good work!

Hugh J.E. Davies, B.Sc.
Hertfordshire, England

Thanks Hugh, and keep up your good work, also.

I just finished reading the Dec. 1981 issue and was interested in the 'CALLER.BAS' article. Congratulations are due to Stephen Fahey for coming up with his solution to free up small buffers.

For a lot of systems, this method would be more than suitable. However, systems with a large number of CCL's and a high amount of activity may find some disadvantages.

1. Two programs (CALLER) plus the normally executed program are run each time a CALL command is entered. The running of an extra program will create some additional overhead in starting the program, disk access and swapping (unless free memory is available).

2. Abbreviated commands (i.e. SY-.STAT, QU-

THESE PRICES • IN STOCK IMMEDIATE DELIVERY

CALL SONJA OR LAURIE AT: (614) 889-0810

SCHERER'S MINI COMPUTER MART

6145 Dolan Place

Dublin, Ohio 43017

BRAND NEW • FULL 90 DAY WARRANTY

CIRCLE 106 ON READER CARD

LETTERS to the RSTS Pro... continued from page 6

Hugh J.E. Davies, B.Sc.
Hertfordshire, England

Thanks Hugh, and keep up your good work, also.

I just finished reading the Dec. 1981 issue and was interested in the 'CALLER.BAS' article. Congratulations are due to Stephen Fahey for coming up with his solution to free up small buffers.

For a lot of systems, this method would be more than suitable. However, systems with a large number of CCL's and a high amount of activity may find some disadvantages.

1. Two programs (CALLER) plus the normally executed program are run each time a CALL command is entered. The running of an extra program will create some additional overhead in starting the program, disk access and swapping (unless free memory is available).

2. Abbreviated commands (i.e. SY-.STAT, QU-

THESE PRICES • IN STOCK IMMEDIATE DELIVERY

CALL SONJA OR LAURIE AT: (614) 889-0810

SCHERER'S MINI COMPUTER MART

6145 Dolan Place

Dublin, Ohio 43017

BRAND NEW • FULL 90 DAY WARRANTY

CIRCLE 106 ON READER CARD

LETTERS to the RSTS Pro... continued from page 6

Hugh J.E. Davies, B.Sc.
Hertfordshire, England

Thanks Hugh, and keep up your good work, also.

I just finished reading the Dec. 1981 issue and was interested in the 'CALLER.BAS' article. Congratulations are due to Stephen Fahey for coming up with his solution to free up small buffers.

For a lot of systems, this method would be more than suitable. However, systems with a large number of CCL's and a high amount of activity may find some disadvantages.

1. Two programs (CALLER) plus the normally executed program are run each time a CALL command is entered. The running of an extra program will create some additional overhead in starting the program, disk access and swapping (unless free memory is available).

2. Abbreviated commands (i.e. SY-.STAT, QU-

THESE PRICES • IN STOCK IMMEDIATE DELIVERY

CALL SONJA OR LAURIE AT: (614) 889-0810

SCHERER'S MINI COMPUTER MART

6145 Dolan Place

Dublin, Ohio 43017

BRAND NEW • FULL 90 DAY WARRANTY

CIRCLE 106 ON READER CARD

LETTERS to the RSTS Pro... continued from page 6

Hugh J.E. Davies, B.Sc.
Hertfordshire, England

Thanks Hugh, and keep up your good work, also.

I just finished reading the Dec. 1981 issue and was interested in the 'CALLER.BAS' article. Congratulations are due to Stephen Fahey for coming up with his solution to free up small buffers.

For a lot of systems, this method would be more than suitable. However, systems with a large number of CCL's and a high amount of activity may find some disadvantages.

1. Two programs (CALLER) plus the normally executed program are run each time a CALL command is entered. The running of an extra program will create some additional overhead in starting the program, disk access and swapping (unless free memory is available).

2. Abbreviated commands (i.e. SY-.STAT, QU-

THESE PRICES • IN STOCK IMMEDIATE DELIVERY

CALL SONJA OR LAURIE AT: (614) 889-0810

SCHERER'S MINI COMPUTER MART

6145 Dolan Place

Dublin, Ohio 43017

BRAND NEW • FULL 90 DAY WARRANTY

CIRCLE 106 ON READER CARD

LETTERS to the RSTS Pro... continued from page 6

Hugh J.E. Davies, B.Sc.
Hertfordshire, England

Thanks Hugh, and keep up your good work, also.

I just finished reading the Dec. 1981 issue and was interested in the 'CALLER.BAS' article. Congratulations are due to Stephen Fahey for coming up with his solution to free up small buffers.

For a lot of systems, this method would be more than suitable. However, systems with a large number of CCL's and a high amount of activity may find some disadvantages.

1. Two programs (CALLER) plus the normally executed program are run each time a CALL command is entered. The running of an extra program will create some additional overhead in starting the program, disk access and swapping (unless free memory is available).

2. Abbreviated commands (i.e. SY-.STAT, QU-

THESE PRICES • IN STOCK IMMEDIATE DELIVERY

CALL SONJA OR LAURIE AT: (614) 889-0810

SCHERER'S MINI COMPUTER MART

6145 Dolan Place

Dublin, Ohio 43017

BRAND NEW • FULL 90 DAY WARRANTY

CIRCLE 106 ON READER CARD
TURNING COUSINS INTO RUN-TIME SYSTEMS

Making RTS's from BP2 programs.

A few issues back we discussed the benefits of creating multi-user tasks out of some of the CUSP's. We also mentioned that the limiting factor in creating sharable (multi-user) tasks was that the resident libraries generated had to be added to load at specific memory addresses. On systems with small amounts of physical memory, this means that only 1 or 2 CUSP's can be made sharable. On systems with large amounts of physical memory, this means that you have to fragment memory.

The long term solution is for DEC to allow us to add resident libraries without specifying the load address like we can with run-time systems. The short term solution is to change a BASIC-PLUS-2 program into a run-time system is to:

- Compile the source program into MACRO.
- Eliminate all of the funny control characters the BASIC-PLUS-2 compiler leaves behind. When the compiler generates it's MAC file it comments the code for literal strings with the literal string. Thus the comment for 'A$ = SYS(CHR$(6) + CHR$(26))' is "CTRL-F CTRL-Z," which will confuse the MACRO assembler, so I use EDT V2 to eliminate all quoted strings from the file. (An SPR has been submitted.)
- Run the program BP2RTS (included in this column) which will separate the read-only code from the read-write code, and generate an ATPK command file. Note that you have to take a guess at the size the run-time system will be. The program defaults to a guess of 16KW.
- Execute the command file, which will:
  - Assemble the 2 macro source files generated by the program.
  - Link the read-write object module.
  - Link the read-only object module including the read-only symbol table.
  - Generate the run-time system from the read-only task using MAKISIL (MAKISIL has a bug in it that prevents it from creating run-time systems larger than 16KW, an SPR has been submitted, and a patch is included.)
- Add the read-only run-time system.
- Link the read-write object module including the read-write symbol table.
- Name the read-write task to the read-only run-time system.
- Delete the files created that are no longer needed.
- Run the read-write task to make sure everything still works.
- Add the commands needed to add the read-only run-time system to your start-up command files.

NOTE

This procedure creates several files. These files are: RO.*, RW.*, RO1.CMD, and cuspname.CMD

Using this procedure we have 'converted' ATPK, SYSTAT, BATRUN, and SPLRUN into sharable run-time systems with the following results:

<table>
<thead>
<tr>
<th>R/W size</th>
<th>R/O (RTS) size</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATPK</td>
<td>3KW</td>
</tr>
<tr>
<td>SYSTAT</td>
<td>17KW</td>
</tr>
<tr>
<td>BATRUN</td>
<td>20KW</td>
</tr>
<tr>
<td>SPLRUN</td>
<td>21KW</td>
</tr>
</tbody>
</table>

These programs were 'converted' because of the high probability that more than one copy may be running at the same time.

This procedure, while not exactly a 'clean' procedure does accomplish the goal of allowing BASIC-PLUS-2 programs to share their read-only segments of code. Good luck.

... continued on page 79
You Deserve An

A+

In Accounting

Software Techniques announces A+. State-of-the-art accounting software for the PDP-11. A system so flexible it adapts to the way you do business, not the other way around.

A+ has multi-company/multi-division capability, comprehensive financial reporting, and complete documentation, plus an optional English-language report writer.

If you'd like to see more, ask us to send you a demo. We think you'll agree it's an A+.

Software Techniques Incorporated

Business tools for the work of Man.

5242 Katella Avenue
Los Alamitos, CA 90720
United States
Phone (714) 995-0533

74/76 Northbrook Street
Newbury, Berkshire RG13 1AE
United Kingdom
Phone 44 (0) 885 30840
The RSTS/E environment is made up of three parts: addressing, the low segment of the task, and the high segment of the task. Each of these areas will be dealt with in this article.

**ADDRESSING**

There are three sets of Active Page Registers (APR) on the PDP-11/70 and 11/45 (two on other types of PDP-11s), kernel mode APRs, user mode APRs, and supervisor mode APRs. The Monitor uses the kernel mode APRs to map itself into memory. The user APRs map the user task into memory. The APR is actually a pair of sixteen-bit registers, the page address register (PAR) and the page descriptor register (PDR).

The page address register defines where the page actually begins in the memory (starting address). The page descriptor register defines the maximum length of the page and how it can be accessed (read or write, read only, etc.).

The sixteen-bit address generated when a program is compiled is treated as a relocatable (virtual) address. It defines which one of the active page registers is to be used to calculate a physical address. It also contains the byte offset within the page.

The PAR of the APR is handled as though it contains bits six through twenty one (bits six through seventeen for PDP-11s other than 11/70 and 11/45) of the 22-bit (or 18-bit) physical address, which is the starting address. The PAR is combined with the byte offset within the page to get the physical address.

Example: Take virtual address 72322 octal and convert it to a physical address, APR 3 is 1460 octal.

<table>
<thead>
<tr>
<th>APR</th>
<th>Offset</th>
<th>Physical Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>12322</td>
<td>160322 octal</td>
</tr>
</tbody>
</table>

The byte offset into the page from the virtual address is thirteen bytes long. This allows addressing of 4096 words. An APR therefore maps 4KW and there are eight APRs so 4KW * 8APR = 32KW program size.

**LOW SEGMENT OF A JOB**

The first one thousand bytes of the user task have special meanings to the Monitor. So the 32KW task area is shortened by one thousand bytes. The figure below indicates what information is contained in this region of the low segment.

<table>
<thead>
<tr>
<th>Virtual Address</th>
<th>!APR ! byte offset with in page!</th>
<th>pointer virtual address to APR</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 13 12 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>! APR ! byte offset with in page!</td>
<td>pointer virtual address to APR</td>
<td></td>
</tr>
</tbody>
</table>

The First 1000 Bytes

<table>
<thead>
<tr>
<th>!controlled by job -- user job image!</th>
<th>! or run-time system</th>
</tr>
</thead>
<tbody>
<tr>
<td>!used by the monitor for job control!</td>
<td>! text information to make job swappable!</td>
</tr>
<tr>
<td>!used by the monitor for hardware!</td>
<td>! floating point context information to make job swappable!</td>
</tr>
<tr>
<td>!default SP stack area!</td>
<td>!mation to make job swappable!</td>
</tr>
<tr>
<td>!Keyword (KEY bits 8 - 15) !</td>
<td>! (USRSP bits 0 - 7) !</td>
</tr>
<tr>
<td>!file request queue block (FIRQB)!</td>
<td>!transfer request block (XRB)!</td>
</tr>
<tr>
<td>!core common area (CORCMN)!</td>
<td>!controlled by job!</td>
</tr>
<tr>
<td>!user-assignable PPN (USRPPN)!</td>
<td>!user-assignable default protection!</td>
</tr>
<tr>
<td>!user-assignable default protection!</td>
<td>!code (USRPR!) !</td>
</tr>
<tr>
<td>!user-assignable default protection!</td>
<td>!user logical device name table!</td>
</tr>
<tr>
<td></td>
<td>! (USRLOG)!</td>
</tr>
</tbody>
</table>

**GENERAL DESCRIPTION**

KEY — (bits eight through fifteen of the keyword) this byte defines the job’s status in the RSTS/E environment. The keyword is refreshed by the monitor at
different points during the timesharing session. The defined bits of the KEY are listed below:

**JFLOCK** Bit 14 — when one indicates that the job does not wish to be swapped.

**JFBIG** Bit 13 — when one indicates that the job can exceed its private memory maximum.

**JFNOPR** Bit 12 — when one indicates that the job is not logged in yet.

**JFSYS** Bit 11 — when one indicates that the job is running with temporary privileges.

**JFPRIV** Bit 10 — when one indicates that the job has permanent privileges.

**JFFPP** Bit 9 — when one indicates that the contents of the hardware floating point unit should be part of the context of this job.

**JFSPRI** Bit 8 — when one indicates that the job is running with the special run priority at \( \frac{1}{2} \) level higher than normal.

**USRSP** — (bits zero through seven of the keyword) is assigned a value of 400 (by COMMON.MAC). The Monitor automatically loads this value into the stack pointer register (R6) when a job is created.

**FIRQB** — is the main communication area between the Monitor and the user for Monitor directives that involve file or device operations. Below is a diagram of the FIRQB area.

---

**FIRQB**

```plaintext
<table>
<thead>
<tr>
<th>Bit  Location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>unused</td>
</tr>
<tr>
<td>1</td>
<td>return status</td>
</tr>
<tr>
<td>2</td>
<td>CALPVAL/ IUO sub</td>
</tr>
<tr>
<td>3</td>
<td>func. (PQFUN)</td>
</tr>
<tr>
<td>4</td>
<td>MSB of file size!channel number * 2</td>
</tr>
<tr>
<td>5</td>
<td>PopIL ! (PQERNO)</td>
</tr>
<tr>
<td>6</td>
<td>project and programmer number ! (PQPPN)</td>
</tr>
<tr>
<td>10</td>
<td>filename (2 words in Radix-50 format) ! (PQNAME)</td>
</tr>
<tr>
<td>14</td>
<td>file extension (in Radix-50 format) ! (PQEXT)</td>
</tr>
<tr>
<td>16</td>
<td>least significant bits of file size ! (PQSZ)</td>
</tr>
<tr>
<td>20</td>
<td>buffer length (PQBUPL)</td>
</tr>
<tr>
<td>22</td>
<td>mode (PQMODE)</td>
</tr>
<tr>
<td>24</td>
<td>status flag (PQFLAG)</td>
</tr>
<tr>
<td>26</td>
<td>protection code !&lt; 0, prt. real code ! (PQFROF)</td>
</tr>
<tr>
<td>30</td>
<td>device name (two ASCII characters) ! (PQDEV)</td>
</tr>
<tr>
<td>32</td>
<td>!&lt;&gt; 0, unit no. ! device unit no. ! real ! (PQDEV)</td>
</tr>
</tbody>
</table>

---

**XRB** — is the main communication area between the Monitor and the user for Monitor directives handling file or device input/output. Below is a figure of the XRB.

---

**XRB**

```plaintext
<table>
<thead>
<tr>
<th>Bit  Location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>buffer size in bytes (XRLEN)</td>
</tr>
<tr>
<td>2</td>
<td>bytes actually transferred ! (XRBC)</td>
</tr>
<tr>
<td>4</td>
<td>!buffer address (XRLOC)</td>
</tr>
<tr>
<td>6</td>
<td>MSB block # !channel number ! (XRBLKM) ! 2 (XRCL)</td>
</tr>
<tr>
<td>10</td>
<td>least significant bits of ! the block number (XRBLK) !</td>
</tr>
<tr>
<td>12</td>
<td>wait time for terminals ! (XRTIME)</td>
</tr>
<tr>
<td>14</td>
<td>device modifier (XRMC)</td>
</tr>
</tbody>
</table>
```

**CORCMN** — this is core common which is used as a common data exchange area when it is
necessary to exchange lengthy data between the monitor and the job, or between programs running under the same job number.

**USRPN** — the project-programmer number used when an "@" is used in the file string scan.

**USRPRPT** — the protection code default used in the file string scan.

**USRLOG** — the user's private logical device name table, three to four logical names can be stored here. A figure for this area if given below.

```
USRLOG  offset
| logical device name | ! 0 |
| in Radix-50         | !   |
| physical name in two| ! 2 |
| ASCII characters    | !   |
| real unit !unit number! | ! 4 |
| number             | ! ! |
```

### HIGH SEGMENT OF A JOB

The run-time system associated with the job is located in the high segment of the job task area. The run-time system takes up multiples of 4K words of virtual address space, due to APR mapping. The BASIC-PLUS run-time system can be generated to take up 13K words but when it is used the user area is not increased by 3KW due to APR mapping (discussed in more detail later).

The Monitor uses certain areas of the high segment to get information from the job defining what work the Monitor is to do for the job, and to pass information to the job. The run-time system sets this area with entry points and values to define itself to the Monitor.

Also contained in the high segment of the task is the pseudo vector. The pseudo vector is used for the run-time system to communicate. In general, the pseudo-vector region contains the following:

1. Values and flags which define the capabilities of the run-time system to the Monitor.

2. Addresses pointing to locations within the run-time system where the monitor is to pass control when certain conditions occur.

### Format of the Pseudo-Vector Region of the High Segment

<table>
<thead>
<tr>
<th>Flags describing the run-time system (P.SIZE)</th>
<th>177732</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum size in K words of user job image (P.MSIZ)</td>
<td>177734</td>
</tr>
<tr>
<td>Trap address for FIS hardware floating point option</td>
<td>177736</td>
</tr>
<tr>
<td>Trap address for trap (P.FISR)</td>
<td>177740</td>
</tr>
<tr>
<td>Trap address for trap (default run-time system only)</td>
<td>177742</td>
</tr>
<tr>
<td>Trap address for trap and T-bit traps</td>
<td>177744</td>
</tr>
<tr>
<td>Trap address for BPT instruction and T-bit traps</td>
<td>177746</td>
</tr>
<tr>
<td>Trap address for IOT instruction (P.IOT)</td>
<td>177750</td>
</tr>
<tr>
<td>Trap address for non-Monitor EMT instructions (P.EMT)</td>
<td>177752</td>
</tr>
<tr>
<td>Trap address for all TRAP instructions (P.TRAP)</td>
<td>177754</td>
</tr>
<tr>
<td>Trap address for FPP or FPU floating point units</td>
<td>177756</td>
</tr>
<tr>
<td>Trap address when user types CTRL-C (P.2CC)</td>
<td>177758</td>
</tr>
<tr>
<td>Trap address when user types one CTRL-C (P.CC)</td>
<td>177760</td>
</tr>
<tr>
<td>Trap address when user types FPP floating point units</td>
<td>177762</td>
</tr>
<tr>
<td>Trap address when user types two CTRL-C (P.CC)</td>
<td>177764</td>
</tr>
<tr>
<td>Trap address when user types one CTRL-C (P.CC)</td>
<td>177766</td>
</tr>
<tr>
<td>Trap address when user types two CTRL-C (P.CC)</td>
<td>177768</td>
</tr>
<tr>
<td>Trap address for various &quot;bad&quot; errors (P.BAD)</td>
<td>177770</td>
</tr>
<tr>
<td>Trap address when user types one CTRL-C (P.CC)</td>
<td>177772</td>
</tr>
<tr>
<td>Trap address when user types two CTRL-C (P.CC)</td>
<td>177774</td>
</tr>
<tr>
<td>Trap address when user types one CTRL-C (P.CC)</td>
<td>177776</td>
</tr>
</tbody>
</table>

### GENERAL DESCRIPTION

**P.FLAG** — this word is set with flags which define the capabilities of the run-time system to the monitor.

**P.DEXT** — the default runnable file extension, when a .RUN is executed and an extension is not given, P.DEXT is used.

**P.MSIZ** — minimum size for a user job in K word for this run-time system.

**P.SIZE** — maximum size that a user job image can be for this run-time system.

**P.FIS** — trap address for the hardware floating point instruction set.

**P.BAD** — Monitor passes control on to the run-time system at the location specified by P.BAD when the following synchronous traps occur:

1. Memory management unit exception
2. Job tries to execute a reserved instruction
3. Job issues an instruction with an odd address

**P.BPT** — contains the trap address for a BPT instruction and for T-bit traps.

**P.IOT** — contains the trap address for an IOT instruction.

**P.EMT** — contains the location to which control is transferred for non-Monitor EMT instructions.

**P.TRAP** — contains location to which control is transferred for all TRAP instructions.

**P.FPP** — contains trap address for FPP (or FPU) hardware floating point units.

**P.CC** — contains location to which control passes when the user types a 1C.

**P.2CC** — contains location to which control passes when the user types a second 1C.
P.CRAS — entry points used only by the system default run-time system.

P.NEW — Monitor passes control to this entry point under the assumption that “new user” or “next request” processing is to be done, as opposes to the P.RUN entry point, where it is known that a specific program is to be run under this run-time system.

P.RUN — Monitor passes control to this entry point when an executable program is to be run for a job under control of this run-time system.

AFTERTHOUGHTS

It is known that RSTS/E has a limitation of 31 KW for user tasks NOT 32KW as prescribed by the use of APRs. The explanation given (by Digital) for this is that there is a problem with the fifteenth bit of the address being used as a sign bit. How can this bit be used as a sign bit if the bits thirteen through fifteen are used to determine the APR? If the physical address is always created by sending the virtual address to memory management then the 32KW of memory MUST always be addressable since RSTS/E would not be concerned with physical addressing. Since the APRs must be used in address calculations there must be some other reason why the 32KW of memory cannot be accessed.

The 32KW of memory is used to communicate between the Monitor and run-time system. This reserved area may prevent the user task from growing into that last kilo-word of memory because the Monitor would start to interrupt the words in the last kilo-word of the task as run-time system entry points and status words. This may explain why RSTS/E will not allow the user task to grow past 31KW.

When using a run-time system, for example BASIC-PLUS, a 16KW run-time system, the user task never exceeds 16KW of memory. The run-time system is mapped by 4 APRs. The user task would be mapped by 4 APRs.

With the introduction of disappearing RSX the entire job area could be used for the user task. The problem of the user task accessing the top 1KW of the task (whatever it may be) becomes important. The 31KW job size maximum is established because RSTS/E cannot go to 32 KW for the user task. In conclusion it is discovered that the 32KW task image that RSTS/E promises is reduced by 2KW. The first one thousand bytes are preassigned by the monitor and the last kilo-word RSTS/E cannot access due to some secret internal problem. The 32KW of the user task cannot be accessed and Digital has indicated that it is a problem dealing with the sign bit. With the information presented here it seems incorrect to say that there is a sign bit problem but rather the fixed locations in the high segment of memory must be used only by a run-time system and not the user task, because these locations have special meanings to the monitor.

Run-time systems must be mapped by the APRs. When a run-time system takes up less than a multiple of 4KW, the memory to the next multiple of 4KW is lost. In other words, if the BASIC-PLUS run-time system is generated with a size of 13KW, the 3KW to the next lower boundary (run-time systems are loaded from the high segment down to the low segment) is lost.

SYSTEM PERFORMANCE ANALYSIS FOR VAX AND RSTS/E USERS

If your system is suffering from slow response, clogged I/O, reduced throughput, then put Raxco's RABBIT-2 on your case. RABBIT-2 locates the trouble spots in your operating system and identifies the source of the problems.

RABBIT-2 will chart your system performance, on an hour by hour, user by user, or program by program basis. RABBIT-2 will quickly sketch a profile of your average system day, your average user demands, and your average program resource requirements.

RABBIT-2 provides the tools you need to investigate system throughput in terms of CPU, I/O, memory, connect, KCTs, etc., over any time period you specify. You can play What If by simulating the removal of the offending program or user and displaying the results of the change.

RABBIT-2 capabilities include:
- Batch and interactive analysis
- Interval or time displays
- Resource consumption diagrams
- User and program investigation
- Graphs and numeric output
- WHO was on WHEN
- Profile analysis of users and programs
- Ratios of resources utilized
- Rankings of users and programs
- Forecasting of future resource consumption

Raxco markets a complete line of operational support, financial planning and data management systems for DEC computing equipment. For a free catalog of these systems, contact:

Suite 200
6520 Powers Ferry Rd.
Atlanta, GA 30339
(404) 955-2553

CIRCLE 33 ON READER CARD
RSTS
Application Software

Accounting Two:
General Ledger
Accounts Receivable
Accounts Payable

Flexible reporting structure
System adapts to you

Table driven software
You make changes easily

Report formatter
End user creates report

Modular and integrated
You enter data only once

Menu selections
Easy to use

On-line support
Free-up your programming staff

Proven successful in Canada and USA
It works now

IAS
Computer Corporation Limited
6080 Young Street
Suite 401
Halifax, Nova Scotia
B3K 5L2

Contact: Pat Hunt (902) 453-4620

CIRCLE 112 ON READER CARD

APR3

other memory

APR4

usable

APR5

beginning of RTS

APR6

BASIC-PLUS RTS

13KW

memory

APR4 contains the starting address of the 4KW segment that contains the first KW of this BASIC-PLUS run-time system. Therefore it is more efficient to generate a 16KW BASIC-PLUS run-time system so that the 3KW of memory is not wasted.

A 17KW BASIC-PLUS run-time system would cut the user task down to 12KW. An additional APR would be needed to map the run-time system. An APR cannot be used to map the run-time system and user task area at the same time.

17KW BASIC-PLUS Run-time System

APR0

APR1 | User task 12KW |
APR2

other memory

APR3

unusable

first 1KW of RTS

APR4

APR5

BASIC-PLUS RTS

17KW

memory

It can be seen from the above figure that a 17KW BASIC-PLUS run-time system reduces the overall task size down to 29KW. 3KW remember are unusable due to APR mapping. In conclusion it should be noted that it is useful to generate a run-time system to a 4KW boundary opposed to reducing it but not reducing it enough to reduce the use of an APR. Sizes of run-time systems must be examined to respect to their APR usage opposed to their memory usage.
Sooner or later you will be using electronic mail. It just makes good sense. When you do, you will want a system that is complete—a delivery system, a scheduling system, and an information manager. Your electronic mail system will become an essential part of your office environment. INTECOM is such an electronic mail system.*

INTECOM’s power is easy to control. It relates to the way you work. Electronic IN, OUT, and HOLD baskets are just what you would expect. You can scan your IN basket, selecting only those message subjects you wish to read. Or, you can place a message into your HOLD basket for a number of days to have it automatically reappear in your IN basket on the appointed day. You can even have INTECOM recall specific messages by providing your own selection criteria. Replying, forwarding, and sending to groups are as easy as can be. And these are just a few of the features in store for you.

You owe yourself a closer look. Write for a brochure or give us a call direct.

INTECOM...the INtelligent TExt COMmunicator.

**INTECOM is currently available on DEC computers using the RSTS operating system. RSTS is a registered trademark of Digital Equipment Corporation. INTECOM is a trademark of Logic eXtension Resources.**

North County Computer Services, Inc.
2235 Meyers Ave.,
Escondido, California 92025
(714) 745-6006, Telex: 182773
RABBIT-4 FILE SECURITY
GUNS DOWN DATA RUSTLERS ON RSTS/E SYSTEMS

Do You Know Who The Pesky Varmints Reading Your Confidential Information Are?

RABBIT-4 will help you track 'em and catch 'em, cause it has a proven record of performance. RABBIT-4 will let you:

□ Log secured file accesses
□ Signal OPSER of violations
□ Roll-out the bandits
□ Freeze system activities with 6 levels of file security to keep your data safe and secure, RABBIT-4 will also:

- Secure up to 64 data files
- Provide 32 user descriptions plus wild cards
- Restrict file access to specified programs
- Identify intrusions and intruders
- Recover automatically from system crashes

SEE OUR TALENTED RABBITS AT THE ATLANTA DECUS MEETING and pick up a free Rabbit Executive Coloring Book Souvenir! We'll have a hospitality suite where we'll demonstrate Rabbit's resourcefulness in:

- Performance Analysis
- Job Accounting
- Resource Accounting
- Data Management
- Financial Planning
- File Security

RAXCO INC.
RAXCO Inc., Suite 200, 6520 Powers Ferry Road, Atlanta, GA 30339 • (404) 955-2553

CIRCLE 21 ON READER CARD

CAMS
(Computer Assisted Medical System)

is now available to RSTS/E Users!

- Designed for RSTS/E Timesharing
- Basic Plus 2
- RMS Files
- VT100 Full Screen

NATIONWIDE INSTALLATIONS
REFERENCES ON REQUEST

CALL TODAY!
(217) 529-8700

New software package handles patient information, billing, all insurance processing. Easy to learn, simple to operate!

155 West Lake Drive
Springfield, IL 62703

CIRCLE 113 ON READER CARD
Create the read-only TCB command file. •
Include the KEK run-time system modules needed. •
Setup the stack. •
Setup the partition. •
Setup the flag word for a non-xmb run-time system. •
Setup a dummy extend section for MAREIL.

5030 OPEN INPUT NAMES + "CMD" FOR OUTPUT AS FILE 1.CHRN.CFG
PRINT 1.CHRN.CFG, "SALLOW NO ERRORS" + CH + LF &
"ANCHOR NO=0" + CH + LF &
"SALLOW WARNING ERRORS" + CH + LF &
"MSW=0" + CH + LF &
"SALLOW NO ERRORS" + CH + LF &
Ignore the undefined symbol error message." + CH + LF &
"SALLOW NO ERRORS" + CH + LF &
"SALLOW NO ERRORS" + CH + LF &
"SALLOW NO ERRORS" + CH + LF &
"SALLOW NO ERRORS" + CH + LF &
Ignore the 4 undefined symbol error messages." + CH + LF &
"SALLOW NO ERRORS" + CH + LF &
"SALLOW NO ERRORS" + CH + LF &
"SALLOW NO ERRORS" + CH + LF &
"SALLOW NO ERRORS" + CH + LF &
Ignore the 4 undefined symbol error messages." + CH + LF &
"SALLOW NO ERRORS" + CH + LF &
"SALLOW NO ERRORS" + CH + LF &
LIST (TEMP_1$) + CH + LF &
"SALLOW NO ERRORS" + CH + LF &
"SALLOW NO ERRORS" + CH + LF &
RUN (1,2)MAKSIL + CR + LP
CLOSE 1.CHRN.CFG &
Create the command file. •

5900 GOTO 32760 &
End of the command file generator. •

142001 &
COPY A UNTIL A STATEMENT IS FOUND. •

14210 DEP* PNGLOBALIZE(TEMP_0$, TEMP_0$) &
LINPUT 4.CHAN.IN, TEMP_0$ &
PRINT TEMP_0$, PNGLOBALIZE(TEMP_0$) &
CLOSE 4.CHAN.IN &
Close all channels (the fast way). •

134001 &
GLOBALIZE STATEMENT LABELS. •

13410 DEP* PNGLOBALIZE(TEMP_2$) &
GOTO 14230 &
IF ASCII(TEMP_2$) <> ASCII.LA &
PRINT 4.CHAN.IN, ASCII(TEMP_2$) &
"ERROR: Syntax error encountered! &
END OF FILE. •
CLOSE 4.CHAN.IN &
End of the function. •

190001 &
Return the line to the user. •

1901 &
Error Handler •

19011 IF ERR = 116 &
THEN PRINT 1.CHAN.ERR, "Bad MAC file format, check for " &
"control character. •
IF ERR = 114 &
RESUME 32760 &
End of file on device. •

19301 IF ERR = 20104 &
OR ERR = 10504 &
THEN PRINT 1.CHAN.ERR, "Bad filename - " &
END OF FILE. •
RESUME 32760 &
In case the user types in some garbage. •

19999 UNDEFINED GOTO 0 &
Give up. •

200001 &
Data Statements •

20010 DATA "160000:020000", "140000:040000", "120000:060000" &
20020 DATA "100000:120000", "060000:120000", "040000:140000" &
20030 DATA "020000:180000" &
Close all channels (the fast way). •

23767 END

The DS 180 provides a total package of performance features for any application where quality impact printing is required. Not a "hobby-grade" printer, the DS 180 is a real workhorse designed to handle your most demanding printer requirements.

High Speed Printing — Bidirectional printing at 180 cps offers throughput of over 200 lpm on average test. A 9-wire printhead generates a 9x7 font with true lower case descenders and underlining.

Non-volatile Format Retention — A unique keypad featuring a non-volatile memory makes programming the DS 180 quick and easy. Top of form, tabs, perforation skipover, communications parameters and many other features may be entered and stored from the keypad. The DS 180 even remembers the line where you stopped printing — eliminating the need to reset the top of form at power-on.

Communications Versatility — Interfaces include RS232, current loop and 8-bit parallel. Baud rates from 110-9600 may be selected. A 1kB buffer and X-on, X-off handshake ensure optimum throughput.

Forms Handling Flexibility — Forms ranging from 3"-15" may be fed from the front or bottom, and an adjustable printhead provides crisp, clear copy on forms with as many as 6-ports.

For more information on how the DS 180's low-cost total printer package can fill your application, contact us at Datasync. The DS 180 is available for 30-day delivery from our sales/service distributors throughout the U.S.

April 1982

CIRCLE 73 ON READER CARD
There's only one accounting, manufacturing software company that speaks the language of VAX... Interactive Management Systems.

IMS has been developing business application software for the VAX family of computers since its introduction by Digital Equipment Corporation. As a result, we know from long experience what you, the VAX-user, need in order to get your accounting and manufacturing functions on-line.

Our truly native mode software is written entirely in VAX-11 Basic, utilizing VAX-11RMS. And that makes it compatible with DEC's Datatrieve query language. You'll also find IMS Software easy to integrate, exceptionally versatile. For more information, see our new IMS Software Portfolio, featuring the most extensive line of Business Applications Software for VAX and RSTS/E.

Clip and mail the coupon below... or call us today at (617) 489-3550.

**Interactive Management Systems**

The software company that speaks your language!

While in Atlanta at DECUS, come talk with us at DEXPO-82, Booth #816!

*We also speak RSTS/E.

DEC, VAX, Datatrieve are trademarks of Digital Equipment Corporation.
THE DEC/RSTS DECISION SUPPORT SOLUTION

When it comes to comprehensive financial, organizational or strategic planning, there is only one choice for the RSTS user:

FCS-EPS

FCS-EPS is the one system that brings the power of the computer directly into the hands of the planning professional.

More than just a modeling system, FCS-EPS is a sophisticated, open-ended system easily applied to virtually any planning task. Over 60 built-in planning-oriented functions allow you to be immediately productive. "What if" analysis, goal-seeking, a customized report writer, hierarchical consolidation, and a built-in financially oriented language make the utility of FCS-EPS virtually unlimited, all without the need to know any cryptic computer language.

The system is also available on the Dcsystem 10 and 20, VAX and over 40 other hardware/operating system combinations.

700 users of FCS-EPS can't be wrong. Find out more about truly user-oriented financial planning systems. Contact EPS today. San Jose, CA 800/538-7578 or 408/292-6212; Toronto 416/279-8711; London (01) 579-6931.
**EXTRACT**

By Stephen Munyan, 135 Brattle St., Holden, MA 01520

EXTRACT is a program which was designed to allow programmers to copy lines of one program into another quickly without having to worry about going through BASIC-Plus immediate mode statements to extract the lines. This program has been used in on our system for over 1 year, and is quite widely used.

As an example of how this program can be used, we will assume that the programmer has written a program which contains several routines which need to be used in a new program under development. For convenience we will call them ROUTINE.BAS and APPLIC.BAS. In this case we want to copy lines 10, 50, 700-750, and line 1000 from ROUTINE.BAS and place them into a new file called APPLIC.BAS. To accomplish this we would issue the following procedure:

**RUN EXTRACT**

**EXTRACT** — Program Line Extraction Program — V7.01

Output = Input [/Append]

*APPLIC.BAS = ROUTINE.BAS

Enter the line numbers to be extracted from the input file separated by commas. A dash may appear between entries to allow ranges of lines to be extracted. Once all lines to be extracted have been entered press CTRL/Z

| 10 50 |
| 700-750 |
| 1000 |
| . |

**Extraction Complete**

In the example above, several line number combinations were entered on the same line. As many entries as desired can be entered on the same line as long as they are separated by commas. For example, if we wanted to enter all of the lines on the same line we could have entered: 10, 50, 700-750, 1000 all on the same line.

If the lines being extracted are to be placed at the end of an existing program, the /APPEND switch can be used. If this switch is used, the program is assumed to be lacking an END statement since when the output file is OLD'ed, it will ignore any statements that were appended to the file after the END statement.

As an optional patch, line 1015 can be updated to use the RECORDSIZE option to allow BASIC-Plus to use larger record sizes on the input file. This will speed the extraction since the program will spend less time waiting for I/O to be processed. Depending on the amount of space allocated to each user, the size of the Record can vary from 4096 to 16384 bytes.

---

**FAST-11**

SYSTEM DEVELOPMENT TOOL FOR RSTS/E BASIC-PLUS

- Automated Definition of Record Layouts
- ISAM, Sequential, Random, and Tape Files
- Version 7 RSTS/E Large File support
- Program generated, RNO compatible Document Facility
- Supports numerous Data Types
- DMS 500/FAM Compatible

CASHER ASSOCIATES INC.
1371 Beacon Street  •  Brookline, MA 02146

CIRCLE 78 ON READER CARD

(617) 232-9111  
(212) 757-2868
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 user's UFD.
- Optimizes cluster sizes.
- Saves accounting data.
- Is easy to use and fully documented.
- Provides logging and statistics.

LAST CHANCE TO BUY DSKBLD AT THE $500 INTRODUCTORY PRICE

☐ Send me more information on DSKBLD

Name __________________________________________
Company _______________________________________
Address _______________________________________
Telephone _______________________________________

Here's What DSKBLD Users Say:

"We have about 130 terminals in the field. The day after I used DSKBLD they called to find out what I had done to fix the response problems!...I used to have a lot of "crazy" weekend hours—now its a simple production job I can trust to my operators"

— New York user

"I would recommend it to anyone with a disk bound system. DSKBLD lets me rebuild a 300 MB disk in an hour and improves system throughput."

— Oklahoma user

"Our users noticed the results after the first use...FIP needed dropped from over 50% to under 20% immediately...After seeing the results, we would pay twice the price for it."

— Minnesota user

"Cache hits went from 45% to 81%...Overall throughput increased almost 100%...cut $15/hour overtime by 75%...have recommended it to others."

— Massachusetts user

"Can rebuild all my disks in one Sunday instead of spending three Sundays each month...definitely met my objectives...am ordering a license for my other CPU."

— Washington user

"I like the safeguards built into DSKBLD—especially for the unsophisticated user...very easy to install."

— Colorado user

"I've already recommended it to two other users."

— Colorado user
RPGED.TEC
By Austin Kinsella, Regional Technical College, Carlow, Ireland

The Regional Technical College in Carlow is one of a number of similar 3rd level institutions in Ireland, providing mainly 2 and 3 years courses in technical subjects. The College has a PDP 11/34 RSTS/E system, on which the bulk of the time is consumed by students on our 2 year data processing course. During their second year, these students spend some time learning RPG. As we cannot afford the overhead of multiple copies of RPGESP, and as the students are already familiar with TECO for editing Cobol and Basic-Plus sources, it was decided to provide them with a simple RPG forms editor in TECO. The listing of this editor is attached. After squeezing, it adds less than 1K to each user's buffer, so that multiple copies can be run without causing swapping.

When running, the editor displays a row of column numbers and the current form mask, with legal fields denoted by I or *. (or C for command) and unused columns by spaces. Cursor movement will not leave the cursor in an illegal column. Facilities are provided for left and right cursor movement, up and down lines, line renumbering, and new line creation with automatic line numbering. To minimize screen updating, only 6 text lines over the mask are displayed. The editor is loaded by EIRPGED, and is run by MR. Exit back to TECO is by IZ, and MR can be re-issued after any intervening TECO commands, for example a search to move to a new position in the file. We have the editor installed with protection <104> in one of our library accounts, but it can go anywhere.

RPGED is not foolproof, and there remains considerable scope for development. The primary design goal was to provide easy RPG forms editing with a low memory overhead, and we feel this has been achieved. Because most of our terminals are VT52s, only VT52 escape sequences are used in the editor, but on a VT100 the editor should run faster in ANSI mode by scrolling the text window up and down over a fixed mask. Other changes that might be desirable would be to make RPGED executable by CCL, to accept numeric arguments on the movement commands, and to retain a column position on macro entry or line change. We have a second version of the editor called SRTED which has the same functions but displays Sort rather than RPG form types.

In conclusion I should point out that I have been aware of the need for this editor for over a year, but it took the arrival of the back issues of RSTS PRO, with the articles on TECO, to spur me to write it.

10 WAYS TO PRINT NOW, PAY LATER.

Call 800-243-9054* for details.

1. CHARGE-A-PRINTER. We'll install the line printer you need today, and defer payment until next quarter.

2. MORE LPM FOR LESS $. The Printer Store sells the identical printers that mini vendors offer, at up to 50% less.

3. UPGRADE YOUR SYSTEM, NOT YOUR BUDGET. Double your system throughput now with a new, high-speed line printer or RLPS.

4. ORDER NOW, GET 3 MONTHS FREE MAINTENANCE. Save on the cost of ownership for a limited time only.

5. CONVERT-A-LEASE. Lease the line printer you need today, with the option to purchase anytime.

6. GUARANTEED TRADE-IN. Trade in your old line printer for an additional 10-20% off our already low prices.

7. TWO LINE PRINTERS FOR THE PRICE OF ONE. Our PXS Switch lets you use your old printer as a back-up (free for a limited time only when you purchase a new printer).

8. SHORT-TERM RENTALS FOR HIGH-SPEED LINE PRINTERS. The most economical way to get increased throughput for peak periods.

9. WE'LL REFURBISH YOUR LINE PRINTER, and provide a "loaner" while it's in for repairs, or sell you a refurbished line printer at 20-30% of the cost of new.

10. LET MA BELL PAY FOR YOUR REMOTE LINE PRINTER. Our Remote Line Printer System pays for itself in line use savings.

Summary of RPGED commands:

- **1B**: Back cursor to last legal column, or start of previous line if at start.
- **1U**: Up to start of previous line, change mask if necessary.
- **1D**: Down to start of next line, change mask if necessary.
- **<sp>**: Cursor Right to next legal column or next line if at end.
- **<cr>**: Make new line of 74 cols, insert mask type, insert number if numbering.

![The Printer Store Digital Associates Corporation](image-url)
Hardware Protection
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.

Hardware Protection

C-XX

For DEC Equipment Users

Temperature limits and approved interface to standard DEC AC power control system.

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

Hardware Protection

C-XX

For DEC Equipment Users

Temperature limits and approved interface to standard DEC AC power control system.

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

Hardware Protection

C-XX

For DEC Equipment Users

Temperature limits and approved interface to standard DEC AC power control system.

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

Hardware Protection

C-XX

For DEC Equipment Users

Temperature limits and approved interface to standard DEC AC power control system.

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

Hardware Protection

C-XX

For DEC Equipment Users

Temperature limits and approved interface to standard DEC AC power control system.

DEC, VAX, and PDP are trademarks of Digital Equipment Corp.
GET ON OUR CASE
For LSI-11, Unibus, and VAX Service
Installation & Maintenance
Mixed Vendor Systems
Peripheral Upgrades
Replacement Components
Add-in/Add-on Memories
Depot Level Repairs

April 1982
RSTS/PROFESSIONAL RSTS/PROFESSIONAL RSTS/PROFESSIONAL RSTS/PROFESSIONAL

UK DECUS and the PDP-11 Commercial Sig Announce A RSTS/E Seminar
CARL MARBACH
DAVE MALLERY
AL CINI
3 MAJOR TOPICS
OVERCOMING FUTURES SHOCK:
VMS from a RSTS Perspective
FROM ALL ENEMIES—FOREIGN AND DOMESTIC:
Protecting a RSTS System
CHOOSING HARDWARE FOR COMMERCIAL APPLICATIONS:
How to Size your System

IN LONDON
Tuesday, June 8, 1982
From 10 A.M. to 7 P.M.
Details available at the DECUS office:
Reading [0734] 58355 ext. 3229
PDP11 Commercial Sig
CIRCLE 126 ON READER CARD

LOGIN

12070 M(3) = M(3) - 1
\ GOSUB 11000
\ M(1), M(2) = 6
\ M(4) = 0
\ M(5) = PROG
\ M(6) = Proj
\ PRINT
\ PRINT " Attaching to job":ATT.JOB
\ CHANGE M# TO LOGINS
\ 25 = SYS(LOGINS)
\ RETURN
\ IF JOB IS DETACHED UNDER THIS ACCOUNT
\ THEN PRINT THE NUMBER OF USERS LOGGED
\ IN UNDER THIS ACCOUNT AND ATTEMPT TO
\ ATTACH TO THE SPECIFIED JOB NUMBER.
19000 : ERROR HANDLING ROUTINE
19005 ES = CVT $$ (RIGHT (SYS (CHR$(63) + CHR$(9) + CHR$(ERR)) , 3) , 4)
\ ES = SYSTEM ERROR MESSAGE
19010 IF ERL = 2040 THEN RESUME 2050
\ IF NO MORE TEMP FILES TO DELETE, THEN
\ TRAP ERROR AND CONTINUE WITH THE PROGRAM.
19020 IF ERR = 5 THEN PRINT ES
\ RETURN 9000
\ IF FILE OR ACCOUNT NUMBER SPECIFIED CAN
\ NOT BE FOUND ON THE DEVICE, THEN PROMPT
\ USER TO THIS FACT.
19030 IF ERR = 52 AND ERL = 31020 THEN RET.LINE$ = 0
\ RETURN 1000
\ DELETE ACCOUNT FROM CORE COMMON
\ IS THERE MORE IN CORE COMMON?
\ IF YES, SEE WHAT IT IS
\ ELSE SET ENTRY TYPE AND PROCEED
19040 IF ERR = 49 AND ERR < 53 THEN PRINT " Illegal job number"
\ RETURN 1030
\ IF JOB NUMBER TO ATTACH TO IS IN
\ ENTERED IN A ILLEGAL FORMAT, TRAP
\ FOR IT AND RESUME
19998 PRINT ES;BELL$; ": at line ": ERL
\ RETURN 9000
\ END OF ERROR HANDLING ROUTINE
29999 : ICCL ENTRY PROCESSING

30000 ACCOUNTS = RIGHT(SYS (CHR$(74)) , 6)
\ GOSUB 10000
\ IF ACCOUNTS = NULLS
\ THEN ENTRIES = 0
\ ELSE ENTRIES = -1
\ GET ACCOUNT # FROM CORE COMMON
\ OBTAIN JOB STATUS DATA
\ DETERMINE IF PROJECT-PROGRAMMER # HAS BEEN ENTERED
30010 GOTO 1000
\ ENTER INTO MAIN PROGRAM
30999 : CHAIN ENTRY PROCESSING
31000 CRS = CHR$(13)
\ RET.PGM$ = NULLS
\ RET.LINE$ = 0
\ COMMONS = SYS (CHR$(74))
\ GOSUB 10000
\ IP = INTR$(14, COMMONS, CRS)
\ IF P4 > 0 THEN
\ THEN 31010
\ ELSE 31020
\ END ENTRY PROCESSING
31000 ENTRY$ = -1
\ ACCOUNTS = LEFT (COMMONS, P4-1)
\ COMMONS = RIGHT (COMMONS, P4-1)
\ IP = INSTR$(14, COMMONS, CRS)
\ IF P4 < 0 THEN
\ THEN 31020
\ ELSE 31030
\ GET ACCOUNT
\ DELETE ACCOUNT FROM CORE COMMON
\ IS THERE MORE IN CORE COMMON?
\ IF YES, SEE WHAT IT IS
\ ELSE SET ENTRY TYPE AND PROCEED
31040 ENTRY$ = -1
\ ACCOUNTS = LEFT (COMMONS, P4-1)
\ COMMONS = RIGHT (COMMONS, P4-1)
\ IP = INSTR$(14, COMMONS, CRS)
\ IF P4 = 0 THEN
\ THEN 1000
\ ELSE 31050
\ GET LINE NUMBER TO CHAIN TO
\ IF NOT, PROCEED
\ ELSE GET LINE NUMBER AND PROCEED

CIRCLE 126 ON READER CARD
CIRCLE 126 ON READER CARD

CIRCLE 126 ON READER CARD
CIRCLE 126 ON READER CARD
RPTMAN — REPORT MANAGER

By Jonathan M. Prigot, Systems Programmer, Polyfbron Division, W.R. Grace and Company, Lexington, MA

The report manager program, RPTMAN, is designed to allow users to organize data from a file and print the organized data.

RPTMAN allows the user to format the data horizontally, vertically, sort on any given field, generate a number of pre-formatted reports, or generate various forms such as cutting tickets, acknowledgements, etc.

1. REPORT MANAGER SELECT SCREEN

RPTMAN is entered from the DCS select screen. It presents the user with the option of generating a horizontal list (HL), vertical list (VL), sorted horizontal list (SL), generate a pre-formatted reports (PL), or generate forms such as cutting tickets, etc. (FM). The abort option (AB) is also provided to allow the user to return to the main DCS program.

2. SELECTION CRITERIA SCREEN

Selecting HL, VL, or SL will bring the user to the SELECT SCREEN. The select screen is used to get general information on the input data file, the output file/device, and whether you wish to limit the range of the report.

The items on the select screen (and their meanings) are:

1. RDF INDEX — The default for this item is LIB:RDF.VIR. If you enter an invalid RDF specification, RPTMAN will erase the invalid entry.
2. DATA FILE — The name of the data file to use (e.g. R756).
3. POINTER FILE — (used only if SL was selected) This is SL's workfile. The default name is the data file name +.PTR.
4. DELETED/QUEST RECORDS — Default is <cr> (i.e. print no deleted records and no questionable records.) If you do wish to display either deleted records and/or questionable records, respond with D and/or Q to this option, else just enter a <cr> (carriage return).
5. OUTPUT FILE — Default is LP: (the main 'line-printer'). Where to 'print' the report. This can either be a file, a terminal screen or a printer. The output can be directed to the printer attached to a Datamedia DTB0/1 terminal by specifying KBZ: NOTE!: If you specify a filename for this item, it will be put in your assigned account.
6. MAXIMUM WIDTH — Default 80 for KBs; 132 otherwise. Maximum width of the output device.
7. FORWARD/BACKWARD LIST — Default is forward. Since records are stored on the system in order of creation date, you can sometimes get your report faster by asking for a backwards listing. This is especially true if the data you desire is recent.
8. HEADING OF LIST — Default is the name of the data file.
9. LIST BLANK ELEMENTS — Default is no. This will suppress the print of blank fields in a VL.
10. LIMIT SEARCH — Default is no. If you wish to limit the scope of the report, respond with Y <cr> to this. It will invoke the SELECTION CRITERIA screen.

Assuming that you responded to the LIMIT SEARCH question with either a <cr> or N <cr>, RPTMAN will then ask you to CONFIRM ALL SELECTIONS. If you respond with anything other than a Y <cr>, RPTMAN will blank the SELECTION CRITERIA screen to allow you to re-enter the data.

3. SELECT FIELDS

The SELECT FIELDS screen is invoked by answering Y <cr> to the LIMIT SEARCH question on the SELECT CRITERIA screen. This screen is used to specify the characteristics the fields within the record must have in order to be listed.

The questions on this screen are:

1. USE CREATE DATE — Default is no. This question allows you to select records created within a certain period of time. If you respond with a Y <cr> to this question, you will be further prompted with:
2. AFTER — This is the date of the earliest record you want. Separate the fields within this question by typing a <cr> after day, after month, and after year. Entering a <cr> alone for the date means use the earliest record in the file.
3. BEFORE — The date of the last record you want. Operation same as AFTER. A <cr> for the day means use the latest date in the file.
4. FIELD NUMBER OR NAME — The user can enter either the field number (e.g. F3001.1.R789) or any part of the field name (e.g. DUE DATE). If there is more than one field containing the specified name, or if RPTMAN cannot find the field you specify, it will so inform you and reposition the cursor for another trial. Entering a <cr> alone forces field specification.
5. BETWEEN — Sets the lowest value allowable.
6. AND — Sets the highest value allowable.

After you enter <cr> to terminate select field specification, the system will then ask you to confirm your selections. If you enter anything but Y <cr>, the system will erase the screen and allow you to redo your selections. Once you confirm your selections, the system will ask you to confirm all your selections. If you respond with anything except Y <cr>, the system will return to the SELECTION CRITERIA screen for re-entry.

4. VERTICAL LISTING (VL) SCREEN

The VL program will inform you that it is [WORKING].
5. HORIZONTAL LISTING (HL) SCREEN

The HL program screen is used to select what fields within the record will be printed, and whether the contents of the field are to be counted or totaled. Those fields that are designated as alphanumeric fields are counted, while those that are numeric are totaled.

The screen prints out the field number and the field name, then waits for your response. The legal responses are:

1. <cr> — Do not list this field.
2. Y — List this field and use the field name on the report.
3. heading — Use this title for the field heading.
4. resp/T — List this field using either the default heading or this heading (per items 2 and 3 above), and give field count or total at the end of the report.
5. LAST — Do not print this field or any field that comes after this field.
6. REST — Print this field and all the other fields that come after.
7. up-arrow key — Back up one field to allow re-selection.

After all desired fields have been listed, the user will be returned to the SELECT SCREEN, unless the report was sent to the user's keyboard, in which case RPTMAN will ask the user to type a <cr> to continue.

6. SORTED LIST (SL) SCREENS

There are two screens associated with the SL program: The SORT SPECIFICATION SCREEN, and the FIELD SELECTION SCREEN.

The SORT SPECIFICATION screen specifies the fields to sort and the direction to sort them in. As in the select screen above, either field number or all or part of the field name can be used to specify the field. The order of the sort can be either ascending or descending for the individual field.

The FIELD SELECTION screen is similar to the HL screen, with the addition of a SUBTOTAL (/S:) option. The /S: option operates in a similar fashion to the TOTAL (/T) option in that it provides a sub-count or sub-total of the field. It is used by appending a /S:trigger-field-name-or-number to the Y or field-heading specification. You may either specify the trigger field's number or a portion of its title. If the field specification is not acceptable, you will be notified, and allowed to re-enter your specification.

During the running of the program, the selected field will be subtotaled whenever the contents of the associated trigger-field changes.

After all fields have been specified, the program will prompt for whether you want the report in report format or tape format. Report format has page numbering and report and field titles. Tape format does not; it is pure data. If you choose tape format, you will be further prompted as to whether to separate the output with spaces or commas between the data. Because the output from the tape option is usually used as input to another program, your response to this question depends on what the next program requires.

BIBLIOGRAPHY

BACK ISSUE OFFER
ALL 10 BACK ISSUES OF THE RSTS PROFESSIONAL
$75.00
Send check to:
THE RSTS PROFESSIONAL
BOX 361
FT. WASHINGTON, PA 19036
PAYMENT MUST ACCOMPANY ORDER

LETTERS

privileges, if any. This can be done in CALLER by adding another DATA element to each command description and dropping temporary privileges if indicated.

Last year we were faced with the problem of not enough small buffers on a system with 30+ jobs and in excess of 200 CCL's. While the number was sufficient at the time, the system was still growing and we had set the number of small buffers to the maximum when doing the systems. At the time, it was decided that regardless of the solution decided upon it was absolutely necessary that it meet the following requirements:

1. Any additional system overhead must be negligible.
2. The change must be transparent to the users.
3. Abbreviated commands were to be allowed.
4. The command must execute the same as a CCL.

The final solution, arrived at mainly due to the necessity of not adding any overhead to the system, was to create a new run-time system which contained the CCL commands and also served as a keyboard monitor. It was made permanently resident and took 2K words of memory. A minor change to the LOGIN program and it became the job's private default run-time system at login time. All of the above requirements were met and upon benchmarking its efficiency, we found it to be faster and used less CPU time than a normal CCL.

Anyone interested in this approach may write to us. We are also enclosing our renewal subscription for the forthcoming year. The articles in the RSTS Professional are excellent and I look forward to seeing it 6 times annually.

Keep up the good work.
D.D. (Bud) Mundy, President
DMD Computer Consultants Inc.
Agincourt, Ontario

Thank you for the honorarium I received for my article in the Dec. 1981 RSTS Professional. It was totally unexpected. As contributing to your magazine was a group effort, I have given this honorarium to my company's Children's Hospital of Pittsburgh charity drive.

Once again, thank you for letting us participate in the RSTS Professional. I look forward to working with you again in the future.
David Froble
Senior Technical Consultant
Transcomm Data Systems, Inc.

I am very grateful for the first copy of the journal 'RSTS Professional' which I have just received. It is indeed a very impressive journal and I am sure that it will serve us well in our work.

Dr. S. Ron, Head of Institute
Tel-Aviv Univ.
Tel-Aviv, Israel

We're here to be of service, Dr. Ron.

P.S. See you at this year's DECUS UK Commercial SIG?
Here we go again!!
P.S. Yes. (And thanks for that great ad above.)

April 1982

The Rsts Professional

DO WHAT SO MANY HAVE DONE:

COME TO SATURN FOR THE ANSWERS

• Faster and more powerful than the stand-alone
• Faster and friendlier than other computer based systems
• More features to solve your word and list processing problems
• Extensive help commands
• Saturn runs on RT-11/TSX, RSX, RSTS, IAS, VMS
• Saturn's ASCII Format interfaces to standard DEC files
• Assembler Language provides maximum speed and efficiency
• User Friendly Prompting helps you master word processing with minimal training
• Extensive List Processing: Forms, prompted data entry, and sort/selector; plus flexible merging of lists and documents

WP SATURN FEATURES

FILE FUNCTIONS: Automatic backup • Users ASCII source files • File directories location, type, date • File Merge • Password security • Print formatted file • Review formatted file • Store formatted file
SCREEN EDITOR: Advance-Forward & Backward • Automatic Carriage Return • Column Operations • Cops Cut, Delete • Cursor Controlled • Display of Operator Actions • Formatted Text Displayed • Vertical and Horizontal Scroll • Load from another document • Paste • Prompts • Replace-Global or Selective • Search
TEXT FORMATTING: Automatic Hyphenation, Formula, User defined dictionary • Automatic Index, Table of Contents • Automatic Outline Formulas, Outline, Decimal • Centering • Date Insertion • Headers & Footers • Justification • Line Indent • Line Spacing • Multilevel Superscripts & Subscripts • Overstrike • Underscore • Redline Format Values • Reformatted Without Retyping • Table Protect (no page breaks) • Tabs, Left, Right, Any character • Variable Spacing • Word Wrapping • Footnotes
LIST PROCESSING: Form Generation • Prompted Data Entry • Multi-Key Sort/Select/Exclude • Record Search • Sequential, Binary • Unrestricted Field Insertion • Field Replacement, Global, Selective

For additional information call toll free 1-800-328-6145

SATURN SYSTEMS
6875 Washington Avenue South, Suite 218, Edina, MN 55435
Call (612) 944-2452

CIRCLE 26 ON READER CARD

Move into the Rsts Range

Thanks for a great magazine.
Nick Wright & Mark Itzcoviz
Piccadilly Computer Services Ltd.
Stammore, Middlesex, UK

P.S. See you at this year's DECUS UK Commercial SIG?
Here we go again!!
P.S. Yes. (And thanks for that great ad above.)

LETTERS... continued from page 71

Send check to:
The RSTS Professional
Box 361
Ft. Washington, PA 19036

PAYMENT MUST ACCOMPANY ORDER

April 1982
NEW 11/70 priced to move quickly. Call for specs (812) 479-6951.

GOOD DEAL on DECNET/E software license. Call Online Data Processing, Inc. (509) 484-3400.

FOR SALE Two RL02's in H9642 cabinet, controller, bootstrap ROM,11 cartridges. 1 year old. DEC maintained, DEC deinstalled. Available immediately. Asking $7000. Call Bret (814) 849-3057.

TIMESHARING AVAILABLE on PDP 11/70 year old. DEC maintained, DEC deinstalled.

FOR SALE Two RL02's in H9642 cabinet, Share in Indiana (317) 448-1686.

RSTS. Software also. Call Wabash Time (317) 636-3223.

April 1982 page 93
NEW 11/44 priced to move quickly. Call for spec (812) 479-6951.

LIST OF ADVERTISERS

Able Computer .......... I.B. Cover
Amcor Computer Corp .. . pp.39-41, 43, 45
Associated Computer Engineers p.69
Associated Computer Technique p.49
Bob Macro Man Meyer p.17
Braingen MPG p.29
C.D. Smith & Associates, Inc. p.36
Cashier Associates, Inc. p.85
Cyde Digital Systems p.2
Com/Design, Inc. p.81
Data Processing Design, Inc. p.52
Data South Computer Corp. p.52
Datatarn Corp. p.96
Datavoice Inc. p.76
Digital Associates Corp. p.87
Digital Management Group p.5
Distributed Logic Corp. p.33
ECC Systems p.40
Emulex Corp pp.19, 21
Enterprise Technology Corp. p.31
EPS Inc. p.84
Evans Griffiths & Hart, Inc. pp.33, 53
FAT Systems Ltd. p.12
Florida Computer, Inc. p.95
IAS Computer Corp. p.78
Infinity Software Corp. p.28
Innovative Data Technology p.70
Interactive Management Systems p.63
Manus Services Corp. p.86
Martin Marietta Data Systems p.37
McHugh, Freeman & Associates, Inc. p.59
MDI Systems, Inc. p.25
MIT p.27
Nassau Systems p.98
National Peripherals, Inc. p.65
NationalData Company pp.10, 50, 57, 60
North County Computer Services, Inc. pp.40, 49, 80
On-Track Systems, Inc. pp.20, 50
Plycom Services, Inc. p.64
Raxco Inc. pp.73, 77, 79, 81
Ross Systems, Inc. I.F. Cover
RTZ Computer Services, Ltd. p.13
RUF Corp. p.17
Saturn Systems, Inc. p.52
Scheuer's Mini Computer Cart p.67, 92, 71
Software Results Corp. p.30
Software Techniques, Inc. p.73
Southern Systems, Inc. p.52
SSPL Inc. p.10
SSP Timeshare p.91
System Industries p.9
System Performance House Inc. p.7
Telecom Computer Systems, Inc. p.47
TransNet Corp. p.79
Unistrone Corp. p.21
W9T Systems, Inc. p.61
Xiren Computing, Ltd. p.23

For further information on WORKER or FINAR, call or write: Michael Hulme, Finar Systems Ltd., 6000 E. Evans, Suite 2-300, Denver, CO 80222, (303) 758-7951.

January, 1982

INTERACTIVE SYSTEMS AND SOFTWARE INTRODUCES EXECUGRAPH

Danvers, MA — ExecuGraph is an inexpensive software package which produces bar charts, line drawings, pie charts and histograms, has been introduced by Interactive Systems and Software, Inc. of Danvers, MA.

Graphs and charts summarize masses of written data into graphic visual form, from which management can make faster and more intelligent decisions. Types of information which ExecuGraph can translate from staid written form to exciting graphic form include current and historical information, trends in sales, profits, growth and marketing efforts.

The presentation prepared by ExecuGraph can be displayed on a CRT type terminal or chart plotter. Depending on output medium, these graphs can be displayed in several colors.

A person does not need programming skills to operate ExecuGraph. A brief question and answer dialogue will produce full documented graphs.

ExecuGraph is capable of graph selection and storage, storage and retrieval of user-defined graph formats and searched data entries for use with user-defined formats. It is also capable of interface definition for external user programs and composite graphs (e.g. forecast vs. actual).

The program is written entirely in FORTRAN-IV and is capable of operating on any computer which has a minimum of 28kw of memory space and graphical output device. The program is supplied with user documentation and examples.

Execugraph is accompanied by a 90 day warranty, an annual subscription update is available for 10% of the license fee.

Hardware required for ExecuGraph is a digital computer system with 28kw program space, a disk operating system with FORTRAN-IV and a graphic output device (plotter, graphic terminal).

Execugraph is offered under a license agreement from Interactive Systems and Software, Inc., P.O. Box 348, Danvers, MA 01923, (617) 774-6703.

February, 1982

ADVANCED VIDEO GRAPHICS ADDED TO RAXCO'S VAX PERFORMANCE ANALYSIS SYSTEM

Atlanta, GA — RAXCO, Inc. announces new graphic performance analysis capabilities by utilizing advanced video features of VT100 terminals under RABBIT-2. RABBIT-2 is RAXCO's interactive performance analysis software for VAX/VMS and RSTS/E environments.

The new features include automatic scaling to maximum height and width of the VT100 terminals, vertical and horizontal bar graphs, reverse image and bar selections. RABBIT-2 will also superimpose multiple graphs on the same display which often makes relative data more meaningful.

RABBIT-2 is a software tool for system managers and operational managers to investigate system bottlenecks, resource consumption, user activities, and program efficiencies. It may be utilized in an interactive, diagnostic mode or through a batch command file. An English-like command language is employed. Graphic output may be directed to a terminal or line printer. RABBIT-2 is priced at $2495 for RSTS/E and $3995 for VMS systems available at $999 and $200/month respectively. RABBIT-2 is sold and supported throughout the U.S.A., Canada and the U.K.

RAXCO provides a wide range of operational software for DEC computer including Resource Accounting, Job Accounting, Data File Security, Data Management and Financial Planning.

For more information contact: RAXCO Inc.,
EMULEX ANNOUNCES 8 MASSBUS COMPATIBLE SUBSYSTEMS FOR END USERS OF VAX 11/750 COMPUTERS

Santas Ana, CA - Emulex Corporation has announced the availability of the first true Massbus compatible storage subsystems for use with DEC VAX 11/750 computers.

The first five include Emulex's recently introduced SC750 disk controller with both fixed and removable media disk drives with capacities ranging from 80 to 675 megabytes. Since functionally emulates the DEC RH7/50 Massbus adaptor and the peripheral Drive Control Logic, it is transparent to VAX software, including UNIX, VMS and DEC diagnostics.

The new subsystems and their list prices are:

- an RM03 equivalent using an 80 MB removable media drive, priced at $17,900;
- an RM05 equivalent using a 300 MB removable media drive, priced at $25,900;
- an RP07 equivalent subsystem using a 675 MB Winchester disk drive, priced at $36,550;
- an RM60 equivalent subsystem using a 160 MB Winchester disk, priced at $17,900;
- and an RM03 equivalent subsystem using an 80 MB Winchester disk drive, priced at $16,200.

All of the subsystems are available now for 30-45 day delivery. Installation is available directly from Emulex and arrangements are being made to include all SC750 subsystems under the existing service agreements between Emulex and Control Data Corporation.

"These new packages represent the first, and only, true Massbus compatible systems available for use with the VAX 11/750," explained Phillip (Flip) Begich, Emulex director of national marketing. "They are not available from anyone else, and for the first time, they provide 11/750 users with an even wider range of disk storage capacities than is available from DEC for either the 11/750 or 11/780.

All of the subsystems employ the SC750 controller which is contained on a single extended hex-size printed circuit board and mounts in one of the three RH750 Massbus adapter slots in the 11/750 backplane. The controller interfaces directly with the 32-bit CPU Memory Interchange (CMI) bus of the 11/750. It contains the memory mapping and registers of the RH750 as well as all the disk control logic for up to eight logical drives.

"We also have an RP06 emulation on the SC750 controller to support the use of a 200 MB drive as an RP06 subsystem, but we do not offer that as a stand-alone subsystem," Begich noted. "The emulation is available, however, and we can make arrangements with one of our dealers/distributors to supply that subsystem package to users who desire it.”

Emulex also offers VAX Unibus versions for all the disk subsystems, using its SC21 controller, for both the 11/750 and 11/780. Most users would probably employ the new Massbus compatible subsystems, however, because the pricing is so close between the two versions that most people would prefer the higher performance and full software transparency gained with the SC750 controller," Begich said. "On the 11/780, however, there is a substantial price difference between the Unibus and Massbus compatible subsystems.

The SC750 subsystems are the latest in a series of products intended specifically for use with DEC VAX computer systems. The SC11/U and SC21/U communications multiplexers provide a DH11 equivalent capability to speed up asynchronous communications on VAX systems.

The SC21/Z multiplexer is also available to VAX users. This product emulates the D211 communications subsystem and provides complete software transparency under VMS, UNIX, and DEC diagnostics. In addition to these communications products, Emulex also offers the TC11/V tape controller which lets users connect any standard half-inch, reel-to-reel 800 or 1600 bit per inch tape drive to any VAX system.

Emulex Corporation, based in Santa Ana, is the leading supplier of disk, tape and communications controllers for use in interfacing a wide variety of peripherals and communications devices to computers made by Digital Equipment Corporation. The company's new Systems Group has also designed and installs complete LS1-11, PDP-11, and VAX 11/780 systems ranging in size from 80 to 675 megabytes, with service provided by Control Data Corporation.

February, 1982

EGH RELEASES V.16 OF DIALUP

Lexington, MA - Evans Griffiths and Hart, Inc. (EGH) announces the release of Version 16 of DIALUP, an asynchronous communication package running under DEC PDP-11 and under the ROSS/V RSTS/E simulator on the VAX. DIALUP uses a standard asynchronous terminal link to link its host system to a remote computer system either via a telephone line or via a null-modem connection (if the systems are close enough together).

DIALUP supports user-directed dialing through automatic calling units, virtual connection of the user's terminal to the remote system, transmission of ASCII text files and from and to the remote system, and, if the remote system is RSTS/E or VMS with ROSS/V, block-mode transmission of binary files (or pieces of files) with CRC16 block checking, block-level retries, and the preservation of attributes.

The binary file transfer module, which is easily bootstrapped to the remote system, is written in machine language. Although machine-language code instead of BASIC-PLUS doesn't reduce RSTS/E monitor overhead for processing asynchronously transmitted data, it does reduce user-mode time by a factor of six and overall CPU time by a factor of 2.5, raising transmission throughput. This is particularly important when two adjacent computers are connected by a null modem operated at 9600 BAUD.

In Version 16 of DIALUP, support has been added for VADIC-style multi-line autoloaders (both DN11- and RS232-controlled), and the binary file transfer module has been enhanced to support the use of command files. DIALUP may thus be used to transfer large quantities of data to and from a remote system without someone having to be present. Also included in Version 16 is a rudimentary command language that allows branching on errors, chaining to and from other programs, and waiting with retries, when, for example, a phone line is temporarily busy. The command language also allows the definition and execution of macros that specify complex sequences of DIALUP commands. Frequently dialed telephone numbers can also be saved as macros.

Version 16 contains several new modules that simplify macro definition in DIALUP. One of these, especially intended for new users of DIALUP, guides the user through the definition of one or more macros that may be used to simplify establishing a connection with a remote system.

First released in 1980, DIALUP is currently in use at over 40 sites. The price of a single CPU license is $1,700.00. For further information, contact Evans Griffiths and Hart, Inc., 151 Waltham St., Lexington, MA 02173. (617) 861-0670.
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

---

<table>
<thead>
<tr>
<th>CONTROLLER</th>
<th>DESCRIPTION</th>
<th>COMPATIBILITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>C03</td>
<td>Cartridge disk controller</td>
<td>RK05</td>
</tr>
<tr>
<td>C33</td>
<td>Cartridge disk controller</td>
<td>RK05</td>
</tr>
<tr>
<td>T03</td>
<td>NRZI mag tape controller</td>
<td>TM11/TU10</td>
</tr>
<tr>
<td>T04/N</td>
<td>NRZI mag tape controller</td>
<td>TM11/TU10</td>
</tr>
<tr>
<td>T04/D</td>
<td>Dual density mag tape controller</td>
<td>TM11/TU10</td>
</tr>
<tr>
<td>T34/N</td>
<td>NRZI mag tape controller</td>
<td>TM11/TU10</td>
</tr>
<tr>
<td>T34/D</td>
<td>Dual density mag tape controller</td>
<td>TM11/TU10</td>
</tr>
<tr>
<td>T36</td>
<td>Dual density mag tape controller</td>
<td>TM11/TU10</td>
</tr>
<tr>
<td>S03/A</td>
<td>80MB/300MB SMD controller</td>
<td>RM02/RM05</td>
</tr>
<tr>
<td>S03/A1</td>
<td>160MB SMD controller</td>
<td>RM02</td>
</tr>
<tr>
<td>S03/B</td>
<td>80MB/300MB SMD controller</td>
<td>RK07</td>
</tr>
<tr>
<td>S03/C</td>
<td>200MB/300MB SMD controller</td>
<td>RP06</td>
</tr>
<tr>
<td>S03/D</td>
<td>96MB CMD controller</td>
<td>RK06</td>
</tr>
<tr>
<td>S33/A</td>
<td>80 MB/300 MB SMD controller</td>
<td>RM02/RM05</td>
</tr>
<tr>
<td>S33/A1</td>
<td>80 MB/160 MB SMD controller</td>
<td>RM02</td>
</tr>
<tr>
<td>S33/B</td>
<td>80 MB/300 MB SMD controller</td>
<td>RK07</td>
</tr>
<tr>
<td>S33/C</td>
<td>200 MB/300 MB SMD controller</td>
<td>RP06</td>
</tr>
<tr>
<td>S33/D</td>
<td>96 MB CMD controller</td>
<td>RK06</td>
</tr>
</tbody>
</table>

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.
Thousands sold worldwide since 1978

Super Max

The single-board, 16-line ABLE DH/DM™ that enhances any UNIBUS system with DH-performance at DZ-prices.

A few years ago, we broke new ground with our DMAX/16™, the original alternative to the DEC DH11. DMAX cut the space requirements from nine slots to three and became an immediate worldwide success. Now we've come up with something even better. This time it's ABLE DH/DM™, an alternative that achieves the optimum cluster size — 16 lines with modem control on a single board. You can compare MTBF, price and throughput and find that ABLE DH/DM beats everything in its class. No one else comes close.

The ABLE DH/DM™ is today's answer to VAX system needs for DMA communications multiplexing and serves all standard UNIBUS systems equally well. Each 16-line ABLE DH/DM™ installs in any standard hex-width slot at only one unit bus load and is DH11 compatible to the diagnostic level. Just plug it in and see it run — up to 19.2K baud using only half the UNIBUS bandwidth of a DEC DH11.

Key ABLE DH/DM™ features include on-board diagnostics with LED display, modem control on all lines, improved on-board silo depth and variable PROM set for proprietary OEM applications.

Keep up with ABLE and optimize your VAX, PDP-11 or System 20. Write or call today for details on our full line of UNIBUS-compatible special-memory, general-purpose and data-communications products plus the MAGNUM™ computer series!

*Now enhances VMS operation!

ABLE the computer experts

ABLE COMPUTER, 1732 Reynolds Avenue, Irvine, California 92714. (714) 979-7030. TWX 910-595-1729 ACT IRIN.

ABLE COMPUTER, ABLE Computer House, London Road, Newbury, Berkshire, England RG13 2QJ. 44(0635) 32125. TELEX 848715 ABLE G.

ABLE COMPUTER GmbH, Forsthausstrasse 1, 8013 Haar (Near Munich), West Germany. 49 089/463080, 463089. TELEX 05213883 ABLE D.

DEC, UNIBUS, VAX and PDP are trademarks of Digital Equipment Corporation.
The Unabridged Addition.

No matter how many additions you’ve made to your VAX, it won’t be complete without WORD-11™. It’s the sophisticated word processing system designed to help you use all the other information you’ve stored. With features like list processing, built-in dictionaries for spelling error detection, automatic table of contents and footnoting that make report writing a snap.

WORD-11 runs concurrently with data processing on multiple terminals. It’s easy to use. And it’s been up and running in hundreds of installations all over the world for years.

WORD-11 is easy to install and operate. It’s cost effective. And it’s supported by a dedicated team of experienced specialists. It could be the most important addition you make to your VAX installation. For details, Call, (714) 993-4160.

WORD-11 is also available for Digital’s RSX-IIIM, RSX-IIIM PLUS™ and RSTS/E™ operating systems.

$10,000.00 COMPUTER DISTRIBUTION

CORPORATE OFFICE
181 W. Orangethorpe, Suite F
Placentia, CA 92670
714-993-4160 Telex 882-278

N Y OFFICE
420 Lexington Avenue, Suite 633
New York, NY 10170
212-687-0904

WASHINGTON D.C OFFICE
4520 East-West Highway, Suite 550
Bethesda, MD 20814
301-657-4098

WORD-11 is a trademark of Data Processing Design, Inc.
VAX, RSX-IIIM, RSX-IIIM PLUS, and RSTS/E are trademarks, and remaining titles are products or trademarks of Digital Equipment Corporation.

CIRCLE 59 ON READER CARD