

# *The Schema*

Newsletter of the Data Management Systems Special Interest Group

July Vol. 3 - No. 4 1981

Please address contributions or inquires to:

Paul D. Clayton, Editor

or

Sat Mohan, Chairman

c/o DECUS

One Iron Way, MR2-3/E55

Marlboro, Mass. 01752

## **WARNING:**

RECEIVED

AUG 27 1981

ACCOUNTING DEPARTMENT

**THIS  
NEWSLETTER  
IS  
UNDER  
NEW  
MANAGEMENT**

Copyright ©. 1981 Digital Equipment Corporation  
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 of Digital Equipment Corporation.

## THE IRONY OF PROGRESS



### 10 WAYS TO KILL A PROFESSIONAL GROUP

- Boycott all meetings.
  - If you don't boycott, come late with a bored look.
  - If the weather is predicted as favorable, hope for 2-feet of snow.
  - Find fault with all officers and contributing members.
  - Never accept an office yourself.
  - However, if you're not appointed for a committee or nominated for office, get sore.
  - Say nothing at meetings, but tell everyone afterwards exactly how things should have been done.
  - Do nothing and when other members start things moving, murmur and howl that everything is run by a clique.
  - Always agitate things when programs go smoothly.
  - Never try to get new members. Instead, discourage present members with intensified and continued effort on the previous nine points.
- For good measure—always start a side discussion while the main event is taking place. Be noisy.

Reprinted with permission from the I.E.E.E.  
Philadelphia Section, "Almanack".

DMS SIG Newsletter Cumulative Index  
July 1981

Sequence No.	Page No.	Brief Description
1.0	1-1	Index
2.0		SIG Business
* 2.1.1	2-1	Letter from new Newsletter Editor, describing new layout and SIG resources. July-81
* 2.2.1	2-2	Letter from Sat Mohan, SIG Chairman, describing current events in the SIG. July-81
* 2.3.1	2-2	Open letter to SIG membership from Steve Pacheco, calling for Menu input and newsletter articles. July-81.
* 2.4.1	2-3	Open request for DMS-500 interest from DOW Chemical. July-81.
* 2.6.1	2-5	DMS SIG Steering Committee Member List. July-81
3.0		RMS Interests
* 3.2.1	3-1	RMS-11 Menu Responses to Spring '81 DECUS. July-81.
* 3.3.1	3-3	Sample Relative RMS-11 program corrected from DEC documentation. July-81.
4.0		FMS Interests
* 4.1.1	4-1	FMS Working Group Survey from Judy Kessler. July-81.
* 4.2.1	4-2	FMS Menu Responses to Spring '81 DECUS. July-81.
* 4.3.1	4-3	Patching FMS-11 Forms for VT-100 Special Features. July-81.
* 4.3.2	4-4	General comments on FMS-11 from Sebern Eng.. July-81.
5.0		DBMS Interests
* 5.2.1	5-1	DBMS-11 Menu responses to Spring '81 DECUS. July-81.
* 5.3.1	5-2	Paper on using DBMS-11 at NASA in spacecraft command and control. July-81.
6.0		TECO Interests
* 6.1.1	6-1	Open letter to TECO users calling for input and positions available. July-81.
* 6.3.1	6-3	Request for information on TECO written in UNIX for VAX. July-81.
* 6.3.2	6-3	Request for information on TECO sources for ver. 36 and VAX support. July-81
* 6.3.3	6-4	Request for answers to problems listed in letter and about a screen editing ver. of TECO. July-81.
* 6.3.4	6-4	Request for documentation for TECO macros distributed with RT11-V4. July-81.

)

)

)

)

)

Letter From The NEW Newsletter Editor  
Paul D. Clayton  
July 8, 1981

I wish to take this time as the new DMS SIG Newsletter Editor, and welcome you to a revised and reorganized newsletter. I assumed the responsibilities of Editor during the Miami Spring '81 DECUS Symposium. When I was considering this position, I was amazed at the resources available to the DMS SIG through its membership and DEC counterparts, and disturbed by the problems in gathering and presenting this information to the SIG membership by phone calls or this newsletter. I found in Miami, the people with answers to problems I encountered in generating two (2) databases using RMS-11 that, combined, took months of my time to solve or develop a work-around. The point of this is that I had no notion of how to use the SIG to get to the people that had the answers. It is with this in mind that I present the following.

How To Use The DMS SIG:

The SIG is a tool, that used wisely can yield information specific to a problem/concern you have. The list of SIG Steering Committee members includes the areas of responsibility for each person. Please understand that these people are NOT a hot-line for debugging programs. If you have tried various approaches to a problem and need new ideas or names and phone numbers of other SIG members this is what they are there for!!!

Also, if you have a "trick" or technique enabling you to do something not thought of before, contact the appropriate Steering Committee member and let them know about it. Call them up on the phone or send a letter!!

What you have to remember is that members of the Steering Committee are not "The Untouchables", they are everyday users of DEC computers that are interested in getting results.

How To Use This Newsletter:

First of all lets face facts, the people who are able, by what ever means, to attend the DECUS Symposium are a very fortunate group. The overwhelming majority of DECUS members are faced with no means to get and/or pass information except through LUG's, which are restricted geographically, (with one notable exception, DeVIAS) from talking to one another. This is where SIG Newsletters fit in!!!! It is for this reason that I would like everyone to realize that this SIG Newsletter is a way to hold a "Mini-Symposium" during the whole year with each issue. Its cheaper than going to a Symposium and with YOUR inputs can be priceless to a lot of people worldwide. Please call me if you have an idea for an article or have an article just waiting for print!! We can talk about the best means of getting YOUR article into this newsletter.

I am reorganizing this newsletter in order to make it more valuable to you regardless of your DMS SIG interest, be it RMS, FMS, DBMS, TECO or what ever in the future. This will be done by setting up this newsletter layout similar to the "Software Dispatch" that DEC distributes for its operating systems. Each major interest, (ie. SIG business, RMS, etc.), will have its own pages that can be pulled out of the newsletter and added to a

notebook. Each article within a particular interest will have a unique identifying number for entry in a cumulative index, which will be updated and printed with each issue. An asterick along side an entry in the cumulative index will designate a new article. The following is a break down of the Newsletter Components and their section numbers.

1.0	Cumulative Index
1.1.X	Newsletter Covers
2.0	SIG Business
2.1.X	Letters from the SIG Newsletter Editor
2.2.X	Letters from the SIG Chairman
2.3.X	Letters from Steering Committee Members
2.4.X	Letters of General Interest from SIG Members
2.5.X	Menu Ballots
2.6.X	SIG Steering Committee Members and Responsibilities
3.0	RMS Interests
3.1.X	Letters from the SIG RMS Product Group Chairman
3.2.X	Menu Responses
3.3.X	Letters from RMS Users
3.4.X	General Questions (and Answers) from RMS Users
4.0	FMS Interests
4.1.X	Letters from the SIG FMS Product Group Chairman
4.2.X	Menu Responses
4.3.X	Letters from FMS Users
4.4.X	General Questions (and Answers) from FMS Users
5.0	DBMS Interests
5.1.X	Letters from SIG DBMS Product Group Chairman
5.2.X	Menu Responses
5.3.X	Letters from DBMS-11 Users
5.4.X	Letters from DBMS 10/20 Users
5.5.X	General Questions (and Answers) from DBMS Users
6.0	TECO Interests
6.1.X	Letters from the SIG TECO Product Group Chairman
6.2.X	Letters from TECO Users
6.3.X	General Questions (and Answers) from TECO Users

Naturally, this numbering scheme is expandable to allow for future interests and additional breakdown of articles printed.

\*\*\*\* CONTEST \*\*\*\*

I am holding a contest to rename this newsletter to something that is more descriptive of the varied interests it now covers. Gone are the days of only DBMS interests!! FMS, RMS and TECO people do not know (or care) what a "Schema" is!! All contributions should be mailed to me (address below) by Sept. 1, 1981. The SIG Steering Committee will judge all entries and choose the winner. All decisions are final. The winner will be taken to dinner by the SIG Steering Committee at the next Symposium, if the winner can make it, otherwise a DEC gift certificate for \$20.00 will be mailed to the lucky person. You must not be a member of the DMS SIG Steering Committee, or related to one, to enter. Mail your entry today!!

Address: Republic Management Systems  
One Neshaminy Interplex, Suite 306  
Trevose, Pa. 19047  
Attn: Paul D. Clayton

Letter From DMS SIG Chairman  
June 19, 1981

As you can see, this newsletter has a new look and contents. We have a new editor, Paul Clayton of Republic Management Systems in Trevoze, Pa.. Paul has been active in DECUS for 2 years, and is well known (notorius?) in IAS circles for associating with the DeVIAS global LUG.

We are aiming for this newsletter to be the best in DECUS within six months. I strongly urge you to go through this issue and make any contributions - technical or organizational - you can to help us serve you better.

One of the areas this SIG addresses is RMS. While we are all interested in functionality of the DEC implementation within their products, not much information exchange has happened regarding how each of us applies it to each application. This could be sharing of your own work in the areas of files design, buckets selection, performance optimization etc. Again, I strongly urge you to send a brief note to Paul at the earliest. Bob Curley, the SIG product groups co-ordinator, is also actively seeking your organizational help.

At the request of the DECUS U.S. Board, we have accepted responsibility for user activities related to TECO. If you like, nay love, TECO you should contact Bob Curley or me at the earliest. More on this next month.

Miami Beach was a very successful meeting for all our activities - DBMS, FMS and RMS. On behalf of all SIG members, I would like to thank DIGITAL for a strong and effective representation in the Data Management area. In addition I want to single out Fred Howell, Anita Moeder, Jim Starkey, Bob Nusbaum, Rick Landau, Bill Harrelson and Chuck Turley for special appreciation of their committment and effort at Miami.

In the July issue of this newsletter, I plan to give you an overview of the SIG's activities, our goals, SIG organization and a brief look at our past accomplishments and failures. By the time you get this issue, a subset of the SIG Steering Committee would have held a planning meeting in Chicago to chart a course and actions for the future. A brief report will be included in the next newsletter.

Hope you are having an enjoyable summer.

Satish Mohan.

ITEMS FOR THE DMS SIG NEWSLETTER

An open letter to the New Newsletter Editor

As the outgoing newsletter editor I would like to extend my best wishes to Paul Clayton in his position of Newsletter Editor for the DMS SIG. Paul will be making some significant changes in the newsletter format and content. We hope these changes will provide a better newsletter for our readers.

I would like to thank all those who supplied material for the newsletter while I was the editor. Without people supplying material to print, there would be no newsletter. Paul and I both hope that our readers continue to supply interesting material to be shared with the entire SIG and DECUS membership.

Though I am no longer the editor, I am retaining my activity within the SIG and you will be hearing from me from time to time.

Again I wish Paul success in his efforts to produce a quality product and hope that he receives sufficient material from which to produce that product.

Thanks for your support,  
Stephen Pacheco

-----  
DMS SIG Menu Procedure

The DMS SIG has established a new menu procedure as of the Miami Symposium. This procedure uses both the symposium and the newsletter as a source of menu items. The menu items will be segregated into categories based on product (e.g., RMS, DBMS-10/20, DBMS-11, FMS) and type of item (e.g., product bug, enhancement, question about use). This will prevent menu items relating to a widely used product from eliminating items of a product with a smaller customer base.

The list of items is presented to the DMS SIG membership for vote in the newsletter or special mailing following a symposium.

The votes received for the items are used to rank the items for submittal to DEC for their response. DEC will supply verbal responses at the next symposium and written responses to be published in the newsletter following the symposium.

DMS SIG Menu Coordinator  
Stephen Pacheco



DOW CHEMICAL U.S.A.

TEXAS DIVISION  
FREEPORT, TEXAS 77541

June 5, 1981

Paul Clayton  
Code-1S Naval Air Development Center  
Street Road  
Warminster, Pennsylvania 18974

cc: Bob Curley  
Dept. of Radiation Therapy  
University of Pennsylvania  
3400 Spruce St.  
Philadelphia, PA 19104

Bob McCormick  
Farm Credit Bank  
PO Box 141  
Springfield, MA 01101

Dear Paul:

We have developed several RSTS/E applications at our location that use a DEC product called DMS-500. It is Basic+ source code that can be appended to Basic+ programs and provides indexed sequential file access to the Basic+ programs. At the Miami Beach DECUS meeting I could find no one in the DMS SIG that knew about DMS-500, nor could I find any DEC people at the educational or software services booths that knew about it.

Bob Curley suggested that I write to you requesting that an inquiry be put in the next issue of SCHEMA soliciting information from anyone that would be interested in the formation of a DMS-500 interest group within the DMS SIG. If we have enough users of the product we should be able to let DEC know that we have such interest and that we do need some technical backup and assistance channels to follow.

Could you place such a request in the SCHEMA? By a copy of this letter to Bob McCormick, I am also asking the RSTS SIG to do the same in THE RSTS NEWSLETTER. Anyone interested could reply to me and I will keep Bob informed as to what happens.

Sincerely,

Harry N. Perk  
Dow Chemical, U.S.A.-Texas Division  
Computer Services-APB Bldg.  
Freeport, TX 77541

Phone: 713/238-1737

bev

AN OPERATING UNIT OF THE DOW CHEMICAL COMPANY

)

)

)

)

)



DMS SIG Steering Committee Members  
July 8, 1981

If you have any questions or concerns contact the person that has the responsibility for the area you want to discuss.

SIG Chairman

Satish Mohan (416) 461-8111 Ext. 269  
TIW Industries Ltd.  
Metals Group  
629 Eastern Ave.  
Toronto, Ontario  
M4M 1E4

Symposia Co-ordinator

Barbara Mann (213) 536-4190  
TRW Inc.  
One Space Park, R3/2030  
Redondo Beach, Calif. 90278

Publications Co-ordinator

Stephen Pacheco (401) 596-1091  
Athena Systems  
37 Forrestal Drive  
Westerly, RI 02891

Newsletter Editor

Paul D. Clayton (215) 441-2708  
Republic Management Systems  
One Neshaminy Interplex, Suite 306  
Trevose, Pa. 19047

Product Groups Co-ordinator

Robert F. Curley (215) 662-3083  
Department of Radiation Therapy  
University of Pennsylvania Hospital  
3400 Spruce Street  
Philadelphia, Pa. 19104

DBMS-11 Product Group Chairman

Michael Antin (617) 684-7308  
Polaroid Corp.  
1265 Main Street W4-2B  
Waltham, MA. 4570

DBMS 10/20 Product Group Chairman

Jack Hill (615) 244-7080  
Nashville Gas Co.  
814 Church Street  
Nashville, TN. 37203

RMS Product Group Chairman

Robert F. Curley  
Same as above

FMS Product Group Chairman  
Judy Kessler (617) 742-3140  
Eye Research Institute of Retina Foundation  
20 Staniford Street  
Boston, Mass. 02114

TECO Product Group Chairman  
Robert F. Curley  
Same as above

Seminars Co-ordinator  
Jack Hill  
Same as above

Performance Working Group Co-ordinator  
Burt Weaver (612) 571-1249  
Consulting Engineer  
Weaver & Associates Inc.  
2852 Anthony Lane South  
Minneapolis, Minn. 55418

User Support Co-ordinator  
Douglas Dickey (703) 827-2700  
CTEC Inc.  
7777 Leesburg Pike  
Falls Church, Va., 22043

Planning Co-ordinator  
Sany Krueger (201) 541-8347  
AMAX Copper Inc.  
833 Roosevelt Ave.  
Cartaret, N.J., 07008

Projects Co-ordinator  
T. Chris Wool (302) 366-4610  
E.I. DuPont  
Louviers Building  
Wilmington, Delaware, 19898

DEC Counterpart  
Anita L. Moeder  
Digital Equipment Corp.  
MK1-2/D03  
Continental Blvd.  
Merrimack, N.H. 03054

↑-----↑  
| d | g | i | t | a | l | I N T E R O F F I C E M E M O R A N D U M  
↑-----↑

TO: Anita Moeder  
CC: Chuck Turley  
Glenn Johnson

Date: 03-JUN-81  
From: Tim Day  
Dept: BSSG; RMS-11  
Ext: 8-264-8236  
Mail Stop: ZK1-3/H1  
DECmail: RUNE;T\_DAY  
Revision: 1, 03-JUN-81  
File: decusmenu,RNO

Subj: Responses to RMS-11 menu items at Spring DECUS

1. Permit an alternate key to be changed without forcing the key to allow duplicates (space reclamation item);

Keying on the words space reclamation, this is only one, albeit visible, function that requires a design change to implement. We are very aware of the less than optimal solution that RMS-11 currently provides in this area, and are actively pursuing an overall strategy for a future release, rather than a piecemeal approach to isolated (from each other) problems. This particular restriction is a "nasty", and will be removed in a future release, independent of other general reclamation issues.

2. Permit segmented keys to be composed of integer data elements.

The extension of allowing per segment data-typing for all supported key types is under investigation for a future release.

3. Provide supervisor mode library support for RMS.

This "enhancement" to the RMS-11 package is seen as one that has a limited benefit to the total RMS-11 community, because the only operating system that currently supports this addressing mode is RSX-11M-PLUS. The current implementation of RMS-11 would require a substantial overhaul of the existing code to provide this function. Therefore, the effort required to enhance the present RMS-11 code to provide supervisor mode support does not seem warranted by the relatively small number of users who would be able to benefit from this solution. It is not currently planned as a future enhancement.

Responses to RMS-11 menu items at Spring DECUS

However, should the circumstance arise during a future release to re-write significant portions of the RMS-11 code, the engineering governing the re-write would certainly aim to include this function as a byproduct of that effort.

4. Read=regardless mode on open, or random or sequential data caching.

The general response to this type of inquiry is that both these functions are privileged, and RMS-11 is not, and doesn't intend to become so.

We have evaluated providing read=regardless (as well as other RSTS/E switch mode options), and to this point have rejected it's inclusion in RMS-11 due to the potential for integrity problems. If read=regardless is enabled, RMS-11 has no means to ensure that any data presented to the "read=regardless" task is valid. While acknowledging the limitations of RMS-11 file sharing for "seat of the pants" excursions into a given file, we remain unconvinced that the RMS file sharing environment is inadequate for successful implementation of a given application.

As regards the data caching, RMS-11 does offer means of both random and sequential caching, which go under the terms multi=block count and multi=buffer count as documented. The RSTS/E implementation of caching is independent of RMS-11, and in keeping with our general policy of not interfacing RMS-11 to system specific features, we have no intentions of interfacing RMS-11 to RSTS/E data caching.

5. Provide an RMS ACP to remove the burden of the RMS routines from the user program.

Again, this solution is not viable for the RSTS/E users. However, we are investigating this as one possibility for the RSX world.

6. Utility to provide a summary of an RMS file. Information to be provided would include the number of buckets, number of buckets full, etc.

This is currently under investigation for a future release.

7. Provide a switch to allow CNV to open the input file shared.

We will look at providing this support in a future release. However, the inclusion of this feature would not ensure that CNV could successfully open the input file. The first accessor into the file determines the access privileges granted to accessors 2+. If the original accessor has the file open exclusively, CNV would not be able to open the file.

8. Provide a backward link under the primary and alternate keys.

This feature would require a complete design change to the present architecture of RMS-11. In addition, it would add a significant amount of overhead to the on disk structures (bucket headers, record headers), swell the code needed in many crucial routines (perish the thought !!), not to mention the performance penalty it would impose on record inserts and updates, and decrease the ability to ensure integrity of file structures across arbitrary program and/or system failures. For these reasons, it is unlikely that we will implement this feature in RMS-11.

9. Complete documentation of on-disk structures of all RMS-11 files.

This is being investigated for a future release. One reason we don't make this available currently is that we anticipate making structure changes in a future release, and will probably time the availability of this documentation to those changes.

10. Make a sequential "get" retrieve by value of KRF (key of reference) for an index. Our experience shows that "get's" retrieve by primary key after a "get" by RFA (Record's File Address). This occurs with KRF and KRZ set to known values.

Although this was categorized as an enhancement on the menu ballot, it really falls into the realm of user error. As documented, RFA access is a method of accessing a record that retrieves on level 0 of index 0 (i.e. your data record). RFA access thus establishes context for future record operations as that of index 0.

Perhaps the intent of this item was to request an extension to the present access methods that would allow the preservation of context of ANY index across disjoint operations. This would essentially allow an access by alternate key 2, an RFA access, and a sequential access to key 2, or any other variation. This would require an extension to the current interface, but nevertheless, is being investigated for a future release.

11. Allow index areas to be placed on a separate disk. Could be done by keeping the name of the file in the prologue and opening it on a separate channel.

We don't believe that we ever want RMS-11 to get into a scheme where portions of a file physically reside on different devices, especially where the underlying operating system does not enforce a software architecture analogous to multi-volume disks. There are a myriad of reasons why this type of implementation would involve great risk, and there

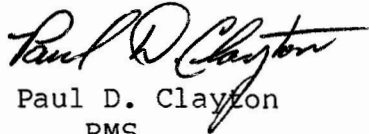
really seems to be little benefit to physically delimiting the levels of the index among devices. For those who are interested, the following is an incomplete list of the problems associated with this type of scheme, listed in no particular priority.

1. How to ensure that the correct disks are mounted in the correct drives at the time the file is opened? What to do if they aren't?
2. How to backup the file in any kind of orderly and reliable manner?
3. What to do about the additional overhead in both the code and RMS data structures to accomplish this? What about the overhead associated with an open file at the operating system level (file headers, etc.)?
4. What to do if one of the set of disks for the file goes offline for whatever reason?
5. How does one copy the file?
6. How do you integrate the file sharing code (presently done on a file/device basis) onto a multi-device scheme,

[End of memo file "decumenu,RND"]

Sample Relative File Program

This program is a corrected version of the sample in Appendix B.3 of the RMS Macro-11 Reference Manual. The original has over 110 errors (syntax and logic combined). Also included is a task build command file to use in building the program.

  
Paul D. Clayton  
RMS

```
.TITLE RELATIVE AND SEQUENTIAL TESTS
.SBTTL MCALL'S + FAB'S + RAB'S DEFINED
;
; FIRST GET ALL THE MACROS NEEDED FOR THIS PROGRAM
;
;
.MCALL $INIT, FAB$B, NAM$B, XAB$B, RAB$B, ORG$, POOL$B
.MCALL $GNCAL, $FBCAL, $RBCAL, $STGDPLY, $GETSTG, EXIT$$
$GNCAL
$FBCAL
$RBCAL
;
.NLIST BEX
;
; OFFSET THE FAB BY 2 BYTES
; THE FABs AND RABs CANNOT START AT "0" ADDRESS IN YOUR
; PROGRAM.
;
.WORD 0
;
; FIRST DEFINE THE FAB FOR THE RELATIVE FILE
;
;
.EVEN
FABAR:
FAB$B ;ALLOCATE FAB
F$BKS 2 ;BUCKET SIZE - 12 RECORDS/BUCKET
F$FAC FB$GET!FB$PUT!FB$DEL ;DOING GETS AND PUTS
F$FOP FB$TMD ;THIS FILE TO BE DELETED
F$LCH 1 ;USE LOGICAL CHANNEL 1
F$MRS 80. ;MAX RECORD SIZE OF 80
F$ORG FB$REL ;RELATIVE FILE
F$RFM FB$VAR ;VARIABLE LENGTH RECORDS
FAB$E
;
; NOW DEFINE THE ASSOCIATED RAB FOR THE RELATIVE FILE
;
.EVEN
RABAR:
RAB$B ;ALLOCATE RAB
R$FAB FABAR ;POINTS TO FAB AREA
```

```

R$KBF    KEYNO                ;WHERE TO PUT RELATIVE KEY NUMBER
R$K SZ   4                    ;KEY IS 4 BYTES
                                   ;KEY MIN. FOR RELATIVE IS 4 BYTES
R$RAC    RB$SEQ               ;WRITE RECORDS SEQUENTIALLY
R$RBF    RECBUF               ;ADDR OF RCD PUT OR GOT
R$UBF    RECBUF               ;SAME AS RBF
R$USZ    150.
RAB$E

;
.PAGE
.SBTTL FAB + RAB FOR TERMINAL

;
;
NOW DEFINE THE FAB FOR THE TERMINAL AS A SEQUENTIAL FILE
;
.EVEN
FABTI:
FAB$B
F$FAC    FB$PUT                ;PUT TO THE FILE
F$FNA    TFNAM                 ;TERMINAL FILE NAME
F$FNS    TFNAMS                ;TERMINAL FILE NAME SIZE
F$FOP    FB$CTG!FB$TMD         ;CONTIGUOUS SPACE, TEMP FILE
F$LCH    5                     ;LOGICAL CHANNEL 5 - USE PDP STANDARD
F$MRS    80.                   ;MAX REC SIZE = 80. BYTES (CHARACTERS)
F$ORG    FB$SEQ                ;SEQUENTIAL FILE
F$RAT    FB$CR                 ;RECS DELIMITED BY CR AND LF
FAB$E

;
;
NOW DEFINE THE RAB TO GO WITH THE FAB FOR THE TERMINAL
;
.EVEN
RABTI:
RAB$B
R$FAB    FABTI                 ;FILE CTRL BLOCK ADDR
R$RAC    RB$SEQ               ;SEQUENTIAL ACCESS
R$UBF    BUFTI                 ;USER BUFFER ADDR
R$USZ    80.                   ;MAX SIZE = 80. BYTES
RAB$E
.PAGE
.SBTTL SPACE POOL AND CONSTANTS

;
;
DEFINE THE SPACE REQUIRED FOR THE RMS-11 POOL
;
.EVEN
POOLAR:
POOL$B
P$BDB    3                     ;NUMBER OF BUFFER DESCRIPTOR BLOCKS
                                   ; MAXBUF = 2
                                   ; MAXREL = 1
                                   ; MAXIDX = 0
P$FAB    2                     ;NUMBER OF FILES OPEN SIMULTANEOUSLY
P$RAB    2                     ;NUMBER OF NON-INDEX FILES
P$BUF    1600.                 ;SIZE OF BUFFERS NEEDED FOR BOTH FILES
                                   ;FOR THE RELATIVE FILE

```

```

; BKS=2, MBC=1, MBF=1
;FOR THE TERMINAL FILE
; BKS=1, MBC=1, MBF=1

POOL$E
;
; LOCAL SYMBOLS
;
CR = 15 ;ASCII CARRIAGE RETURN
LF = 12 ;ASCII LINE FEED
PRAMTR: .BLKW 1 ;PARAMETER TO SEND INFO TO PRINT
HOLDIT: .BLKB 2 ;HOLD NUMBER FOR ASCII CONVERT
KEYNO: .BLKB 4 ;RELATIVE KEY
KEYTST: .BLKB 2 ;KEYTST TO COMPARE RECORD RETRIEVED
;WITH RECORD PUT
DELTST: .BLKB 2 ;TEST FOR DELETED RECORDS
TSTFOR: .BLKB 1 ;TO TEST FOR FOURTH TEST
ERR: .BLKB 1 ;TEST FOR TEST 4 ERRORS
RNFSET: .BLKB 1 ;TEST FOR RNF ERROR
.EVEN
BUFTI: .BLKB 80. ;TERMINAL FILE BUFFER
;
; NOW DEFINE THE DATA THAT IS TO BE WRITTEN TO THE RELATIVE FILE
;
RECBUF:
RECKEY: .BLKW 1
.ASCII /MISC. DATA/ <CR><LF>
RECLEN = . - RECBUF
;
; DEFINE THE NAME OF THE TERMINAL FILE
;
.EVEN
TFNAM: .ASCII /TI:/
TFNAMS = . - TFNAM
;
;
.PAGE
.SBTTL ERROR MESSAGES
;
; FOLLOWING ARE THE POSSIBLE ERRORS THAT COULD OCCUR DURING THE TEST
;
ERR1: .ASCII /ERROR CREATING TTY FILE / <CR><LF>
ERR1LN = . - ERR1
;
ERR2: .ASCII /ERROR CONNECTING TTY FILE / <CR><LF>
ERR2LN = . - ERR2
;
ERR3: .ASCII /ERROR CREATING RELATIVE FILE/ <CR><LF>
ERR3LN = . - ERR3
;
ERR4: .ASCII /ERROR CONNECTING RELATIVE FILE/ <CR><LF>
ERR4LN = . - ERR4
;
.EVEN ;SET BOUNDARY

```

```

MSG5: .ASCII /TEST /
TESTNO: .BLKB 1
SUC5: .ASCII / SUCCESSFUL/ <CR><LF>
MSG5LN = . - MSG5
;
ERR6: .ASCII /ERROR PUTTING RECORD TO FILE/ <CR><LF>
ERR6LN = . - ERR6
;
.EVEN ;ALIGN
ERR7: .ASCII /ERROR DELETING RECORD NUMBER /
DELNUM: .BLKW 1
.ASCII <CR><LF>
ERR7LN = . - ERR7
;
ERR8: .ASCII /ERROR GETTING RECORD FROM FILE/ <CR><LF>
ERR8LN = . - ERR8
;
ERR9: .ASCII /ERROR CLOSING FILE/ <CR><LF>
ERR9LN = . - ERR9
;
ERR10: .ASCII /ERROR REWINDING FILE/ <CR><LF>
ERR10LN = . - ERR10
;
ERR11: .ASCII /ERROR DISCONNECTING FILE/ <CR><LF>
ERR11LN = . - ERR11
;
ERR12: .ASCII /RETRIEVED WRONG RECORD IN FILE/ <CR><LF>
ERR12LN = . - ERR12
;
.EVEN ;ALIGN
ERR13: .ASCII /ERROR FINDING RECORD NUMBER /
FINDNO: .BLKW 1
CRLF13: .ASCII <CR><LF>
ERR13LN = . - ERR13
;
ERR14: .ASCII /ERROR - OVER 40. RECORDS WRITTEN ON TEST1 / <CR><LF>
ERR14LN = . - ERR14
;
.EVEN ;ALIGN
ERR15: .ASCII /DID NOT GET RECORD NUMBER /
TST5NO: .BLKW 1
CRLF15: .ASCII <CR><LF>
ERR15LN = . - ERR15
;
.EVEN
ERR16: .ASCII /ERROR DISCONNECTING FILE/ <CR><LF>
ERR16LN = . - ERR16
;
; CREATE AN INDEX TABLE OF ADDRESS TO THE ABOVE ERROR MESSAGES
; FOR EASY ACCESS TO THEM
;
.EVEN
ERRTBL: .WORD ERR1
.WORD ERR1LN

```

```

.WORD ERR2
.WORD ERR2LN
.WORD ERR3
.WORD ERR3LN
.WORD ERR4
.WORD ERR4LN
.WORD MSG5
.WORD MSG5LN
.WORD ERR6
.WORD ERR6LN
.WORD ERR7
.WORD ERR7LN
.WORD ERR8
.WORD ERR8LN
.WORD ERR9
.WORD ERR9LN
.WORD ERR10
.WORD ERR10LN
.WORD ERR11
.WORD ERR11LN
.WORD ERR12
.WORD ERR12LN
.WORD ERR13
.WORD ERR13LN
.WORD ERR14
.WORD ERR14LN
.WORD ERR15
.WORD ERR15LN
.WORD ERR16
.WORD ERR16LN
.PAGE
.SBTTL ORGS AND MAIN PROCESSING LOOP

```

```

;
;
;
THERE ARE TWO "ORGS" REQUIRED FOR THIS PROGRAM. ONE FOR THE RELATIVE
FILE AND ONE FOR THE SEQUENTIAL FILE.

```

```

ORG$ REL,<CRE,FIN,GET,PUT,DEL>
ORG$ SEQ,<CRE,PUT>

```

```

;
;
;
NEXT IS THE MAIN CONTROL LOOP FROM WHICH EACH TEST IS CALLED
IN SEQUENCE. IF ALL WENT WELL, THE PROGRAM WILL EXIT HERE
ALSO AFTER ALL TESTS HAVE BEEN RUN.

```

```

;
;
;
START:
$INIT ;GO INITIALIZE THE RMS-11 FILES
CALL CRETTY ;CREATE, CONNECT TTY FILE
CALL CREATE ;CREATE, CONNECT RELATIVE FILE
CALL TEST1 ;TEST 1 BEGINS
CALL TEST2 ;TEST 2 BEGINS
CALL CLOSE ;DISCONNECT AND CLOSE RELATIVE FILE
CALL TEST3 ;TEST 3 BEGINS
CALL TEST4 ;TEST 4 BEGINS
CALL TEST5 ;TEST 5 BEGINS
CALL CLOSE ;DISCONNECT AND CLOSE RELATIVE FILE

```

```

EXIT$$
.PAGE
.SBTTL CREATE BOTH FILES
;
;
;
CREATE AND CONNECT THE TTY FILE
;
CRETTY:
MOV #0,PRAMTR ;SET UP TTY CREATE ERRMSG
$CREATE #FABTI,#ENDIT ;CREATE TTY FILE
MOV #4,PRAMTR ;SET UP TTY CONNECT ERR
$CONNECT #RABTI,#ENDIT ;CONNECT TTY FILE
RETURN ;BACK TO THE CALLER
;
;
CREATE THE RELATIVE FILE
;
CREATE:
MOV #RABAR,R4 ;INITIALIZE RAB POINTER
$STORE #RECLN,RSZ,R4 ;SET UP RECORD SIZE
CLR R1 ;RO R1 SET UP 2 WORD MRN
MOV #40.,R0 ;SET UP RO FOR MRN
MOV #FABAR,R5 ;SET UP FAB POINTER
$STORE R0,MRN,R5 ;INITIALIZE 40. AS MRN
MOV #10,PRAMTR ;SET UP CREATE REL FILE ERR
$CREATE R5,#ENDIT ;CREATE THE FILE
RETURN ;BACK TO CALLER
;
;
SET UP THE THE CONNECT AS A SUBROUTINE
;
CONNECT:
MOV #14,PRAMTR ;SET UP CONNECT REL FILE ERR
MOV #RABAR,R5 ;SET UP RAB POINTER
$CONNECT R5,#ENDIT ;CONNECT REL FILE
RETURN ;BACK TO CALLER
;
;
REWIND THE FILE
;
REWIND:
MOV #44,PRAMTR ;SET UP FOR REWIND ERROR
$REWIND #RABAR,#ENDIT ;GO TO B-O-F FOR NEXT TEST
RETURN ;BACK TO CALLER
;
;
DISCONNECT AND CLOSE THE FILE
;
CLOSE:
MOV #74,PRAMTR ;SET UP FOR DISCONNECT ERROR
$DISCONNECT #RABAR,#ENDIT ;DISCONNECT FILE
MOV #40,PRAMTR ;SET UP FOR CLOSE ERROR
$CLOSE #FABAR,#ENDIT ;CLOSE FILE
RETURN ;BACK TO CALLER
.PAGE
.SBTTL ERROR HANDLING ROUTINE
;
;
THE FOLLOWING ROUTINES HANDLE THE ERROR TRAPPING FROM ANY ACCESS

```



```

;          TO A RMS-11 FILE
;
;
; PRINT THE MESSAGES
;
PRINT:
MOV     #0,R2          ;CLEAR R2
MOV     PRAMTR,R2      ;SET UP INDEX WITHIN TABLE
ADD     #ERRTBL,R2     ;POINTER TO ERROR MESSAGE
MOV     #RABTI,R3      ;POINTER TO ERROR TABLE
$STORE (R2),RBF,R3     ;STORE ADDRESS OF MESSAGE
$STORE 2(R2),RSZ,R3    ;STORE SIZE OF MESSAGE
$PUT   R3,#BOMB        ;TTY MSG - BOMB ON ERROR
RETURN  ;BACK TO CALLER

;
; IF WE GET HERE THERE IS NO HOPE SO PRINT ERROR MESSAGE THEN...
;
ENDIT:
CALL    PRINT          ;PRINT ERROR MESSAGE

;
; BOMB OUT WITH BPT TRAP BY SYSTEM. ONLY GET HERE ON SEVERE ERRORS
;
BOMB:
BPT     ;BOMB

;
; .PAGE
; .SBTTL TEST 1
;
;-----
;TEST 1
; THIS IS A TEST OF MRN. MRN SET TO 40.
; WILL ATTEMPT TO WRITE 50. RECORDS.
; SHOULD ENCOUNTER ER$MRN ERROR.
;-----
;
TEST1:
CLR     R1              ;INITIALIZE R1
CALL    CONECT          ;GO CONNECT THE RELATIVE RAB TO THE FAB

PUTIT:
INC     R1              ;NEXT RECORD
$PUT   R4              ;WRITE THE RECORD
$COMPARE #SU$SUC,STS,R4 ;PUT ERROR?
BNE     TIERR           ;IF YES, GO TO TIERR
CMP     #50.,R1         ;WRITTEN 50 RECORDS YET?
BHI     PUTIT           ;NO, GO WRITE SOME MORE
MOV     #64,PRAMTR      ;SET UP FOR TEST 1 ERROR
JMP     ENDIT           ;BOMB OUT-WITH ERROR MESSAGE PRINT

TIERR:
MOV     #24,PRAMTR      ;SET UP FOR PUT ERROR
$COMPARE #ER$MRN,STS,R4 ;MRN ERROR?
BNE     ENDIT           ;NOT MRN ERROR--BOMB
MOV     #20,PRAMTR      ;SET UP FOR SUCCESS MSG
MOVB   #61,TESTNO      ;SET UP FOR PRINTING TEST 1

```

```

CALL    PRINT           ;GO PRINT TEST 1 SUCCESS MESSAGE
CALL    REWIND          ;RETURN RELATIVE FILE TO B-O-F
RETURN  ;BACK TO CALLER

;-----
; IF TEST 1 IS SUCCESSFUL, RECORDS HAVE BEEN WRITTEN
; UP TO MRN. THAT IS, 40 RECORDS HAVE BEEN WRITTEN.
; THE FILE HAS BEEN REWOUND, READY FOR TEST 2.
;-----
;
; .PAGE
; .SBTTL TEST 2
;
;-----
;TEST 2
; THIS TEST ATTEMPTS TO RETRIEVE RECORDS
; SEQUENTIALLY BEYOND MRN.
; SHOULD ENCOUNTER AN ER$EOF ERROR.
; IT ATTEMPTS TO RETRIEVE 50. RECORDS.
; AFTER THE 40.TH RECORD, AN ER$EOF ERROR
; SHOULD RESULT.
;-----
;
TEST2:
MOV     #RABAR,R4      ;SET UP RAB POINTER
CLR     R1              ;INITIALIZE RECORD COUNTER
MOV     #34,PRAMTR     ;SET UP FOR GET ERROR
MOVB   #62,TESTNO      ;SET UP FOR PRINTING TEST 2

T2INC:
INC     R1              ;INCREMENT RECORD COUNTER
$GET   #RABAR          ;GET ANOTHER RECORD
$COMPARE #SU$SUC,STS,R4 ;SUCCESSFUL GET?
BNE     GETERR         ;IF NOT, GO TO GETERR
CMP     #50.,R1         ;GOTTEN 50 RECORDS YET?
BNE     T2INC          ;IF NOT, CONTINUE
RETURN  ;BACK TO CALLER

GETERR:
$COMPARE #ER$EOF,STS,R4 ;EOF ERROR?
BNE     ENDIT           ;NOT AN EOF ERROR - BOMB
MOV     #20,PRAMTR      ;SET UP FOR TEST SUCCESS MESSAGE
CALL    PRINT          ;PRINT SUCCESS MESSAGE
CALL    REWIND          ;REWIND RELATIVE FILE AGAIN
RETURN  ;BACK TO CALLER

;
; .PAGE
; .SBTTL TEST 3
;
;-----
;TEST 3
; CREATE A NEW FILE.
; PUT EVERY OTHER RECORD RANDOMLY UP TO AN MRN OF 40.
; REWIND, AND GET SEQUENTIALLY.

```

```

; SHOULD GET ONLY THOSE RECORDS JUST PUT.
;-----
;
TEST3:
CALL CREATE ;CREATE NEW RELATIVE FILE
$STORE #RB$KEY,RAC,R4 ;RELATIVE ACCESS MODE
CALL CONNECT ;CONNECT TO THE FILE
MOV #1,KEYNO ;INITIALIZE THE RELATIVE KEY
CLR KEYNO+2 ;CLEAR 2ND WORD OF KEY
CLR RECKEY+2 ;CLEAR 2ND WD OF KEY IN FILE
;
;SET UP FILE - WRITE EVERY OTHER RECORD
;
SETUP:
MOV #24,PRAMTR ;SET UP FOR PUT ERROR
MOV KEYNO,RECKEY ;PUT KEY NUMBER IN RECORD
$PUT #RABAR,#ENDIT ;PUT RCD; ERROR = MSG, END.
ADD #2,KEYNO ;SKIP A RECORD
CMP #40.,KEYNO ;FILE FULL?
BHI SETUP ;IF NOT, WRITE SOME MORE
MOV #20,PRAMTR ;SET UP FOR TEST 3 SUCCESS
MOVB #63,TESTNO ;SET UP FOR PRINTING TEST 3
;
;***** COMMON ROUTINE FOR TESTS 3 AND 4 *****
;
;
; REWIND AND REINITIALIZE
;
SEQGET:
CALL REWIND ;RETURN TO BEGINNING OF FILE
MOV #-1,KEYTST ;SET UP TO TEST FOR CORRECT RCD
$STORE #RB$SEQ,RAC,R4 ;GET SEQUENTIALLY
;
; GET THE RECORDS SEQUENTIALLY
;
GETIT:
$GET R4 ;GET RECORD
$COMPARE #SU$SUC,STS,R4 ;SUCCESSFUL GET?
BNE GETERR ;IF NOT, TEST FOR E-O-F
;GETERR IN TEST 2
;
RECTST:
ADD #2,KEYTST ;EVERY OTHER RECORD
CMP KEYTST,RECKEY ;RIGHT RECORD?
BEQ GETIT ;IF SO, LET'S GET ANOTHER
;
BADREC:
CMPB #1,TSTFOR ;IS IT TEST 4?
BNE BADCON ;IF NOT, WRONG RCD
ADD #6,DELTST ;HOLD KEY OF NXT DELETED RCD
CMP DELTST,KEYTST ;IS IT A DELETED RECORD?
BEQ RECTST ;YES, ADD 2 TEST AGAIN
;
BADCON:
MOV #54,PRAMTR ;WRONG RECORD = MSG
JMP ENDIT ;PRINT OUT MESSAGE, BOMB
;

```

```

.PAGE
.SBTTL TEST 4
;
;-----
;TEST 4
; TEST 4 USED THE FILE CREATED IN TEST 3.
; SOME RECORDS ARE DELETED.
; GET SEQUENTIALLY AND SEE IF DELETED RECORDS
; ARE FOUND.
;-----
TEST4:
MOV #RABAR,R4 ;SET UP RAB POINTER
$STORE #RB$KEY,RAC,R4 ;DELETE USING RANDOM
MOV #-5,KEYNO ;INIT REL KEY FOR DELETES
MOVB #1,TSTFOR ;SET UP FOR FOURTH TEST
MOV #-5,DELTST ;INIT FOR DELETED RCD KEY
;
NXTDEL:
ADD #6,KEYNO ;DELETE EVERY 4TH RECORD
CALL FINDIT ;GO DELETE THE RECORD
CMP #37.,KEYNO ;LAST RECORD
;TO BE DELETED BEFORE EOF?
;IF NOT, DELETE SOME MORE
;ANY ERRORS ENCOUNTERED?
BHI NXTDEL ;NO ERRORS, GET SEQUENTIALLY
CMPB #0,ERR ;ERRORS FOUND - BOMB
BEQ GET4
JMP BOMB
;
GET4:
MOV #20,PRAMTR ;SET UP FOR TEST 4 SUCC MSG
MOVB #64,TESTNO ;SET UP FOR PRINTING TEST 4
CALL SEQGET ;READY TO GET SEQUENTIALLY
RETURN ;BACK TO CALLER
;
; CODE TO FIND A RECORD IN THE RELATIVE FILE
;
FINDIT:
$FIND #RABAR
$COMPARE #SU$SUC,STS,R4 ;FIND SUCCESSFUL?
BEQ DELETE ;IF IT WAS, DELETE RECORD
MOV #60,PRAMTR ;SET UP FOR FIND ERROR
MOV KEYNO,R0 ;SET UP FOR KEY NO.
;CONVERSION TO ASCII
CALL CONVER ;GO CONVERT REC. NO. TO ASCII
MOV HOLDIT,FINDNO ;PUT REC. NO. IN ERRMSG
CALL PRINT ;PRINT ERROR MESSAGE
RETURN ;BACK TO CALLER
;
; CODE TO DELETE A RECORD ONCE ITS BEEN FOUND
;
DELETE:
$DELETE #RABAR ;DELETE RECORD
$COMPARE #SU$SUC,STS,R4 ;SUCCESSFUL DELETE?
BEQ RETURN ;IF SUCCESSFUL, CONTINUE
MOV #30,PRAMTR ;SET UP FOR DELETE ERRMSG
MOV KEYNO,R0 ;SET UP FOR ASCII CONVERT

```

```

CALL CONVER          ;CNV KEY NO. TO ASCII FOR ERRMSG
MOV  HOLDIT,DELNUM  ;PUT RCD NO. IN ERRMSG
CALL  PRINT         ;GO PRINT ERROR MESSAGE
;
RETURN:
RETURN              ;BACK TO CALLER
;
.PAGE
.SBTTL CONVERT BINARY TO ASCII
;
CONVERSION TO ASCII TO PRINT OUT RCD NBR IN ERRMSG
CONVERTS TO 2 ASCII DIGITS
;
CONVER:
MOVB #1,ERR          ;NOTE THAT ERROR OCCURRED
MOV  #RABAR,R4       ;SET UP RAB PNTR AFTER ERR
$STORE #1,STS,R4     ;RESET TO GOOD STATUS
MOV  #1,R3           ;SET UP R3 FOR INDEXING
MOV  #2,R2           ;SET UP R2 TO LOOP 2 TIMES:
                          ;ONCE FOR EACH BYTE
;
LOOP:
MOV  R0,-(SP)        ;SAVE REMAINDER FOR CNV
BIC  #-10,R0         ;CLEAR ALL BUT LOW ORDER 3 BITS
ADD  #60,R0          ;CONVERT TO ASCII
MOVB R0,HOLDIT(R3)  ;STORE IT IN RECORD
MOV  (SP)+,R0        ;RESTORE CONTENTS
ASR  R0              ;SHIFT RIGHT 3 BITS
ASR  R0              ;TO WORK ON EACH DIGIT
ASR  R0
DEC  R3              ;DECREM. INDEX TO OUTPUT
SOB  R2,LOOP        ;DIGIT IN EACH BYTE??
RETURN              ;BACK TO CALLER
.PAGE
.SBTTL TEST 5
;
;-----
;TEST 5
THIS TEST WILL TEST THE ROP FIELDS OF
KGT AND KGE.
AFTER SUCCESSFUL COMPLETION OF TEST 4,
THE FILE WILL NOW HAVE OCTAL RECORDS:
3,5,11,13,17,21,25,27,33,35,41,43,47.
USING THIS INFORMATION, WE WILL TEST
'KEY GREATER THAN' AND 'KEY GREATER THAN
OR EQUAL TO'.
;-----
;
TEST5:
MOV  #RABAR,R4       ;SET UP RAB POINTER
$STORE #RB$KEY,RAC,R4 ;USING RANDOM
$STORE #RB$KGT,ROP,R4 ;SET UP KEY GREATER THAN
CALL  KGT            ;DO KEY GREATER THAN TEST
$STORE #RB$KGE,ROP,R4 ;SET UP KEY GTR'N OR = TO

```

39

```

CALL REWIND          ;REWIND TO B-O-F
CALL  KGE            ;DO GREATER OR = TO TEST
CMPB #0,ERR          ;ANY ERRORS DETECTED?
BEQ  GOODS5          ;NO ERRORS = SUCCESS MSG
RETURN              ;BACK TO CALLER
;
GOOD5:
MOVB #65,TESTNO     ;MOV 5 INTO SUCCESS MESSAGE
MOV  #20,PRAMTR      ;SET UP FOR TEST 5 SUCC MSG
CALL  PRINT         ;PRINT MESSAGE
RETURN              ;BACK TO CALLER
;
KGT:
MOV  #5,KEYNO        ;WE WANT THE NEXT REC GT 5
MOV  #11,R0          ;IT SHOULD BE RECORD 11
CALL  GET5           ;GET THE RECORD
MOV  #20,KEYNO       ;TRY NEXT REC GTR'N 20
MOV  #21,R0          ;WE SHOULD GET RECORD 21
CALL  GET5           ;GET THE RECORD
MOV  #27,KEYNO       ;WANT THE NEXT REC GTR'N 27
MOV  #33,R0          ;WE SHOULD GET RECORD 33
CALL  GET5           ;GET THE RECORD
MOV  #47,KEYNO       ;GET REC GT 47
MOVB #1,RNFSET       ;SET UP TO TEST FOR RNF
CALL  GET5           ;GET THE RECORD
RETURN              ;BACK TO CALLER
;
; CODE TO DO GREATER THAN OR EQUAL TO TEST
;
KGE:
MOV  #5,KEYNO        ;WE WANT THE REC. GE 5
MOV  #5,R0           ;SHOULD GET RECORD 5
CALL  GET5           ;GET THE RECORD
MOV  #20,KEYNO       ;WE WANT THE REC. GE 20
MOV  #21,R0          ;WE SHOULD GET RECORD 21
CALL  GET5           ;GET THE RECORD
MOV  #27,KEYNO       ;WE WANT THE REC. GE 27
MOV  #27,R0          ;SHOULD GET RECORD 27
CALL  GET5           ;GET THE RECORD
MOV  #47,KEYNO       ;WE WANT THE REC. GE 47
MOV  #47,R0          ;SHOULD GET RECORD 47
CALL  GET5           ;GET THE RECORD
RETURN              ;BACK TO THE CALLER
;
; CODE TO GET A RECORD FROM THE FILE
;
GET5:
$GET  R4              ;GET THE RECORD
$COMPARE #SU$SUC,STS,R4 ;SUCCESSFUL GET?
BEQ  COMPAR          ;YES = TEST FOR CORRECT RCD
CMPB #1,RNFSET       ;TESTING FOR RNF ERROR?
BNE  GT5ERR          ;NO = CONT ERR PROCESSING
MOVB #0,RNFSET       ;RESET FOR NO RNF ERROR
$COMPARE #ER$RNF,STS,R4 ;DID WE GET AN RNF ERROR?
BNE  GT5ERR          ;NO, THERE'S REALLY AN ERROR
RETURN              ;BACK TO CALLER

```

```

GT5ERR:  MOV     #34,PRAMTR      ;SET INDEX FOR GET ERR
        $$STORE #1,STS,R4    ;RESET TO GOOD STATUS
        MOVB   #1,ERR        ;SET ERR BYTE
        CALL  PRINT          ;GO PRINT THE ERROR MESSAGE
        RETURN              ;BACK TO THE CALLER
;
; CODE TO COMPARE THE RECORD FOUND WITH THE ONE REQUESTED
; R0 CONTAINS THE RECORD NUMBER SOUGHT
;
COMPAR:  CMP     R0,RECKEY     ;CORRECT RECORD?
        BNE   ERROR5        ;IF NOT GO TO ERROR ROUTINE
        RETURN              ;BACK TO THE CALLER
ERROR5:  MOV     #70,PRAMTR    ;SET INDEX IN TABLE FOR ERROR MSG
        CALL  CONVER         ;CONVERT REC. NO. TO ASCII FOR MESSAGE
        MOV   HOLDIT,TST5NO  ;PUT RCD NBR IN MSG
        CALL  PRINT          ;GO PRINT ERROR MESSAGE
        RETURN              ;BACK TO CALLER
;
;
.PAGE
.SBTTL  SAMPLE MAC AND TKB COMMAND STREAMS
;
; THE FOLLOWING ARE SAMPLE COMMAND STREAMS THAT ARE USED TO ASSEMBLE
; AND TASK BUILD THIS PROGRAM USING RMS-11. THESE STREAMS
; LEND THEMSELVES TO USING COMMAND FILES AS THEY ARE
; LENGTHY.
;
; FIRST IS THE COMMANDS TO ASSEMBLE THIS PROGRAM. THE ASSUMPTIONS ARE:
;
; 1) THE NAME OF THIS FILE IS "RELSEQ.MAC"
; 2) YOUR ACCOUNT IS SHOWN AS "[X,Y]"
; 3) THE RMS MACRO LIBRARY IS IN [1,1]
;
; THE RESULTING COMMAND STREAM IS:
;
RELSEQ,RELSEQ/CR/-SP=[1,1]RMSMAC/ML,[X,Y]RELSEQ
;
; NEXT IS THE TASK BUILD COMMAND STREAM FOR THIS PROGRAM. THE SAME
; ASSUMPTIONS AS ABOVE ARE IN EFFECT.
;
; THE RESULTING COMMAND STREAM IS:
;
RELSEQ,RELSEQ/-SP=RELSEQ,[1,1]RMSLIB/LB
;
; YOU NOW HAVE AN EXECUTABLE TASK IMAGE THAT WILL PERFORM THE TESTS
; CODED IN THIS PROGRAM AND PROVIDE A BASE FOR DEVELOPING
; YOUR OWN RELATIVE AND SEQUENTIAL FILES USING RMS-11.
;
;

```

```

;
;
; .EVEN
; .END  START

```

The following is the output of the program when run on a PDP 11/70 under IAS Ver. 3.0.

```

RUN  RELSEQ
19:33:41
TEST 1 SUCCESSFUL

TEST 2 SUCCESSFUL

TEST 3 SUCCESSFUL

TEST 4 SUCCESSFUL

TEST 5 SUCCESSFUL

19:33:47  Size: 11K  CPU: 0.86

PDS>>

```

**\*\* FMS WORKING GROUP SURVEY \*\***

Judy Kessler, Chairman

During the Miami DECUS '81, we distributed survey forms to current and prospective FMS users. Our goal is to promote active interchange of knowledge in this (relatively) new product area. Here are some of the responses:

Special Interest Areas:

1. Process control front end.
2. Menu picking.
3. Real time sytem status display.
4. Shared libraries and commons.
5. Accounting applications.
6. On line inquiry.
7. Transaction entry.
8. General purpose transaction package (menu driven) which performs data conversion, range checking, special validations.
9. Help forms (reference guide on-line).
10. CAI (Computer Aided Instruction).
11. Manufacturing data entry.
12. Realtime pipeline control.
13. Display generation and update.
14. Double height/double width display.
15. Banking applications MIS data entry, report writer.
16. Data entry for Datatrieve.
17. Samples of add, delete and change programs.
18. Placing FMS on a distributed processor to offload host laboratory data entry and display.

Specific Questions:

1. How can I modify video attributes of a field in a displayed form at run time?
2. Does anyone have an FMS-11 front end to OMSI PASCAL?
3. When will FMS-RSTS be released?
4. How easily does FMS move from PDP-11's to VAX's? Vice-Versa?
5. Are many commercial users using FMS for their financial applications?
6. If I use FMS, will I have an overhead problem in also using RMS?
7. How can we improve swapping time (RSX-11, IAS)?
8. Does anyone else have problems fitting FMS and/or RMS and a user program in 32K? If so, how were they solved?

Please send your questions, comments and answers to:

Judy Kessler  
Retina Foundation  
20 Staniford Street  
Boston, Mass. 02114

I would like to take this opportunity to DENY that I am now, or ever have been, the chairman of the Retina Foundation !!!

Digital Responses

To

FMS Menu

DECUS Symposium Spring 1981  
Miami Beach, Florida

<u>ITEM</u>	<u>DESCRIPTION</u>	<u>RESPONSE</u>
53	Multiple contexts/streams per terminal.	A worthy and ambitious long-term goal. Not likely in next release.
48	Support non-DEC terminals or provide specs/interfaces so users can.	Unlikely.
56	Provide a single reference manual for all O/S (e.g., Fortran IV Plus).	Unlikely, but we will make the various implementations even more identical and document all compatibility/portability issues.
44	Single character I/O interfaces with swapping.	Has top priority for next release.
50	Data Conversion (Integer and Real).	We'll consider it.
58	Source licenses?	We don't have them.
47	Support VT100 line-drawing character set.	Probable for next release. See answers for Double High/Wide.
51	More field validation (eg., range checks, table look-up).	Next version will probably add some more validations and make it easier for you to add your own.
45	Alternate keypad invocable on a Per-Key Basis.	Unlikely for next release.
52	Support more than one terminal/task.	We'll consider it for next release.
46	Support VT100 Double Height and Double Width Characters.	Probable for next release. FDV works if you can figure out how to patch your forms. Ian Darwin <sup>1</sup> has done this successfully.
49	Provide Graphics Support.	Full Graphics would make FMS a different product and is therefore unlikely. Mark Sebern <sup>2</sup> has ideas on ways to build such a system on top of FMS.

<u>ITEM</u>	<u>DESCRIPTION</u>	<u>RESPONSE</u>
54	Hardcopy Snapshots of Terminal Activity.	Probably in next release. Short-term alternatives include SWS modifications (already done for RSX, in process for VMS) and VT1XX-AC printer port option.
41	Split screen support in Horizontal and Vertical directions.	Lack of horizontal scrolling is terminal HW characteristic that would kill performance if emulated.  Some amount of this can be achieved today via scrolling, clever overlaying of forms, clever use of video attributes etc.
42	Call like FGETAL to start in middle of form and continue to end.	Will consider for future release.
55	Reduce program overhead requirements.	Top priority for next release.
39	Document the use of FMS-11 with Pascal.	We will consider the use of FMS with any DIGITAL Pascals that are available at the time of the next release. We recommend the documentation of non-DIGITAL Pascals as an activity for the FMS working group of the DMS SIG.
57	Support BASIC-11 under RSX for easier migration from BASIC-11/RT-11.	Unlikely. BASIC-PLUS-TWO is our main offering in this space. Besides, moving programs back and forth is quite simple - we do it all the time.
43	126th form in a library is not accessible.	To be fixed in the next release of FMS.
34	Hardcopy terminal support for line-by-line dialogue and messages.	Not in next release, maybe later.
40	Demos distributed with the kits. Unsupported is OK!	We have done this VAX-11 FMS. In the future we will try to eliminate the need for this by providing more examples and techniques and formal Ed. Svcs. training.
1.	Ian Darwin Univ. of Toronto Computing Services 10 King's College Road Toronto, Ontario M5S 1A1	2. Dr. Mark Sebern Sebern Eng. Inc. W55 N015 Cedar Ridge Drive Cedarburg, Wisc. 53012 414-375-2200

Patching FMS-11 Forms for VT100 Special Features  
Ian Darwin

There are two ways of modifying FMS-11 forms to make use of the VT100 line drawing set and the double-height, double-width lettering. One is by patching escape sequences into text segments on the form; the other is by patching the form attributes for a text segment. The first way is easier, but uglier. Neither way is supported by DEC. We have used the first means extensively.

There is also a way to get just boxes on a form. This way uses FED only; it is fully supported by DEC but not quite as elegant.

1.0 Patching Text

To patch the text fields, you must start by making up a form, called for example ESCFRM, and then make all your forms from it. You patch ESCFRM, then make up new forms with the patched text. This relies on the (undocumented?) feature that the forms driver will pass escapes which it finds in the text without clobbering them. THIS IS NOT GUARANTEED TO CONTINUE, in fact I don't even warrant that it works on any Forms Driver except RT-11, although it probably does.

Your ESCFRM would look something like this:

```
$#3This is double height (top half).
#$4This is double height (bottom half).
#$5This is normal text.
#$6This is double width single height.
$(0This enters graphics mode.
$)0This leaves graphics mode.
```

ESCFRM now contains the ANSI escape sequences needed to put the VT100 into the specified modes (you could use VT-52 compatible escape sequences). Just type ESCFRM up using FED and typing a dollar sign where there ought to be an escape. Then run whatever DUMP program is on your system to get an ASCII-and-byte dump of the form file. Finally, use whatever binary-file-patch program is on your system (PATCH on RT-11 V3, for example) to convert the "\$" characters to 233 octal (use 233, not 033, to work on most systems).

Once you have done this, use FED to look at the ESCFRM form to make sure that the escape sequences are correct (I got them from a manual, not from a working copy of ESCFRM, which is on another system right now) and that you have patched them correctly.

Now make up a new form from ESCFRM, using FED's cut-and-paste feature to move the escapes in front of the text in your form. For graphics, see the VT100 manual for the translation from lower case letters to line drawing set. YOU WILL HAVE TO DO SOME EXPERIMENTING because FED behaves somewhat strangely. Use CTRL/W whenever you have put the escapes in a different place. Remember that FED thinks it has 80/132 column lines when in fact some of the lines will be displayed with half that many characters.

Have fun with this, but don't call me (or DEC) when it breaks!

2.0 Patching Attributes

Patching the attributes is a more sophisticated way of doing the same thing. You edit the form with FED and then use PATCH (or better yet write an intelligent program to do the same thing) to turn on the attribute bytes for things like Double Height Double Width letters for the appropriate text segment. This is not described here in detail because I HAVEN'T DONE IT YET although the FMS developers have told me that it works for some of the attributes (I'm not sure which work and which don't). To use this, you will have to read the article "FMS-11 Internal Form Structure", which will be in the next issue of the SCHEMA, to get the details of the internal form format.

3.0 If boxes are all you want...

There is a way to draw boxes on a VT100 with AVO (Advanced Video Option) that is fully supported and will continue to work in future releases of FMS-11. The only tool you need is the standard Forms Editor FED. The method is to use the reverse video to make the box around your text. The boxes are bigger than those made with the line set, but they are much faster to make up.

Basically you SELECT a region where you want the outline of your box to be, and mark it VIDEO REVERSE. You then select an inner region (one line below the top and above the bottom, and two columns in from either side), and mark it VIDEO CLEAR. It's actually easier to do this than to describe it; hopefully the following crude diagram will clarify it.

The screen after writing the region:

```
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

Then after clearing the inside:

```
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XX                               XX
XX                               XX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

It looks much better on a real VT100; I have used X's to show where the VT100 will use solid background because my printer doesn't have a good way of doing solid blobs.

4.0 Summary

You can make boxes or use extra features of your terminal by patching the forms; in so doing you may be boxing yourself in later. If boxes are all you really need, make them with FED and save yourself time now and hassles later.

SEBERN ENGINEERING INC.

W55 N815 CEDAR RIDGE DRIVE ■ CEDARBURG, WISCONSIN 53012  
(414) 375-2200

13 June 1981

Paul Clayton  
Newsletter Editor  
DECUS Data Management SIG

Dear Paul:

At your request, I am putting into writing the following comments regarding FMS-11 and its application:

Integration of Graphics with FMS-11

Having sold a client on the virtues of FMS-11 for a new medical data management system (including talking DEC into letting them be a field test site so as not to blow a tight schedule), the next question was "How do we work in the graphics?". After some work, the following approach emerged:

1. Use one of three methods for the actual display of the plot information, depending on the requirements:
  1. Horizontal bar characters from VT100 line drawings character set.
  2. A custom character set (in EPROM) with a set of 121 characters. Each character consisted of a left half and a right half; each half was either blank or contained a single horizontal line (scans 1-10). This set provided twice the vertical and horizontal resolution, and could handle simple f(x) plots.
  3. An add-in bit map graphics board. The one chosen by this client was from Selanar, but the Retrographics unit could have been used if you don't mind mushy smooth scroll. Other units may be available by now. This particular client had no use for the VT105, but that might be an option also.
2. Use FMS-11 for all text labeling, data entry, etc.
3. Integrate the graphics with the FMS form by using the NAMED DATA feature of FMS-11. For example, the dimensions and location of the plot area was specified in the named data. This allows the form to be changed even to the extent of moving the plot around. Other named data items indicated preferred scale factors for

various parameters, etc. FMS-11 display-only fields were used for displaying the parameter names, and similar information.

4. The operation, then, was to have the form driver display the form, and then do direct terminal I/O to put up the plot. So as not to confuse FMS-11, it is normally necessary to save and restore the cursor position when doing the direct I/O; this is done with the VT100 escape sequences. If an add-in graphics board is used, it may be necessary to clear or disable it before displaying the next form.

FMS-11 Problems: Architecture and Scrollins

All in all, most FMS-11 users I know are pretty happy. Still, there are a few problems in the current version. Those that bother me most are what I consider to be violations of the basic FMS architecture. Examples are the FPUTAL and FRETAL functions (except for the default-restoring version of FPUTAL). These functions require a knowledge of the arrangement of fields in the form; the ability to be ignorant of such details is one of the main ideas behind FMS-11 in the first place. Luckily, most applications need not use these functions.

Unfortunately, the situation is the same in another area, and it's not so easy to get around. The problem area: scrollins regions. If you want scrollins to look right, it is necessary to pass an entire scrollins line with the FPFT function. To do this, you have to know the layout of fields in the scrollins region, which makes the application task dependent on form definition. To avoid this problem:

1. One possible solution. Have the application task tell FMS-11 the format it is using for scrolled region information, and let the form driver rearrange it so it falls in the right place. For example, the task would give the form driver a list of field names and sizes in the order that they were packed in the task's data buffer; the form driver would look up the field names, pick up the data from the user, and display it (truncating or filling as necessary if the lengths didn't match). This is the technique used in a "poor man's RMS" multikey file system in use here and at several client sites. (Note that this method could be used with FRETAL and FPUTAL as well.)



2. A temporary work-around. Make the application task do the work, but in a more general way. For example, put the necessary form dependent information into NAMED DATA. A named data item like

```
[SCRL1 ] [QUAN PARTNO DESCR PRICE AMOUNT]
```

would mean that scrolling region number one consisted of the fields QUAN, PARTNO, DESCR, PRICE, and AMOUNT (in that order). The FRETN or FLEN functions can set the actual field lengths to control padding or truncation.

There are other problems with scrolling regions, but for which I'm not sure I can offer better solutions. The introduction of a scrolling region makes the FGETAL function illegal and sets the application task more involved in messy details of form operation. To set around this might require something like co-routine calls back to the user (e.g., to retrieve data to be scrolled in), and that isn't too pleasant, either. For forms with small "virtual" scrolling regions, it might be possible to pass all the data to FMS, including the part not visible; this method breaks down with large data files, however. Luckily, the extra task code required with the current FMS implementation can be written in a fairly general way, and transported from one application to another as needed.

One slight pain in the current scrolling region implementation is the need for the application program to keep two running pointers, one an index into the scrolling region ('what line am I on?') and the second an index into the real data base being displayed. Depending on the state of these pointers, one must use different forms of the FPFT operation and/or refuse to accept the operator's scroll terminator. While I haven't thought this all out, it seems that a scrolling region could be treated more like an indexed set of fields. In such an implementation, the form driver would maintain the current "field index" for the line containing the cursor. This index would continue to increment as the operator scrolled down through the data, and would thus correspond directly to the application's own data index. (If the data base had more records than the index value could represent, this could be a problem.) When the operator tried to scroll off the top or bottom line of the scrolling region, the form driver could return a special terminator indicating the fact so the program would pick up the required data before calling FPFT. The form driver could automatically exit the scrolling region upward from data base entry 1, and do the same downward if the application kept it informed of the highest legal data index value. It seems that such an approach would do away with much of the pointer management now done by FMS application tasks, and the task would not even need to know the size of the form's scrolling region. (There are some minor details to be handled here for scrolling region initialization when the form is first

put up.) Most of the required information is already present in the form driver; it need only be made available to the program in a useful way.

#### Full Duplex Terminal Driver Overhead

Using the RSX-11M full duplex terminal driver for form driver I/O is like . . . (insert your favorite expression for overkill). On an 11/23 and DLV-11J (or other DLV-xxx), the output interrupt overhead alone is about 240 usec, not including QIO queueing and setup time. It is not hard for two 9600 baud terminals doing output to bring the system nearly to a halt. This is particularly unfortunate since the form driver's I/O is really little more than single character input and uninterpreted string output (like RT-11). While we have yet to do it, one client threatens to write a small, resident driver for FMS-only terminals, optimized for speed. If you have a DH-11 or equivalent with DMA output, this is much less of a problem.

#### Hard Copy FMS-11

A real weakness in the current FMS package is the inability to produce hard copy reports in the same "form independent" way. This omission, while not really a part of FMS's operator interaction function, makes it more difficult to design systems using FMS. While FMS takes away a good part of the drudgery of CRT interaction, the pain reappears when it comes time to print a report, thus lessening the perceived value of FMS itself. One quick and dirty solution that would work for a certain subset of hard copy needs is:

1. Use the form editor to create an image of a hard copy form, with the restriction that it consist of a top part (header), a bottom part (footer), and a middle part (an FMS scrolling region).
2. Write a subroutine (similar to the form driver) which would read the above form from a form library, fill it with data, and print it out. The header would appear at the top of the printout, the footer at the bottom, and the scrolling region data would be "stretched" to fill the center of the printout (more than the 23 lines on the scope). Provisions could be made for multiple page reports (e.g., additional forms for subsequent pages).
3. Use the form's NAMED DATA to handle things like the actual size of the hard copy form, etc.

Obviously, this technique would not satisfy all possible application needs, but might work well enough to save some grief in certain cases. Things like LAXXX character pitch, lines

longer than 80/132 characters, etc., would be problems that need work. Documentation from DEC on internal form structure would help, if they don't choose to provide a hard copy FMS themselves.

#### DEC Source License Policy

-----

Now to my pet FMS-11 peeve. Unless it has changed recently, DEC's policy is that no FMS-11 source code is available. As an outsider, I can only guess that this policy rests on one or more of the following assumptions:

1. FMS must be tied to VT100's in order to get people to buy VT100's.
2. Making FMS hard to adapt to 'foreign' terminals will hurt the competition.
3. FMS isn't good enough that people will be willing to pay for it (even for use with foreign terminals).

My observations are:

1. Until recently (if now), DEC couldn't make VT100's fast enough to satisfy the demand.
2. More and more foreign terminals are VT100 compatible, and can be used in any case.
3. DEC has always claimed to compete on 'quality and price (value)', not by dirty tricks aimed at the competition.
4. Users love FMS, and are happy (well, would you believe willing?) to pay for it.
5. VT100's are good enough to compete on their own merits.
6. DEC's 'no source' policy results in hardships for their own best customers, not for the competition:
  1. A client needed to change the 'last line' location to allow for system-specific alarm and warning messages and to modify some forms for double width lines. Both of these changes were much more difficult than necessary, since well commented sources were not available.
  2. Another client chose to go with a large number of VT100's in a new system, but still had some old CRT's (manufactured in-house) on the predecessor system. Desiring to have compatible software on both systems, this company faces needless difficulty in supporting its old terminals under FMS during the remaining life of the old systems.

This client chose to go with FMS on the new system anyway, but I'm not sure I would have blamed them if they chose not to.

3. At the Chicasso FMS user panel, one user told DEC's Cheryl Vedoe that they had made modifications to the form editor that were necessary in their environment. Cheryl asked how they did this without sources; the answer: "we disassembled it". Is this how DEC computers aid productivity?
7. Judging from reaction at the Chicasso DECUS FMS panel, I don't think I'm alone in my criticism of the "no sources" policy. If you feel the same, help apply the heat. I have to believe that DEC hasn't totally given up it's past dedication to meeting user needs.

I hope that these comments are of interest to readers of the DMS SIG newsletter. Suggestions regarding the FMS work-arounds and enhancements are welcome.

Sincerely,



Mark J. Sebern  
President

Before answering the specific issues in the menu, I must mention that the most important problem facing DBMS-11 is getting the features in version 1.8 to customers on IAS. The value of new features must be weighed against the delay they would cause the next release. Menu items that have little impact on the whole system are more likely to appear in the next release than those that affect many parts, regardless of their relative priority in the menu. Those items that can not be done for this release will be considered in future releases.

We appreciate the time you put into formulating and ordering the menu.

To respond to specific items:

1. "Provide support of I\*4 sort keys as well as R\*4 and R\*B.  
Adding the code to handle new sort keys is not difficult. Unfortunately it involves changing the subschema data type representation, which affects most of the utilities. Although we recognize that this is a valuable feature, and one that the SIG has asked for often, it is unlikely to appear in the next version of DBMS-11.
2. "Provide a function to force an "End of Volume" condition on a database Journal. This would allow an operator to change tapes at his convenience."  
It may be possible to add a command to DBQ to cause DBM to go to the next volume of the Journal. If time allows the command will be added.
3. "Provide display capability for COMP-1 and COMP-2 items."  
If time allows, DRQ will display real and all integer data types in the next release.
4. "Provide support for DBQ STOP by access mode. This would allow the control of update versus retrieval operations. In addition, with update prevented a database backup can be done while being accessed for retrieval."  
Adding support for STOP by access mode is possible, although less likely than either "EOV" support or DBQ display enhancements. We will investigate the possibility of modifying DRQ DUMP to allow it to run if updating programs are locked out.
5. "Modify DBQ to process lower case and floating point values properly"  
This improvement is likely since it involves only DRQ.

6. "Improve recovery to reduce the number of passes over the Journal tape. This would have a significant effect on JTFIX, RLFWRD, and RLBACK."  
Using JTFIX in two file mode (in which it creates a new copy of the Journal rather than "fixing" the existing copy) reduces the number of passes it makes over the tape. We are unlikely to change the way Journaling works beyond that, and making the recovery code more robust.
7. "Allow shared update and automatic recovery, still insuring recovery integrity."  
This menu item involves changes far beyond the scope of the current release. To make automatic recovery possible with shared update would involve very serious changes to DBM with a considerable risk of making the system less reliable.
8. "Access to current database status from user program. This would allow the user program to determine if the database is quiescent. (This is important for certain maintenance operations)."  
If this can be done without significantly affecting the DBCS, we will do it for the next release.
9. "Provide performance measuring tools for the DBMS-11 product"  
We will investigate making the "finish statistics" available to the user program on demand, rather than just at a finish.

USE OF DBMS-11 IN SPACECRAFT  
COMMAND AND CONTROL AT NASA

John K. Nieberding  
Westinghouse Electric Corporation  
Baltimore, Maryland

ABSTRACT

Westinghouse designed and developed a baseline, modular, Data Base-driven software system on PDP 11/70 computers for the command and control of several spacecraft currently being supported by NASA's Goddard Space Flight Center. The Digital Equipment Corporation's DBMS-11 was utilized for the Data Base Management Subsystem.

Each spacecraft application has a unique Data Base composed of telemetry, command, and display parameters which had been specified by spacecraft personnel. The raw Data Base inputs are stored in an "Information Data Base" using DBMS-11. This data is retrieved, also using DBMS-11, and reformatted into efficient Data Base files, called the "Operations Data Base", for rapid access to support real-time spacecraft command and control functions. Each spacecraft's Information and Operations Data Bases are then integrated into the modular software system to produce each unique spacecraft software system.

INTRODUCTION

NASA's Goddard Space Flight Center (GSFC), which is located in Greenbelt, Maryland, is the center of a world-wide ground system that provides for the command and control of several orbiting scientific spacecraft. In 1977, GSFC contracted the Westinghouse Electric Corporation for complete hardware/software computer systems to (1) develop an operations control center for the command and control of the third High Energy Astronomy Observatory satellite (HEAO-3) and (2) modernize the existing Multi-Satellite Operations Control Center (MSOCC-I) which has command and control responsibilities for as many as twelve other scientific satellites.

During this contract, seven Digital Equipment Corporation (DEC) PDP 11/70 computer systems were installed in these control centers along with a

baseline, modular, Data-Base-driven software system which is utilized as the nucleus of all the individual real-time spacecraft software systems. Each spacecraft software system is developed by starting with this baseline software nucleus, adding any spacecraft-unique software modules or modifications, as required, and incorporating a spacecraft-unique Data Base.

DEC's DBMS-11 was chosen as the Data Base Management System for the generation and maintenance of each spacecraft Data Base. This paper describes the Data Base philosophy of the real-time spacecraft software systems and the use of DBMS-11 to implement this philosophy. Prior to this discussion, a brief overview of a GSFC control center and its functions and equipment is presented. It is assumed that the reader is familiar with DBMS-11 and the CODASYL

Data Base terminology as described in the DEC manual "Data Base Administrator's Guide (DEC-1100DABA-B-D)".

GSFC Operations Control Centers Overview

Figure 1 presents a functional diagram of a typical real time command and control application in a GSFC operations control center. Spacecraft telemetry data is first received at one of the NASA ground stations located throughout the world. This serial bit stream from the satellite is formatted by the station into 4800-bit blocks and transmitted over the NASA Communication System (NASCOM) to GSFC where it is routed to the appropriate operations control center.

The telemetry blocks may then be input to a front end preprocessor for any required data reformatting. A PDP 11/34 computer system is presently used as the preprocessor. The telemetry blocks are then sent on to one of the PDP 11/70 computer systems (called "Application Processor") where the software system for this spacecraft is currently active. Each software system contains telemetry modules which utilize information from the Data Base to deblock and process the telemetry. Telemetry processing includes the following:

- (a) decommutation - detection and extraction of specific telemetry discrete or analog values out of the raw telemetry bit stream
- (b) quality checking and time-tagging of the data
- (c) limit checking of analog telemetry values to determine if they are in their proper operating ranges, and generating operator alarms for each value which is too high or too low
- (d) checking of discrete (status) telemetry values to determine if the spacecraft and its payload are in the expected configuration, and generating operator alarms for each discrete in an unexpected state
- (e) computing of equations (algorithms) to generate additional data values for display and monitoring

Requested display data and alarms are generated and updated, and transmitted to CRT

displays in a Master Control Room, where control center personnel monitor the health and safety of the spacecraft and its payload. Display formats are defined in the Data Base for CRT displays as well as line printers and strip chart pen recorders.

Commands for the spacecraft are initiated in one of two ways. The control center personnel can manually key-in each command request to a PDP 11/70 Application Processor from the CRT's in the Master Control Room, or they can activate a previously-developed Procedure which will automatically issue commands at predefined times. The System Test and Operations Language (STOL) is a NASA-defined language which provides for both this manual man-machine interface and the automatic Procedure execution. The STOL processor is part of the baseline software system and executes in each Application Processor.

The requested command bits, as obtained from Data Base, are formatted into blocks and transmitted from an Application Processor to a ground station via NASCOM. The station deblocks the command data and transmits it to the spacecraft.

The "Support Functions" illustrated in Figure 1 refer to the fact that each PDP 11/70 Application Processor may send data to, and receive data from, an IBM 370 computer. Telemetry data is sent to the IBM computer for more detailed processing and analysis, and spacecraft command loads are received from the IBM computer for subsequent transmission to the spacecraft.

Application Processor Hardware

Figure 2 illustrates a block diagram for each Application Processor in the MSOCC-I Operations Control Center. Each Application Processor features a PDP 11/70 computer with 768 Kbytes of main memory. The peripherals include an LA36 Decwriter, two 600 lines/minute Data Products line printers, four TEL6 magnetic tapes, a 1 Mbyte R504 disk, and an 88 Mbyte RP04 disk. A 176 Mbyte RP06 disk has recently been added to each system.

Each PDP 11/70 computer includes an interface to an IBM 370 computer, and an interface to receive and transmit the 4800-bit telemetry and

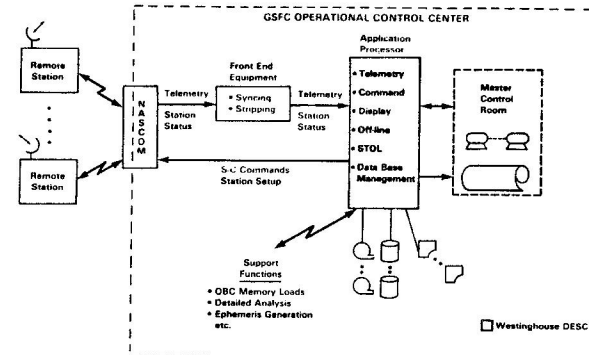


Figure 1. GSFC Operational Control Center Functional Diagram

command blocks. Each computer can also be connected to one or two PDP 11/04 computers which transmit display data to strip chart pen recorders.

Up to 15 keyboard CRT terminals (KCRT's) can be electrically switched into each computer. ADDS 980 terminals are utilized as the KCRT terminals in the Master Control Room, as well as in the computer equipment area, for spacecraft monitoring and control.

Data Base Philosophy

Each application software system operating in an Application Processor is Data Base-driven in that all telemetry information, all command information, and the "canned" display formats utilized by the software system are defined in a spacecraft-unique Data Base. The information for a spacecraft Data Base is generated initially by Project personnel familiar with the telemetry and command characteristics for that particular spacecraft. The Project personnel are responsible for generating a magnetic tape with this Data Base information in predefined formats.

An initial Data Base is supplied by the Project to the Software Development Contractor who is developing and testing a particular spacecraft software system. The Project updates the Data Base as required to produce a launch-ready Data Base for inclusion in the software system which

will support launch and flight operations. Then, for the lifetime of the mission, the Project supplies updated Data Bases to the NASA Data Base Administrator for inclusion in the software system when required.

Figure 3 illustrates a functional block diagram of the off-line Data Base subsystem and

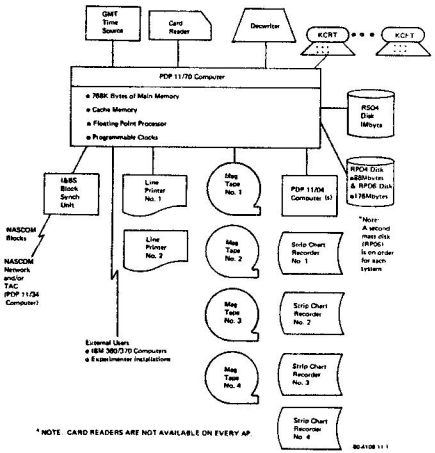


Figure 2. Application Processor Block Diagram

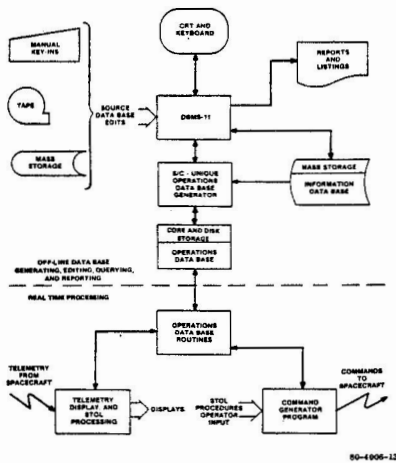


Figure 3. Data Base Functional Block Diagram

how it interfaces with the real time processing within each spacecraft software system. Source lines are usually input from a Project-supplied magnetic tape but may also be input from disk or from manual key-ins. Each line in the Data Base is eighty ASCII characters in length and, therefore, is easily generated by the DEC commercial text Editor on the Application Processor or with another computer elsewhere.

The application software system utilizes the DEC Data Base Management System (DBMS-11) to store the input information (called the "Information Data Base") on the RPO4 disk. Listings with diagnostics and cross-reference indexes are also produced at this time. The information in the Information Data Base can also be queried from an operator request at the keyboard using the DEC utility program DBQ.

This Information Data Base is not, however, in an efficient format for real time accessing. Current typical spacecraft software systems must decommutate and limit check real-time telemetry blocks with up to 400 telemetry parameters at rates up to 4 blocks per second. Thus, it could require as many as 1600 disk accesses per second

to individually access decommutation and limit check information out of the Information Data Base for each telemetry parameter. This figure far exceeds the 125 possible accesses per second off of even the high speed disk. Similarly, the use of the Information Data Base for display and command processing would burden the software system with too many disk accesses. Therefore, software tasks exist in the Data Base subsystem to generate files to be stored in core and on disk in formats for efficient real-time access. This form of the Data Base is called the "Operation Data Base".

As illustrated in figure 3, the telemetry, command, display, and STOL modules which operate in real time access the efficient files of the Operation Data Base to obtain the required parameters for their Data Base-driven modules.

Data Base Implementation

Westinghouse selected DBMS-11 because it was a field-proven CODASYL-standard Data Base Management System for the PDP 11/70 computer. It also completely satisfied the GSFC requirements. The FORTRAN Data Manipulation Language (DML) was selected over the COBOL version of DML because the FORTRAN IV PLUS language was being utilized in the rest of the software system.

Figure 4 presents an overview of the Data Base Subsystem of the GSFC baseline software system. This subsystem executes in an off-line mode and performs the following functions:

- (1) The Data Base source lines are input as 80-character ASCII lines and stored as next files on the RPO4 disk. A separate Raw Source text file is set up for each of the six major components of the Data Base as follows:
  - (a) DSCFIL.RDB - spacecraft discrete/digital telemetry records
  - (b) ANAFIL.RDB - spacecraft analog telemetry records
  - (c) COMFIL.RDB - spacecraft command records
  - (d) EQUFIL.RDB - equation results, specifications, and constants records
  - (e) PAGFIL.RDB - CRT and Line Printer display format records

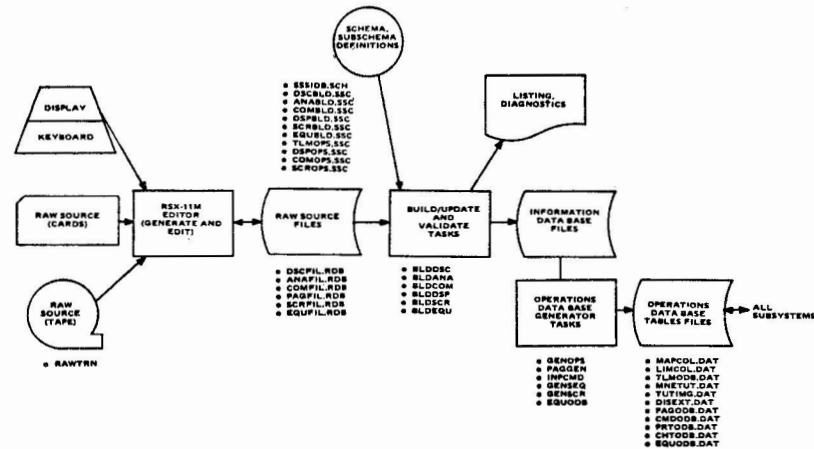


Figure 4. Data Base Subsystem Overview

(f) SCRFIL.RDB - strip chart pen recorder display format records

Figure 5 summarizes the data items of the source input and presents examples of each. The formats of the data items are the same for each spacecraft application with the exception of some command items labeled "spacecraft-unique". The format of these data items is dependent on the individual characteristics of each spacecraft's command decoding equipment and the manner in which the Project personnel choose to express these characteristics.

If the input source is cards, or a magnetic tape generated on a PDP 11/70 computer, these files are set up by using DEC's PIP utility. If the input source is CRT key-ins, these files are set up by using DEC's EDITOR utility. If the input source is a magnetic tape generated on another computer, these files are set up by a special RAWTRN module which reads the tape and generates the required files

- (2) The six Build/Update tasks are activated by the Data Base Administrator each time he wishes to generate the Information Data Base from the six spacecraft Raw Source files. These tasks utilize the Schema Definition to define the exact format of the various areas, records, and sets comprising the Information Data Base. These tasks utilize the Subschema Definitions to define the access criteria for each Data Base area. These Schema/Subschema Definition Files had been previously generated utilizing the DBMS-11 Data Definition Language (DDL). Validation of the Information Data Base is performed during the Build/Update process. Listings with diagnostics are also generated. A Data Base area is generated via the FORTRAN DML language for each of the six components, as well as a seventh area for the generation of a cross reference concordance listing of all Data Base entries.

Figure 6 presents a graphic representation of the Information Data Base illustrating its seven areas, twenty-five record types, and fourteen set relationships.

(3) There are six Operations Data Base Generation tasks which are activated by the Data Base Administrator to produce the eleven files which constitute the Operations Data Base. These tasks utilize the FORTRAN DML language to retrieve the required information from the Information Data Base and generate the efficient files for subsequent use in

real time. The efficiency is derived from the fact that the files are organized to provide all the information needed to process telemetry, command or display requests in convenient contiguous 256-word blocks for rapid access with block I/O statements (QIO) in FORTRAN. Also, the data within these files is in a form convenient for direct FORTRAN processing.

#### Operations Data Base Utilization

Ideally one would prefer to have all the

#### DATA ITEM

#### EXAMPLES

- |  |   |   |
|--|---|---|
| (1) Discrete/Digital Telemetry Measurand Record    | <ul style="list-style-type: none"> <li>o Mnemonic Designation</li> <li>o Location(s) in Telemetry Stream</li> <li>o Number of Bits in measurand</li> <li>o Raw Value-Descriptor Display Conversions</li> </ul>  | XMTRA (Transmitter A's status)<br>S017120-1, E120000-1/2<br>1<br>0/ON, 1/OFF  |
| (2) Analog Telemetry Measurand Record              | <ul style="list-style-type: none"> <li>o Mnemonic Designation</li> <li>o Location(s) in Telemetry Stream</li> <li>o Engineering Units for Display</li> <li>o Delta Limit (check for spikes)</li> <li>o High/Low Limit Pair(s)</li> <li>o Raw Value-Engineering Value Conversions</li> </ul> | BAT1VOLT (Battery 1 voltage)<br>S209117, etc.<br>V (for "volts")<br>5<br>4.94 (low) to 5.46 (high)<br>130/4.79, 140/4.94, 171/5.46, etc.  |
| (3) Spacecraft Command Record                      | <ul style="list-style-type: none"> <li>o Mnemonic Designation</li> <li>o Command Type</li> <li>o Bit Structure</li> <li>o Discrete Verifier/Expected Value</li> <li>o Critical Command Indicator</li> <li>o Critical Command Message</li> </ul>   | BAT1DISC (Battery 1 disconnect)<br>P (for "pulse" - spacecraft-unique)<br>12/103/164 (spacecraft-unique)<br>BATDSCN/0 (Battery 1 status)<br>YES<br>BAT1DSC MUST BE APPROVED BY etc. |
| (4) Equation Constant Definition Record            | <ul style="list-style-type: none"> <li>o Mnemonic Designation</li> <li>o Constant Type/Display Units</li> <li>o Constant Value</li> </ul>   | DELTAX<br>5/RD (Floating Point/Radians)<br>.0078  |
| (5) Equation Result Definition Record              | <ul style="list-style-type: none"> <li>o Mnemonic Designation</li> <li>o Result Type/Units/Notation/# of Dec. Places</li> </ul>   | DISCUR1 (Battery 1's discharge current)<br>5/AH/N/3 (Floating Point/Amp-Hours)  |
| (6) Equation Specification Record                  | <ul style="list-style-type: none"> <li>o Mnemonic Designation</li> <li>o Equation Outputs</li> <li>o Equation Inputs</li> </ul>   | EQPROP (Equation Name)<br>PROPRESA/R, PROPRESB/R<br>TNKATEMP/I, TNKBTEMP/I, etc.  |
| (7) CRT/Snapshot Page Display Record               | <ul style="list-style-type: none"> <li>o Mnemonic Designation</li> <li>o Position/Parameter/Display Format</li> </ul>   | POWER<br>1-1/BAT1VOLT/E, 2-1/BAT2VOLT/E, etc.   |
| (8) Sequential Printout (Histogram) Display Record | <ul style="list-style-type: none"> <li>o Mnemonic Designation</li> <li>o Position/Parameter/Display Format</li> </ul>   | THERMAL<br>1/BAT1TEMP/E, 2/BAT2TEMP/E, etc.   |
| (9) Strip Chart Display Record                     | <ul style="list-style-type: none"> <li>o Mnemonic Designation</li> <li>o Pen Number/Parameter/Scale Factor/Offset</li> </ul>  | SCRBAT1<br>01/BAT1VOLT/, 02/BAT1TEMP/   |

Figure 5. Summary of Data Base Inputs

81-4292-6

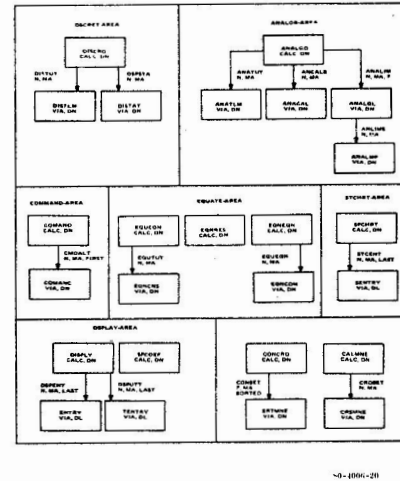


Figure 6. Graphic Representation of IDB

Operations Data Base files in main memory for rapid access by all the real time software modules. However, the total size of all of these files precludes this. It was decided to store two frequently-used files in main memory. These are copied into a 12 Kbyte resident partition as part of each software system's bootstrap procedure. All tasks using these files link to them via FORTRAN COMMON statements and the Task Builder's COMMON option. Consequently, these tasks access the data directly as FORTRAN arrays.

Also, to speed up access of the telemetry extraction information stored in a 3 Mbyte telemetry mapping file and reduce the number of RPO4 disk accesses, the particular 524 Kbyte portion of this file needed during a particular real time support requirement is copied onto the RS04 high-speed disk during system initialization. The remainder of the Operations Data Base files are kept on the RPO4 for access when needed.

Software tasks requiring access to an Operation Data Base file on the RS04 or RPO4 disk, use the "CALL QIO" statements in FORTRAN to read in or write out the required number of 256-word blocks

from the specified file. The data in these blocks had already been converted to the required format during the Operations Data Base Generation process. This format includes byte, integer, floating point, double precision, and ASCII text strings - whatever is necessary to reduce the processing time in the real time modules accessing the Data Base.

Figure 7 illustrates a simplified block diagram of the Telemetry, Command, and Display subsystems of all spacecraft software systems developed from the baseline software system. The ten Operations Data Base files are utilized as follows:

- (a) MAPCOL.DAT - contains information used by the Telemetry User Table Generator module to decommutate the telemetry bit stream into discrete and analog raw values for storage in a global Telemetry Users Table. MAPCOL.DAT is the file which is partially stored on the RS04 disk.
- (b) TURING.DAT - contains an image of the Telemetry User Table. This file is copied into main memory and includes slots for all discrete and analog telemetry values as well as for all equation results and constants. This file is used by most of the modules in each spacecraft software system.
- (c) DISEXT.DAT - contains information about the location and size of all discrete telemetry values in the Telemetry User Table. This file is also copied into main memory and is also used by most modules in each spacecraft software system.
- (d) LIMCOL.DAT - contains analog limits used by the Limit Check module to determine if the analog telemetry values are within their expected operating ranges.
- (e) TLMODB.DAT - contains descriptive information on all discrete and analog telemetry values and all equation values. This file contains conversion parameters for analog values and text equivalents for discrete values. This file is used to build temporary (wild card) display formats by the Format module and to obtain descriptive information for alarm messages by the Limit Check module.

- (f) MNETUT.DAT - contains a correspondence table that relates a telemetry or equation mnemonic to its location in the Telemetry Users Table in main memory. This is used to build wild card displays and to access a telemetry or equation value from an operator or automatic Procedure request within the STOL Processor Module.
- (g) CMDODB.DAT - contains command-related information for every command mnemonic defined on the Data Base. This information includes the command descriptor, the actual bits to be transmitted to the spacecraft, and the telemetry values to be checked to verify that the commands were executed on board the spacecraft. This file is used by the CMD module.
- (h) EQUODB.DAT - contains a list of inputs

and outputs for each equation processor module to obtain data from, and store data into the Telemetry User Table.

- (i) PAGODB.DAT, PRTODB.DAT - contain display formats for CRT displays and line printer displays respectively. These formats include the telemetry and equation parameters to be displayed and any text information for display along with the location on the display for each. These displays may be requested by operator or automatic Procedure request and the requests are processed by the PAGE and PRINT modules.
- (j) CHTODB.DAT - contains display formats for the strip chart pen recorders. These formats include the raw telemetry values to be strip charted. These displays may be requested by operator or automatic Procedure request of the CHART module.

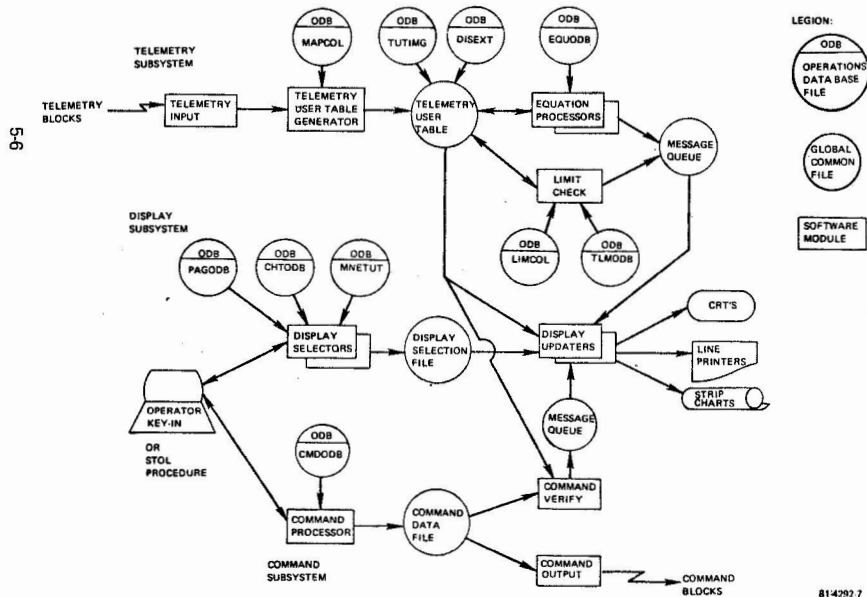


Figure 7. Software System Block Diagram

#### Sizing Information

This paragraph presents the sizes of the raw source input, Information Data Base, and Operations Data Base for a representative spacecraft software system developed using the baseline software system. The system chosen is the one developed to support the launch, experiment activation, and flight operations of the High Energy Astronomy Observatory (HEAO-3) spacecraft. The HEAO-3 Data Base raw source input contains the following number of record occurrences for each of the record types:

Discrete/Digital Telemetry Measurements	973
Analog Telemetry Measurements	510
Spacecraft Commands	1446
Equation Constants	133
Equation Results	127
Equation Specifications	15
CRT/Snapshot Page Display	78
Sequential Printout Displays	14
Strip Chart Displays	24

The sizes of the RP04 disk files generated from this input to produce first the Raw Source text file, then the Information Data Base and, finally, the Operations Data Base for HEAO-3 are as follows:

Raw Source Text File	0.84 Mbyte
Information Data Base Files	6.88 Mbyte
Operations Data Base Files	3.68 Mbyte

As stated previously, during actual real-time operations a 12 Kbyte portion of the Operations Data Base is loaded in resident main memory, and a 524 Kbyte portion of the Operations Data Base is loaded on the high speed RS04 disk for more rapid access.

#### Comments on DBMS-11

The following comments represent some of our thoughts and experiences with DBMS-11 during this effort.

- (1) The DBMS-11 training course and documentation provided by DEC provided us with the information necessary for a timely implementation of Data Base subsystem into the baseline software system.
- (2) Although we were one of the first users of the FORTRAN version of the DML language, we experienced no errors in the DML software. Furthermore, we experienced no errors in any other portion of DBMS-11.
- (3) Because of a lack of good report generator capability in DBMS-11, it was necessary for us to write our own listing programs. The integration of DEC's DATATRIEVE package into DBMS-11 would overcome this deficiency.
- (4) Based on our experiences with DBMS-11, we would not hesitate to use it again when a data base management system is required on a PDP 11 computer.

#### ACKNOWLEDGEMENTS

The author wishes to acknowledge Mr. David Carey of Westinghouse who was one of the principal designers of the Data Base philosophy for the baseline software system and who then developed the major portion of the Data Base subsystem. The author also wishes to acknowledge Mr. Jerold Hahn of NASA's Goddard Space Flight Center who was the Technical Officer on this program and provided suggestions and support throughout.





# UNIVERSITY of PENNSYLVANIA

SCHOOL OF MEDICINE

17 June 1981

DEPARTMENT OF RADIATION THERAPY

Computer Facility

Robert F. Curley, Director  
Robert E. Wallace

Hospital of the University of Pennsylvania  
3400 Spruce Street  
Philadelphia, Pennsylvania 19104  
(215) 662-3083

Dear TECO User,

The TECO SIG has faded away! There has been increasing inactivity for too long. This may be our last shot at resuscitation.

There will be no place in DECUS for TECO users to gather unless YOU volunteer to write articles, organize and deliver sessions at symposia, edit a newsletter, fix bugs or any of the many other chores that make a fruitful Users' Group work.

Contribute at the national level! Positions immediately available:

TECO User Group Manager  
PDP-8 TECO Coordinator  
PDP-11 TECO Coordinator  
PDP-10 TECO Coordinator  
Newsletter Organizer

Write to me; your ideas, offers to work, newsletter contributions, suggestions.

Robert F. Curley  
P.O. Box 322  
Flourtown, Pa. 19031

TECO is at a crossroads, please help steer it down the right road.

Sincerely,

A handwritten signature in cursive script that reads "Robert F. Curley".

Robert F. Curley  
Executive Steering Committee  
Data Management Services SIG

RFC/dlp



DIGITAL EQUIPMENT COMPUTER USERS SOCIETY  
ONE IRON WAY, MR2-3/E55  
MARLBORO, MASSACHUSETTS 01752

BULK RATE  
U.S. POSTAGE  
PAID  
PERMIT NO. 129  
NORTHBORO, MA  
01532

INSTALLATION

**MOVING OR REPLACING A DELEGATE?**

Please notify us immediately to guarantee continuing receipt of DECUS literature. Allow up to six weeks for change to take effect.

- Change of Address
- Delegate Replacement

DECUS Membership No.: \_\_\_\_\_

Name: \_\_\_\_\_

Company: \_\_\_\_\_

Address: \_\_\_\_\_  
\_\_\_\_\_

State/Country: \_\_\_\_\_

Zip/Postal Code: \_\_\_\_\_

Mail to: DECUS - ATT: Membership  
One Iron Way, MR2-3  
Marlboro, Massachusetts 01752 USA

Affix mailing label here. If label is not available, print old address here. Include name of installation, company, university, etc.