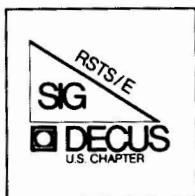


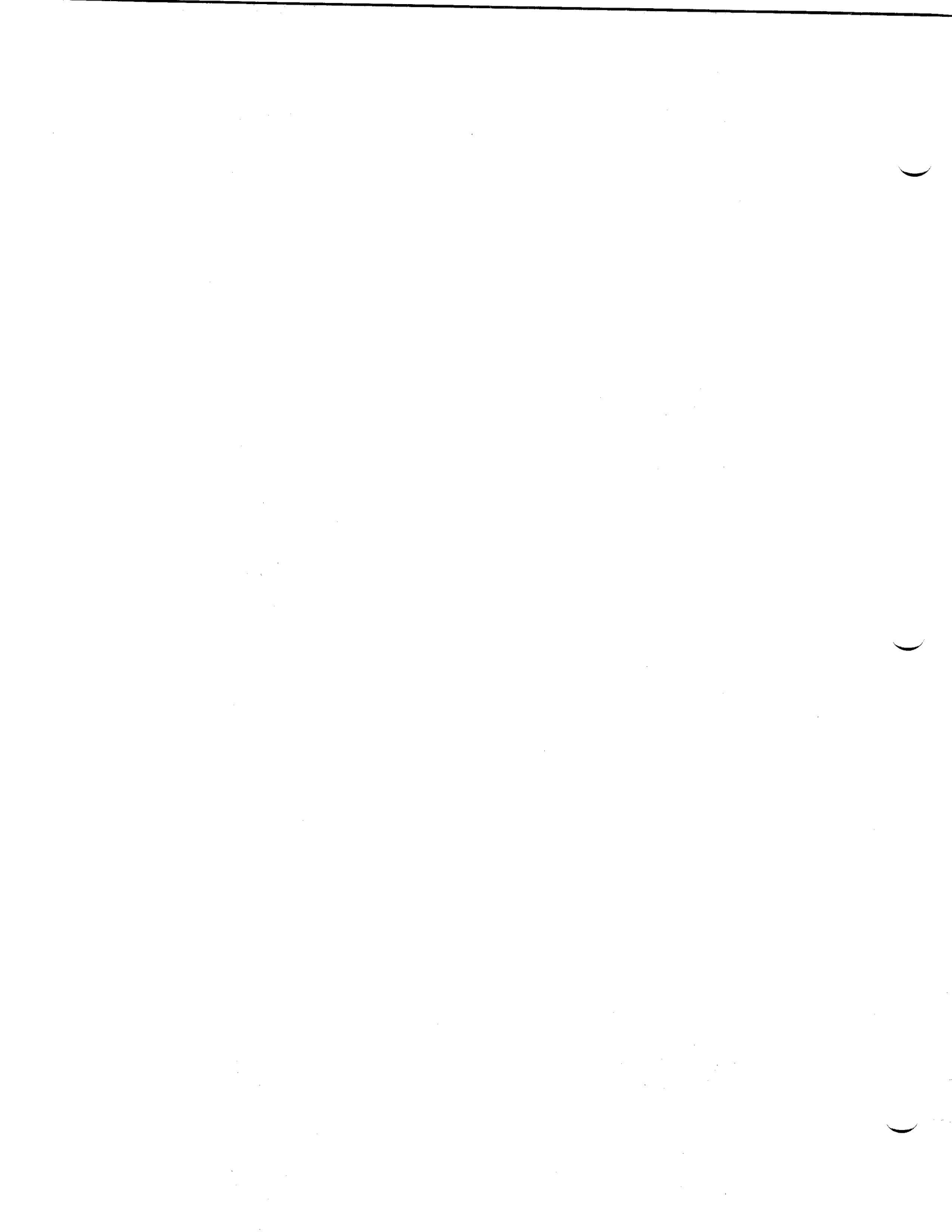
The Cache Buffer

*

APRIL 1985 ISSUE

DECUS
Subscription Service

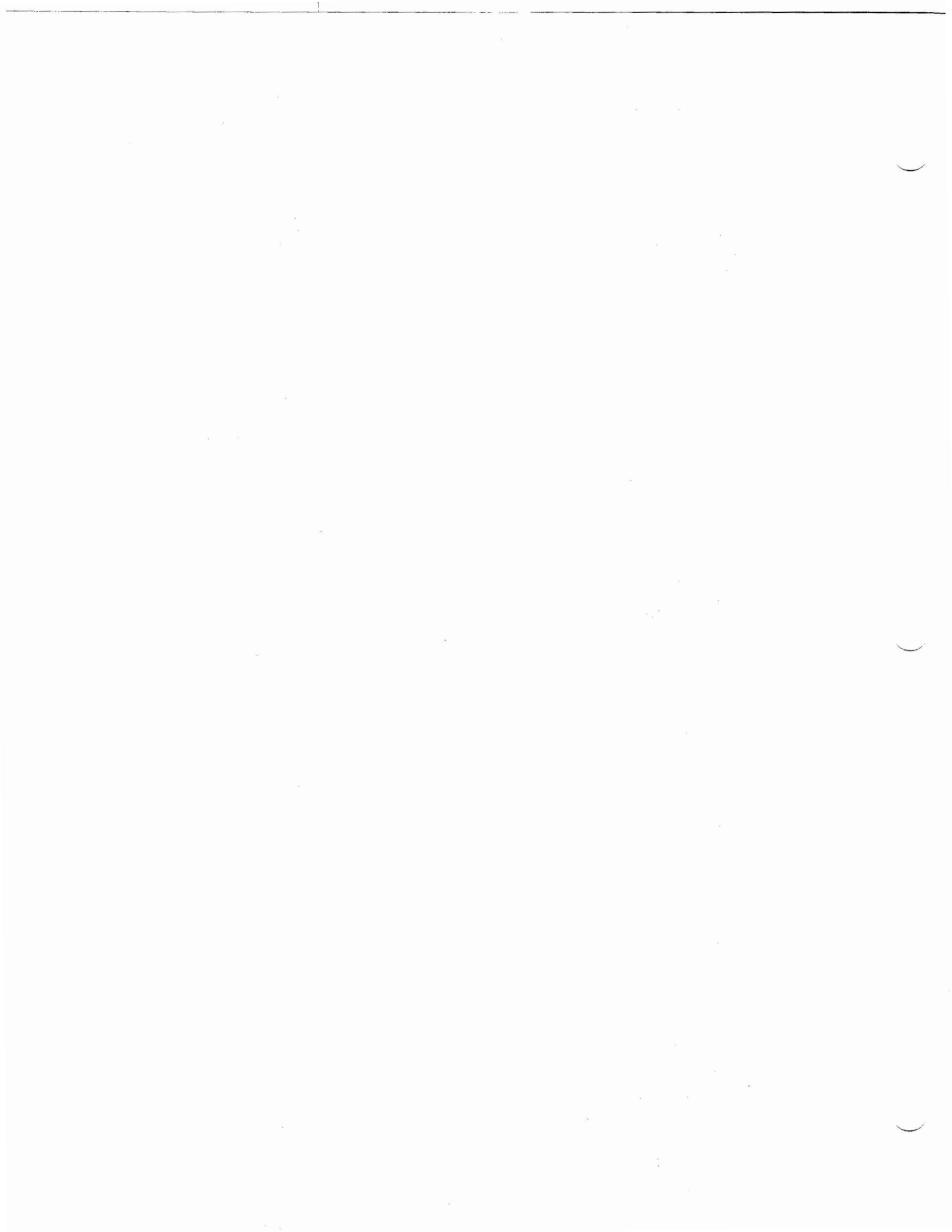




The CACHE BUFFER
RSTS/E Special Interest Group
Newsletter - Winter/Spring '85

In this issue:

<u>page</u>	<u>content</u>	<u>author</u>
1	From The Editor	W. Hobbs
2	Letters To The Editor	
5	Preview Of Spring '85 Symposia	S.W. Pandorf
9	RSTS/E- BASIC +2 Magic	Steven Edwards
14	RSTS Sig Tape Dist. Tree	Philip Hunt
17	Excerpts From Fall '84 Sysmposia Sessions	RSTS Development Team Members



From The Editor-

Well suprise, suprise, here's yet another version of the ever popular RSTS/E SIG newsletter, THE CACHE BUFFER.

As with the last edition, the submissions are rather scarce, but unlike the last edition, we do have some. Hopefully, this copy has reached your desk prior to the symposia in New Orleans. Included with this edition, you will find articles on what's planned for New Orleans, BASIC +2 Magic, a collection of the handouts and material presented at Anaheim by the RSTS development team, and other articles of interest.

Unfortunately, we did not have a submission for the RSTS ANTIQUES section but we will shoot for one in the summer edition. We also have not got our "ASK THE DEVELOPMENT MANAGER" section off the ground, but I have been assured by Mr. Joel Arker, RSTS Development Manager, that if you send in some questions, we will have some answers to publish in the next edition.

To those who have submitted, my sincere thanks and keep up the good work. To those who have yet to submit, this is your newsletter, it's only as good as you make it.

Looking forward to seeing you all in New Orleans, have a safe spring and early summer.

Bill Hobbs
Cache Buffer Editor

ps: The address for submissions is still

W.H. Hobbs
Director of Operations and Services
ComMand Inc.
6535 E. 82nd. St. Suite 102
Indianapolis, In. 46250

Letter to the Editor:

RSTS, RT11/TSX, AND A-TO-Z

If you have not heard by now, Digital has announced a new interface standard called A-TO-Z. This software is now running on RSX and is rumored to be moving to VMS. A-TO-Z is a sophisticated user interface or menu. The concept behind A-TO-Z is that any application can run under A-TO-Z and present the same user interface. The application developer who follows the A-TO-Z standard can bundle other A-TO-Z applications with his and make it look like one integrated solution. A-TO-Z is a very good idea.

What does A-TO-Z mean to a RSTS or RT11/TSX user? Not much on the plus side. Under current Digital thinking RSTS and RT11 are NOT going to get A-TO-Z. The excuse is that A-TO-Z supports a job/task rollout-rollin feature which RSTS and RT11 currently cannot support. I will leave aside the issue of how difficult it really would be to implement this under RSTS and RT11. Even if we accept the statement that job rollout-rollin cannot be implemented under RSTS and RT11, the most valuable part of A-TO-Z, the standardized user interface, can be implemented. I will go so far as to state that it could be implemented without much effort. So here we have Digital using a smoke screen excuse for not giving RSTS and RT11 a very valuable tool.

You may say 'so what - I can still sell my application.' But the plot thickens. Digital is planning on setting up a network of application developers who use A-TO-Z. The idea is that a local OEM who does not have a general ledger but has an MRP package will contact Digital who will then put him in touch with another A-TO-Z developer with an accounting package. The local OEM now sells a complete solution. The problem is that if you have been selling that complete solution under RSTS or RT11, i.e. you invested the time to do it right, you now have competition you never had before. Even more interesting is that it will be Digital endorsed competition. A-TO-Z is much more than just a another software product.

The end result is that you may be forced to move to RSX for no valid technical reason. The question becomes is A-TO-Z being restricted from RSTS and RT11 for valid technical reasons or desire on the part of some product manager to promote RSX.

If you are marketing applications under RSTS or RT11/TSX I need to hear from you. I want to develop a profile of persons marketing software which uses RSTS or RT11/TSX. We need to know who you are, if you want to move to and why or why not, and how many sites you have installed. Please write to:

JEFFREY J. KILLEEN
INFORMATION DESIGN AND MANAGEMENT, INC.
SUITE 2
41 NO. MAIN ST.
SHERBORN, MA. 01770

617-655-0877

PURDUE
UNIVERSITY AGRICULTURAL DATA NETWORK

March 26, 1985

Mr. W. H. Hobbs
Director of Operations & Services
Computerized Management Decisions, Inc.
6535 E. 82nd Street, Suite 102
Indianapolis, IN 46250

Dear Bill:

The "RSTS Antiques" column for the Cache Buffer newsletter is a nice touch. It made me feel old, however, when the first contribution was a Software Product Description of RSTS version 6A. I'm still using version 6A.

We have a PDP-11/45 that has been running RSTS 6A since it was released. It provides services to research scientists in the Purdue School of Agriculture and is part of a hierarchical network including CDC 6000's and a CYBER 205 at the top end and intelligent laboratory instruments at the bottom. Services include timesharing, data and virtual terminal communications to other computers at Purdue, and realtime data collection through another antique, a PDP-11/15 running a home grown operating system. In 1979, a second 11/45 running RSTS 6A was added since the first machine had become saturated with use.

The reason for not upgrading from 6A is a story as long as the history of RSTS itself. The data collection activities predated the availability of realtime systems like RSX by three to four years, so a large amount of special software was developed, including modifications to internal RSTS (using blind patches since sources were not available). Upgrading was therefore very labor intensive, and difficult to justify, since the existing system was in production and people were relying on it. Laboratory data collection has become more sophisticated in the last decade and central data collection is no longer required, but computer services are still needed by our users. We are currently planning the upgrade of these services. The 11/45's will probably be turned off in 1986, after fourteen years in production, most of which was conducted with RSTS version 6A.

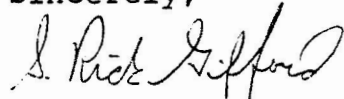


Smith Hall
West Lafayette, Indiana 47907

Someone once told me that something can't be called an antique unless it is still being used. People sleep in antique beds that cost more than new beds. People drive antique cars that cost more than new cars to maintain. Other computers on campus run the latest version of RSTS and other operating systems, but our little RSTS systems keep right on doing their job very reliably. I guess "antique" is not such a bad label.

I would be interested in finding any other sites using old RSTS/E. If you want a relic, I'll go through my files. I think I still have an SPD for RSTS version 5A.

Sincerely,



S. Rick Gifford
Ag Data Network Software Manager
Smith Hall 105
Purdue University
West Lafayette, IN 47907

PREVIEW OF THE SPRING 1985 DECUS SYMPOSIUM

S.W.Pandorf, RSTS SIG Symposium Coordinator

I would like to take this opportunity to invite you to attend the Spring 1985 DECUS Symposium in sunny, balmy, New Orleans, Louisiana, May 27-31 1985. DECUS Symposia are always a good place to catch up on the 'latest and greatest', or to just get your questions answered. We feel that we will again have an excellent symposium for all attendees, with RSTS sessions ranging from version 9 status updates, to 'how to' sessions.

The SIG is again sponsoring the pre-symposium seminar given by Bruce Gaarder a version 9 conversion tutorial.

This session will be chaired by Bruce Gaarder of Macalester College, a past RSTS SIG chairman, and long time RSTS and layered product field test site. This tutorial is designed for those attendees who would like to be prepared for the next release of RSTS (version 9), which should be released to the Software Distribution Center about the time of the symposium. While the session is not designed as a version 9 features overview it will touch on many of the version 9 enhancements, and how the various field test sites worked with, and adjusted to, these new features.

This session was well received in Anaheim, and we encourage all those planning to convert to version 9 to register and attend.

Beginning Monday, May 27th, we have a very full week of RSTS sessions planned, a roadmap follows, but here are a few of the highlights:

Monday morning 9:00am (you did want to get up didn't you!) the SIG roadmap and business meeting, there are changes in your SIG leadership which will be announced at this session, as well as any last minute changes to the overall schedule. Following this through the day are various RSTS sessions given by both the RSTS development team, and the SIG. Monday ends with the amazing, incredible, RSTS Magic, Horror Stories, and Tales from Beyond.

Tuesday mornings sessions are 'how to' sessions, while Tuesday afternoons sessions are on Micro RSTS, and DECNET/E.

Of special note Wednesday is the RSTS version 9 installation demo, we will have a live RSTS system, and large screen display facilities so the development team can do an 'on-line' installation for all to see, and answer any questions for the attendees.

New Orleans marks the 15th Birthday of RSTS, and OF COURSE we'll have a party Thursday night to celebrate, cake, a special present, and generally a good time will be had by all.

Friday morning, for those that stick it out, has a good session by Bruce Gaarder on writing version 9 DCL command procedures, and a session on what's on the SIG tape.

Once again, plan to attend, we'll look forward to seeing you in New Orleans.

SWPandorf Indianapolis, IN March 1985.

ROADMAP FOR NEW ORLEANS

This is the RSTS SIG Roadmap for the Spring 1985 DECUS Symposium in New Orleans. It outlines all of the sessions (DEC,USER,SIG) sponsored by the RSTS SIG. This document is intended to be a starting guide for the Spring symposium, it is not exhaustive, nor does it indicate any session changes or BOF (Birds-of-a-Feather) sessions that may be scheduled at the symposium.

SWPandorf Indianapolis, IN March 1985

RSTS SPRING 1985 SYMPOSIUM

Sessions by Day and Time

Session	Title	Day	Time	Source
RS005	RSTS SIG Roadmap and Business Meeting	Mon	09:00	SIG
RS023	RSTS/E Announcements and Futures	Mon	10:30	DEC
	-- Lunch --			
RS028	Print/Batch Services in Version 9	Mon	13:30	DEC
RS024	RSTS/E Version 8 to 9 considerations	Mon	14:30	DEC
RS022	Caching Well on RSTS	Mon	16:00	USER
	-- Dinner --			
RS033	MODULA II Migrating from RT11 to RSTS	Mon	17:00	USER
RS012	RSTS Tech Tips I (SPR)	Mon	19:00	SIG
RS020	RSTS/E Magic, Horror Stories, etc.	Mon	20:30	USER
RS020	RSTS Damaged Disk Recovery	Tue	09:00	USER
RS031	How to add UUO's to RSTS	Tue	10:00	USER
RS002	Well Structured Backups under V9	Tue	11:00	USER
	-- Lunch --			
RS027	Next version of RSTS overview	Tue	16:00	DEC
RS003	Remote Site Support for RSTS	Tue	17:00	USER
RS030	DECNET/E Version 3 features	Tue	18:00	DEC

RSTS SPRING 1985 SYMPOSIUM

Sessions by Day and Time

Session	Title	Day	Time	Source
RS026	RSTS/E Layered Product Panel	Wed	09:30	DEC
RS013	RSTS Tech Tips II (Bits and Bytes)	Wed	11:00	SIG
	-- Lunch --			
RS025	RSTS/E Version 9 Installation Demo	Wed	15:00	DEC
	-- Dinner --			
RS016	DECWORD/DP Product Panel	Thu	09:00	DEC
	-- Lunch --			
RS010	RSTS Wishlist Session	Thu	12:00	SIG
RS021	The Ins and Outs of RSTS Device Drivers	Thu	13:30	USER
RS009	RS232 Magic - Understanding Cables	Thu	16:00	USER
	-- Dinner --			
RS029	Announcing RSTS Version 4a	Thu	18:30	SIG
RS032	RSTS 15th Birthday Party	Thu	19:30	SIG
RS001	RSTS/Basic SIG Tapes What's on them	Fri	09:30	SIG
RS006	DCL Command Procedures for Version 9	Fri	10:00	USER

All Sessions will be held in the New Orleans Hilton.

The RSTS Suite (there will be no campground) will be in the Hilton, watch the boards for specific room.

DECUS is a participatory organization, that is, you the member/attendee are DECUS. Your participation is needed and desired. Digital sponsors many sessions during the week, but many more are user sponsored, and those users are **YOU**. There will be many chances throughout the symposium to contribute and participate. Take some time and consider participating at the next symposium. RSTS Steering committee members are easily identified by their badges, seek one of us out, and offer to participate.

Bill Hobbs
Cache Buffer

March 18, 1985

Dear Bill,

Sorry for the extreme delay in forwarding this material to you. The RSTS magic enclosed is used as follows:

The first bit of magic is a patch to the BASIC-PLUS-2 object library module "STMSC" that causes it to "beep" whenever the string compactor or "garbage collector" is invoked. The patch is installed in the same fashion as any Digital supplied patch. It is suggested that this patch not be applied against your production object library.

The second bit of magic is an addition to the Basic-Plus run-time system that implements the un-implemented SysCalls (10 & 13). The module is assembled with COMMON.MAC and linked into the run-time system. The example implementation of Sys(Chr\$(10)) allows the user to dump the current image to a specified disk file. The example implementation of Sys(Chr\$(13)) executes a machine code instruction sequence passed by the user program as a string.

The third bit of magic showed how to execute machine code from a BASIC-PLUS-2 program without the use of sub-programs. Unfortunately, this bit got away from me and I don't have the original anymore. The magic can be described as follows: create a mapped array in your program with the name of "FOO". In the body of the program, assign each word of the array with the decimal opcode for the machine code instruction sequence you want executed. After the array is loaded, execute the statement "CALL FOO". The trick is that the module must be compiled with the "/MAC" qualifier, then assembled by the macro assembler. The assembler will resolve the call address to the first address in the map.

For both the second and third bits, don't forget that the code must be position independent and end with a "RTS PC".

Good luck

Steven L. Edwards
Software Techniques, Inc.

; Patch to cause the BASIC-PLUS-2 object-time system to 'beep'
; when the garbage collection routine is called.

; Copyright (c) 1982, 1983 by Software Techniques, Inc.
; Los Alamitos, CA 90720

.TITLE SSTMSC

.IDENT /SOFTEC/

.PSECT BP2OTS,RW,I,LCL,REL,CON

\$\$\$\$\$\$ = . ; SAVE THE START OF THE PSECT.
. = \$\$\$\$\$\$+46 ; OFFSET TO THE CALL FOR THE
 ; STRING COMPACTOR.
CALL BP2BEL ; CALL THE BEEPER.
. = \$\$\$\$\$\$+610 ; OFFSET TO THE ROUTINE WE
 ; REPLACED.
COMPAC: ; (MAKE UP A LABEL.)

.PSECT BP2BEL,RW,I,LCL,REL,CON

BEL: .WORD 7 ; BEEP.

BP2BEL:

MOV R1,-(SP) ; SAVE R1.
MOV #442,R1 ; ADDRESS OF XRB.
MOV #1,(R1) ; LENGTH OF BEL.
MOV (R1)+,(R1)+ ; " " "
MOV #BEL,(R1)+ ; ADDRESS OF BEL.
.REPT 4
CLR (R1)+ ; ZAP A WORD.
.ENDR
EMT 4 ; .WRITE (ASSUME NO ERRORS).
MOV (SP)+,R1 ; RESTORE R1.
JMP COMPAC ; COMPACT THE STRINGS.

.END

.Enabl LC

Title UI,<BASIC-PLUS UN-IMPLEMENTED SYS CALLS>,01,30-NOV-81,<SLE>

; Written by: STEVEN L. EDWARDS
;
; Date: 30-NOV-81
;
; Package: In-House
;
; Description: IMPLEMENT THE UN-IMPLEMENTED SYS CALLS

; Copyright (C) 1981
; Software Techniques
; Los Alamitos, CA 90720
;

; This software is furnished under a license for use only on a
; single computer system and may be copied only with the inclusion
; of the above copyright notice. This software, or any other
; copies thereof, may not be provided or otherwise made available
; to any other person except for use on such system and to one who
; agrees to these license terms. Title to and ownership of the
; software shall at all times remain in Software Techniques.
;

; The information in this document is subject to change without
; notice and should not be construed as a commitment by Software
; Techniques.
;

; This software is un-released and Software Techniques has no
; commitment to support it at this time, unless stated elsewhere in
; writing.
;

.Sbttl Calling Format

; UI
; --
; TEMP.0\$ = SYS(CHR\$(13) + FILENAME\$)

; Arguments:

; Name Description
; ----
; FILENAME\$ FILE NAME TO DUMP IMAGE TO.

; UI2
; ---
; TEMP.0\$ = SYS(CHR\$(10) + CHR\$(0) + MC\$)

; Arguments:

; Name Description
; ----
; CHR\$(0) ALIGN MC\$ ON WORD BOUNDRY.
; MC\$ MACHINE CODE STRING.

.Sbttl Modification History

; Ver/Edit Date Reason (Who)

```

; -----
; V7.0-01          30-NOV-81      Initial conception.
;
; .Sbttl          Program Description
;
; THIS MODULE IMPLEMENTS THE UN-IMPLEMENTED BASIC-PLUS SYS
; CALLS. THE FIRST CALL (SYS 10) DUMPS THE CURRENT PROGRAM TO THE
; SPECIFIED FILE. THE SECOND CALL (SYS 13) EXECUTES THE MACHINE
; LANGUAGE CODE STRING PASSED TO US.
;
; .Sbttl          Assembly instructions:
;
; MACRO UI = COMMON/P:1, UI
;
; .Sbttl          Global Symbols
;
; .Globl UI2, B.4
;
; .Sbttl          Variable Description and Initialization
;
; .Psect UI, RW, I, GBL, REL, OVR
;
CHAN      =      17*2          ; CHANNEL 15.
FATAL    =      200          ; FATAL ERROR BIT.
LENGTH   =      4            ; OFFSET INTO STRING HEADER.
;
; .MACRO CHKERR ?A
; TSTB @#FIRQB          ; ERROR?
; BEQ A                ; GOOD.
; TRAP FATAL+1         ; LET BASIC HANDLE IT.
;
A:
; .ENDM CHKERR
;
; .Sbttl          SYS CALL 10.
;
; .Psect UI, RW, I, GBL, REL, OVR
;
UI:      PUSH <R0,R1,R2,R3,R4,R5> ; SAVE THE REGISTERS.
; .STAT                ; GET JOB STATS
; MOV @#XRB+XRLEN,R2   ; OLD CORE SIZE.
; CALL SETXRB          ; CLEAR XRB.
; MOV LENGTH(R1),(R0)  ; LENGTH OF FILENAME.
; DEC (R0)              ; ADJUST FOR FUNCTION CODE.
; MOV (R0)+,(R0)+      ; " " "
; MOV R3,(R0)          ; ADDRESS OF FILENAME.
; CALL SETFQB          ; CLEAR FIRQB.
; .FSS                 ; SCAN THE FILENAME.
; CHKERR               ; CHECK FOR AN ERROR.

```

```

MOV#B      #CREFQ,(R0)+      ; CREATE FILE FUNCTION CODE.
MOV        #CHAN,(R0)       ; CHANNEL 15.
ASL        R2                ; OLD IMAGE SIZE TIMES 2
MOV        R2,@#FIRQB+FQSIZ ;   FOR FILESIZE.
CALFIP
CHKERR                                           ; CHECK FOR AN ERROR.

CALL      SETXRB              ; CLEAR XRB.
ASH      #11,R2              ; OLD IMAGE SIZE TIMES 2048
MOV      R2,(R0)             ;   FOR BYTE COUNT.
MOV      (R0)+,(R0)+         ;   "   "   "   "   "
TST      (R0)+               ; STARTING AT 0.
MOV      #CHAN,(R0)         ; CHANNEL 15.
.WRITE
CHKERR                                           ; CHECK FOR AN ERROR.

CALL      SETFQB              ; CLEAR FIRQB.
MOV#B      #CLSFQ,(R0)+     ; CLOSE CHANNEL FUNCTION CODE.
MOV        #CHAN,(R0)       ; CHANNEL NUMBER.
CALFIP
CHKERR                                           ; CHECK FOR AN ERROR.

RETURN:   POP      <R5,R4,R3,R2,R1,R0> ; RESTORE THE REGISTERS.
          RETURN   ; BACK TO BASIC(S).

SETFQB:  PUSH     <#FIRQB+FQFUN,R1>    ; SAVE R0, R1.
          MOV      #FIRQB,R0          ; POINT AT FIRQB
          MOV      #<FQBSIZ/2>,R1     ; SIZE OF FIRQB IN WORDS.
          BR       CLEAR              ; JOIN COMMON CODE.

SETXRB:  PUSH     <#XRB+XRLEN,R1>      ; SAVE R0, R1.
          MOV      #XRB,R0           ; POINT AT XRB.
          MOV      #<XRBSIZ/2>,R1     ; SIZE OF XRB IN WORDS.
          .BR      CLEAR              ; JOIN COMMON CODE.

CLEAR:   CLR      (R0)+             ; ZAP A WORD.
          SOB      R1,CLEAR          ; UNTIL WE ARE DONE.
          POP      <R1,R0>          ; RESTORE R0, R1.
          RETURN   ; BACK TO MAINLINE CODE.

;
; .Sbttl1          SYS CALL 13.
;

.Psect   UI2, RW, I, GBL, REL, OVR

UI2::   BIT      #1,LENGTH(R1)      ; ODD STRING LENGTH?
          BEQ     10$,                ; BRANCH IF EVEN.
          TRAP   FATAL+B.4           ; FATAL OUT WITH ODD ADDRESS.
10$:    PUSH     <R0,R1,R2,R3,R4,R5> ; SAVE THE REGISTERS.
          CALL   1(R3)               ; CALL THEIR CODE.
          JMP    RETURN              ; BACK TO BASIC(S).

.End

```


Directions for use:

Above this sheet is a diagram showing boxes with numbers and letters inside. Please send copies of enclosed tapes in DOS format at 1600 BPI to the people that are connected to your box by an arrow. PLEASE SEND A COPY (EXTRAS ENCLOSED) OF THE DISTRIBUTION LIST TO PEOPLE YOU SEND TAPES TO and let them know that this file is on the Spring '84 tape as [1,1]distr.lis.

Note: The most tapes anybody must send out is 2 copies, and some people do not have to send any. If the tape is returned or you would like to let the person know a tape is coming, I have included phone numbers when possible so you may contact them. Please send these tape AS SOON AS POSSIBLE!!!!!! Any questions, please contact me at the above number.

Any errors or omissions or updates, please contact me and I will update the list. Thank-you.

Gene Alpern
Saber Computer Services
1975 Johns Drive
Glenview, ILL 60025
312-998-5950

B Doug Bickford
Univ of Vermont
Academic Computing Center
Cooke Science Bldg
Burlington, VT 05401
802-656-3190

Hank Vander Waal
Prime Metals
3910 Roger Chaffee
G.R., MICH 49508
616-241-3451
David Taft
USGS Box 25046 - MS 978
Denver Federal Center
Denver, CO 80025
303-234-6479

2 Jim Bivins
TD Industries
13737 North Stemmons Pkwy
Dallas, TX 75234
214-620-1511
4 Gene Dugger
Harding University
Box 753
Searcy, ARKANSAS 72143
501-268-6161 x266

Steve Lorentzen
555 Fourth + Battery Bldg

Seattle, WA 98121
206-223-6453

6 Dan Tipton
ORFMA
117 Flint Road
Oakridge, Tn 37830
615-482-5000

Lee Gilbreath
SignAd, Inc.
PO Box 8626
Houston, TX 77249
713-861-6013

8 Guy Dunbar
Wm. E. Davis and Sons, Inc
73 NW 122nd Street
Oklahoma City, OK 73114
405-751-4660

Jerry Ethington
Federal Land Bank
201 W. Main St/PO BOX 32390
Louisville, Ky 40232

1 Tom Haase
SOHIO
4440 Warrensville CTR Rd
Cleveland, OH 44128
216-581-5724

3 Frank Atkinson
1900 E. Dublin-Granville Rd
Suite 100
Columbus, OH 43229
614-889-0841

5 Robert Walraven
Univ of California
Applied Science
Davis, CA 95616
916-752-0360

7 Tom Gerhard
Advanced Data Management
15 Main St
Kingston, NJ 08528
609-799-4600

9 Roger Engleman
Computer Partners, Inc.
2995 N. Cole Suite 140
Boise, ID 83704
208-377-2070

1 Robert Fairchild
Nebraska Wesleyan Univ
50th & St. Paul
Lincoln, NE 68504
402-466-2371

3 Pierre M. Hahn
SUNY - HSC - T10

Stony Brook, NY 11790-8101

5 Jim Reed
Federal Land Bank of Wichita
151 N. Main
Wichita, KS 67202

7 ***OPEN***

10 Robert Perry
Tektronix Inc
Box 500, M/S 19-333
Beaverton, Oregon 97077
503-627-5410

12 Kathy Furtik - Computer Svcs
Lockwood Green Engineers
PO Box 491
Spartanburg, SC 29304
803-578-2000

14 Scott Matthews
Dow Chemical-Tx Div
A-2708 Railroad Dept
Freeport, Texas 77541
409-238-4816

16 Micheal Newell
Florida Inst of Technology
150 W University Blvd
Melbourne, Fl 32901
305-723-3701 x255

18 Marion Fauber
PO Box 1042 MS 4-07

Dayton, Oh 45401
513-258-7707

20 Richard Wrenn
Washington Univ-Biochemistry
Campus Box 8094
St. Louis, MO 63116
314-362-3354

22 Virgil Larson
Univ of Minnesota
Box 732 Mayo, Surgery Dept
Minneapolis, MN 55455
612-376-8280

24 Rocky Hayden
Userware International
2235 Meyers Ave.
Escondido, Ca 92025

26 ***OPEN***

NOTICE

The following pages are excerpts from the handouts and presentation material used by the RSTS/E development team during presentations given at the Anaheim Fall 84 symposia. Due to the various stages of development of the products mentioned in these pages, Digital, nor the Cache Buffer editor, assume any responsibility for the accuracy or completeness of the material presented. Use this material as a guideline for what may be expected in future releases of the products mentioned but not as a overview of the final released products.

FUTURE RSTS HARDWARE SUPPORT

New Hardware supported with RSTS V8.0-7

New Hardware features supported in RSTS V9.0

New Hardware Support
in RSTS V8.0-7 Update Kit "E"

MicroPDP-11/73

- o J11 based processor
- o 11/70 and 11/44 memory management
(Instruction and data space)
- o FP11 floating point instruction set
- o 8Kb cache memory
- o PDP-11/70 CPU registers
(CPU error and Cache control registers)

New Hardware Support

in RSTS V8.0-7 Update Kit "E"

RD52

- o 5 1/4" fixed media disk
- o 31.9 Mbyte capacity
- o QBUS only
- o Same electronic connections as RD51
- o MSCP protocol, device designator "DU"
- o Controller initiated Bad Block Replacement

New Hardware Support

in RSTS V8.0-7 Update Kit "F"

RC25/RCF25 disk subsystem

- o 8" fixed and removable disk subsystem
- o 26Mb fixed platter/26Mb removable cartridge
- o QBUS and UNIBUS interface
- o 2 drives/controller (1 master/1 slave)
- o MSCP protocol, device designator "DU"
- o Host initiated Bad Block Replacement

New Hardware Support

in RSTS V8.0-7 Update Kit "G"

RUX50

- o UNIBUS version floppy disk controller
- o Interfaces RX50 subsystem to any PDP-11 system
- o One quad-height board
- o 18-bit addressing
- o MSCP protocol, device designator "DU"

New Hardware Support
in RSTS V8.0-7 Update Kit "E"

TK25

- o 1/4" cartridge tape subsystem
- o stand-alone drive in a table-top enclosure
- o approximately 45Mbytes per cartridge
- o 10Trk, 55inches/second
(serial recording in a serpentine pattern)
- o QBUS only
- o Can be used as a bootable device.
- o TS11 protocol, device designator "MS"

(TK25 media will not be offered for
Digital software distribution.)

- o TK25 Performance in V8.0-7;
 - Approximately 1/2 - 1Mb per minute
(depending upon record size and operation)
 - WRITE operations will stream the TK25
 - READ operations will cause tape repositioning

o Offline SAVRES	+-----+-----+	without verify		with verify	
RD51 to TK25	+-----+-----+	14 minutes		35 minutes	
RD52 to TK25	+-----+-----+	41 minutes		105 minutes	

Verify is highly recommended with cartridge tapes.

New Hardware Features
supported in RSTS V9.0

- o Streaming tape support
- o USER MODE I and D space
- o Virtual Disk

RSTS V9.0 will provide streaming capabilities for the TU80, TSV05 and TK25 tapes.

- o TU80 - (UNIBUS) 1600 bpi tape
(Top loading, manual thread)
Controller will automatically switch into 100 ips if I/O requests are issued fast enough.
- o TSV05 - (QBUS) 1600 bpi tape
(Front loading, auto-thread)
Driver will select 100 ips if I/O requests are issued fast enough.
- o TK25 - (QBUS) cartridge tape
(Front loading, cassette tape)
TK25 will always operate at 55 ips.

Asynchronous I/O in RSTS

- o All disk drivers
- o All "MS" class tapes (TS11, TSV05, TU80, TK25)
- o Simulated for "MM" and "MT" tapes
- o Only new BACKUP utility will use Asynchronous I/O
(no streaming support in PIP or SAVRES in V9.0)

RSTS V9.0 hardware features

USER I and D SPACE

- o User developed MACRO, BP2, and FORTRAN/77 programs can expand to 64K words using separate I and D space memory mapping.
- o Programs must be built using the RSX task builder.
- o Restricted to non-overlaid tasks.
(for V9.0)

Programming restrictions

- o Instruction space can contain only instructions, immediate operands, absolute addresses and index words.
- o The MARK instruction cannot be used.
- o Instruction space cannot contain subroutine parameters which are data (i.e. "CALL SUBRTN,R5,DATA")

RSTS V9.0 hardware features

THE VIRTUAL DISK

- o Allows the use of host memory as a disk device.
(Temporary storage of data requiring fast access)
- o Contains the same structures as any other physical disk (SATT.SYS, BADB.SYS, etc.).
- o Memory allocated by using DEFAULT option of INIT.
(Memory Table suboption: "same as XBUF")
- o Need at least 4Kw for file-structured use
(1Kw = 4 blocks).
- o Must be initialized after each system start-up.
(DSKINT, Data is lost when system is shut down)

Suggestions on Virtual Disk usage

- o 1Kw - 2Kw can be used in Non-File-structured mode as a system-common read/write area
- o Large Memory Systems:
 - Temporary work files for SORT/MERGE utilities
 - BP2 compiler placement
 - OVR.SYS or ERR.SYS placement.

Major advantage over data caching during writes.
Already in memory; caching never done.

Asynchronous I/O in RSTS

- o Asynchronous I/O directives
 - .READA
 - .WRITA(additional information is passed to monitor in XRB and FIRQB)
- o Program execution continues while I/O request is being serviced by driver.
(not "stalled" as with synchronous I/O)
- o I/O completion is notified to the user's program through an asynchronous program trap, or AST.
(program flow is interrupted by monitor to notify user of I/O completion, "RT style")
- o Program flow is transferred to an Asynchronous I/O completion routine within user's program.
(AST routine specified in FIRQB at .READA/.WRITA)
- o I/O buffer address can be within the user's program or a Resident Library.
(Job or Resident Library will remain locked in memory until all I/O is complete)
- o Maximum of 14 outstanding Asynchronous I/O requests per job.
- o Asynchronous DISK I/O requests are not guaranteed to complete in the same order as issued.
(User supplied parameter stored in FIRQB)

ASYNCHRONOUS I/O IN RSTS/E V9.0

GENERAL CONCEPTS

- o Same basic function as synchronous I/O, asynch I/O moves data between a device and a user program
- o Synchronous I/O stalls user program until RSTS driver is done
- o Asynch I/O does not stall user program, continues to run while driver services the I/O request
- o Main difference is in the completion handling

ASYNCH PROGRAM TRAPS (AST's)

- o I/O Completion routine within user program or resident library
- o Executes when I/O has completed
- o Brief, suggested use is error checking, buffer management

NEW .READA/.WRITA DIRECTIVES

- o .READA (EMT 102) performs asynchronous read
- o .WRITA (EMT 104) performs asynchronous write
("TUNE" privilege is required for both)

ASYNCHRONOUS DEVICES

- o All disks
- o "MS" type magtape devices

TS11 TS05 TU80 TK25

ASYNCHRONOUS I/O in RSTS V9.0
General Concepts (continued)

"TRUE" ASYNCH VS. "SIMULATED" ASYNCH

TRUE Asynch I/O

- o User program issues .READA/.WRITA to an asynchronous device:

ALL Disks MS:

- o User program will not stall
- o Completion routine (AST) executes when done

SIMULATED Asynch I/O

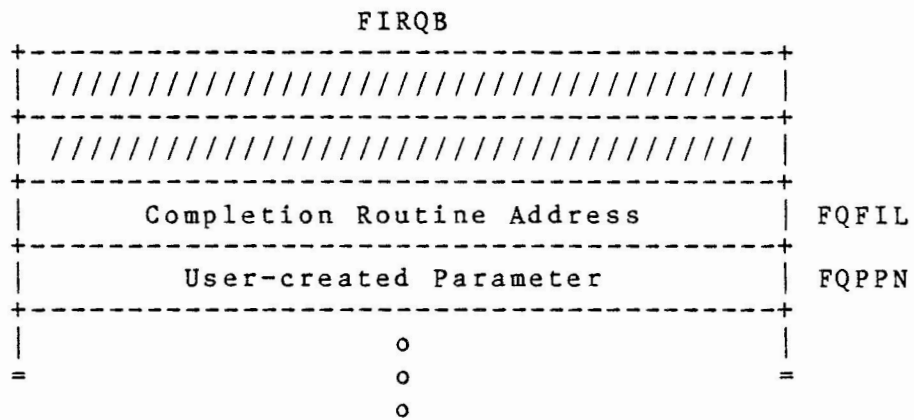
- o User program issues .READA/.WRITA to a synchronous device:

MM: MT:

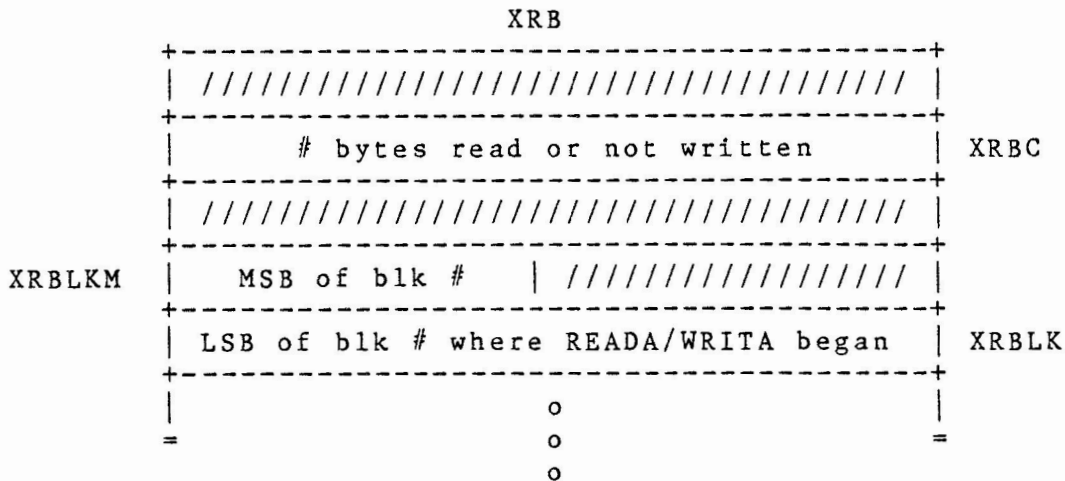
- o User program stalls like synch I/O
- o Completion routine (AST) executes when done

NEW DIRECTIVES

- o Data passed is mostly the same as conventional synchronous I/O .READ/.WRITE directives
- o XRB is the same as synchronous I/O
- o ADDITIONAL data passed:



- o All other fields are reserved and should be initialized to zero
- o Completion routine address = 0 means no completion routine to be executed
- o Data returned from .READA/.WRITA:



- o Information returned at I/O completion time

ASYNCHRONOUS I/O in RSTS V9.0
 New Directives (continued)

READ CHECK

- o No I/O transfer
- o Compares specified user buffer with data at specified block number of disk
- o XRMOD flags read check request (ignored for tape I/O)
- o Data passed:

XRB		
	length of buffer in bytes	XRLEN
	(must=0)	XRBC
	buffer starting address	XRLOC
XRBLKM	MSB of blk # Channel # *2	XRCI
	LSB of beginning blk #	XRBLK
	////////////////////////////////////	
	//////////////////////////////////// 1=Read check	XRMOD
	o	
	o	
	o	

XRMOD = 0 will perform normal .READA transfer
 XRMOD = 1 performs read check - NO TRANSFER

- o Possible error:

DATERR "Data error on device"

MORE ABOUT AST'S...

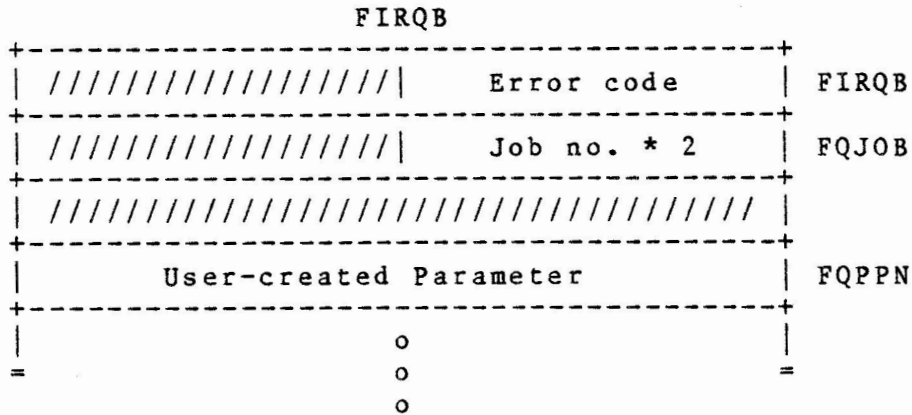
- o AST routines do not interrupt each other
- o Multiple I/O completions cause AST's to be queued
- o User programs should use interrupt capabilities intelligently
- o Generally, keep AST's brief
- o General registers that are used must be saved/restored before exiting

ERROR REPORTING TO THE JOB

Errors reported to user can be during one of two phases:

- 1) Immediately after issuing the .READA or .WRITA directive
- 2) At I/O completion time

o Data returned at directive time:



- o Errors at directive validation time (immediately after .READA/.WRITA)

EXAMPLE User program slice:

```

.FREADA
TSTB    @#FIRQB
BNE     ERR
< more stuff >

```

EXPECTED directive errors:

- INUSE Account or device in use
- BADCNT Illegal byte count for I/O
- NOBUFS No buffers space available
- PRVIOL Protection violation ("TUNE" privilege required)

ASYNCHRONOUS I/O in RSTS V9.0
Error Reporting to User Job (continued)

o Errors at completion time (AST execution)

EXAMPLE User program slice:

```
R.AST:  TSTB    @#FIRQB  
        BNE     ERR  
        < more stuff >  
        .ASTX
```

```
ERR:  
<report error>  
.ASTX
```

EXPECTED completion errors:

```
NOROOM  No room for user on device  
EOF     End of file on device  
HNGDEV  Device hung or write locked  
DATERR  Data error on device
```

EXAMPLE PROGRAM

```
.INCLUDE      /SY:[1,198]COMMON.MAC/  
.SBTTL  Asynchronous I/O Disk-to-disk copy
```

```
;  
; COPYRIGHT (c) 1974, 1984 BY  
; DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.  
;  
; THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  
; ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE  
; INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER  
; COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  
; OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY  
; TRANSFERRED.  
;  
; THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE  
; AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT  
; CORPORATION.  
;  
; DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS  
; SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.  
;  
;  
; DKCOPY.MAC (Asynchronous disk-to-disk copy)  
;  
; This program performs a fast copy operation between two RL02 disks.  
; The program originally issues 3 .READA requests on the first disk  
; and issues corresponding .WRITA's when the reads complete. In turn,  
; when each .WRITA completes, another advanced .READA is issued,  
; assuring that there is always one outstanding input read and at  
; at least one outstanding write request. Each I/O issued within the  
; AST uses the same buffer address of the finishing request, which is  
; an easy way to keep track of the completion order.  
;  
; Note that for sake of simplicity, the program does not handle bad  
; blocks gracefully, nor does it prompt for Input/Output disk?,  
; etc. Any errors are treated as fatal.  
;  
; ALSO NOTE THAT DUE TO PRE-EMPTIVE SCHEDULING, THIS PROGRAM WILL  
; "HOG" THE CPU. RUN AT PRIORITY -16 DURING NORMAL TIMESHARING.  
;  
; Register Usage:  
; R2 = Input block number  
; R4 = Buffer number (carried in the User-created parameter field)  
;
```


ASYNCHRONOUS I/O in RSTS V9.0
 Example Program (continued)

```

.PSECT

START::      CALL      CLRFBQB          ; Clear FIRQB

;
; Open input disk device = DL0:
;
      MOVB      #OPNFQ,FIRQB+FQFUN      ; Function code
      MOVB      #3*2,FIRQB+FQFIL        ; Set input channel=3
      MOV       #"DL,FIRQB+FQDEV        ; on an RL02
      MOV       #377*400+0,FIRQB+FQDEVN ; unit 0, device real
      MOV       #200*400+8192.,FIRQB+FQMODE ; Read-only mode
      CALFIP                      ; Go for it
      MOV       FIRQB,R0              ; Error?
      BNE      FATAL                  ; Stop and check it out

;
; Open output disk device = DL1:
;
      MOVB      #OPNFQ,FIRQB+FQFUN      ; Function code
      MOVB      #4*2,FIRQB+FQFIL        ; Set output channel=4
      MOV       #"DL,FIRQB+FQDEV        ; on an RL02
      MOV       #377*400+1,FIRQB+FQDEVN ; unit 1, device real
      CALFIP                      ; Go for it
      MOV       FIRQB,R0              ; Error?
      BNE      FATAL                  ; Stop and check it out

;
; Issue the first three initial reads
;
      CALL      CLRFBQB          ; Clear FIRQB  XRB

      MOV       #3,R1              ; One read, 2 block read-ahead
      CLR      R2                    ; R2 = virtual block zero
      MOV       #BUFFER,R4          ; Load first buffer address

10$: MOV       #512.,XRB+XRLEN        ; Set buffer size
      CLR      XRB+XRBC              ; XRBC = zero
      MOV      R4,XRB+XRLOC          ; Input buffer address
      MOVB     #3*2,XRB+XRCCI        ; Input channel * 2
      INC     R2                      ; Next input block number
      MOV     R2,XRB+XRBLK          ; Read that one
      MOV     #R.AST,FIRQB+FQFIL    ; Read AST address
      MOV     R4,FIRQB+FQPPN        ; Buffer #
      .READA                      ; Asynchronicity!
      MOV     FIRQB,R0              ; Directive error?
      BNE     FATAL                  ; Yes, stop
      ADD     #512.,R4              ; Get next buffer address
      SOB     R1,10$                ; Issue next read-ahead

30$: BR       30$                    ; Do some processor work

```

ASYNCHRONOUS I/O in RSTS V9.0
 Example Program (continued)

```

;
; All .READA's come here at completion time
;
R.AST:      MOV      FIRQB,RO                ;; Error?
           BEQ      60$                      ;; No, continue
           CMPB     #13,RO                   ;; End of disk (EOF) ?
           BEQ      CLSIN                    ;; Yes, go close disks
           BR       FATAL                    ;; No, unexpected error, stop

60$: MOV     #512.,XRB+XRLEN                ;; Reset buffer size
      MOV     #512.,XRB+XRBC                ;; Want to write entire buffer
      MOV     FIRQB+FQPPN,XRB+XRLOC        ;; Use the same buffer address
      MOVB    #4*2,XRB+XRCI                ;; Output channel * 2
                                           ;; XRBLK has correct block #
      MOV     #W.AST,FIRQB+FQFIL           ;; Write completion routine addr
      .WRITA                                     ;; Issue the write for this block
      BR     ASTDON                          ;; Join common code

;
; All .WRITA's come here at completion time
;
W.AST:      MOV      FIRQB,RO                ;; Error?
           BEQ      70$                      ;; No, continue
           CMPB     #13,RO                   ;; End of disk (EOF) ?
           BEQ      CLSIN                    ;; Yes, go close disks
           BR       FATAL                    ;; No, unexpected error, stop

70$: MOV     #512.,XRB+XRLEN                ;; Set buffer size
      CLR     XRB+XRBC                      ;; XRBC = zero
      MOV     FIRQB+FQPPN,XRB+XRLOC        ;; Keep the same buffer
      MOVB    #3*2,XRB+XRCI                ;; Input channel * 2
      INC     R2                            ;; Inc to next block
      MOV     R2,XRB+XRBLK                 ;; Read this block number
      MOV     #R.AST,FIRQB+FQFIL           ;; Read AST address
      .READA                                     ;; Asynchronicity!

;
; AST common code
;
ASTDON:     MOV      FIRQB,RO                ;; Any directive error?
           BNE      FATAL                    ;; Yes, stop
           .ASTX                              ;; No, exit AST(s)

```

ASYNCHRONOUS I/O in RSTS V9.0
 Example Program (continued)

```

    .SBTTL  FIRQB/XRB clearing routines
    .ENABL  LSB

CLRFX:   PUSH    <R0,R1>                ; Save R0, R1
        MOV     #FIRQB,R0                ; Point to FIRQB.
        MOV     #<<FQBSIZ+XRBSIZ>/2>,R1 ; Compute words to clear.
        BR     200$

CLRFB:   PUSH    <R0,R1>                ; Save R0, R1
        MOV     #FIRQB,R0                ; Point to FIRQB.
        MOV     #<FQBSIZ/2>,R1          ; Compute words to clear.
        BR     200$

CLRFRB:  PUSH    <R0,R1>                ; Save R0, R1
        MOV     #XRB,R0                  ; Point to XRB
        MOV     #<XRBSIZ/2>,R1          ; Compute words to clear.

200$:   CLR     (R0)+                    ; Zero it out
        SOB    R1,200$                  ; 'til all done.
        POP    <R1,R0>                  ; Restore R0, R1
        RETURN

;
; Errors (code in R0) come here
;
FATAL:   BPT                            ; All errors come here
OUT:     .EXIT                          ; All done

;
; Close input disk (may still have outstanding .READA's)
;
CLSIN:   CALL    CLRFB                  ; Clear FIRQB
        MOVB   #CLSFO,FIRQB+FQFUN      ; CLOSE Function
        MOVB   #3*2,FIRQB+FQFIL        ; Set channel=3
        CALFIP                                ; Do it
        MOV    FIRQB,R0                 ; Error?
        BEQ    CLSOUT                   ; No error; close output disk
        CMP    #3,R0                    ; Expected INUSE error?
        BEQ    CLSIN                     ; Yes, try it again
        BR     FATAL                    ; No, something unexpected!

;
; Close output disk (may still have outstanding .WRITA's)
;
CLSOUT:  CALL    CLRFB                  ; Clear FIRQB
        MOVB   #CLSFO,FIRQB+FQFUN      ; CLOSE Function
        MOVB   #4*2,FIRQB+FQFIL        ; Set channel=4
        CALFIP                                ; Do it
        MOV    FIRQB,R0                 ; Error?
        BEQ    OUT                       ; No error; We're all done!
        CMP    #3,R0                    ; Expected INUSE error?
        BEQ    CLSOUT                   ; Yes, try it again

```

ASYNCHRONOUS I/O in RSTS V9.0
Example Program (continued)

```
.DSECT 3000,NOCREF

BUFFER:      .BLKB  512.                ; Buffer one
              .BLKB  512.                ; Buffer two
              .BLKB  512.                ; Buffer three
BUFEND:      ; Address for EOB

.END        START
```

ASYNCHRONOUS TAPE I/O

- o Asynch tape requests guaranteed to finish in order of submission (unlike disk)
- o Data passed to .READA/.WRITA tape requests exactly same as disk requests
- o XRMOD is ignored (no read check)
- o Tape I/O completions handled same as disk

ASYNCH TAPE ERROR HANDLING

- o What is done when an asynch hard error (EOT) occurs on tape?

RSTS

- 1) Driver encounters a hard error (EOT)
- 2) Driver marks the troubled unit/controller as "error locked"
- 3) Successive requests to the unit (there may be several) get answered with "Device hung or write locked"
- 4) Driver continues to answer requests with errors until user program "fixes the problem"

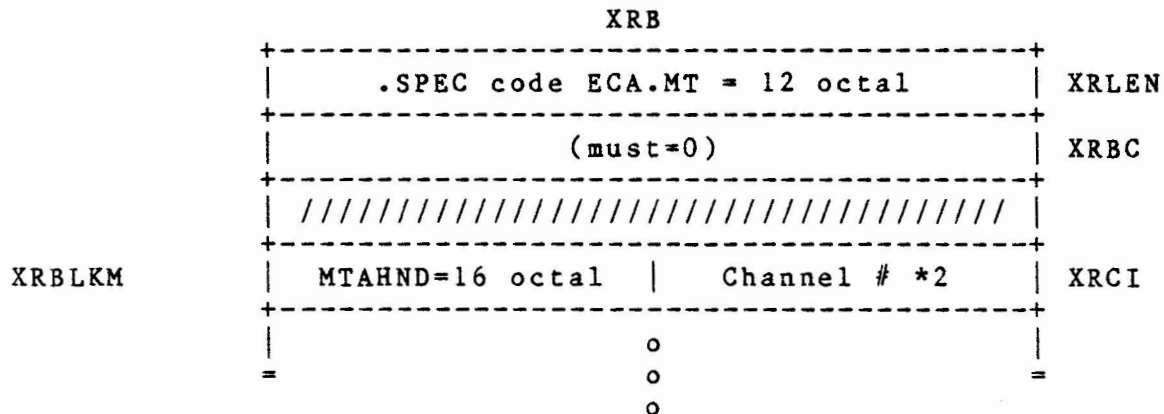
USER PROGRAM

- 1) Notes which requests are coming back with errors
- 2) Stops sending data to the device
- 3) Prompts operator to mount another tape
- 4) Issues new .SPEC function = ECA.MT "Error Condition Acknowledged"
- 5) Continues with output to unit

ASYNCHRONOUS I/O in RSTS V9.0
 Asynchronous Tape I/O (continued)

NEW .SPEC FUNCTION - ECA.MT
 "Error Condition Acknowledged"

o Data passed:



o Issue ECA when physical recovery is needed by the operator

o Errors which will require user ECA:

- NOROOM No room for user on device
- EOF End of file on device
- HNGDEV Device hung or write locked
- MAGSEL Magtape select error