DECUS

# The DeVIAS Letter

SIG IAS DECUS U.S. CHAPTER

SIG RSX DECUS U.S. CHAPTER

# The RSX Multi-Tasker

April 1984

Printed in the U.S.A.

## IAS

## Did not submit material for this issue

Send your submissions for the next issue to:

**IAS NEWSLETTER**

John W. Drummond
Ontario Hydro
700 University Ave.
Toronto, Ontorio
Canada M5G 1X6

The following are trademarks of Digital Equipment Corporation:

| | | |
|---|---|---|
| DEC | DIBOL | PDT |
| DECnet | Digital Logo | RSTS |
| DECsystem-10 | EduSystem | RSX |
| DECSYSTEM-20 | IAS | UNIBUS |
| DECUS | MASSBUS | VAX |
| DECwriter | PDP | VMS |
| | | VT |

UNIX is a trademark of Bell Laboratories.

Copyright © Digital Equipment Corporation 1984
All Rights Reserved

It is assumed that all articles submitted to the editor of this newsletter are with the authors' permission to publish in any DECUS publication. The articles are the responsibility of the authors and, therefore, DECUS, Digital Equipment Corporation, and the editor assume no responsibility or liability for articles or information appearing in the document. The views herein expressed are those of the authors and do not necessarily express the views of DECUS or Digital Equipment Corporation.

# RSX MULTI-TASKER

## TABLE OF CONTENTS

## From the Editors

The RSX Woods meeting was held in Boston on March 1 and 2. At that time Ralph Stamerjohn resigned as chairman of the RSX SIG. On behalf of Allen and myself, I would like to thank Ralph for his leadership during the past months and his continuing help and guidance (he is remaining on the Steering Committee). The new SIG chairman is Terry Medlin. Liz Bailey has accepted the newly created position of vice-chairman. A more detailed report on the RSX executive board meeting will appear in a future issue.

The Multi-Tasker is looking for columnists. If you have an idea for a column or would like to write one, please contact the editors. Note: a column need NOT appear in every issue. If you would prefer to write a column every other month that´s fine too. A column we are starting in this issue is "It´s on the SIG Tape". Twice each year, a treasury of free, high quality software is distributed by the SIG. Help us sort through this storehouse. If you are using a program from the tape, let us know, if you encountered problems or work-arounds the membership can use this information.

I´ve saved the toughest subject for last: the Multi-Tasker subscription fee. A lot is going on within DECUS concerning publication fees. No one wants them, but it is a fact of life. Don´t be alarmed, no company will become overextended by subscribing (about $20 per year) to this Journal. Whether or not this is high depends upon what the Multi-Tasker is worth to you. If the Multi-Tasker helps you only one time during the year, answers one question, or saves a few hours of research, the cost of a subscription has paid for itself many fold. If you are a user of RSX, the Multi-Tasker doesn´t cost money, it saves money.

## Help Yourself

The Editor
The Multi-Tasker
C/O One Iron Way
MR2-3/E55
Marlboro, Ma 10752

Dear Sir,

At the Spring DECUS meeting, we are going to have a session called "Just a Modest Proposal". This letter is a call for RSX users to participate in this session. The abstract is as follows.

> Let´s face it, at one time or other each of us has thought (privately, to be sure!) that DEC did it all wrong, or really doesn´t understand the needs of our particular interest group, or is missing the opportunity for a great product (hardware/software). Well, here is your chance to present to DEC and the RSX community Your Very Own Modest Proposal. Proposals should be limited to about 5 - 10 minutes in length. Viewgraphs, slides, movies or any other presentation aid are encouraged. The entries may be serious, or not so serious. A distinguished panel of DEC Developers, Product managers and the attendees will judge the proposals in three categories; 1)"Noted" ie DEC likes it, 2)"UPG" ie DEC deserves it, 3) "Potential for Cult Following" ie the RSX community likes it.

The session is not designed to be a "gong" show, or a forum for beating up on DEC. It is, however, a chance for those of us who must exist in a Love/Hate relationship with RSX, to present those "If only...." ideas, or perhaps some thoughts that gently poke fun DEC (Remember UPG?). The success or failure of this session, is up you. I have a few ideas (some serious, some funny) but certainly not enough to fill up a session.

Sincerely,

James Downward
KMS Fusion, Inc
P.O. Box 1567
Ann Arbor, MI 48106
(313)-769-8500

February 6 1984

National Research Facility
for Submicron Structures
Knight Laboratory
Cornell University
Ithaca, New York 14853

Multi-Tasker
DECUS
MR02-3/E55
One Iron Way
Marlboro, MA 01752

In answer to question 2 from the November issue, DUPLEX does have many problems. I have used a similar program written at Cornell for ´Virtual Terminal´ operations between an 11/60 and 11/34, and found that it is completely faster than 1200 baud. Fortunately, there is another alternative. Kermit is a program written originally at Columbia University, which provides file transfer and virtual terminal support for DEC-20, VMS, RT, IBM, and other mini/mainframe environments, in addition to supporting many micros. There should be a new RSX implementation on the Fall 83 RSX SIG tape.

I´ve used the VMS version for virtual terminal support (including RMD, EDT, etc.) to an 11/34 running RSX, and it has been successfully used for file transfers between VMS systems, VMS to/from DEC-30, etc.

If you need something sooner (or Kermit-RSX does not work), I have a package called CADnet which works fairly well for RSX and VMS systems.

Sincerely,

Dennis P. Costello
Computer System Manager

November 29, 1983

RCA Service Company/Government Services
Box 3935 U.S Naval Station
F.P.O
Miami, Florida 34051
(809) 865-7314

Multi-Tasker
DECUS
One Iron Way
Marlboro, MA 01752

Dear Multi-Tasker:

Please include the following in your Multi-Tasker Help Yourself Column

Ramtek 9400 Cross Assembler

We need a cross assembler for a Ramtek 9400 display generator which will run under RSX-11M. The assembler should accept Ramtek mnemonics and create binary or symbolic data record (file) which can be incorporated by reference into a FORTRAN program.

The intent is to generate Ramtek display commands or display lists which will then be manipulated as data within the FORTRAN program and output to the display device as appropriate.

Please contact:

Linda A. Slawson
RCA Service Company
Box 3935 USNS
FPO Miami, Fl 34051

Thank You.

Sincerely yours,

Linda A. Slawson
Manager, Software Support

## Questions from the North Texas LUG Newsletter V7.0

Jeanine McDermott of Electrospace Systems wishes to know if there is a program somewhere called TIDY that is used to reformat FORTRAN files to make them look consistent and pretty. (There is a program on one of the SIG tapes called RESEQ which will resequence FORTRAN source files to provide consistent labeling in routines)

If you have SORT-11 or COBOL-81 for RSX-11M, please contact Bill Hancock of SOHIO. He would like to talk with you.

To get in touch with the North Texas LUG contact:

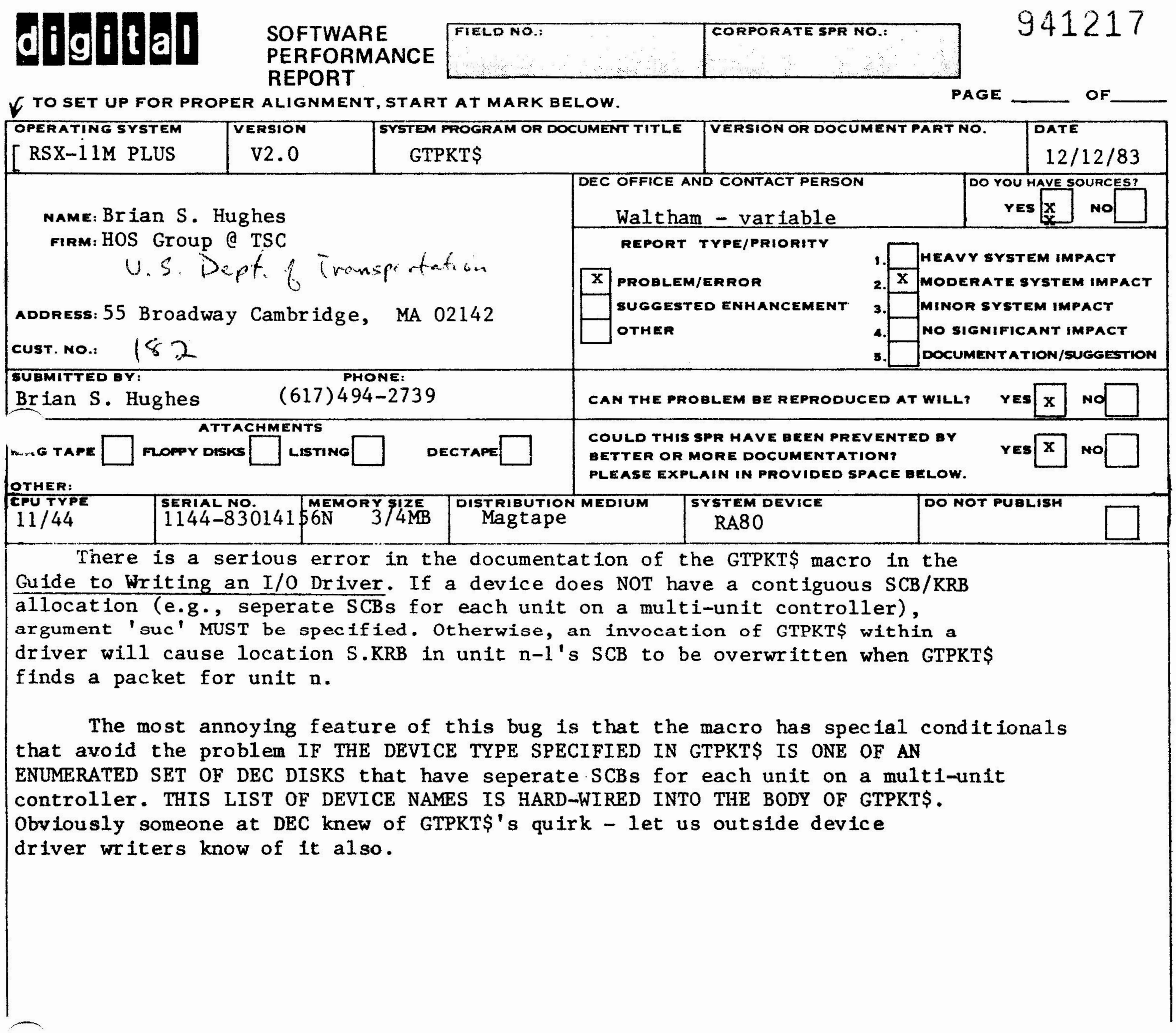
Mike Drabicky
Rockwell International
1200 N Alma Way MS 406-280
Richardson, Texas 75081
(214) 996-7532

## Error in documentation for GTPKT$ macro

Brian S. Hughes
HOS Group @ TSC
U.S. Dept. of Transportation
55 Broadway
Cambridge, MA 02142

There is a serious error in the documentation of the GTPKT$ macro in the _Guide to Writing an I/O Driver._ If a device does NOT have a contiguous SCB/KRB allocation (e.g. separate SCBs for each unit on a multi-unit controller) argument ´suc´ MUST be specified. Otherwise, an invocation of GTPKT$ within a driver will cause location S.KRB in unit n-1´s SCB to be overwritten when GTPKT$ finds a packet for unit n.

The most annoying feature of this bug is that the macro has special conditionals that avoid the problem IF THE DEVICE TYPE SPECIFIED IN GTPKT$ IS ONE OF AN ENUMERATED SET OF DEC DISKS that have separate SCBs for each unit on a multi-unit controller. THIS LIST OF DEVICE NAMES IS HARD-WIRED INTO THE BODY OF GTPKT$. Obviously someone at DEC knew of GTPKT$´s quirk -- let us outside device driver writers know of it also.



digital SOFTWARE PERFORMANCE REPORT

FIELD NO.:

CORPORATE SPR NO.:

941217

PAGE ______ OF______

TO SET UP FOR PROPER ALIGNMENT, START AT MARK BELOW.

| OPERATING SYSTEM | VERSION | SYSTEM PROGRAM OR DOCUMENT TITLE | VERSION OR DOCUMENT PART NO. | DATE |
|---|---|---|---|---|
| RSX-11M PLUS | V2.0 | GTPKT$ | | 12/12/83 |

NAME: Brian S. Hughes
FIRM: HOS Group @ TSC
U.S. Dept. of Transportation
ADDRESS: 55 Broadway Cambridge, MA 02142
CUST. NO.: 182

DEC OFFICE AND CONTACT PERSON: Waltham - variable

DO YOU HAVE SOURCES? [✓ YES] NO

REPORT TYPE/PRIORITY
- [✓ PROBLEM/ERROR]
- SUGGESTED ENHANCEMENT
- OTHER

1. HEAVY SYSTEM IMPACT
2. [✓ MODERATE SYSTEM IMPACT]
3. MINOR SYSTEM IMPACT
4. NO SIGNIFICANT IMPACT
5. DOCUMENTATION/SUGGESTION

SUBMITTED BY: Brian S. Hughes PHONE: (617)494-2739

CAN THE PROBLEM BE REPRODUCED AT WILL? [✓ YES] NO

ATTACHMENTS: MAG TAPE FLOPPY DISKS LISTING DECTAPE
OTHER:

COULD THIS SPR HAVE BEEN PREVENTED BY BETTER OR MORE DOCUMENTATION? PLEASE EXPLAIN IN PROVIDED SPACE BELOW. [✓ YES] NO

| CPU TYPE | SERIAL NO. | MEMORY SIZE | DISTRIBUTION MEDIUM | SYSTEM DEVICE | DO NOT PUBLISH |
|---|---|---|---|---|---|
| 11/44 | 1144-83014156N | 3/4MB | Magtape | RA80 | |

There is a serious error in the documentation of the GTPKT$ macro in the Guide to Writing an I/O Driver. If a device does NOT have a contiguous SCB/KRB allocation (e.g., seperate SCBs for each unit on a multi-unit controller), argument 'suc' MUST be specified. Otherwise, an invocation of GTPKT$ within a driver will cause location S.KRB in unit n-1's SCB to be overwritten when GTPKT$ finds a packet for unit n.

The most annoying feature of this bug is that the macro has special conditionals that avoid the problem IF THE DEVICE TYPE SPECIFIED IN GTPKT$ IS ONE OF AN ENUMERATED SET OF DEC DISKS that have seperate SCBs for each unit on a multi-unit controller. THIS LIST OF DEVICE NAMES IS HARD-WIRED INTO THE BODY OF GTPKT$. Obviously someone at DEC knew of GTPKT$'s quirk - let us outside device driver writers know of it also.

## Library News

Recent Submissions to the DECUS Library for RSX

Allen Watson, Editor
From weekly DECUS Library reports

The following lists a few things that have appeared in the weekly reports I receive about new library submissions and changes in older programs in the library. Program catalog numbers are referred to in the form "11-nnn".

11-260 SRD -- No longer available; out of date version.

11-655 Tektronix Emulator for RSX-11 -- New program as of March, 1983. Requires FORTRAN and MACRO-11, plus standard VS11 RSX-11M Device driver. Allows users of Tektronix 4014's to run their existing software on the VS11 with its advantages of extra performance, flexibility and color.

11-SP-50 Preliminary "C" -- This version of the "C" language is being confused with the latest release of the "C" language system, DECUS No. 11-SP-18. Therefore it is being removed as a library offering.

11-SP-47 Portacalc -- Glenn Everhart spreadsheet for RSX, IAS, and VMS. Later versions on SIG tapes from Las Vegas; see also patch from Glenn in this issue.

11-SP-56 Portacalc XL: Floppy Version -- for IAS, RSX.

11-597 DTC: Desk Top Calendar -- from Mitch Wyle and Glenn Everhart.

11-549 File Management Utility (INQ and CTL) -- Author: Jean-Paul Denis, Brussels, Belgium. Uses Files-11 but allows record access by number or by key, changes to records, hard copy, record selection on print or copy, etc. Menu-driven. Currently under evaluation and not yet available.

11-679 SPELL for RSX11M -- Spelling error checker by Jeff Hamilton.

11-680 RSX-11M-PLUS System Accounting Reports with DTR -- All the Datatrieve structures presented in the January, 1984 Multi-Tasker, plus other helpful information on using them.

PRO-103 C Language System (binary) for RT-11 on PRO/300 -- This is included here because the very fact of its existence should be of interest to people using DECUS "C" on a PDP-11 or VAX.

PRO-104 RUNOFF M02.4H for RT-11 on PRO/350 -- same reason for inclusion as above.

## PRO/300 Software

There is a fair amount of software being submitted to the library for the Professional 300 series; much of it seems to be based on RT-11. Readers interested should contact the DECUS library directly for more information. Entries include TECO V36, a SORT, VT100 library, FORTRAN cross-referencer.

11-SP-47 Portacalc: A 3D spreadsheet/database program -- Revision as of February 13, 1984, from Glenn Everhart. For IAS, RSTS/E, RSX (all), and VMS. Faster screen display; entire new PDP-11 version with all capabilities formerly in VAX version only; matrix math; several bug fixes.

11-683 RUNOFF for RSX-11 and RSTS/E -- works on VMS in compatibility. From Charles Spalding III; reviewed in last two issues of Multi-Tasker.

11-SP-60 Symposium tape for RSX SIG, Fall 1983, Las Vegas -- See previous Multi-Taskers for another partial listing of contents. If you can't get this tape through a LUG you can purchase it from the library.

Highlights of 11-SP-60 are:

Swedish Pascal updated for RSX-11M V4.0 and V4.1, FCB program fro RSX-11M-PLUS V2.0, CRT library for VT100 control, RUNOFF from the RSX SIG working group, Multi Trek (multi-user Startrek game), RSX-11M/M-PLUS Activity monitor, written for V3., COMLIB module patch to get BRU to write files from tape to VAX disks. DGT - read DG, Unix TAR, IBM, many more tapes; writes some too. FIXED up ORC object disassembler. Resubmission of IAS virtual disks and pseudo tty drivers. Multi-file or multi-disk virtual disk package with security enhancements. FX: memory virtual disk package for M/M+. Desk Top Calendar. Appt/Mtg sched, cal. display. Spelling checker, signal processing programs, tape copier, SCCS, day-of-week subroutine and more.

Rice University RUNOFF (John Clement). Many word processing enhancements and letter quality output support. Similar to DSR. Interr-interrogate a central DECnet node for new network control info in big networks. RSX-11M/M+ User monitor. network time coordinators. KMSKIT (RSX accounting and performance enhancements). CCL V9.0 Concise Command Language...better catchall task, very user friendly. Machine-readable versions of Hows and Whys of ASTs in RSX; MSCAN; Cluster and Resident Libraries; and RSX SYSGEN session. Online Pool analyzer for RSX-11M V.

SRD working group SRD. This is the definitive version of SRD, a useful directory/file maintenance utility.

Manuscript and files for developing an ACP for RSX-11M in a Higher Order Language. Supermac in FORTRAN. Datatrieve structures needed to make RSX-11M+ accounting reports. SFGL70 graphics package enhancements. Complete distribution of KERMIT communications for many micro, mini and mainframe systems (1/12/84, fixes bugs of 10/21/83 version of VAX Fall ´83 tape). Very complete comm program, runs on DEC, MSDOS, CP/M, Unix, VM/370, Apple, etc., etc. All sources included. DG ADS Kermit (in RATFOR and FORTRAN). RSX and RSTS kermit.

Adventure compiler and runtime for RSX. 3D Plotting package for Tektronix 4014. TED full screen and line editor.

## It's on the SIG Tape

### EMPIRE

EMPIRE is a simulation of a full scale war between two emperors, the computer and you. Naturally, there is only room for one, so the object of the game is to destroy the other. The computer plays by the same rules that you do.

The map is a rectangle 600 by 1000 miles. The resolution is 10, so the map you see is 60 by 100 miles. The map consists of sea='.', land='+', uncontrolled cities='*', computer controlled cities='X' and your dominated cities='0. Each emperor gets 1 move per round and moves are done sequentially.

The map is displayed on the player's screen during movement. Each piece is represented by a unique character on the map. With a few exceptions, you can only have ONE piece on a given location. On the map, you are shown only 8 squares adjacent to your units. This information is updated before and after every move. The map displays the most recent information known.

The game starts by assigning you one city and the computer one city. Cities can produce new units. Every city that you own produces more pieces for you according to the cost of the desired unit. The typical play of the game is to issue the Automove command until you decide to do something special. During movement in each round, the player is prompted to move each piece that does not otherwise have an assigned function.

This game is available on the Fall 83 RSX SIG tape under UIC [356,50]

This article submitted by the North Texas LUG Newsletter V7.0

## Speak Out

## Comments on RUNOFF Comparison

John Clement
T. W. Bonner Nuclear Lab
Rice University
Box 1892
Houston TX 77251
February 20, 1984

Dear Allen:

I received your letter and the copy of the comparison document between the three leading versions of RNO. I was pleased by your comments. My attitude toward this software is that we needed this product so I produced it and future developments will be slower. I might be interested in adding the few features that people want most. I probably will add the .PRINT command and investigate terminal input. The table of commands is easy to modify, to add alternate command names and spelling. I could be persuaded to perform this simple task to add more compatibility. If someone else wishes to do all of the future development I am willing to give him all the rights to do so, with of course retaining the right to act as a critic. I have some other projects that will take up most of my time in the future.

I appreciate the list of bugs you sent, and some have already been fixed. I have never seen the FOOTNOTE problem so I would appreciate an example of the problem preferably on 800/1600 BPI magnetic tape, with both input and output files. The missing "END" statements puzzle me, as there are error messages to indicate the probability of missing "END" among other errors. Some specific examples of this would also be nice.

You asked for a .COLUMNS command. This is not necessary in RNO if you use tabs in the following way. Issue the commands .FILL .NO JUSTIFY and then set the left margin to line up with the first column you wish. Set the tab stops to line up with the other columns. Put a tab AFTER each item. Items may be on separate lines or strung together on a single line. Never begin a line with a tab or space. I mention this in the documentation, but it may not have been clear.

I would like to point out a few omissions from your comparison. Some of the incompatibility may be alleviated by creating a prefix file with .DEFINE COMMAND to create the missing or differently named commands. Your comparison completely ignored the fact that my version has a set of alternate commands which are considered nonstandard. This was done to retain compatibility with the older versions of RNO. The non-standard commands are contained in an appendix to the Bonner Lab Runoff manual. The most notable example is .[NO] HYPHENATION. I use it as a synonym for .[ENABLE/DISABLE] HYPHENATION. Digital´s DSR also can handle a variety of abbreviations (.HEADER or .HEADERS are equivalent). In addition I believe it can handle some other variants. The .STANDARD command also exists in my version. The non-standard commands and commands which I consider to be not recommended are omitted from the help file, though they appear in the .DOC file. By the way you incorrectly claimed that .RIGHT doesn´t cause a break. I must confess that the appendix "LIST of commands" may have more errors which can confuse users as it did you. You can get terminal input by .REQ if you specify a file name in addition to a device (.REQ ´TT:INPUT´). The same trick may work with DSR. The .NO PERIOD command does exist contrary to your table.

The latest version of BL4.0 has been submitted to the DECUS library. It is now RT compatible and it has a few more commands and several bug fixes. I am sending a copy of the README file to let you know what has been fixed, or enhanced. You should publish your comparison quickly before it becomes hopelessly outdated.

The following changes have been implemented and will be available on the Spring tape. These were relatively easy so I took your suggestions. The changes marked 4.0 were implemented for the DECUS version.

1. .HEADERS is the standard command, but .HEADER is still recognized. (4.0)

2. .TYPE has been added as an analog to .PRINT. The name was changed to avoid confusion since other .PRINT commands refer to the .DOC output. If this causes problems .DEFINE COMMAND may be used to define .PRINT as .TYPE.

3. .REQUIRE "TI:" is now legal.

4. Minus parameters work for .RIGHT. (4.0)

Finally I would like to let you know how pleased I am with the general reception to my version of RUNOFF. I repeat that I would be very happy to let someone else do the future development. Perhaps DEC could be persuaded to include the most desirable features in DSR. The DECUS developers should have a method of talking to the DEC developers. Many DECUS members may not be able to go to the symposia, so their voices are not heard.

Sincerely,

John Clement
Sr Research Scientist

## On RSX's 10th Anniversary

## Whither Goest the SIG?

(Fair warning to the reader: It has been some time since anyone has done this in the Multi-Tasker, so I will shortly get up on my hind legs, bark, kick the dog (or the DEC) around, and even abuse him somewhat.)

It is now comfortably past the official RSX 10th anniversary celebration at the Las Vegas Symposium (which was, as always, indeed a good time), and perhaps a little of the time in the interim I have spent in mellow reflection causes me to wonder about the directions being taken by both RSX in general and by the SIG.

RSX as an operating system is now at least 10 years old. I don't think that it is quite possible for newcomers to RSX, such as myself, to comprehend the significance of this achievement. 10 years is a long, long time in the computer world for an operating system, much less a computer architecture, to be in popular use. Yes, surely there are still users of even earlier systems such as FOCAL/PDP-8s; but here I mean in general use and widely accepted by a large community of machine owners.

This is partially due, no doubt, to the enduring excellence of PDP-11 architecture. The PDP-11 is a classic architecture - one might defensibly say the classic architecture - and is still widely accepted as one of the better general purpose architectures for minicomputer class machines. The PDP-11 is without doubt the most widely taught and studied basic architecture in colleges of engineering and computer science, and this is justifiably so; it is the most widely used and emulated general purpose minicomputer in the world.

Despite the hardware hacks (UMRs, KISARs, PARs, etc.) which have become necessary with expanded computing requirements, and which have grown like warts (or cancers?) on the "basic 11", the PDP-11 is still a simple, beautiful, orthogonal - elegant - architecture. Sell me a 32-bit direct-addressing PDP-11 with FPP (no, not a VAX!) and RSX to run on it, and I'll marry it. I'm sure that many other users feel the same way.

Another reason for the widespread acceptance of RSX is its truly incredible flexibility. RSX may not be the most widely used 11-series operating system (RT-11 is, apparently; but RSX will no doubt pass it in the near future); but it is certainly the most flexible of all 11-series operating systems. It can support timesharing, batch processing, several varieties of networking, realtime control and data acquisition, database query and maintenance, text processing, high-reliability transaction processing - and it does all of these things well, and in many cases simultaneously. Among the DEC operating systems, RT-11 does realtime incomparably well; RSTS timeshares without peer; VAX/VMS is without parallel for number crunching, MUMPS is a wonderfully friendly database system - but only RSX offers them all simultaneously to the user.

Let us not forget, however, that the ability of RSX to endure - nay, to prevail - is also due to the past and continuing commitment of Digital to support RSX into the "foreseeable" future. (More on this later.) A dedicated and enthusiastic support and implementation team are a large part of why RSX endures in popularity; without such support from DEC, it is difficult for any operating system to endure. (Forget "NIX; I'm talking real operating systems, not funny languages with double backslashes and ampersands where there should be honest BEQs and EMTs.)

And, without a doubt, the success of RSX is also due to the support and interest of its user community. The RSX SIG has been active and instrumental in the implementation of new features for the operating system in the past; this will surely continue into the future. We may take justifiable pride in the fact that no other SIG takes such an interest in their product and the improvement thereof. There is no other SIG within DECUS with such an active and interested membership. More on this later, too.

Yes, RSX is a gem even among other excellent DEC operating systems; it is the joat, the wizard, the Renaissance man of the PDP-11 series. It has been honed to a fine edge by the continuing support of DEC, an excellent development team, and by the support and demands of its large user community. It is pleasant to work with, friendly to the user to an extreme found in no other system, readily optimizable for almost any application, and generally easy to use. It is so successful that it is the template on which VAX/VMS was written and is the foundation of P/OS, the Professional series canned operating system.

That is where RSX stands today. Where do we go from here? Is it possible that we now stand at the pinnacle of achievement, with nowhere to go but down? (i.e., to VMS, an immature system?) Is even M-Plus a "mature" product? Most importantly, what is in store for the RSX SIG?

Such questions have been nagging at me for some time. At a local DECUS conclave, a few of the Las Vegas attendees sat down and discussed the situation for a while. The unspoken consensus was that the SIG is losing its ... fire, joie de vivre, mandate, the je ne sais quoi of technical excellence and mass user interest which, up until now, has characterized and justified its existence.

Now, a lot of people out there may be thinking that even if RSX is slacking off due to competition from VMS, the SIG will shortly receive a shot in the arm from the fact that the Professionals run a bastardized version of RSX, and therefore Professional users are going to be interested in RSX; ditto for Microlls and Micro/RSX. Wrong. The average Professional or Microll owner could care less about RSX. What they want is a system which requires minimal smarts to use; their interests are grossly different from ours. And as for the "RSX" on the Professionals ... friends, they don´t call it P/OS for nothing. No, if we are to continue in the grand manner of the past, we must save ourselves without help from that direction.

Consider: What have we, as a SIG, done in the last few years to continue the trailblazing done in the past? Pre-symposium sessions? Sure. We thought them up. Magic sessions? We were there firstest with the mostest (and ours are still the best). New user Q&A? You bet. The software clinics? Another first.

But ... that´s symposium stuff; a lot of people don´t get to the symposia; and anyway, I mean recently. What about generally useful stuff one might find on the SIG tapes or in the Multi-Tasker? Ralph Stamerjohn´s monumental ACP manual is already a few years old, and though it is extremely well done and still without peer among user-written documents, it can only be done once. Virtual disks have exploded into general use; yes, even into DECnet and main memory disks ... but can only be done once. The KMS kit is "mature", with the maturing of RSX-11M. Joe Sventek´s Software Tools package is a truly colossal achievement ... once. There are others.

What have we turned out recently on that scale? Is there anything left to offer us a challenge worthy of the abundant expertise of this SIG? Are there still mountains for us to swim, oceans to climb, people to go, places to do, things to meet, code to hack?

Most of all, and what I fear most, is the SIG going to die an unbecoming, lackluster death for lack of a continuing challenge as DEC keeps pushing us to migrate into bigger and "better" (ha) (i.e., VAX) machines? Nay, it would be far better to dissolve the SIG and follow our individual lights, with only pleasant remembrances of our grand and beautiful edifice at the heights of glory, than to permit such an ignominious fate. Who wants to go to endless Magic sessions where all that is done is mindlessly repeat tattered tales of past glories? A Falcon with Micropower/Trashcal and an ASR33 can do <u>that</u>.

The dinosaurs (e.g., PDP-14s) didn't meet or make new challenges. There are no dinosaurs (well, only a few) today.

In short, where we are going, and in what direction does our best possible future lie?

Now, this is a tough question, and I cannot presume to speak for even a small fraction of the membership of the SIG. My interests lie along system lines, which is decidedly untrue of the majority of SIG members. However, for what they're worth, here are some suggestions and thoughts on things which may be interesting and profitable for us to take up. At the very least, perhaps these will stimulate some discussion and thought.

Where is multiprocessor RSX? If the RSX implementation team likes it well enough that they're keeping their 11/74, it must be good! (How many people out there know what an 11/74 is? Raise your hands. One, two ... Bezeredi has his hand up twice, that's cheating, ... five ... seven?)

N-processor RSX-11M-Plus is considerably more feasible with the introduction of the 11/73 cards. Is DEC withholding this goodie from us deliberately? (One hates to say it, but must ... to sell VAXes? More on this later.) Is the problem <u>really</u> the lack of the I**2T, the much-lamented interprocessor insanity timer card? That's a facile excuse, but it doesn't hold up; if DEC can make limited-use items like the DMV-11 (and sell them at a profit), it can certainly make IIT cards so we can run multiprocessor RSX.

Another quick excuse is that nobody knows how the 11/74 works anymore. Sorry, I don't buy that, because you can still buy prints for it (part number MP-00822 ). Or check out pp. 398-400 in the DEC "Computer Engineering" book for a good discussion of the 11/70mP. No, the problem isn't <u>really</u> due to lack of hardware.

The problem may be the lack of an enthusiastic and demanding RSX user community lustily screaming at DECUS for multiprocessor RSX; and I think this to be likely. Give me a quad processor 11/73 and I'll assuredly whip a 780 for realtime and for almost any other performance benchmark. WHY DON'T WE HAVE MULTIPROCESSOR RSX?! I've already got a dynamite logo for it ... maybe you'll see it in Anaheim this fall.

Tied in with that question ... where is a truly high-performance 11-series machine? Get any three wizards together and they can throw together a super-performance machine with minor mods to the hardware. Example: Separate kernel and user cache memory. Why not cache the Exec separately? That's where the system spends a lot of time handling QIOs and directives, isn't it? Where is an IIL or ECL 11/80, a super11 strictly for users who require max performance? Where is a DEC-supported auxiliary mathematics processor for scientific processing, or an auxiliary database processor to get Datatrieve moving?

How many of you know that the SBI was originally intended for such a super-11? "The SBI was originally conceived in 1974 for use on a PDP-11 processor ..." p. 279, ibid. Or that a 32-bit PDP-11 architecture was actually finalized back in 1974 - a "natural, easy extension"? And what did DEC do with it? "Fortunately, the project was discontinued." p. 407, ibid. Fortunately? For who? Not for us, certainly; maybe for the VAX designers. I could just scream.

Come on, the 11-series architecture is well defined now; how much effort can it possibly take to push it to the max in new silicon? They did it with the J-11 in CMOS, right? Why not a K-11 in ECL?

Speaking of the J-11: The 11/34 is probably the most popular Unibus processor ever made. At the Las Vegas DECUS, any number of people were asking for a J-11 based card to replace the M8265/M8266 CPU cards in the 11/34 to give the power of an 11/44 and (maybe) 22-bit memory addressing capability to the 11/34. (Did you know that the MS11-LD is a 22-bit card? Surprise, surprise.) What was the answer from DEC? "We don't think there is enough interest in such a product." I don't understand ... if people are asking for it, there must be interest in it, right?

And if only better use was made of the hardware that we have ... let me see CIS support in editors and in Datatrieve, please! The 11/44 supports it, the 11/23 and 11/73 will support it; if that is where things are going, then let's make good use of it. CIS should not be restricted to Cobol users when it is such a powerful tool for text and record handling.

Who knows why RMS has to occupy 40K of memory? 40K?!! 40K is bigger than a lot of tasks! How much time has been spent on optimizing RMS for size and performance? Functionality does not equal efficiency. Does anyone really understand RMS? Yes, assuredly many of us use it, but how seldom do we see RMS hacks in the Multi-Tasker - probably because nobody understands it. (Occam's Razor.) Perhaps we need an "Up Your RMS" manual to make that puppy haul.

Now for my and many other users' pet peeve: How about the RSX System Logic Manual? There has not been a new version since RSX-11M V3.1, and I'll bet the majority of RSX users have never even seen one. This is a disgrace, and an outrage. If everybody out there filed a top priority SPR stating that they wanted a new issue of the System Logic Manual, do you think we'd get it? Bet your sweet IOT we would!

If you do nothing else after reading this, file a short, top priority SPR demanding a new revision of the System Logic Manual.

Who the hell wants directive commons in their Exec? How much POOL does any specific RSX system need to run effectively, and if we know that, how about moving some of that directive common code back into the Exec for systems which can afford it? And for the nth time, when will someone do a real evaluation of how the SYSgen and dynamic system tuning parameters affect operation of an RSX system in various modes of operation?

These days, I seldom see truly elegant little pieces of code in the Multi-Tasker, whether they be functional or just neat. Where is a spiffy little task to rotate loadable drivers through GEN in lieu of rotating console lights? Where are zaps for M-Plus to let me run it on an 11/45? With I and D space enabled? What has happened to the days when the Multi-Tasker regularly went 20, 30, 40 half-size pages on cheap offset paper?

If it's what we deem necessary, let's reef on DEC to give us a 32-bit architecture PDP-11. Tack 16 bits on to each word and let's go for a 32-bit direct addressing bus. Gimme a full 7680 Kword of main memory. Forget I and D space. I'm emphatically not talking about cheap VMS; I want to see real RSX in a new environment. RSX-32 could beat the pants off a VAX with a properly designed bus, IIL logic, multiprocessors, and cache in the right places. And it won't require a fancy new architecture or VAX's 87 addressing modes, of which indeed only the PDP-11 basic 8 are used most of the time.

DEC Employees

Please do not read the following two paragraphs. It will make you angry, and I don't like to have DEC people angry at me. My machine sometimes does strange things after a PM when DEC is mad.

Why don't we have a 32-bit PDP-11? Because we let DEC sell us out. We let them tell us we needed VAXes, and we believed them. (Remember what P.T. Barnum said?) Well, we sure got what we deserved; not an improved, upward compatible PDP-11; we got an incomprehensible, user-unfriendly monstrosity that runs a Virtually

Monstrous System. Ever try using HELP on a VAX? Virtual memory doesn´t make everything right, people: It´s nowhere near the HELP we have on RSX. Serves us right. We let them do it! And then we let them get away with it!

Now DEC is telling us that RSX is "mature", that PDP-11s are "traditional products", and that VAXes are "changing the way the world thinks"; and like the vacuum-headed stuffed dogs you see in back windows of cars, everybody is nodding their heads up and down, going "yup, yup", and putting in orders for VAXes.

DEC: "I am your DEC sales rep. You are getting sleepy."  
User: "Yes I am."  
DEC: "PDP-11s are obsolete."  
User: "Yes they are."  
DEC: "You want a VAX."  
User: "Yes I do."  
DEC: "You don´t need to see stringent performance comparisons for your application."  
User: "No I don´t."  
DEC: "Trust me. We are DEC and we know best."  
User: "You are DEC and you know best; I am going to buy me a VAX and chuck my 11/70."  
DEC: "Sign here."  
User: "For sure."

It serves you _right_ you sorry suckers, we not only let them do it to us, we´re actually still helping them and doing it to ourselves!

Now, you may think that this boy is off the deep end and that DEC isn´t really trying to cram VAXes into everybody´s site. OK. Try this simple experiment: Tell your sales rep that you have an RSX application on an 11/44, but need more power to support just a few more users. See whether he (a) suggests you look into a used 11/70, (b) suggests you add more memory or faster disks to prevent or speed up swapping, (c) suggests you add another 11/44 and buy add-on Class D licenses for it, or (d) suggests you go to a "baby" VAX (and buy all new VAX software) because they´re "so cheap and _completely_ upward compatible with RSX." I will bet you dollars to doughnuts that you get a spiel on how great VAXes are.

Another thing: What was really novel or interesting at the last Symposium you attended? Was there anything that knocked you back and made you think? Or are we instead becoming like a fat, complacent old dog, content to lie in the sun, snapping only occasionally at mildly irritating fleas which would be killed off sooner or later by an official DEC $1500/month Flea Circus collar?

So, is the RSX SIG indeed eventually to be reduced to arguing about who told the floating point story at the ´81 Los Angeles Symposium? Will the Multi-Tasker be reduced to no better than a comic book containing nothing but symposium Q&A session notes, cute indirect command files and continuing pleas from the Editor for user articles? (Tune in next week ...)

Seriously, folks, this hurts me most: It is not fitting for peons like me to criticize their betters, and those higher in the scheme of things; but is it not a symptom of growing lassitude in the SIG when our very own Chief Wizard says, in print, that there will probably never be a 20th anniversary celebration? Ladies and gentlemen, life begins after 4.0.

Am I wrong? Am I right? Both? I don´t know; I sincerely hope that I´m wrong at least about the SIG´s future, which to me looks mighty dark at this point. So help me out - shucks, help yourselves out - and prove me to be wrong. Write me a nasty letter of rebuttal and make it stick. Copy it to the Multi-Tasker. Show me that there are more than half-a-dozen people in the SIG who know from you-know-what and are willing to share it (no, not the you-know-what) with the rest. Prove to me that DEC has an ongoing committment to making the PDP-11 series better, bigger and faster. Write a Multi-Tasker article on something novel, something to challenge my interest, something to point the way and hold high the torch for those of us who are stumbling around in the dark, looking for the light.

Point out the way. Lead; I´ll follow - or vice versa. Push. Pull. Or get out of the way.

If nothing else, file that SPR and tell DEC you want a new System Logic Manual.

But let´s collectively and individually do something - a whole bunch of somethings - if for no other reason than to spite those poor suckers who bought the DEC sales line and own VAXes, and to demonstrate to them that RSX has a long way to go yet before we, the rightful parents and family of RSX, lay it finally, with tears, pride, and honor, to rest.

"Do not go gently into VMS,
The SIG should flash and flame at close of day;
Rage, rage against the dying of RSX."

Respectfully,

Bruce R. Mitchell
Machine Intelligence and Industrial Magic
PO Box 601
Hudson, WI 54016

## Working Group News

Jeff Hamilton
Working Group Coordinator
(214)457-4175

Date of this report: 08MAR84
The working group chairmen are as follows:
RSX-11M Unsupported Versions:
  Bill Burton
  Texas Research Institute
  1300 Moursand
  Houston, Texas 77030
System Performance and Accounting
  Paul Sorenson
  AEP, Interactive Graphics
  1 Riverside Plaza
  Columbus, Ohio 43215
DECUS Library
  Bruce Zielinski
  RCA
  Marne Highway M/S 138-2
  Moorestown, N. J. 08057
SIG Tape Collection
  Glen Everhart
  RCA Government Systems Division
  Route 38
  Cherry Hill, New Jersey 08358
SRD
  Bob Turkelson
  NASA/Goddard Space Flight Center
  Mail Code 935
  Greenbelt, Maryland 20771
Data Acquisition, Simulation and Process Control (DASPC)
  Allen J. Bennett
  Clark Equipment Co.
  P.O.Box 3000
  Battle Creek, Mich. 49016
Runoff
  Chuck Spalding
  Adept Technology Inc.
  1202 Charleston Rd.
  Mountain View, Calif. 94043

The Unsupported Version working group is currently planning sessions for the Spring Symposium for past unsupported versions of 11M/11M+. An article is being submitted to the Multitasker about the current work being done by the working group.

## RSX MULTITASKER

The System Performance and Accounting working group is continueing its work in preparing the index of the past RSXSIG tapes in regards to System performance and accounting. There is a session being planned for the Spring Symposium for the System Performance and Accounting working group to discuss further work to be done. No other work is reported for the last month.

The DECUS Library working group have continued efforts to construct a tape to provide to the DECUS library of the best software off of the past RSXSIG tapes. A form letter was sent out to the 60 members of the working group and 30 replies were returned. These people are being used in the evaluation of a sample 2400´ 800 bpi tape that will be submitted to the DECUS library.

The RSX83B Sig tape is starting out across the country and some few replies are back. It has also been distributed to the DECUS library as number 11-SP-60, and sent to Europe and Australia. Ralph Stamerjohn has volunteered to index the SIG tapes from day 0 to the present and he received his tape last week. Nothing else new has been done.

The SRD working group has continued work on the improved SRD supplied to the Fall 83 RSXSIG tape. Enhancements and continued bug fixes are being considered.

The DASPC working group has continued its efforts to lobby DEC for more real time development into RSX systems. This will continue into the future. Representatives from the LABS SIG, VAX Realtime Working Group and DASPC continued in their efforts to form a new SIG, but the status is unknown as of this writing. A seperate session is also planned for this working group.

The Runoff group has continued its effort to consolidate desirable features of several versions of Runoff into an "official" version. In the near future an article will appear in the MULTI-TASKER doing a comparison of each of the versions that exist along with DSR the DEC Standard Runoff. Telephone discussions have been held to review the charter that is held for the Runoff working group. No sessions other than the general session is planned for the Spring Symposium.

If you are interested in providing information to a special working group concerning problems or ideas in that area, please get in touch with the working group chairman of that group.

## Hints and Things

## A Patch for RSX-11M-Plus HELLO

One of the problems encountered on M-Plus systems with a large number of privileged users and/or printing terminals is that when a privileged user logs in on a printing terminal with a HELLO line of the form HELLO name/password, a permanent record of his login name and password is left on the terminal for anyone to read. This might charitably be said to compromise security.

The enclosed patch to [12,10]HELLO.MAC on the M-Plus distribution kit solves this problem by overstriking the login line if the user logs in using a line of this form. It has only been tested for RSX-11M-Plus, but should be readily adaptable to RSX-11M.

To apply and use the patch:

1. Create the HELLO.SLP file.
2. Copy it to [12,10] on the distribution kit.
3. Apply the patch using SLP (SLP @HELLO.SLP).
4. Assemble the new HELLO file (MAC HELLO=LB:[1,1]EXEMC/ML, [11,10]RSXMC/PA:1, [12,10]HELLO).
5. Modify the proper HELLO .ODL build file in [1,24] to refer to the new HELLO.OBJ in [12,10] rather than the [1,24]MLTUSR/LB:HELLO library.
6. Rebuild HELLO using the HELBLD, HELFSLBLD, or HELRESBLD command files in [1,24] (TKB @HELxxxBLD).
7. VMR the old HELLO out of the system and VMR in the new HELLO.

```
HELLO.MAC=HELLO.MAC
-109,,/;BRM001/
;  BRM001  09-JAN-84 STRIKEOVER ON LOGINS OF FORM XXX/PASSWORD
;
-137,,/;BRM001/
```

```
     SOVMSK:     .ASCII   /MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM/<15>
        .ASCII   /WWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWW/<15>
        .ASCII   /314159265348978323847264338327/<15>
        .ASCII   /                                /<15>
     SOVLEN = . - SOVMSK
     -187,,/;BRM001/
     BRMFG0:     .WORD    0                 ; Flags login of type XXX/password (0=false
     BRMFG1:     .WORD    0                 ; Flags login on a batch terminal (0=false
     -201,,/;BRM001/
     BRMDPB:     QIOW$    IO.WVB,LUN1,EFN1,,,,<SOVMSK,SOVLEN,0> ; Strikeover DPB
     -437,439,/;BRM001/
     41$:        CLR      BRMFG0            ; Assume "normal" prompted password
        CMPB     -1(R0),#'/       ; PASSWORD COMMING?
        BNE      42$              ; NO, SYNTAX ERROR
        MOV      R0,PSWDBF        ; SAVE PASSWORD ADDRESS
        INC      BRMFG0           ; Password given; set the flag
     -444,444,/;BRM001/
     42$:        CLR      BRMFG1            ; Assume no batch job
        CLR      PSWSKP           ; SET TO CHECK PASSWORD
     -472,,/;BRM001/
        INC      BRMFG1           ; Show that this is a batch job
     -498,499,/;BRM001/
     46$:        TST      BRMFG0            ; Was the login of form xxx/password?
        BEQ      4601$            ; If not, simply proceed
        TST      BRMFG1           ; Was the login for a batch job?
        BNE      4601$            ; If so, don't print the overstrike
        DIR$     #BRMDPB          ; Print the overstrike message

     4601$:      CALL     $SWSTK,465$       ; SWITCH TO SYSTEM STATE
     /
```

# A PERFORMANCE HINT FOR SPECIAL I/O DRIVERS

Terry Medlin
GEJAC, Inc.
P.O. Box 188
Riverdale, MD 20737
301-864-3700

A typical sequence of operations involved in servicing I/O requests in RSX goes something like this:

1. The program (or its run-time system) issues a QIO$ directive

2. The exec checks the DPB and the I/O parameter block for validity

3. A QIO packet is allocated and queued to the appropriate driver

4. The program waits for the I/O to complete by synching with some event flag

5. The program recycles to the first step

The amount of code executed in the second and third steps is usually not a problem. However, for realtime systems with very large sample rates, e.g. 50K or higher, some applications might benefit from the elimination of those two steps.

One way to accomplish this objective is the following approach:

1. The program issues a (super) QIO$ and passes a list of buffers, lengths, event flags, status arrays, etc. in the I/O parameter block

2. The driver performs validity checking as required and starts the I/O

3. The program waits for the I/O to complete by checking the event flag

4. The program recycles to the preceding step

The driver is this case fills the first buffer. When it is complete, it does NOT call $IOFIN as per normal; rather, it passes back status info and sets event flags itself. The next buffer is then selected and the driver continues. I am glossing over a few details but the basic concept is, I hope, clear. You will need to define how the driver tells the program to stop at which point the program can issue an IO.KIL to clear the I/O and the driver.

Since this mechanism is NOT normal, it should only be used on special drivers. Note that the I/O never completes in the above. This can have ramifications on certain RSX options such as shuffler activity and swapping. The cases where I have used this technique were on practically dedicated systems and I did not care who got hurt as long as I could handle the data rate.

## EXTENDING RSX ON THE FLY

Terry Medlin
GEJAC, Inc.
P.O. Box 188
Riverdale, MD 20737
301-864-3700

As most RSX hackers know, there are many useful extensions or site specific requirements that can be addressed by modifying or patching the source of RSX. Following such changes one needs to regenerate the system by either going through an entire sysgen or reassembling the module altered, replacing the module in RSX11M.OLB, and relinking the exec and/or privileged programs. While sysgen has been made almost trivial with the advent of saved answer files and auto-configure, they are still time-consuming. Those of you who missed the opportunity to gen earlier versions of RSX really don´t know what you missed.

The purpose of this article (which Allen finally got me to write) is to describe a procedure that can be used to modify the memory copy of RSX. While it is moderately non-trivial to setup, future changes for autopatch or new releases are almost trivial.

For RSX-11M and RSX-11M-PLUS without support for D-space, you have a fairly simple scenario:

1. Allocate a block of POOL

2. Cram a segment of position independent code (PIC) into this block of POOL

3. Disable interrupts

4. Replace the code in portions of RSX to jump (JSR style) to the "new" code in POOL. This new code must execute your extensions plus any code replaced by the JSR instruction.

5. Enable interrupts

Obviously, a privileged program is required for this operation.

For RSX-11M-PLUS systems with D-space support the situation is more complex. First, POOL space is mapped by the D-space registers and not by the I-space registers. Therefore code placed into POOL will not execute in the normal system. Second, most of the exec is mapped by the I-space registers but not by the D-space registers. Therefore you cannot replace instructions without altering the mapping registers. However there is a way to accomplish our goal:

1. Allocate a block in the ICB area which is always mapped by APR 0 in both I-space and D-space.

2. Allocate a block of POOL for the "new" PIC segment you want to execute and copy your PIC segment into that block

3. Cram a piece of code into the ICB area to remap POOL so as to allow code execution. Since the code moved into the POOL area is PIC, you can map it by any APR that does not conflict with other operations. A reasonable choice is APR 5.

4. Disable interrupts

5. Modify the exec to JSR to the ICB area which will then do a remap and transfer you to your "real" code in POOL. Note that you must alter the D-space mapping to modify the exec as explained earlier.

6. Reenable interrupts

The above approach sounds more cumbersome than it really is. It does work and the overhead is nil. Also, note that instead of allocating POOL, you could just as easily allocate secondary pool or even use a predefined partition. The key to this method is the ICB area which is typically small. However, it can be enlarged by another sysgen if you really need additional space.

## Letter from U.K. RSX-SIG Newsletter Editor

Paul Phillips
South Western Regional Transfusion Centre
Southmead
Bristol BS10 5ND.

Date: 22-June-1983

The U.K. RSX-SIG has just relaunched its own newsletter, of which, for my sins, I have become the editor.

I am enclosing an article submitted to us by Graeme Miller, concerning a problem he experienced in converting from V3.2 to V4.0. I hope it might be of interest to you, for publication in a future edition of the Multi-Tasker.

Yours sincerely,

Paul Phillips

## RSX11M V3.2 TO V4.0 CONVERSION PROBLEMS

Graeme Miller
Tarmac Roadstone Holdings Limited
Roadstone House
P.O. Box 44
50 Waterloo Road
Wolverhampton
WV14RU

I would like to relate a problem that we found in moving one of our application systems from V3.2 to V4.0 of RSX11M. I do this in the hope that those of you who have not yet made the conversion (as you are probably too busy writing software for users) and are using the same features of V3.2 as we were, will be forewarned.

We have a number of 11/23's running at remote sites where there are no computer-sentient beings. The software must, therefore, be very robust and tamper-proof. The type of message produced by RSX on the console log would not be accepted by the people using these machines with any great sympathy, let alone understanding. We have console logging enabled to a log file, but with the console terminal disabled. All system messages and user program errors are reported to the console log file. The console terminal is freed for normal applications use after the startup command file has completed.

We also have the terminals running the application set as slave terminals. I refer to the V3.2 meaning of "slaved", where unsolicited input is held in the typeahead buffer for use when required, not the V4.0 meaning of "slaved" where unsolicited input is discarded. The former can be implemented on V4.0 by means of a patch to TTICH in the October 1982 Software Dispatch. Our application programs are selected from the terminal by means of a menu task MNU. A copy of this task runs from each application terminal. it has a low priority and checkpoints out when it has spawned the application task selected by the operator. When there are slack periods in the operation of our system, the menu task sits waiting for input from the terminal. Slack periods vary for each terminal according to the time of day.

We wrote a small utility called XQT, which enables you to force a MCR command at any terminal in the system. It has the same syntax as BROADCAST, e.g.

```
XQT TT5:BYE
```

would log TT5: off. This utility has the spawn directive at the heart of it. We wrote XQT to accept pseudo-device names and we found that the command

```
XQT CO:HELLO 1/54/PASSWD
```

did indeed log the pseudo device CO: on to 1,54. This meant that we had created something that was a natural for use as a batch facility and continued to use it as such, with commands of the form

```
XQT CO:@BATCHPROC
```

All was well on V3.2. The tasks running on CO: used the console log file as their TI: and left the physical console terminal TT0: (to which console logging was disabled) alone.

Along with many other improvements, V4.0 brought us,

a whole new meaning to the word "slaved"
a new console logger
a new version of the indirect command file processor.

We overcome the first of these three improvements with the patch mentioned previously. However, when we came to run command files on our, by now, indispensible batch terminal, we found that occasionally the indirect command file would start and then just go into a wait-for state.

With some detective work and some good fortune we found that the ICP only did this when the physical console terminal was running a program, was in slaved mode, and was waiting for input. If you hit 'return' on TT0: the command file on CO: then happily carried on. Moreover, if you transferred the ownership of COT... to any other physical terminal via the commands

```
SET /COLOG/COTERM=TTn:
SET /COLOG/NOCOTERM
```

the problem then transfers to that physical terminal, i.e. you must hit 'return' on that terminal to release a command file running on CO:

As near as I can come to it (and I would gladly welcome any suggestions to the contrary) this is what is happening;

the spawn directive successfully invokes the ICP on pseudo-device CO:

the ICP realizes it hasn't got a physical terminal for input and in its panic tries to attach to the physical terminal that owns COT... Why it should need to attach to a terminal it is not going to use, is not clear. In fact, a command file that contains only the .EXIT directive will produce this undesirable effect.

the terminal which owns COT... is busy. it is busy because it is slaved and is waiting for input.

as soon as an entry is made on the physical terminal, the terminal is freed, the ICP can attach to it in order not to use it and can proceed as normal using the log file for its output.

We explained the problem to DEC UK in the form of our local office and the telephone support centre. DEC were most sympathetic and helpful, but suggested we were using an unsupported feature of RSX. They correctly pointed out that the executive reference manual for RSX (both 3.2 and 4.0) specifies that the spawn directive is for use with physical terminals or virtual terminals if you have 11M+. Unfortunately the device CO: does not come into either of these classifications.

Our SPR has been forwarded to the States, so no doubt, in the fullness of time we will receive a solution to our problem.

In the meantime, we were advised to make use of the DECUS virtual terminal driver, which I am now obtaining. I assume we can use this to facilitate our batch stream and leave CO: to its proper function. so, to whoever wrote the virtual terminal driver, thank you in anticipation.

To those of you who are doing such outlandish things as this with RSX and have solved this problem already, could you please publish the solution.

To those of you who are not, try logging the CO: device on as a batch terminal. it is very useful so long as TT0: is not slaved and waiting for input.

## DEC Utilities Have Problems with RP11 Controller

Michael E. Mazzoni, President
Process Control Systems, Inc.
1300 S. Calhoun Road
Brookfield WI 53005

Re: RSX Q & A, Nov. 1983 Multi-Tasker

I have a thought on question 38 from Mike Nei, Essex. I had the same problem with most new RSX utilities and layered products whenever I ran them on an RSX system with an RP11 controller and an RP03 disk as the system device.

Before I realized that the problem was RP03 related I would dutifully report my problems to DEC and get the same stock answers:

´...can;t reproduce the problem...´
´...it works just fine on our system...´
´...not a known problem...´

Then one day I talked to an experienced DEC developer and he explained the problem.

It seems that the RP11 controller always transfers in multiples of 2 words (I suspect that the RP11 is a reworked RP10 since the buffer register light display runs from bits 00 to 35). Tasks which attempt to transfer an odd number of words suffer unexpected results. Despite the fact that the RP11 dates back to 1971, some software developers still don´t know about the characteristics of this device.

Some of the DEC RSX products that fail as distributed are RPT for 4.0 and 4.1, RTEM-11 V1.0, RTEM-11 V1.1, BP2 compiler V1.6, BP2RUN V2.0, and BP2RUN V2.1.

Fixes have been published for BP2 V1.6 and RPT for V4.0. When latest release kits were prepared, RPT for V4.1 failed to incorporate the V4.0 update, but an unpublished patch exists.

In answers to my SPR´s DEC has acknowledged the problem for RTEM and BP2RUN and "...it may be fixed in the next release" or "...it may be fixed in the future. As a workaround, run from another device." But since DEC has announced that support for the RP11 ends with RSX-11M V4.1 (and presumably RSX-11M PLUS V2.1), I am not comfortable with those answers.

Although DEC hasn´t marketed an RP03 since 1977, some manufacturers of disk controllers emulate the RP11/RP03 with their current devices. if the emulation is faithful, the problem will be the same as with a real RP11.

## Switchable RP06 Disk Drives

B.J. Stafford
Lloyds of London Press
Sheepen Place
Colchester, Essex
CO3 3LP
United Kingdom

Having a dual PDP 11/70 configuration with a switchable RP06 drive causes a hassle if you boot in RSX11M V4.1 with the drive switched to the other machine (RSX very kindly sets the drive offline and to RP04 despite sysgen RP06 default!).

In order to use the disk drive on both machines (as an RP06) the following patch (marked BJS001) to [12,10]SAVSIZ.MAC is suitable:-

```
;
; SET THE INITIAL VALUE
;
INISIZ: MOV     DEVSIZ(R2),U.CW2(R4)             ; SET HIGH SIZE
        MOV     DEVSIZ+2(R2),U.CW3(R4)           ; AND LOW SIZE
```

```
         RETURN
;
; RP04/05/06 SIZER
;
DBSET:   MOV      #40,10(R1)               ; CLEAR THE MASSBUS CONTROLLER
         MOV      R0,10(R1)                ; SELECT DRIVE
         MOV      26(R1),R0                ; READ DRIVE TYPE
         BIT      #10000,10(R1)            ; DRIVE EXIST?
         BNE      5$                       ; NO
         BIT      #400,12(R1)              ; DRIVE AVAILABLE TO THIS PORT?
         BNE      10$                      ; YES
         BR       20$                      ; NO - LEAVE IT ALONE **BJS001**
5$:      BISB     #US.OFL,U.ST2(R4)        ; NO, SET OFF-LINE
         BR       20$                      ;
10$:     CALL     INISIZ                   ; SET UP INITIAL VALUE
         BIT      #2,R0                    ; IS IT AN RP04?
         BEQ      20$                      ; YES
         MOVB     #5,U.CW2(R4)             ; NO, IT'S AN RP05
         MOV      #12990.,U.CW3(R4)        ; SET ITS SIZE
20$:     RETURN
```

Compile using the following line to MAC:-

```
SAVSIZ=[11,10]RSXMC/PA:1,[1,1]EXEMC/ML,[12,10]SAVTRC,SAVSIZ
```

Then rebuild SAV task in the normal priveleged task way:-
Replace SAVSIZ in SAV.OLB

```
          LBR [1,24]SAV/RP=[12,10]SAVSIZ
          TKB @[1,24]SAVBLD
```

and then VMR new SAV into the system

# NanoRSX - Getting M-Plus on One RX50 Floppy

Hans J. Jung
The Associated Press
New York, NY

At one time or another, we have all been faced with fitting 10 pounds of <fill in the blank> into a five pound bag. Trying to fit a bootable RSX-11M Plus system onto a 800 block RX-50 diskette is such a problem.

Shortly after the fall symposium, Carl Freidberg and Paul Mort asked if they could use my site to load M-Plus onto a Micro-11. Paul was lucky enough to get one of the first Micro-11s delivered in this area. This was the first the local field service folk had seen.

We use a number of Q-bus systems at AP, running both M and M-Plus. With most of our systems using third party winchester disks and controllers, I´ve developed some facility at cross loading software.

At the time, there was no official M-Plus distribution available on RX-50´s, though Micro-RSX had been announced at symposium. We expected no problems, thinking that we could easily plug one of my controllers into the Micro-11, boot and transfer the system to the Micro´s RD50 10 meg winchester. Two or three hours at most, with time for a nice dinner afterward. Wrong.

The Micro-11 uses the 11/23-Plus processor board, one that I don´t use. The Plus board, a KDF-11B, has boot proms and two terminal ports on a quad board along with the 11/23. The standard 11/23, the KDF-11A, only has the processor and memory mapping hardware on a dual board. The Micro´s boot proms wouldn´t boot my RM03 lookalike controller, and none of the DEC documentation described how to turn off the boot on the processor board.

This brings us to the problem that plagued us for the next two weekends, inadequate-to-non-existent-documentation. The hardware documentation supplied with the Micro-11 gave no information as to switches and jumpers on the processor or disk controller boards. Further, while there is a section on interpreting error messages during boot and self test, most solutions involve calling an 800 number to ask for help. Not too useful on a Sunday afternoon.

Without going into a blow-by-blow description, we finally plugged the Micro´s disk controller into one of my processors and were able to move a system onto the RD50 hard disk. In the process, we discovered problems with the RX-50 floppy drive, which would later turn out to be controller problem.

By now it was late Sunday night, and we had spend two full days on a three-hour project. Oh well.

Paul called DEC to fix the floppy problem, and along the way, the field-service folks wiped the winchester clean. Damn. But at least now we knew how to hook everything up.

This time the hardware gremlins moved to the hard disk. After reloading the system, we began to get disk errors while testing it. Errors during read, and so on. The mediocre documentation essentially advised calling the 800 number. Now I have no objection to phone calls, but it was obvious that our problems were not nice simple ones like inserting the floppy the right way around.

The problem seemed to be bad format data on the RD50. We tried to reformat the RD50 using FORMAT, but M-Plus requires you mount a disk foreign before you can format it. MOUNT tries to read the first block of a device even if you are mounting foreign. Since the first block gave a read error, we couldn´t mount. Catch 22.

Sunday afternoon. None of us relished the prospect of a third weekend in the AP computer room.

Ok, maybe we can´t load the winchester. But since we had a complete copy of the Micro-11 system on one of AP´s disks, we could store it on a stack of RX-50 floppies. DEC even modified BRU to allow multiple floppies on a save, ala magtape. Unfortunately, we discovered that stand-alone BRU from M-Plus V2 wouldn´t run properly on the 11/23. We never discovered why.

Well, if stand-alone BRU won´t work, why not build a bootable system on an RX-50 and run regular BRU from it? Carl played with the VMR file for a while, but couldn´t get the total number of blocks required below about 1200. An RX-50 holds 800 blocks, period.

We were ready to give up when I remembered a article about SAVE in an old "Multitasker." It said that once task images were loaded and fixed in the save file, the original tasks no longer needed to be on the disk. Thus was born Nano-RSX, a bootable M-Plus system on one 800-block diskette.

The basic technique was to copy a virgin system-image file and some of the minimally required task files to the RX-50, then run a partial VMR to load the tasks into the system image file. The loaded tasks were then deleted from the disk, and the remaining tasks copied and VMR´ed in.

This is the first command file to initialize the diskette, copy the first set of tasks and VMR them in:

```
ALL DU2:
INS $INI
MOU DU2:/FOR
INI DU2:NANORSXMPLUS/INDX=BEG/INF=20.
DMO DU2:
MOU DU2:NANORSXMPLUS/VI
UFD DU2:[1,54]
SET /UIC=[1,54]
PIP DU2:RSX11M.SYS/NV/CO/BL:216.=DR0:RSX11M.TSK
PIP DU2:/NV/CO/CD=DR0:DIR11M.TSK,DR211M,LDR,MCD,TKN
PIP DU2:/NV/CD=DR0:DIR11M.STB,DR211M,RSX11M
PIP DU2:/NV/CO/CD=DR0:TTDRV.*,DUDRV.*,NLDRV.*,RDDRV.*
ASN DU2:=SY:
INS DR:[3,54]VMR
ASN DU2:=LB:
VMR @DR0:MINVMR.VMR
```

MINVMR.VMR is about half the normal VMR command file. The major change is fixing each task. This allowed us to delete the task files. This is MINVMR, less comments to keep it short:

```
RSX11M.SYS
SET /SYSUIC=[1,54]
SET /LIBUIC=[3,54]
SET /POOL=700
SET /PAR=SECPOL:*:120:POOL
SET /PAR=SYSPAR:*:*
INS DIR11M/RON=YES/PAR=SYSPAR
FIX DIR11M/DIR
INS DR211M/RON=YES/PAR=SYSPAR
FIX DR211M/DIR
INS LDR/XHR=NO/PAR=SYSPAR
FIX ...LDR
INS MCD/XHR=NO/PAR=SYSPAR
FIX MCR...
INS TKN/PAR=SYSPAR
FIX TKTN
SET /TOP=SYSPAR:-*
SET /PAR=DRVPAR:*:*
LOA TT:/SIZE=12500
LOA DU:
LOA NL:
LOA RD:
SET /TOP=DRVPAR:-*
SET /PAR=GEN:*:*
SET /RNDC=6
SET /RNDL=1
SET /RNDH=150.
SET /SWPC=20.
SET /SWPR=5
```

```
PAR
TAS
CON DISP
SET /POOL
```

The second command file deletes the task files we´ve loaded and copies the remaining ones we need:

```
PIP DU2:TTDRV.*;*,DUDRV.*;*,NLDRV.*;*,RDDRV.*;*/SD
PIP DU2:DIR11M.STB;*,DR211M;*/SD
PIP DU2:/NV/CO/CD=DR0:FCPLRG.TSK,INS,MCR,MOU,HRC,SAV
VMR @DR0:MINPART2.VMR
ASN =
```

The second VMR file now loads these tasks:

```
RSX11M
INS FCPLRG/PAR=GEN/IOP=NO/CKP=NO          ! F11ACP
;
; Install the tasks into GEN
;
INS INS/IOP=NO              ! Task installer
INS MCR                               ! MCR command line interpreter
INS MOU                               ! Volume mounter
INS HRC/IOP=NO              ! Online reconfiguration control task
INS SAV                               ! System save task
PAR
SET /POOL
```

After completing all the mechanics, which actually was quite easy, we were left with a bootable system. After booting and saving the system, SAV was deleted also. Subsequent boots give a dire warning about SAV not being present, but the system runs ok.

Given a bootable floppy, we can load task images from the other floppy in the RX-50 pair. One nice touch we had to pass up was installing and fixing BRU in the system image. BRU is now overlaid.

Once it was running, we BRU´d the system onto 10 RX-50. The help files went on six more.

There were probably more task files we could have deleted, the directive commons as a start.

It was worth it. DEC fixed the system, this time replacing the RD50. Paul successfully loaded and booted his hard disk.

I don´t know if DEC is shipping Micro-RSX yet. I know if it had been availible Nano-RSX would never have been created. But the whole tale makes a great war story.

Now, if we had made secondary pool even smaller, and fixed a few more tasks, maybe EDT and TKB would have fit. And then we would have room for ...............

# An Approach for Systemizationing Generation of RSX-11M Systems

Barry T. Miller
Chemistry Division
Argonne National Laboratory
9700 S Cass Ave.
Argonne, Il. 60439
(312) 972-3690

## 1.0 ENVIRONMENT

The Chemistry Division operates a modest sized DECnet network to support real-time experiments within the Division. The root level of the three-level hierarchical network is a VAX 11/780. LSI microcomputers form the lowest level of the network. They are located in the scientist´s laboratories. These remote systems support research experiments by performing data acquisition, experiment control and data reduction functions. The VAX is used for both analysis of experimental data and program development. The second level is a set of routing nodes that act as DECnet front-end concentrators for the remote systems. The front-ends also provide auxiliary disk storage for the remote systems.

There are approximately twenty remote LSI´s within the Division. Initially, most all of these remote systems ran RT-11. However, a few experiments required larger multi-tasking systems, ala RSX. The number of RSX systems has already grown to eight. With the announcement of both the 11/73 and DECnet-11M/11 M PLUS support of ethernet, the Division expects the number of RSX systems to continue to increase.

Unfortunately most all of these RSX systems have different peripheral configurations. In the past, this has required separate RSX/DECnet sysgens for each different laboratory system. To avoid creating the job position "Chief RSX Sysgener", we decided to systemize the RSX system generation/distribution process. Initially, only RSX-11M systems would be supported but the process needed to handle 11M PLUS also.

The following objectives were established for the project

1. Develop a consistent approach for RSX system creation which can be followed by all members of the Computer Systems Group (CSG)
2. Reduce the manual effort to produce and maintain systems.
3. Reduce the number of different systems within the user community.
4. Automate the record keeping process, ie. keep track of what was generated, at what level, when and for whom.
5. Centralize and stardardize the location of and access to systems software.
6. Develop conventions and procedures for RSX-11M which could be easily extended to RSX-11M PLUS.

The systems creation process developed is described in the following paragraphs.

## 2.0 RSX SYSTEMS CREATION PROCESS OVERVIEW

The first step in the process is a definition step. During definition, one selects a pre-built RSX system, and defines RSX drivers, a DECnet system and DEC layered products. The output of the definition step is a set of build command files. The build steps mount the necessary disks, make appropriate assignments and then copy/build software components. To a large extent, the system build is a copy process. The final output is a bootable RSX system that is a unique node on the Chemistry network.

The definition step establishes an interactive dialogue with the terminal. Definitions are grouped into sections. For each section, a series of questions is asked and responses noted. In typical DEC fashion, the user is given the opportunity to <cr> - continue, R - repeat, or E - exit at the end of each section.

There are five definition sections. They are:

1. Preliminary setup, eg. target processor node name, device setup, etc.
2. RSX-11M system defintion
3. RSX-11M driver definition
4. Decnet end node definition
5. DEC layered product definition

Definition sections are implemented as separate command files. The output of a definition section is a build command file. Auxiliary command files exist to mount volumes and log sysgen activity. Definition sections and their corresponding builds are described below.

## 2.1 RSX System Creation

Three RSX-11M systems have been generated. These pre-built RSX systems include two 18 bit systems and one 22 bit system. While this reduces the number of RSX executives and privileged tasks to a small number, it also limits the selection of options to those supported by the pre-built systems. The only drivers included in the pre-built systems are the terminal driver and the non-physical (psuedo) devices. The Chemistry RSX system software creation process selects a system from this set of pre-built RSX systems.

Some level of customization is also possible when defining an RSX system. This includes:
1. Selection of ANSI mag tape support
2. Selection of queue manager support (QMG)
3. Selection of RMS support
4. Selection of HELP files (discussed below)

Inclusion of HELP files is optional. During the definition run, any level of HELP files can be built. However, HELP files require substantial disk space on an RL02. To minimize system software disk requirements on small remote systems, DECnet's RMT can be used to access front-end system HELP files.

During the definition, a special definition file, RSXNODE.DEF, is created for each target system. Each pre-built RSX system has a definition file that corresponds to it. During the definition, this file is copied to directory [200,200] on the target disk. Definition customization options and system startup options are appended to it. During a boot, the standard startup command file refers to RSXNODE.DEF to dynamically configure system options.

RSXNODE.DEF contains:

1. Size of system checkpoint space
2. Appended RSX customization selections (mentioned above)
3. Appended system startup options for the target system. These include:
   1. Activation of console logging (CO:)
   2. Establishing DCL as a command line interpretor
   3. Activation of error logging
   4. Installation of RMS tasks
   5. Automatic loading of DECnet at system startup
   6. Specification of a user startup command file

A virtual disk is used to store the pre-built RSX systems. RSX systems are assigned different group UIC's on the virtual disk. To create a system, files under the appropriate UIC's for that system are be copied to a target disk.

All systems support the resident FCS library. Therefore, only one copy on the RSX non-privileged tasks (utilities) is needed. They are kept in a separate directory, [1,53]. Like the Executive and privileged tasks, they, too, are copied to a target disk during the build step. This also allows us to fit an RSX/DECnet system on an RL02 and still leave ´some´ space for the scientist.

## 2.2 RSX-11M Driver Definition

As stated earlier, the only drivers defined for the systems are the full duplex terminal driver, and the psuedo drivers. Drivers for DEC Devices or DEC-emulated devices are created as loadable drivers with loadable data bases. This allows a unique set of peripherals to be built for each Chemistry node

This is done using the approach developed by Dave Strait and documented in the August, 1982 issue of the MULTI-TASKER. Device type and number of controllers are defined during the definition step and are written to the build file. During the driver build process, the following steps are completed for each driver.

1. Controller vector, CSR and peripheral units are defined and the loadable driver database is created. Several RSX sysgen command files including the pre-built RSX system answer file are invoked during this process.
2. The driver and database are assembled.
3. The driver object modules are then task built for the target system and placed in [1,54] on the target disk.

To include the driver in the target system, a new RSX system image must be created with VMR. In this process, each driver is included in DRVPAR by manually inserting a ´LOA´ command for each device.

## 2.3 DECnet End-node Definition

Pre-built DECnet end-node software exists for each pre-built RSX system. The build file created during definition selects a pre-built DECnet system based on the ID of the selected RSX system as defined in RSXNODE.DEF. DECnet systems are also located on the same virtual disk containing the RSX systems.

The DECnet build step makes only minor changes to these pre-built systems. During definition, the user is asked for the node number and the DECnet line characteristics. Currently, all our remote LSI systems are connected to the network via 9600 BAUD 20 ma asychronous serial lines. After coping the DECnet system to the target disk, the following customizations are made with CFE.

1. Define new executor characteristics (node name and number)

2. Define new line characteristics (vector and CSR address)
3. Define other Chemistry nodenames

## 2.4 DEC Layered Products And Other Distributed Files

Our current DEC layered products portfolio is:

1. Fortran 77
   - F77 compiler with FCS OTS, and/or
   - F77 compiler with RMS OTS
2. Datatrieve-11 (requires RMS)
3. RGL/11 (requires FORTRAN objects to be placed in SYSLIB.OLB to use the overlay facility)

All DEC layered products are kept on a separate disk. If a generation or installation procedure is supplied with the layered product, as DEC frequently does, an attempt is made to use it. The product installation procedure is invoked from the layered product build file. The build file mounts the source disk, makes necessary logical assignments, sets the proper UIC and invokes the product's installation command file. Upon completion of the product installation, it also dismounts the disk, deassigns logical names, etc. While this suffers from the approach that certain information has to be re-entered, eg. target disk, it does reduce the command file writing effort. It also makes installation of new versions of the layered product easier.

## 3.0 CONCLUSION

The final output of the system creation process is a bootable RSX system that is a unique node on the Chemistry network. The entire system creation process takes approximately one hour. Additionally, a software generation log file is automatically maintained. It contains entries for each DEC software product generated. Each entry includes the Chemistry node name, DEC product name, product version level, and creation, or build, date. We are hopeful that the process can be easily extended to support RSX-11M PLUS systems.

When we were finally ready to go, RSX-11M V4.1 update B arrived. But, then, that's how it goes in this business.

# USER-WRITTEN LOADABLE DIRECTIVES

Philip Miller
Century Computing, Inc.
Laurel, Maryland
(301) 953-3330

## Introduction

This article describes how to implement "loadable directives" under RSX-11M Version 4.0. The directives are contained in a task separate from the RSX monitor and may be loaded, tested, unloaded, changed, and re-loaded without repeated SYSGENs. The technique is inspired by the loadable driver capability of RSX.

## Why New Directives?

In general, we are reluctant to make changes to RSX. The system is quite good as it is and home-grown changes invite reliability and maintenance problems. (We know of several large application systems that are forever locked into some 3.x version of RSX because of uncontrolled, excessive RSX tinkering.)

In our own case, we were faced with converting a large real-time system from RSX-11D to RSX-11M. The -11D system had project-specific directives and our goal was to avoid non-trivial changes to the user mode software. Thus, we were faced with implementing a number of new directives; the loadable directive technique was devised to facilitate development of the new directives.

## Overview

The logic for the directives is contained in a privileged, executive-mapped task named LOADIR. When LOADIR is executed (the "loading" of the directives), the task stuffs some pointers into RSX to "announce" the presence of the new directives. From this point forward, RSX will enter the task--in system state--to process user requests for the directives.

The RSX changes to support loadable directives consist of several lines added to the directive dispatcher (DRDSP.MAC) and a new module (LDENTR.MAC) containing eight instructions.

The implementation here assumes that the executive common feature is selected during SYSGEN, i.e., that the DRDSP module is mapped via APR5 and is contained in a region separate from the main RSX monitor region.

## Directive Dispatcher Changes

The following changes were made to the standard RSX DRDSP.MAC module:

1. The directive dispatch table is defined to be a global symbol so that LOADIR can stuff new entries into the table.

2. A new dummy entry is defined in the dispatch table to reserve room for the entries stuffed by LOADIR.

3. The BYTTAB assembly flag is forced to be undefined. (The BYTTAB option creates a compressed form of the dispatch table that is not compatible with the loadable directive technique.)

4. New logic is required to detect loadable directives and branch to the LDENTR.MAC module in the main RSX region.

The DRDSP changes are listed below.

## The LDENTR Module

LDENTR is a bridge between the DRDSP module and the directive processing in LOADIR. When DRDSP detects a loadable directive, DRDSP branches to LDENTR in the main RSX region. LDENTR maps APR5 to LOADIR and calls LOADIR to process a directive.

LDENTR plays the same role for loadable directives that the RSX $OTHR1 subroutine plays for standard directives that are located in an executive common region separate from DRDSP.

The LDENTRY.MAC module is listed below.

## LOADIR

LOADIR is a privileged task and is mapped to the executive. Initially, LOADIR executes as a task to announce its existence to RSX. After initialization, LOADIR is entered from LDENTR to process the loadable directives as an extension of RSX.

LOADIR must be task built with the /PR:5 option and must be linked to the RSX11M.STB file and the EXCOM1.STB file.

LOADIR initialization consists of the following steps:

1. The T.IOC ("count of outstanding I/O operations") field of the TCB is bumped. This prevents RSX from shuffling LOADIR. (Note that fixing a task does not guarantee that the task will remain unshuffled.)

2. New entries are added to the directive dispatch table.

3. The $$LDA5 word in the LDENTR module is stuffed with the APR5 mapping of the LOADIR task.

4. The $$LDEP word in the LDENTR module is stuffed with the entry point in LOADIR of the directive processor.

5. LOADIR executes a STOP$ directive.

Steps 1 through 4 are performed in system state (via the SWSTK$ macro) to avoid race conditions with tasks that issue the loadable directives.

The task remains in the stopped state while directives are processed. If LOADIR is unstopped by the operator, the pointers in LDENTR are cleared, the T.IOC field is decremented, and LOADIR terminates. Unstopping LOADIR "unloads" the directives.

The dispatch table entries for the loadable directives have the directive identification code in the first word (as opposed to the address of the directive dispatcher that the standard RSX entries have). The directive code is always less than 256 and serves as a flag that the directive is to be processed in the LOADIR module.

The DIRENT entry point in LOADIR is entered from LDENTR when a loadable directive is executed. DIRENT uses the directive code (saved in $$LDDI by LDENTR) to determine the address of the proper directive processor in LOADIR.

A directive processor in LOADIR is entered with the exact register settings as for a standard RSX directive processor. The rules for writing directives are beyond the scope of this article (but may be inferred from studying the standard directives).

A sample listing of LOADIR appears below. The sample implements a simple directive that requests the version number of RSX and the processor model number.

## Refinements

The following are considerations for refinements to the loadable directive technique:

1. Support for RSX-11M+. (The technique is probably directly applicable, but no analysis has been done to verify compatibility.)

2. Conditional assembly logic to support systems without executive common.

3. A more "pool-efficient" scheme for the LOADIR task. After initialization, LOADIR takes up pool space for task tables even though LOADIR is not really needed as a task.

4. There is no abort protection for LOADIR. If the task is aborted, the directives are still available (because T.IOC is greater than zero), but the unstop operation is not accepted.

## SLP Updates to Version 15.35 of DRDSP.MAC

```
-2
   .IDENT  /LD101/          ;loadable directive  update
-459
   MAC     199.,0,0         ;reserve table space for loadables
-564,564
-584
$$DSPT::                    ;directive table as global
-906
   CMP     (SP),#256.       ;is this loadable directive?
   BHI     57$              ;brif not loadable
   JMP     $$LDEN           ;branch to root and enter LOADIR
57$:
/
```

## LDENTR.MAC Module

```
   .TITLE  directive dispatch for loadable directives
   .IDENT  /205/
   .MCALL  HWDDF$
   HWDDF$
;
;  This module is linked into the RSX root and receives
;  control to establish mapping for and call the
;  loadable directive module (LOADIR).
;
;  The following words are stuffed by LOADIR when attaching
;  to define new directives.
;
$$LDEP::.WORD       0       ;loadir entry point stuffed here
$$LDA5::.WORD       0       ;APR5 mapping for LOADIR stuffed here
;
;  The following is reached via a jump in the DRDSP module
;  when DRDSP determines that the directive being processed
;  is a special directive.
;
;  The mapping done here requires that this module be in the
;  RSX root and not in one of the executive commons.
;
   .GLOBL  $XCOM1             ;APR 5 mapping for EXCOM1 region
   .GLOBL  $DRSOK             ;entry in DRDSP for "directive ok"
   .GLOBL  D.RS99             ;bad directive code
;
$$LDEN::TST         $$LDA5           ;has LOADIR attached here?
   BNE     10$                ;brif yes
   DRSTS   D.RS99             ;bad directive; LOADIR not loaded
10$:
   MOV     (SP)+,$$LDDI       ;save identification code
   MOV     $$LDA5,@#KISAR5 ;establish mapping to LOADIR
   CALL    @$$LDEP            ;call LOADIR to process directive
   MOV     $XCOM1,@#KISAR5 ;re-establish mapping to EXCOM1
   JMP     $DRSOK             ;enter DRDSP for "directive ok"
;
$$LDDI::    .BLKW   1         ;directive code for use by LOADIR
   .END
```

<u>LOADIR.MAC</u> <u>Module</u>

```
   .TITLE  LOADIR  -- Loadable RSX directives
   .IDENT  /211/
   .SBTTL  initialization
   .MCALL  SWSTK$,TCBDF$,HWDDF$,DRERR$,EXIT$S,STOP$S
        HWDDF$
   TCBDF$
   DRERR$
   .GLOBL  $TKTCB,$$LDEP,$$LDA5,$$DSPT
;
LOADIR::
   MOV     $TKTCB,R0                  ;R0=TCB for LOADIR
   INCB    T.IOC(R0)                  ;bump to avoid shuffle
   SWSTK$  50$                        ;enter system state to
                                      ;serialize table access
   MOV     #$$DSPT,R0                 ;r0=RSX dispatch table ptr
        BIC         #160000,R0                ;clear page number
   BIS     #140000,R0                 ;force it to page 6 and...
   MOV     $XCOM1,@#KISAR6            ;map page 6 to EXCOM1
   ADD     #<LOWDIC*2-2>,R0           ;r0=ptr to first new entry
   MOV     #SPCENT,R2                 ;r2=from buffer
   MOV     #DIRWDS,R1                 ;r1=words to move
10$:
   MOV     (R2)+,(R0)+                ;stuff directive table
   SOB     R1,10$
   MOV     @#KISAR5,$$LDA5            ;stuff mapping word
   MOV     #DIRENT,$$LDEP             ;stuff entry point
```

```
        RETURN                              ;return from system state
    50$:
        STOP$S                              ;stop task; when unstopped by
                                            ;operator...
        SWSTK$  100$                        ;disconnect from DRDSP:
        CLR     $$LDA5                      ;directives now invalid
        CLR     $$LDEP                      ;
        MOV     $TKTCB,R0                   ;r0=tcb ptr for...
        DECB    T.IOC(R0)                   ;allow task to be exited
        RETURN                              ;return to...
    100$:
        EXIT$S                              ;terminate
```

LOADIR.MAC Module (cont'd)

```
        .SBTTL  Directive Entry Point
    ;
    ;   LDENTR calls us here in system state to process a special
    ;   directive. Registers are set as for an RSX directive
    ;   function:
    ;
    ;   r3 = mapped pointer to first word in DPB
    ;        (i.e., the word following the DIC/SIZE word)
    ;   r4 = pointer to task header of current task
    ;   r5 = pointer to TCB of current task
    ;
    ;   A return means successful directive execution.
    ;
        .GLOBL  $ACHKP                      ;buffer mapping/checking
        .GLOBL  $$LDDI                      ;dic of special directive
    DIRENT::
        MOV     $$LDDI,-(SP)                ;(sp)=directive code
        SUB     #LOWDIC,(SP)                ;(sp)=offset in table
        ADD     #BRTABL,(SP)                ;(sp)=ptr into branch table
        MOV     @(SP),(SP)                  ;(sp)=ptr to processor
        JMP     @(SP)+                      ;enter directive processor


        .SBTTL  INFO -- sample directive to get RSX information
    ;
    ;   This directive expects a DPB of the following form:
    ;
    ;   .byte   199.,2
    ;   .word   buffer
    ;
    ;   The buffer must be 3 words long to be filled as follows:
    ;
    ;   word 0: two-character base level id of the RSX system
    ;   word 1: two-character base level revision
    ;   word 2: processor model number (binary value)
    ;
```

```
;
INFO:
   MOV      (R3),R3                  ;pointer to user buffer
   MOV      #6,R1                    ;bytes in user buffer
   CALL     $ACHKP                   ;map r3 to user buffer
   MOV      $SYSID,(R3)+             ;set base level words
   MOV      $SYSID+2,(R3)+           ;
   MOV      $PRMOD,(R3)              ;set processor model
   RETURN
```

<u>LOADIR.MAC Module (cont´d)</u>

```
   .SBTTL  Tables
.PSECT     DATA,RW,D                         ;data areas
.PSECT     BRTAB,RO,D                        ;branch table
BRTABL::                             ;(assembled by DIR macro)
.PSECT  DATA                                 ;resume data areas
;
;  The directive dispatch table entries here are moved
;  into RSX during LOADIR initialization.  Each entry
;  has the same format as for a standard RSX directive,
;  except the first word is the directive code (and not
;  the directive processor address).
;
   .MACRO  DIR ENTRY,DPBSIZ,MASK
   .WORD   CURDIC                    ;directive code
   .BYTE   DPBSIZ*2                  ;dpb size
   .BYTE   MASK                      ;flags for RSX
   .PSECT  BRTAB                     ;switch to branch table
.=BRTABL+<CURDIC-LOWDIC>
   .WORD   ENTRY                     ;create branch table entry
   .PSECT  DATA                      ;back to data psect
CURDIC=CURDIC+2
   .ENDM
;
;  These mask bits are copied from DRDSP module.
;
ACHKDB=200          ;Check PLAS block
CEFNCL=100          ;Next DBP arg is efn
GEFUSE=40           ;Group global flag
CEFNMT=20           ;efn required
DFCTSK=10           ;task name may default to current
MUPCHK=4            ;multi-user protection checks
SRSTCL=2            ;no special action; next arg is task
;
;
LOWDIC=199.                 ;lowest loadable directive id code (DIC)
CURDIC=LOWDIC               ;current DIC
;
;  loadable directive entries for dispatch table.
;  one entry per loadable directive
;
```

```
    SPCENT::
       DIR      INFO,2,0              ;199. Get RSX info
    DIRWDS=<.-SPCENT>/2                     ;words in above tables
       .END     LOADIR
```

# Multi-user F11ACP for RSX-11M V4.1A

Multi-user F11ACP for RSX-11M v4.1a

C. H. Stockley
Sandia National Laboratories
Division 8411
Livermore, CA 94550

This is an addendum to an article titled:

F11ACP Performance Measurements for RSX-11M
by Joseph S. Sventek
Oct-81 MULTI-TASKER, Vol.15, No.4, pg 145.

If you have more than one disk drive and use dedicated F11ACP's for each drive, you should consider using Multi-user F11ACP's.

The standard FCPLRG ACP consists of 68% R-O sharable code and 32% R/W non-sharable code. Therefore each additional ACP only costs you a 32% increase in required memory.

The following table compares the number of standard FCPLRG ACP's versus multi-user MUFCP ACP's for the same amount of memory.

| | | | | |
|---|---|---|---|---|
| # of FCPLRG | 1 | 2 | 3 | 4 |
| # of MUFCP | 1 | 4 | 7 | 10 |

To build a multi-user F11ACP there are two utilities you will need that are available on the DECUS RSX-SIG tape RSX-F83.

| | |
|---|---|
| ORC (object module disassembler) | [312,315]ORC.ARC |
| LBL Tools kit (new release) | /BAC:LBLTOOLS |

There are four steps to build a multi-user F11ACP:

1. Modify module DISPAT to force the program entry point into the R/W segment.
   This is the only module that needs to be modified.

   a. Extract module DISPAT from FCP.OLB
   b. Disassemble using ORC to get a .MAC file
   c. Edit DISPAT.MAC changing:
      ```
      .IDENT /0254/  to  .IDENT /0254MU/
      .PSECT $1$COD  to  .PSECT DISPAT,RW,I
      ```
   d. Assemble as DISPATMU.OBM

2. Taskbuild and get a map. (See Appendix D)

   a. Assign TK: & MP: and TKB @FCPLMUBLD.TKB
   b. Look at FCPMU.MAP for the sizes of the two sections of code.

      ```
      R/W mem limits: 120000 133325 013326 05846.
      R-O mem limits: 140000 170477 030500 12608.
      ```

3. Run the LBL Tools kit to split the .TSK image into ROFCP.TSK, ROFCP.STB, & MUFCP.TSK

   a. Set /uic=[1,54] (wherever FCPMU.TSK is located)
   b. Run MUBLD under the Shell:

      ```
      % SHL -C MUBLD -V FCP
      .       pip mufcp.tsk=fcpmu.tsk
      .       fcpro.mac
                  |
                  |
      .       mac dummy=dummy
      .       tkb @dummy.tkb
      % ^Z
      ```

4. Run VMR to create the partitions and install:

   a. Check FCPMU.MAP for R-O & R/W partition sizes.
   b. Refer to MCR MOU command for explanation of default ACP names.
   c. Be sure and include the /PRI=149. sw on INS command, otherwise it will default to /pri=50.

      ```
      SET /MAIN=ROFCP:*:305:COM        ! R-O sharable
      INS ROFCP
      ;
      SET /MAIN=FCPPAR:*:(134 for each acp):SYS
      ;
      INS MUFCP/TASK=ddnnF1/PAR=FCPPAR/-CKP/PRI=149.
      FIX ddnnF1
      ;
      INS MUFCP/TASK=ddnnF1/PAR=FCPPAR/-CKP/PRI=149.
      FIX ddnnF1
      ```

```
;
INS MUFCP/TASK=ddnnF1/PAR=FCPPAR/-CKP/PRI=149.
FIX ddnnF1
```

## Appendix D - Taskbuild file for Multi-user F11ACP (RSX11M v4.1a)

```
;       [1,24]FCPLMUBLD.TKB
;
; Build entirely memory resident, multiuser F11ACP (FCPLRG)
;       for RSX11M v4.1.
;
; Module DISPAT is the only module that has been modified to
;               force the PRG xfr address into the RW section.
;
;       Name - FCPMU
;
;  NOTE: ****   must asn TK: & MP:
;
TK:[1,54]FCPMU/MU/AC:5/-IP/MM/-CP/-FP/PM,MP:[1,34]FCPMU/-SP/MA=
;
SY:[1,24]FCP/LB:F11ACP:F11CM:F11BUF:SMCOM:DIRBUF:FIOSUB
SY:[1,24]DISPATMU.OBM
SY:[1,24]FCP/LB:ARWVB:INIT:MXQIO:SMRVB
SY:[1,24]FCP/LB:ALLOC:BLXIO:CKSUM:CLACC:CLDAC:GTFID
SY:[1,24]FCP/LB:ATCTL:CRFID:DATIM:RATCM:RWATT:WATCM
SY:[1,24]FCP/LB:CLATT:CLCRE:CLDEL:CLDIR:GTMAP
SY:[1,24]FCP/LB:CLEXT:DARITH:DLBLK:DLHDR:DRGET:DELCK
SY:[1,24]FCP/LB:INFCB:MPHDR:MPVBN:NXHDR:PROCK:RDHDR:RLEAS
SY:[1,24]FCP/LB:RLFCB:RW1LB:SCFAC:SCFCB:WITRN:WRHDR:WTRN1
SY:[1,24]FCP/LB:DRINI:DRPAC:DRSEF:DRVLB:DRWRT:DWPND:FDRMV
SY:[1,24]FCP/LB:LOCAT:MATNAM:SMALC:SMDEL:SMNXB:WACCK
SY:[1,24]FCP/LB:CRFIL:DLMRK:DLFIL:TRUNC:ACCESS:DEACC
SY:[1,24]FCP/LB:CLNUP:CLFCB:CLCOM:EXTEN:EXINI:EXCOM:EXCMP
SY:[1,24]FCP/LB:IXEXT:EXTHD:RDATT:WRATT:RWVB:RWVBL
SY:[1,24]FCP/LB:FNDNM:DREX:RMVNM:ENTNM:DRACC:INWIN
SY:[1,24]FCP/LB:DREXT:DRALC:DRCPY:DREOF:DMOUNT
SY:[1,24]FCP/LB:CNTRL:MOUNT:MOUNT0:MOUNT1:MOUNT2
SY:[1,24]FCP/LB:CDTTA:CATDT:MONTB:DIVD:DIVQ
LB:[1,1]EXELIB/LB
SY:[1,54]RSX11M.STB/SS
LB:[1,1]SYSLIB/LB
/
;
TASK=F11ACP
STACK=50
UNITS=1
UIC=[1,1]
PRI=149
PAR=FCPPAR:00:00
IDENT=05.00
```

```
TSKV=.SSTVC:7
;
; ===================================================
; Allocate a separate buffer for index file bitmap.
;
; If you've memory to burn, enable the EXTSCT=$$BUF0:1006.
;
EXTSCT=$$BUF0:1006
;
; ===================================================
; Allocate a separate buffer for file headers.
;
; Always enable the EXTSCT=$$BUF1:1006 if possible.
;
EXTSCT=$$BUF1:1006
;
; ===================================================
; Allocate private FCB pool space.
;
; The value of the EXTSCT determine the size of the
; internal FCB pool which doesn't help performance
; but cuts down on the use of system pool.  The optimum
; size is the size of an FCB (about 54(8)) times the
; number of files open plus preaccessed directories (LRU).
;
EXTSCT=$$AFR1:3000
;
; ===================================================
; Extend the directory buffer.
;
; The EXTSCT=$$BUF3 determines the number of additional
; blocks of directory buffer to allocate.  Determine the
; size of the most frequently used directories and then
; extend $$BUF3 by that value minus one times the block
; size (1000(8)).
;
EXTSCT=$$BUF3:2000
;
/
```