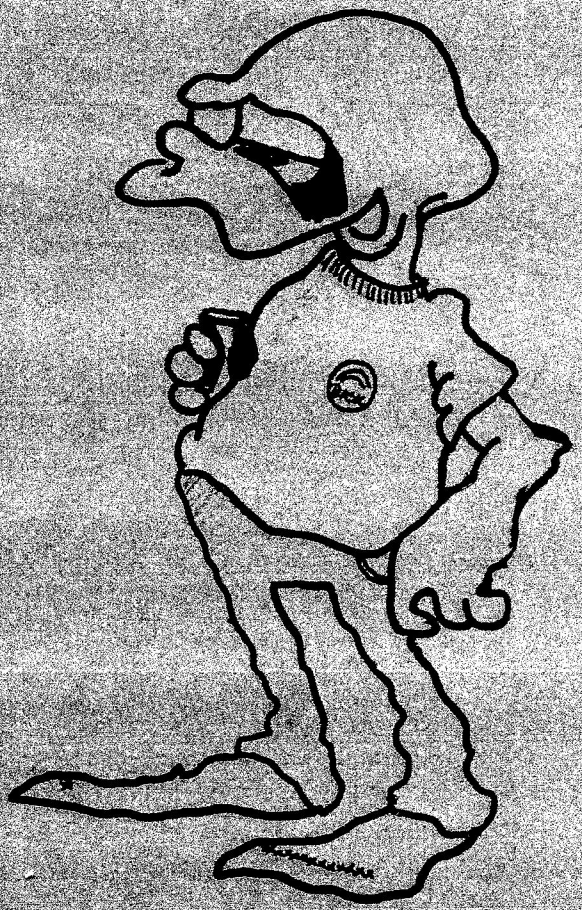
BASIC SIG

OCTOBER 19

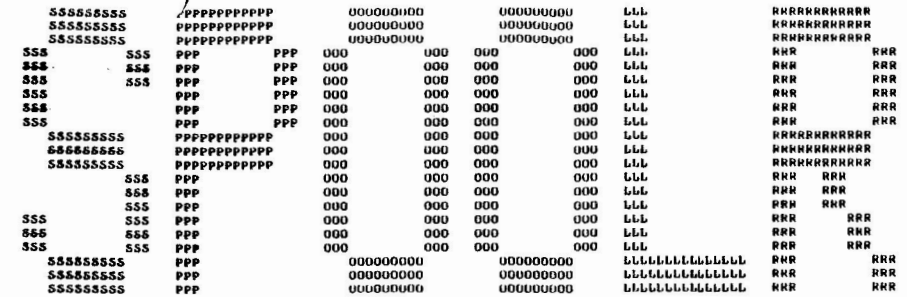
GREAT T-SHIRT

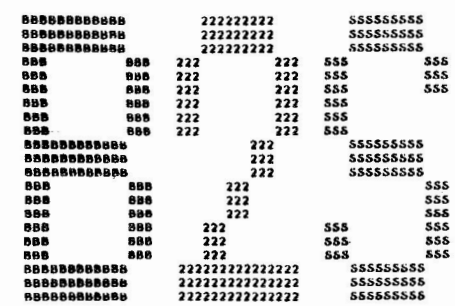


GIVE-AWAY

Printed in the U.S.A.

```
!               FLAGS%      - BIT FLAGS AS FOLLOWS:
!
!                                  1% - FTN CARRIAGE CONTROL (QUE/TYP:FTN)
!                                  2% - RESTART (QUE/RE)
!                                  4% - DELETE (QUE/DE)
!                                  8% - BINARY (QUE/BI)
!                                 16% - END (QUE/END)
!                                 32% - NO HEADER (QUE/NH)
!
!                               THE FLAG VALUES MAY BE COMBINED.
!
!       EXIT:   RESULT%     - RETURNED ZERO IF OPERATION OK ELSE RETURNED
!                             AS THE ERR VALUE OF THE ERROR WHICH OCCURED
!
!       COMMON: NONE.
!
!       MAPS:   NONE.
!
!       CALLS:  NONE.
!END-DOCUMENTATION


10000   ON ERROR GO TO 32000                            ! TRAP ERRORS

\       FS$ = SYS( CHR$(6%) + CHR$(-10%) + FILE.NAME$ ) ! SCAN FILE NAME

!               NOW PICK UP QUEUE INFORMATION

\       Q$ = EDIT$( QUEUE$, 2%+4%+32% )                 ! DROP BLANKS, ET

\       Q$ = 'LP' IF Q$ = ''                            ! DEFAULT TO LP

\       Q.NAME$ = LEFT( Q$, 2% )                        ! GET LP, BA
\       GO TO 10100 IF LEN(Q.NAME$) <> 2%               ! IF BAD
\       Q.UNIT% = -1%                                   ! NO UNIT

\       Q$ = RIGHT( Q$, 3% )                            ! GET REST
\       GO TO 10200 IF Q$ = '' OR Q$ = ':'              ! IF NO UNIT

\       QU$ = LEFT( Q$, 1% )                            ! GET UNIT
\       GO TO 10100 IF QU$ < '0' OR QU$ > '7'           ! IF INVALID

\       Q.UNIT% = VAL( QU$ )                            ! GET UNIT NUM
\       GO TO 10200

10100   !       INVALID UNIT NUMBER, SET RESULT% TO BAD DEVICE (6)

\       RESULT% = 6%                                    ! RETURN ERROR
\       GO TO 32767

10200   !       SET UP THE SPOOL CALL AND DO IT

        SP.CALL$ = CHR$(6%) + CHR$(-28%) + CHR$(0) + CHR$(0)
                 + MID( FS$, 5%, 8% )                   ! PPN, FILE
                 + Q.NAME$                              ! QUEUE NAME

\       SP.CALL$ = SP.CALL$ + CHR$(Q.UNIT%) + CHR$(-1%) ! IF UNIT
                   IF Q.UNIT% >= 0%
\       SP.CALL$ = SP.CALL$ + CHR$(0%) + CHR$(0)        ! IF NO UNIT
                   IF Q.UNIT% < 0%

\       SP.CALL$ = SP.CALL$ + CHR$(0) + CHR$(0)         ! UNUSED
                 + CVT%$( SWAP%(FLAGS%) )               ! FLAGS
                 + CHR$(0) + CHR$(0)                    ! UNUSED
                 + MID( FS$, 23%, 4% )                  ! FILE DEVNN:
                 + STRING$( 4%, 0% )                    ! UNUSED

\       R$ = SYS( SP.CALL$ )                            ! SPOOL FILE

\       RESULT% = 0%                                    ! NO ERRORS
\       GO TO 32767

32000   RESULT% = ERR                                   ! SAVE ERROR
\       RESUME 32767

32767   SUBEND
```





Example of how to call Subprogram to Send a file to the Spooler to print at the System Line Printer. Set Queue$ to 'LP:' or system logical for lineprinter.

```
20000!*************************************************************&
!                                                                  &
!              SEND A FILE TO THE SYSTEM SPOOLER                   &
!                                                                  &
!*************************************************************&
!                                                                  &
\       RESULT% = 0%                                               &
!                                                                  &
\       CALL SPOOL ( OUT.FILE.NAME$, QUEUE$, FLAG$%, RESULT% )    &
!                                                                  &
\       IF RESULT% <> 0%                !SOME KIND OF ERROR        &
         THEN ERL.% = 32000%            !SET ERROR LINE            &
           \     GO TO 30000            !ERROR ROUTINE             &

20900   RETURN
```

new
CPM-133

Vol. 33 Search & Rescue Programs

Version: April 1984

Author: Various

Submitted By: Digital Equipment Corporation

Operating System: CP/M-80

Source Language: BASIC, BASIC-E, CBASIC

Memory Required: 64KB-128KB

The programs on this diskette are associated with SAR (Search and Rescue) planning. Given below are descriptions of some of the programs on the diskette:

ELTPOD	- Calculates and prints tables of aerial search ELT probability of detection as a function of terrain, altitude and track spacing.
ADDPOD	- Calculates and prints a table that allows the user to combine two visual pod's or two ELT pod's.
RJCASP	- SAR resource allocation program that uses sophisticated algorithms to assist the mission coordinator in placing his search forces in the optimum search areas.
CASPPLU	- Expanded version of RJCASP which includes an extensive visual search analysis routine.
CAESAR	- Sophisticated log keeping program for air operations.
BITCASP	- Enhancement of RJCASP which uses a tablet digitizer to transfer map information for POS analysis.
CASPGAME	- Training game which challenges the user to find an unknown target on the map by using POD's.
CAPSERCH	- Game that teaches fundamentals of proper utilization of various search capabilities.

This package was developed on a Z80 chip. It was not developed on a Digital Equipment Corporation personal computer. In some cases, the source code might make specific calls to the hardware which would require changes to the sources.

There are no quarantees that this software will run "AS IS" across the Rainbow, the DECmate II (with CP/M option), or the Professional-300 (with CP/M option), series of computers.

Documentation on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Keywords: Simulation, CP/M - Utilities
Operating System Index: CP/M-80

August 6, 1984

new
CPM-137

Vol. 37 CBASIC2 Programs

Version: April 1984

Author: Various

Submitted By: Digital Equipment Corporation

Operating System: CP/M-80

Source Language: BASIC

Memory Required: 64KB-128KB

The following is a brief description of some of the programs to be found on this diskette:

MATH PROBLEM GENERATOR SYSTEM - Can be used to tutor students in mathematics problems. It also has metric problems and generates a grade report file. Requires CBASIC2 to compile and run.

CRAPS - Very elaborate craps playing program written in CBASIC2.

JRNL - Ledger-type program to keep track of expenses and income. Written in BASIC-E. Presently set up for business usage, could be modified for individual needs.

PASSWORD - Program to change keywords in your BASIC interpreter. Unclear as to which BASIC's will execute this program.

This package was developed on a Z80 chip. It was not developed on a Digital Equipment Corporation personal computer. In some cases, the source code might make specific calls to the hardware which would require changes to the sources.

There are no guarantees that this software will run "AS IS" across the Rainbow, the DECmate II (with CP/M option), or the Professional-300 (with CP/M option), series of computers.

Source code only is included. Documentation on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Keywords: Educational,
Software Collections,
Mathematical
Operating System Index:
CP/M-80

August 6, 1984

new
CPM-139

Vol. 39 Music Programs

Version: April 1984

Author: Various

Submitted By: Digital Equipment Corporation

Operating System: CP/M-80

Source Language: BASIC/E, ASSEMBLY

Memory Required: 64KB-128KB

This diskette contains three major sets of music playing programs

MUSIC contains a three voice music program with the ability to load and save programs in CP/M files. This program takes music entered in a hexadecimal notation and compiles (scores) it into a series of instructions which wiggle the interrupt enable line fast enough to produce three voices. The source code is kept in memory in a line numbered format (a la BASIC) and edited with a built-in editor. Has a range of about 2 1/2 octaves, supports stacatto, long and short articulation, dotted notes and whole thru sixty-fourth notes. Source code is not available, but good documentation is included. Requires simple hardware, such as an amplifier and a speaker.

MUSIC4 is a program to play music. Seems to have the ability to produce four different voices. Requires a Z80 CPU and an 8 bit digital-to-analog converter. BASIC-E is needed to run some of the support software for generating song files.

MUSPAT is an overlay for a three voice music program written by Software Technology. Program requires a Processor Technology SOL computer for loading and saving of the music files. Source was not supplied.

This package was developed on a Z80 chip. It was not developed on a Digital Equipment Corporation personal computer. In some cases, the source code might make specific calls to the hardware which would require changes to the sources.

There are no quarantees that this software will run "AS IS" across the Rainbow, the DECmate II (with CP/M option), or the Professional-300 (with CP/M option), series of computers.

Complete sources are not included. Documentation on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Keywords: CP/M Music
Operating System Index:
CP/M-80

August 6, 1984

revision
11-708

RSTS System Utilities from the University of Tennessee

Version: June 1984

Author: Harry Flowers, University of Tennessee, Memphis, TN

Operating System: RSTS/E V7.1, 7.2, 8.Ø

Source Language: BASIC-PLUS

Memory Required: Varies (MONITR requires 17KW)

Other Software Required: SPLRUN patch requires large system spooling package, MONITR assumes OPSER running, but not necessary.

This package contains four separate utilities. Following is a brief description of each:

SPLRUN patch - We have patched SPLRUN to count pages. If you use the big spooling package and would like to keep track of the pages you print, this patch works fairly well. One known problem with it is it comes up one short if there is no job burst page printed...shouldn't be too hard to fix if it matters to you. On our system, SPLRUN sends a message to our online accounting program. For purposes of this patch, the information is being sent to the OSC through OPSER.

SPLRUN.PAT patch to SPLRUN (BASIC-PLUS)
SPLPAT.CMD ATPK command file for patch and compile (read before you execute)

System monitoring package (MONITR) - The package performs similar to a combination of DYNPRI and KBMON. If you have both of these programs running, you can save a job slot with MONITR. MONITR also has some additional features which can be very useful. Package consists of:

MONITR.BAS	monitoring program (runs detached)
KBOARD.BAS	keyboard report and maintenance
KBOARD.MTR	keyboard data file (created by KBOARD.BAS)
SNDMTR.BAS	message sender to MONITR
MONITR.DOC	documentation for this package
MONITR.CMD	command file for CUSP compiling

For further details, see MONITR.DOC documentation.

Billboard - public notes system which acts as a billboard. Care was taken to write this program without cursor control so that it may be run from any terminal. See source code for further details. You will probably wish to modify the help screen, as it contains references to UTCHS.

BILBRD.BAS billboard program
BILBRD.BAS must be compiled with the privileged bit set in the protection code, as [232].

Password changers - programs which will change the passwords to accounts. You are prompted for the old password, then the new password twice to make sure it's right.

PASWRD.BAS changes password to any user account
PRIVP .BAS changes the password to all privileged accounts
PASWRD.BAS must be compiled with the privileged bit set in the protection code, as [232].
PRIVP.BAS should NOT have the privileged bit set, to force running it from a privileged account.

Changes and Improvements: MONITR: System Monitoring Package has been changed to monitor all logged-out users, and keyboard usage time has been correctd.

Restrictions: SPLRUN gives one less page than was printed if no job burst page was specified. For MONITR a maximum of 128 configured keyboards, assumes OPSER online, but will run without OPSER. See documentation for further details.

For SPLRUN only patch is included. Documentation on magnetic media.

Media (Service Charge Code): 600' Magtape (MA)

Format: DOS-11

Keywords: RSTS/E - Utilities,
RSTS/E - System Management
Operating System Index:
RSTS/E

August 6, 1984

new
CPM-143

Vol. 43 Osborne CBASIC2 Accounts Payable and Accounts Receivable

Version: April 1984

Author: Various

Submitted By: Digital Equipment Corporation

Operating System: CP/M-80

Source Language: CBASIC2

Memory Required: 64KB-128KB

This extensive package of programs will keep track of current payables and receivables, as well as ageings and payments against both. There are programs for transaction entry, ledger, sort, and printing for both accounts payables and receivables. There are also programs for a check calculator, register and writer, among others.

These programs require CBASIC2 to compile. There are known bugs that you will have to work out. Changes will be required in some programs if other than a Hazeltine terminal is used.
This package was developed on a Z80 chip. It was not developed on a Digital Equipment Corporation personal computer. In some cases, the source code might make specific calls to the hardware which would require changes to the sources.

There are no guarantees that this software will run "AS IS" across the Rainbow, the Decmate II (with CP/M option), or the Professional-300 (with CP/M option), series of computers.

Associated Documentation: Osborne/McGraw-Hill "Accounts Payable & Accounts Receivable (CBASIC)" by Lon Poole, Mary Borchers, Martin McNiff, Robert Thomson.

Source code only is included. Documentation is not included on the magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Keywords: Business Applications
Operating System Index: CP/M-80

August 20, 1984

new
CPM-144

Vol. 44 Osborne/McGraw-Hill General Ledger Programs

Version: April 1984

Author: Various

Submitted By: Digital Equipment Corporation

Operating System: CP/M-80

Source Language: CBASIC2

Memory Required: 64KB-128KB

This is the general ledger series of programs published by Osborne/McGraw-Hill. It compiles and runs on a Hazeltine terminal, but will require modification if you use a different terminal. Familiarity with CBASIC2 programming is helpful.

BUDGET1 operates on 1-20 expense records, which are input by the user. Each of these records may be subdivided into several (max of 4) categories. The program checks the addition used in allocating these categories. These are then output onto disk in records that contain the day of the transaction, payee, and category/amount for 1-4 categories. The record length is 64 characters, so that random access to the files is possible. The program will add new expenses to the disk immediately after any current expenses already logged.

LEDGER1 gives, by category, the date of payment, payee, and individual amount. It gives a total for each category for the month; it ends with a total of the month's payments. You may do more than one month at a time, and you can specify output to disk, terminal, or printer.

ANNTOT1 reads the monthly categorized totals from disk, and outputs them in ledger form for optional printing on hardcopy or output to the terminal.

BUDGETCH makes changes in erroneous entries made with BUDGET1.

There are no guarantees that this software will run "AS IS" across the Rainbow, the DECmate II (with CP/M option), or the Professional-300 (with CP/M option), series of computers.

Associated Documentation: Osborne/McGraw-Hill "General Ledger CBASIC" by Lon Poole, Mary Borchers, Martin McNiff, Robert Thomson

Source code only is included. Documentation is not included on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Keywords: Business Applications
Operating System Index: CP/M-80
August 20, 1984

new
CPM-145

Vol. 45 Osborne/McGraw-Hill Payroll with Cost Accounting

Version: April 1984

Author: Various

Submitted By: Digital Equipment Corporation

Operating System: CP/M-80

Source Language: CBASIC2

Memory Required: 64KB-128KB

Other Software Required: DECUS No. CPM-144

This diskette contains Osborne/McGraw-Hill Payroll with Cost Accounting programs. It consists of programs to do general information and employee master file maintenance, federal and state tax file maintenance, payroll transaction entry, file sorting, payroll deduction and check register, journal file, quarterly report, printing of W-2 forms and insurance reporting, as well as other transactions.

There are some routines that are needed for this program but they have not been included. They can be found in the book "Payroll With Cost Accounting" published by Osborne/McGraw-Hill.This package was developed on a Z80 chip. It was not developed on a Digital Equipment Corporation personal computer. In some cases, the source code might make specific calls to the hardware which would require changes to the sources.

There are no quarantees that this software will run "AS IS" across the Rainbow, the DECmate II (with CP/M option), or the Professional-300 (with CP/M option), series of computers.

Associated Documentation: "Payroll With Cost Accounting" by Osborne/McGraw-Hill.

Source code only is included. Documentation is not included on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Keywords: Business Applications
Operating System Index: CP/M-80

August 20, 1984

new
PRO-122

General Purpose Database Package

Version: V1.4, February 1984

Author: R. J. Welldon

Operating System: P/OS V1.7

Source Language: PRO/BASIC

Memory Required: 64KB

Other Software Required: PRO-BASIC, PROSE

Special Hardware Required: LA50 or LA100 printer

GPDB is a simple menu-driven database package. The database files must reside on diskette. The data fields can be either character, numeric or date. The output is only sorted by one field.

Restrictions: The database to reside on a diskette.

Documentation is not included on the media and must be ordered separately.

Media (Service Charge Code): Write-Up (AA), 5 1/4" Floppy Diskette (JA)

Format: FILES

Keywords: Data Base Management
Operating System Index: P/OS

August 20, 1984

new
11-742

LEAP: Library Electronic Acquisition Program

Version: V3, February 1984

Author: Saskatchewan Technical Institute, Moose Jaw S, Saskatchewan, Canada

Operating System: RSTS/E V8.0

Source Language: BASIC-PLUS2

This system aids the Library in keeping track of the acquisitions function. It looks after a book from the ordering to the receiving. The correct fund is updated and funds reporting is done on request.

LEAP consists of nine sub-programs; each sub-program performs a function in the system.

1. Adding a record.
2. Printing purchase orders.
3. Receiving or cancelling an order.
4. Searching for a record.
5. Modifying a record.
6. Updating the fund account.
7. Fund account reporting.
8. Updating the vendor file.
9. Acquisitions listing.

Documention on magnetic media.

Media (Service Charge Code): Manual (EC), 600' Magtape (MA)

Format: DOS-11

Keywords: Data Acquisition, Business Applications, Library (Book)
Operating System Index: RSTS/E

August 20, 1984

new
CPM-121

Vol. 21 Microsoft BASIC Programs

Version: April 1984

Author: Various

Submitted By: Digital Equipment Corporation

Operating System: CP/M-80

Source Language: BASIC

Memory Required: 64KB-128KB

The following is a brief description of some of the programs to be found on the diskette:

Microsoft BASIC games such as ACYDUCY, APPOLO, BANNER, BLKJK, CHASE, CHESS, DIAMONDS, FURS, HORSE, LANDER, MANDALA, MAZE, ROCKET, RUSSIAN, SNOOPY, STRTRK, TACOS, TAXMAN, TRAP, WUMP.

This package was developed on a Z80 chip. It was not developed on a Digital Equipment Corporation personal computer. In some cases, the source code might make specific calls to the hardware which would require changes to the sources.

There are no guarantees that this software will run "AS IS" across the Rainbow, the DECmate II (with CP/M option), or the Professional-300 (with CP/M option), series of computers.

Source code only is included. Documentation on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Keywords: Games
Operating System Index:
CP/M-80

July 23, 1984

new
CPM-122

Vol. 22 Monstrous Startrek Games

Version: April 1984

Author: Various

Submitted By: Digital Equipment Corporation

Operating System: CP/M-80

Source Language: BASIC

Memory Required: 64KB-128KB

Well documented games for people with lots of memory and lots of time:

BIGTREK.ASC	STARTREK.TXT trimmed to load under TDL disk BASIC with 64K memory.
BIGTREK.BAS	Compacted version of BIGTREK.ASC for much faster load under TDL disk BASIC.
STARTREK.TXT	Starting point of BIGTREK. Purported to work with MITS 8K BASIC.
STRTRK/2.ASC	Another Startrek program.
TREKINFO.DOC	Detailed rules and features of STARTREK.TXT and BIGTREK.
TREKMOD.ASC	BIGTREK trimmed some more and able to load with microsoft disk BASIC.

This package was developed on a Z80 chip. It was not developed on a Digital Equipment Corporation personal computer. In some cases, the source code might make specific calls to the hardware which would require changes to the sources.

There are no quarantees that this software will run "AS IS" across the Rainbow, the DECmate II (with CP/M option), or the Professional-300 (with CP/M option), series of computers.

Source code only is included. Documentation on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Keywords: Games
Operating System Index:
CP/M-80

July 23, 1984

new
CPM-126

Vol. 26 Microsoft BASIC & FORTRAN Games & Utilities

Version: April 1984

Author: Various

Submitted By: Digital Equipment Corporation

Operating System: CP/M-80

Source Language: BASIC, FORTRAN

Memory Required: 64KB-128KB

The following is a brief description of some of the programs to be found on the diskette:

Microsoft BASIC games such as BACCRRT, BASEBALL, BIRTHDAY, BLACKJACK, CHESS, CLOUD-9, CRAPS, CRAZY-8, GALAXY, SWARMS, AND WEATHER. It also contains executable code for the FORTRAN game OTHELLO.

This package was developed on a Z80 chip. It was not developed on a Digital Equipment Corporation personal computer. In some cases, the source code might make specific calls to the hardware which would require changes to the sources.

There are no guarantees that this software will run "AS IS" across the Rainbow, the DECmate II (with CP/M option), or the Professional-300 (with CP/M option), series of computers.

Source code only is included. Documentation on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Keywords: Games, CP/M-80 - Utilities
Operating System Index:
CP/M-80

July 23, 1984

new
CPM-127

Vol. 27 Microsoft BASIC Games

Version: April 1984

Author: Various

Submitted By: Digital Equipment Corporation

Operating System: CP/M-80

Source Language: BASIC

Memory Required: 64KB-128KB

The following is a brief description of some of the programs found on the diskette:

Microsoft BASIC games such as ANTONYMS, DISSAMBR, FOOTBALL, GOLF, GREEKRTS, HANGMN-1, HIDESEEK, MASTERMD, MAZE, MEMBRAIN, ROULETTE, SNOOPY, STARTREK, and MEGATREK.

This package was developed on a Z80 chip. It was not developed on a Digital Equipment Corporation personal computer. In some cases, the source code might make specific calls to the hardware which would require changes to the sources.

There are no guarantees that this software will run "AS IS" across the Rainbow, the DECmate II (with CP/M option), or the Professional-300 (with CP/M option), series of computers.

Source code only is included. Documentation on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Keywords: Games
Operating System Index:
CPM-80

July 23, 1984

new
CPM-128

Vol. 28 BASIC-E Utilities, Games, Database, ALGOL-Like Language

Version: April 1984

Author: Various

Submitted By: Digital Equipment Corporation

Operating System: CP/M-80

Source Language: BASIC, ALGOL-M

Memory Required: 64KB-128KB

This package contains a BASIC-E maillist utility, a Database system, and an ALGOL-like-language called ALGOL-M.

The main theme of the database system is to provide a common set of programs that help the user create, modify, and access data files for a variety of needs. In this way, the system can be better tailored for a particular sitation, and yet the different parts of it can also be much more compatible. The sequence of operation is normally to first run the DBSETUP program to define the name and structure of the file, then run the DBENTRY program to make the initial entries, and last run the DBQUERY program to access the files.

ALGOL-M was modeled after ALGOL-60. This was done intentionally in order to provide a language which would be best suited to the needs of applications programmers using microcomputer systems. However, the basic structure of ALGOL-M is similar enough to ALGOL-60 to allow simple conversion of programs from one language to the other.

This package was developed on a Z80 chip. It was not developed on a Digital Equipment Corporation personal computer. In some cases, the source code might make specific calls to the hardware which would require changes to the sources.

There are no quarantees that this software will run "AS IS" across the Rainbow, the DECmate II (with CP/M option), or the Professional-300 (with CP/M option), series of computers.

Sources may or may not be included. Documentation on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Keywords: Games, CP/M - 80
Utilities, Programming Languages
Operating System Index: CP/M - 80

July 23, 1984

new
CPM-130

Vol. 30 - BASIC-E Version 1.4 Floating Point Part Two

Version: April 1984

Author: Various

Submitted By: Digital Equipment Corporation

Operating System: CP/M-80

Source Language: BASIC-E

Memory Required: 64KB-128KB

Other Software Required: DECUS Part No. CPM-129 to obtain rest of files for the Floating Point Package.

The following is a brief description of the programs to be found on the diskette:

CATALOG.30	Contents of CP/M group volume 30.
BASCOM.LIT	Literal equates.
BASIC.PLM	Version 1.4 BASIC-E compiler modified for CP/M.
BASPAR.PLM	Parser module.
BASSYN.PLM	Symbol table and code generator module.
BASIC.COM	Executable compile module.
RUN.PLM	Run module.
RUN.COM	Executable run module.
BUILD.PLM	Invoked when run called to build internal tables from int file.

This package was developed on a Z80 chip. It was not developed on a Digital Equipment Corporation personal computer. In some cases, the source code might make specific calls to the hardware which would require changes to the sources.

There are no guarantees that this software will run "AS IS" across the Rainbow, the DECmate II (with CP/M option), or the Professional-300 (with CP/M option), series of computers.

Note: This package contains some of the files for the Floating Point Conversion Package. Order DECUS Part No. CPM-129 to obtain the rest of the Floating Point Package.

Sources may or may not be included. Documentation on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Keywords: Floating Point Routines
Operating System Index:
CP/M-80

July 23, 1984

revision
11-654

Student Terminal Management System

Version: V2, April 1984

Author: William B. Leng, Southern Connecticut University,
New Haven, CT

Operating System: RSTS/E

Source Language: BASIC-PLUS2

Memory Required: 17KW

Other Software Required: Uses RSTS/E SYS calls

Special Hardware Required: Uses VT100 series cursor commands

A terminal management system to automatically handle scheduling of student terminals on a first-come, first served (one-hour-on, one-hour-off) basis. Provisions are made to send messages to all STUDENT terminals and to ascertain who's on them and what they are running. Terminal usage can be formatted for printout to teachers or usage percentage can be plotted on a VT100 with hard-copy backup to use for justification of resource changes. The available terminal list can be dynamically changed at any time.

Documentation on magnetic media.

Media (Service Charge Code): Write-Up (AA), Floppy Diskette (KA)
Format: RT-11, 600' Magtape (MA)
Format: DOS-11

Keywords: Tools - Applications Development, Terminal Resource Management, Utility - System Management, Terminal Management
Operating System Index: RSTS/E

July 23, 1984

new
11-739

GRADES: Course Management Program

Version: V1, April 1984

Author: Ronald S. Daniel, California Polytechnic University, Pomona, CA

Operating System: RSTS/E V7.2 and 8.Ø

Source Language: BASIC-PLUS

Memory Required: 128KB

Special Hardware Required: Printer desirable but not required.

The software package described in this document is a grade keeping system developed at California State Polytechnic University, Pomona. The package is written in the BASIC PLUS language running under the RSTS operating system in a PDP-11/7Ø minicomputer.

This document is strictly written for an experienced programmer to understand the foundation of the system in order to be able to modify it, enhance it, or eliminate features from it.

The software is formed by different modules which perform a function on a database stored in the computer memory. The names of these modules and a short description of the function that they perform are:

INIT	Initializes the master file of one of three different courses that the software can support.
ENTER	Is the "main scheduler" program. This module is the one that "calls the different specialized modules to perform their function on the database based on the user's response to the main menu.
NAMES	This program is run at the beginning of the quarter to initialize the database for any of the three supported courses.
SCORE	This program is acessed whenever scores for an evaluation are going to be entered.
PRINT	This program will produce a hard copy printout of a course database.
PRINT1	This version of PRINT will output the report to a file in the computer memory for later queing to the fast speed printer.
CORET	This module is used to correct any kind of data that there is found to be wrong in the students' records.

Note: This program has been tested only on RSTS/E versions 7.2 and 8.Ø.

Restrictions: Limited to three sections (courses) of enrollment of 300 each or less.

Media (Service Charge Code): Write-up and Listing (DB), 600' Magtape (MA)

Format: DOS-11

Keywords: Business Applications, Educational
Operating System Index: RSTS/E

July 23, 1984

new
VAX-9Ø

DECAL to DAL Translator

Version: V1.Ø, May 1984

Author: Digital Equipment Corporation

Operating System: VAX/VMS V3

Source Language: VAX-11 BASIC

The DECAL to DAL translator (DECALDAL) is used to migrate lessons created with DECAL (DIGITAL'S RSTS/E Authoring Language) to DAL which is a component of the Courseware Authoring System (C.A.S.) running under VAX/VMS.The tape contains the following files:

ØREADME.TXT	This file you are reading.
DECALDAL.ABS	An abstract describing the functionality of the software.
DECALDAL.BAS	The source code for the translator which is written in VAX-11 BASIC.
DECALDAL.COM	The command procedure for using RUNOFF to create a print file for the manual.
DECALDAL.MEM	RUNOFF output of the DECAL-DAL Translator Definition manual ready for printing.
DECALDAL.OBJ	Object module for DECALDAL.
DECALDAL.RNO	RUNOFF input file for the DECAL-DAL Translation Definition manual.

To obtain an executable version of the translator, you must follow these steps:

1. Use BACKUP to copy all of the files from the tape to disk.

2. Link the .OBJ to generate a DECALDAL.EXE file.

3. Run DECALDAL using the instructions in the DECAL/DAL Translator Definition document.

Documentation on magnetic media.

Media (Service Charge Code): 6ØØ' Magtape (MA)

Format: VAX/ANSI (Blocked at 2Ø48)

Keywords: Educational, CAI - Computer Assisted Instruction
Operating System Index:
VAX/VMS

July 23, 1984

new
CPM-112

Vol. 12 Pilot Interpreters Patched for CP/M

Version: April 1984

Author: Various

Submitted By: Digital Equipment Corporation

Operating System: CP/M-80

Source Language: BASIC

Memory Required: 64KB-128KB

The following is a brief description of the programs to be found on the diskette:

CATALOG.12	Contents of CP/M vol 12
GOLDI.PLT	Source program for ZPILOT
HIPILOT.PLT	Source program for ZPILOT
PILOT.ASM	Patched version of #7.2. See PILOT.DOC
PILOT.COM	See PILOT.DOC
PILOT.DOC	Description for CP/M implementation of PILOT
PILOT.TST	Source program for PILOT. Type "PILOT PILOT.TST" to run
PMON.ASM	CP/M interface used on PILOT.COM
WEIRD	WIERD.PLT re-coded for the PILOT.COM syntax for comparison purposes only.
WEIRD.PLT	Source program for ZPILOT
ZPILOT.COM	Object of ZPOLOT.Z80
ZOUKIT.Z80	ZILOG mnemonic source of a pilot interpreter. No doc!!

This package was developed on a Z80 chip. It was not developed on a Digital Equipment Corporation personal computer. In some cases, the source code might make specific calls to the hardware which would require changes to the sources.

There are no quarantees that this software will run "AS IS" across the Rainbow, the DECmate II (with CP/M option), or the Professional-300 (with CP/M option), series of computers.

Source may or may not be included. Documentation on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Keywords: Interpreters
Operating System Index:
CP/M-80

June 11, 1984

Vol. 13 BASIC-E/CBASIC and Microsoft BASIC Programs & Games

Version: April 1984

Author: Various

Submitted By: Digital Equipment Corporation

Operating System: CP/M-80

Source Language: BASIC

Memory Required: 64KB-128KB

The following is a brief description of the programs to be found on the diskette:

CATALOG.13	Contents of CP/M volume 13.
15/PUZ.ASC	Program in Microsoft BASIC.
1500.ASC	Program in Microsoft BASIC.
23MATCH.BAS	Program in BASIC-E/CBASIC.
BAGELS.BAS	Program in BASIC-E/CBASIC.
BIORYME.ASC	Program in Microsoft BASIC.
BLACKJACK.BAS	Program in BASIC-E/CBASIC.
BULLSEYE.BAS	Program in BASIC-E/CBASIC.
CHECKERS.BAS	Program in BASIC-E/CBASIC.
CHIEF.BAS	Program in BASIC-E/CBASIC.
CONVERT.BAS	Program in BASIC-E/CBASIC.
DICE.BAS	Program in BASIC-E/CBASIC.
KINGDOM.BAS	Program in BASIC-E/CBASIC.
NFL.BAS	Program in BASIC-E/CBASIC.
ROCKET.BAS	Program in BASIC-E/CBASIC.
RUSSIAN.BAS	Program in BASIC-E/CBASIC.
SWARMS.BAS	Program in BASIC-E/CBASIC.
SWARMS2.ASC	Program in Microsoft BASIC.
TRAP.BAS	Program in BASIC-E/CBASIC.
WUMPAS.BAS	Program in BASIC-E/CBASIC.
ZOSO.2	Review of Programs

This package was developed on a Z80 chip. It was not developed on a Digital Equipment Corporation personal computer. In some cases, the source code might make specific calls to the hardware which would require changes to the sources.

There are no guarantees that this software will run "AS IS" across the Rainbow, the DECmate II (with CP/M option), or the Professional-300 (with CP/M option), series of computers.

Sources may or may not be included. Documentation on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Keywords: Games,
Software Collections
Operating System Index:
CP/M-80
June 11, 1984

new
CPM-114

Vol. 14 Various CP/M Utilities

Version: April 1984

Author: Various

Submitted By: Digital Equipment Corporation

Operating System: CP/M-80

Source Language: BASIC, ASSEMBLY

Memory Required: 64KB-128KB

The following is a brief description of the programs to be found on the diskette:

CATALOG.14	Contents of CP/M volume 14.
ARTICLE	Text for SECTEST.BAS.
BDC-DRVR	Driver to convert ASCII to that funny reverse BCD used by IBM 2740 terminals.
DLDHI.ASM	High portion of DLOAD - See DOC.
DLOAD.ASM	Patches to put MITS 3.2 8K BASIC up on CP/M with CSAVE/CLOAD to disk.
DLOAD.DOC	Implementation notes for DLOAD and comments on MOVE, LIST32 and the use of MITS 8K BASIC vers 3.2 after patching.
DUMP.COM	Running version of DUMP below, supplied as
DUMP.ASM	coded for TDL assembler (8080 OK).
DUMP.MAC	Fantastic disk viewer program. Can address files, CP/M groups or sectors directly, uses standard console output, and displays in DDT DUMP form with HEX and ASCII simultaneously.
LIST32.ASM	Program to recover ASCII file from internal storage MITS 3.2 form - see DLOAD.DOC.
MOVE.ASM	A PIP to transfer files without the problems of [CTL Z]'S in files with non-COM type names, such as BASIC-/CBASIC int files - see DLOAD.DOC.
PUT.ASM	Used to load a file at any memory address, and optionally start to run it. Useful for poking odd drivers and monitors into memory for those with no front panel.
REL1.ASM	Instructive re-construction of RELOC/CPM program. See RELHOW.DOC. Note that relocation table is not included.
REL256.COM	RELOC for INTEGER K-100H system.
REL512.COM	RELOC for INTEGER K-200H system.
REL768.COM	RELOC for INTEGER K-300H system.
RELHOW.DOC	Implementation notes for relocating CP/M version 1.3 at 100H increments instead of 400H as supplied.
SECTEST.BAS	CBASIC program for testing context comprehension and recall. Uses the ARTICLE file.

SECTEST.DOC	Instructions for SECTEST.BAS.
SEDY.ASM	Disk peeking program.
SEDY.COM	Compilation of SEDY.ASM written for TDL assembler (8080 OK).

This package was developed on a Z80 chip. It was not developed on a Digital Equipment Corporation personal computer. In some cases, the source code might make specific calls to the hardware which would require changes to the sources.

There are no guarantees that this software will run "AS IS" across the Rainbow, the DECmate II (with CP/M option), or the Professional-300 (with CP/M option), series of computers.

Sources may or may not be included. Documentation on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Keywords: CP/M-80 - Utilities
Operating System Index: CP/M-80

June 11, 1984

new
CPM-116

Vol. 16 Assemblers, other Utilities and FOCAL

Version: April 1984

Author: Various

Submitted By: Digital Equipment Corporation

Operating System: CP/M-80

Source Language: BASIC

Memory Required: 64KB-128KB

The following is a brief description of the programs to be found on the diskette:

CATALOG.16	Contents of CP/M group vol. 16
VOLUME16.DOC	Comments on certain programs
ASMX.COM	Assembler which recognizes Z-80 OPS See VOL.DOC [Careful: with correct syntax (ASMX FILENAME.AAA) This does work. With faulty syntax the program takes revenge on the disk directory.] Runs OK on 8080
COPYDSK.ASM	Disk copy program. See VOL.DOC
COPYDSK.MAC	As 16.2 for TDL Assembler
CPMUTIL.ASM	CP/M subroutines useful generally and employed as part of Z80ASM 16.17
EDIT.COM	Intel-like editor. Does LFB and -B much faster than ED.COM. See VOL.DOC
EDUCATOR.ASM	8080 instruction set tutor from byte of July 1976
FOCAL.ASM	FOCAL language interpreter. See VOL.DOC
MACASM.COM	MACRO assembler. See VOL.DOC
MOVDOWN.ASM	Program to load file which operates below 100H
SEEK.ASM	Set disk track from front panel during alignment
SPAT1.ASM	Rewrite of 1.29 to generalize console from original VDM dependancy
TASMIO.DOC	DOC for TASMIO patch to put TDL tape assembler up on CP/M
TASMIO.HEX	See TASMIO.DOC
TASMIO.MAC	See TASMIO.DOC
TEST1A.ASM	Successful test for Z80ASM 16.17
TEST2.ASM	Unsuccessful test for Z80ASM 16.17
Z80ASM.COM	ZILOG mnemonic assembler. Runs on 8080. See Z80DOC.DOC 16.18
Z80DOC.DOC	DOC for 16.17
Z80MAIN.ASM	See 16.17
Z80OPCDS.ASM	See 16.17
Z80SUBS.ASM	See 16.17

This package was developed on a Z80 chip. It was not developed on a Digital Equipment Corporation personal computer. In some cases, the source code might make specific calls to the hardware which would require changes to the sources.

There are no quarantees that this software will run "AS IS" across the Rainbow, the DECmate II (with CP/M option), or the Professional-300 (with CP/M option), series of computers.

Source may or may not be included. Documentation on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Keywords: CP/M-80 - Utilities, Assemblers
Operating System Index: CP/M-80

June 25, 1984

new
CPM-120

Vol. 20 - Basic-E/CBASIC Programs, Pictures

Version: April 1984

Author: Various

Submitted By: Digital Equipment Corporation

Operating System: CP/M-80

Source Language: BASIC-E/CBASIC

Memory Required: 64KB-128KB

The following is a brief description of the programs on the diskette:

CATALOG.20	Contents of CP/M Group Vol. 20
ZOSO.20	Our talented and modest reviewer pays tribute to a worthwhile set of submissions
BLACKJAC.BAS	Program in BASIC-E/CBASIC. See ZOSO.20
CIVILW.BAS	Program in BASIC-E/CBASIC. See ZOSO.20
FOOTBALL.BAS	Program in BASIC-E/CBASIC. See ZOSO.20
GOLF.BAS	Program in BASIC-E/CBASIC. See ZOSO.20
GUNNER.BAS	Program in BASIC-E/CBASIC. See ZOSO.20
LUNAR1.BAS	Program in BASIC-E/CBASIC. See ZOSO.20
PINUP.PIC	Picture
PINUP1.PIC	Picture
POKER.BAS	Program in BASIC-E/CBASIC. See ZOSO.20
SNOOPY.PIC	Picture
STARTREK.BAS	Program in BASIC-E/CBASIC. See ZOSO.20
STMASTER.BAS	Program in BASIC-E/CBASIC. See ZOSO.20
STMASTER.DOC	See ZOSO.20
STRTRK/1.BAS	Program in BASIC-E/CBASIC. See ZOSO.20
TREKINST	See ZOSO.20
TWEETY.PIC	Picture

This package was developed on a Z80 chip. It was not developed on a Digital Equipment Corporation personal computer. In some cases, the source code might make specific calls to the hardware which would require changes to the sources.

There are no guarantees that this software will run "AS IS" across the Rainbow, the DECmate II (with CP/M option), or the Professional-300 (with CP/M option), series of computers.

Sources may or may not be included. Documentation on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Keywords: CP/M - BASIC,
Operating System Index: CP/M-80

June 25, 1984

new
CPM-251

GRADING: An Elementary School Teacher's Gradebook for the Rainbow Series

Version: V1.0, March 1984

Author: Robert A. Malseed, Albuquerque, NM

Operating System: CP/M-86/80 V1.0 (1.1)

Source Language: M-BASIC-86

Memory Required: 64KB

Other Software Required: M-BASIC-86 Interpreter

Special Hardware Required: Serial Printer - LA50 recommended

GRADING consists of two Microsoft MBASIC programs for the Rainbow 100 which are used to initialize and to maintain an elementary school teacher's gradebook. Up to 14 grades in seven subjects may be recorded for 24 students.

The Gradebook is initialized with the GRADINIT program and then the GRADING program is used to maintain the gradebook. The GRADING program is a menu driven program. Student names may be added, deleted, or changed. Test dates and scores may be entered and changed if necessary. Test scores must be numerical in the range of 0 to 100. An "A" for "absent" may be entered if a student did not take a particular test.

Summaries of student grades and class and student averages for each subject may be printed. In addition, a class report card with student overall averages may be printed.

Documentation on magnetic media.

Media (Service Charge Code): Write-Up and Listing (DA),
5 1/4 " Floppy Diskette (JA)

Keywords: CPM-86 -
Educational
Operating System Index:
CP/M-86

June 25, 1984

new
11-730

Steinmetz High School Card Reader Monitor

Version: V8.0, December 1983

Author: Francis W. Harsey, Jr., Steinmetz High School, Chicago, IL

Operating System: RSTS/E V7.0 or V8.0 required

Source Language: BASIC-PLUS

Memory Required: 16KB

Other Software Required: RSTS/E Batch processing and spooling package (OPSER, BATCH, QUEMAN, SPOOL, QUE), PSEUDO keyboard

Special Hardware Required: Card reader, line printer

The card reader monitor package is a package consisting of three files:

READER.BAS is a program which runs detached and monitors the card reader at 10 second intervals. In addition, READER.BAS includes 4 modes of processing: queueing of control files, creation of files anywhere on the system disk, listing of cards on the line printer, and a special feature which allows the user to disable the reader program for 60 seconds to allow the user to use the card reader for his/her own applications (I.E. 'PIP', 'RUN CR:', ETC.).

SENDER.BAS allows the system manager to communicate with the reader program while it is online to either shutdown the reader program, suspend the reader program for a period of time, change the message console, or resume reader operations after a suspend command was issued.

READER.DOC contains complete documentation necessary to use the package.

Restrictions: QUE program must be defined as a CCL command, (i.e. Run $UTILITY and then type: CCL QU-EUE=$QUE.*;PRIV 3000)

Documentation on magnetic media

Media (Service Charge Code): 600' magtape (MA)

Format: DOS-11

Keywords: RSTS/E - Utilities
Operating System Index: RSTS/E

June 25, 1984

new
11-733

PLIBR: A Library Control Program

Version: V2.1, April 1984

Author: Thomas Leih, University of Wisconsin-Parkside,
Kenosha, WI

Operating System: RSTS/E V7.0, 7.2, 8.0

Source Language: BASIC-PLUS

Memory Required: 16KW

PLIBR allows you to combine many (up to 31) small or large files into one library file. You have complete control to list, extract, delete, rename and replace entries in a library and to condense a library. This is especially useful with disks with large cluster sizes and/or many small files.

Restrictions: Files stored in a library loose all attributes and protection code information.

Documentation on magnetic media.

Media (Service Charge Code): Write-Up (AA), 600' Magtape (MA)

Format: DOS-11

Keywords: RSTS/E Libraries,
Operating System Index:
RSTS/E

June 25, 1984

new
11-734

ASCII Driven MENU for RSTS/E

Version: V1.0, April 1984

Author: Kevin Davidson, Rose-Hulman Institute of Technology,
Terre Haute, IN

Operating System: RSTS/E V7.0

Source Language: BASIC-PLUS, BASIC-PLUS2

Memory Required: 128KB

Other Software Required: BASIC-PLUS or BASIC-PLUS II with ECHO control, RSX run-time system, ONLPAT

This is menu system which is driven from text option files. It is based upon the RSX run-time system menu patch where a program is executed upon entry of this modified RSX RTS (called MENU). Each option can be protected by a user specification. Any valid run command of CCL command can be executed from the menu prompt as long as it does not match a menu option. You can specify what line number to enter on and also put data in core common before the chain to another program.

The modification to the RSX run-time system is included.

Restrictions: Set up to run on VT-52 compatable terminals. You have to switch run-time systems to exit menu. On our system this is done through a LOGIN Command File used by the LOGIN Program.

Documentation on magnetic media.

Media (Service Charge Code): 600' Magtape (MA)

Format: DOS-11

Keywords: Tools -
Applications Development, Menu Control
Operating System Index:
RSTS/E

June 25, 1984

new
11-735

RSTS/E File List and Scan Utilities

Version: April 1984

Author: Susan M. Abercrombie, Ventrex Laboratories Inc., Portland, ME

Operating System: RSTS/E V7.2 or 8.0

Source Language: BASIC-11

Two utilities are supplied. SPLIST reformats source files for output by the system spooling package, with page headers identifying the files, and doing page feed to keep basic or bp2 lines together. FLSCAN searches source files for lines containing matches for a specified string. The output may be to the terminal or to a disk file which optionally may be spooled. Included also is a patch file for the RSTS V8 SYSTAT program. With this patch you can do SY/W or SY/O with output limited by job number, keyboard, or account.

Note: Two programs are supplied with complete sources. In addition a patch file is supplied for the RSTS SYSTAT utility.

Restrictions: Will not run on V6C or earlier because wildcard account scan is used.

Documentation on magnetic media.

Media (Service Charge Code): Listing (BA) 600' Magtape (MA)

Format: DOS-11

Keywords: Tools - Applications Development, RSTS/E - Utilities, Spool
Operating System Index: RSTS/E

June 25, 1984

new
VAX-85

Bibliography System

Version: April 1984

Author: Tim Baird, Harding University, Searcy, AR

Submitted By: Stephen Baber, Harding University, Searcy, AR

Operating System: VAX/VMS V3.4

Source Language: VAX-11 BASIC

Memory Required: 290 pages working set

This program uses the index-sequential capabilities of VAX-11 BASIC to set up a cross-reference system for books, periodicals, etc. One application for use is to store the information on all books in a particular field that are owned by departmental faculty as well as the University's library. Inquiries can then be made by subject, author, reference, or title.

Documentation on magnetic media

Media (Service Charge Code): Write-up and Listing (DB), 600' Magtape (MA)

Format: VAX/ANSI (Blocked at 2048)

Keywords: Data Base Management
Operating System Index: VAX/VMS

June 25, 1984

BASIC *MAGIC*

BASIC SIG

```
10      EXTEND
20      DIM ARRAY_FOR_CHAR$(256%)
30      ARRAY_FOR_CHAR$(0%)   = "NUL"
        ARRAY_FOR_CHAR$(1%)   = "SOH"
        ARRAY_FOR_CHAR$(2%)   = "STX"
        ARRAY_FOR_CHAR$(3%)   = "^C "
        ARRAY_FOR_CHAR$(4%)   = "EOT"
        ARRAY_FOR_CHAR$(5%)   = "ENQ"
        ARRAY_FOR_CHAR$(6%)   = "ACK"
        ARRAY_FOR_CHAR$(7%)   = "^G "
        ARRAY_FOR_CHAR$(8%)   = "BS "
        ARRAY_FOR_CHAR$(9%)   = "HT "
        ARRAY_FOR_CHAR$(10%)   = "LF"
        ARRAY_FOR_CHAR$(11%)   = "VT "
        ARRAY_FOR_CHAR$(12%)   = "FF "
        ARRAY_FOR_CHAR$(13%)   = "CR "
        ARRAY_FOR_CHAR$(14%)   = "SO "
        ARRAY_FOR_CHAR$(15%)   = "^O "
        ARRAY_FOR_CHAR$(16%)   = "DLE"
        ARRAY_FOR_CHAR$(17%)   = "^Q "
        ARRAY_FOR_CHAR$(18%)   = "DC2"
        ARRAY_FOR_CHAR$(19%)   = "^S "
        ARRAY_FOR_CHAR$(20%)   = "DC4"
        ARRAY_FOR_CHAR$(21%)   = "NAK"
        ARRAY_FOR_CHAR$(22%)   = "SYN"
        ARRAY_FOR_CHAR$(23%)   = "ETB"
        ARRAY_FOR_CHAR$(24%)   = "CAN"
        ARRAY_FOR_CHAR$(25%)   = "EM "
        ARRAY_FOR_CHAR$(26%)   = "^Z "
        ARRAY_FOR_CHAR$(27%)   = "ESC"
        ARRAY_FOR_CHAR$(28%)   = "FS "
        ARRAY_FOR_CHAR$(29%)   = "GS "
        ARRAY_FOR_CHAR$(30%)   = "RS "
        ARRAY_FOR_CHAR$(31%)   = "US "
        ARRAY_FOR_CHAR$(32%)   = "SP "
        ARRAY_FOR_CHAR$(I%)    = CHR$(I%) FOR I% = 33% TO 126%
        ARRAY_FOR_CHAR$(127%) = "DEL"
        ARRAY_FOR_CHAR$(128 + 0%)   = "NUL"
        ARRAY_FOR_CHAR$(128 + 1%)   = "SOH"
        ARRAY_FOR_CHAR$(128 + 2%)   = "STX"
        ARRAY_FOR_CHAR$(128 + 3%)   = "^C "
        ARRAY_FOR_CHAR$(128 + 4%)   = "EOT"
        ARRAY_FOR_CHAR$(128 + 5%)   = "ENQ"
        ARRAY_FOR_CHAR$(128 + 6%)   = "ACK"
        ARRAY_FOR_CHAR$(128 + 7%)   = "^G "
        ARRAY_FOR_CHAR$(128 + 8%)   = "BS "
        ARRAY_FOR_CHAR$(128 + 9%)   = "HT "
        ARRAY_FOR_CHAR$(128 + 10%)   = "LF"
        ARRAY_FOR_CHAR$(128 + 11%)   = "VT "
        ARRAY_FOR_CHAR$(128 + 12%)   = "FF "
```

```
        ARRAY_FOR_CHAR$(128 + 13%)  = 'CR '
        ARRAY_FOR_CHAR$(128 + 14%)  = 'SO '
        ARRAY_FOR_CHAR$(128 + 15%)  = '^O '
        ARRAY_FOR_CHAR$(128 + 16%)  = 'DLE'
        ARRAY_FOR_CHAR$(128 + 17%)  = '^Q '
        ARRAY_FOR_CHAR$(128 + 18%)  = 'DC2'
        ARRAY_FOR_CHAR$(128 + 19%)  = '^S '
        ARRAY_FOR_CHAR$(128 + 20%)  = 'DC4'
        ARRAY_FOR_CHAR$(128 + 21%)  = 'NAK'
        ARRAY_FOR_CHAR$(128 + 22%)  = 'SYN'
        ARRAY_FOR_CHAR$(128 + 23%)  = 'ETB'
        ARRAY_FOR_CHAR$(128 + 24%)  = 'CAN'
        ARRAY_FOR_CHAR$(128 + 25%)  = 'EM '
        ARRAY_FOR_CHAR$(128 + 26%)  = '^Z '
        ARRAY_FOR_CHAR$(128 + 27%)  = 'ESC'
        ARRAY_FOR_CHAR$(128 + 28%)  = 'FS '
        ARRAY_FOR_CHAR$(128 + 29%)  = 'GS '
        ARRAY_FOR_CHAR$(128 + 30%)  = 'RS '
        ARRAY_FOR_CHAR$(128 + 31%)  = 'US '
        ARRAY_FOR_CHAR$(128 + 32%)  = 'SP '
        ARRAY_FOR_CHAR$(128 + I%)   = CHR$(I%) FOR I% = 33% TO 126%
        ARRAY_FOR_CHAR$(128 + 127%) = 'DEL'
1000    ! PROGRAM TO PRINT DISK FILES IN ASCII BINARY FORMAT
1010    ! SELECTIVE BY BLOCK

1020    ON ERROR GOTO 32000

1030    PRINT 'ENTER FILE NAME        [<CR> TO EXIT]';
1040    INPUT LINE F$
1050    F$ = LEFT(F$,LEN(F$)-2%)
        GOTO 32000 IF F$ = ''
1060    DIM TED$ (16)

1070    INPUT 'OUTPUT DEVICE                    <KB:>';P$
1080    IF P$ = '' THEN P$ = 'KB:'

1090    INPUT 'MODIFY MODE                      <NO> ';Y$
1100    IF Y$ <> 'YES' GOTO 1150

1110    INPUT 'INPUT BLOCK TO MODIFY [<CR> TO EXIT] ';B%
        GOTO 32000 IF B% = 0%

1120    INPUT 'DO YOU WANT IT PRINTED         <YES> ';L$
1130    S% = B%
        E% = B%
1140    GOTO 1160

1150    INPUT 'STARTING BLOCK NUMBER [<CR> TO EXIT] ';S%
        GOTO 32000 IF S% = 0%
        INPUT 'ENDING BLOCK NUMBER                  ';E%
```

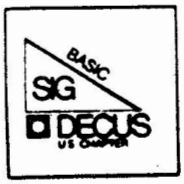
BASIC SIG

```
1160    OPEN P$ FOR OUTPUT AS FILE 10%
1170    OPEN F$ FOR INPUT AS FILE 1%, MODE 4096% IF Y$ <> 'YES'
1180    OPEN F$ FOR INPUT AS FILE 1%, MODE 1%    IF Y$ =  'YES'

1190    GET #1%, RECORD S%
1200    IF L$ = 'NO' THEN 1300%
1210    FOR X% = 0% TO 31%
1220            FOR X1%= 0% TO 15%
1230                    FIELD #1%,(X%*16+X1%) AS TED$, 1% AS TED$ (X1%)
1240            NEXT X1%
1250            S1% = X%*16%+X1%
1260            PRINT #10% USING '##### ###  ',S%,S1%;
1270            PRINT #10% USING ' ###',X1%; FOR X1% = S1%-15% TO S1%
                PRINT #10%
                PRINT #10%, TAB(11%);
                PRINT #10% USING ' ###',ASCII(TED$ (X1%)); FOR X1% = 0% TO 14%
                PRINT #10% USING ' ###',ASCII(TED$ (15))
                PRINT #10%, TAB(14%);
                PRINT #10%  USING '\  \',ARRAY_FOR_CHAR$(ASCII(TED$ (X1%))); &
                        FOR X1% = 0% TO 14%
                PRINT #10%  USING '\  \',ARRAY_FOR_CHAR$(ASCII(TED$ (15%)))
                PRINT #10%
1290    NEXT X%
1300    IF Y$ = 'YES' GOTO 1340
1310    S% = S% + 1%
1320    IF S% <= E% GOTO 1190
1330    IF Y$ <> 'YES' GOTO 1490
1340    INPUT 'ENTER BYTE NUMBER TO CHANGE   (<513)';A%
1350    IF A% < 0% OR A% > 512% GOTO 1340
1360    FIELD #1%, (A%-1%) AS TEDDY$, 1% AS BEAR$
1370    BEAR% = ASCII(BEAR$)
1380    PRINT 'THE VALUE YOU ARE ABOUT TO CHANGE = ';BEAR%;
        PRINT 'CHARACTER = ';ARRAY_FOR_CHAR$(BEAR%)
1390    INPUT 'ENTER NEW VALUE                (<257)';V%
1400    IF V% < 0% OR V% > 256% GOTO 1390
1405    PRINT 'CHARACTER ';ARRAY_FOR_CHAR$(BEAR%);' WILL BE CHANGED TO CHARACTER '; &
              ARRAY_FOR_CHAR$(V%)
1410    INPUT 'DO YOU WANT IT CHANGED ?       <NO> ';Y1$
1420    IF Y1$ <> 'YES' GOTO 1470
1430    LSET BEAR$ = CHR$(V%)
1440    PUT #1%, RECORD S%
1450    PRINT 'BLOCK ';S%;' MODIFIED '
1460    GOTO 1090
1470    PRINT 'NO MODIFICATION MADE'
1480    GOTO 1110
1490    CLOSE 1%
1500    CLOSE 10%
1510    GOTO 1150
```

```
32000   CLOSE I% FOR I% = 1% TO 12%
32767   END
```

BASIC SIG

```
1       !>>PROGRAM    : COMBP2.BAS
        !  DATE       : 24-AUG-80
        !  PROGRAMMER : TED A. BEAR
        !  DESCRIPTION:

4       EXTEND
25       P9$ = 'COMBP2'
         P8$ = 'BASIC PLUS-2 COMPILER BATCH '
         ON ERROR GOTO 32000
         Z$=SYS(CHR$(6%)+CHR$(-7%))
         N$=''
         L$=CHR$(13%)+CHR$(10%)
         KB%=12%
         EOS%=-1%
         EOL%=-2%                          !>>PROGRAM NAME
                                           !  TRAP ERRORS
                                           !  ENABLE CC TRAP
                                           !  N$=NULL STRING
                                           !  L$=<CR><LF>
                                           !  KB%=KEYBOARD CHANNEL
                                           !  EOS%=CLEAR TO EOS
                                           !  EOL%=CLEAR TO EOL
30      OPEN 'KB:' AS FILE KB%
         X3%=1%
         V%=95%
                                           !>>OPEN KEYBOARD FOR INPUT
                                           !  CURSOR POSITIONING
                                           !  PAINT CHARACTER=UNDERSCORE
90      PRINT #KB%, L$; FNV$(1%,1%,N$,EOS%,V%);
               FNV$(1%,1%,P8$+'  '+DATE$(D9%)+'  '+TIME$(0%),0%,V%)
                                           !>>PRINT THE BANNER
100     RETURN IF BACK_FROM_QUEUE%

1000    INPUT 'PROGRAM NAME <WITHOUT .BAS> ';PROGRAM.NAME$
        INPUT 'BATCH QUE                   ';BQ$
        GOTO  32767% IF PROGRAM.NAME$ = '/E'
        OPEN PROGRAM.NAME$ + '.CTL' FOR OUTPUT AS FILE 1%

1010    PRINT #1% '$JOB/CCL/NOLIMIT'
        PRINT #1% 'BP2'
        PRINT #1% 'OLD ' + PROGRAM.NAME$
        PRINT #1% 'COMP'
        PRINT #1% 'EXIT'
        PRINT #1% 'TKB ' + PROGRAM.NAME$ + '=' + PROGRAM.NAME$ + ',LB:BP2OTS/LB'
        PRINT #1% '$EOD'
        PRINT #1% '$EOJ'

1020    CLOSE I% FOR I% = 1% TO 12%
```



BASIC SIG

```
2000    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
        !!!              QUE REQUESTS                                      !!!
        !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

2010    G$  = CVT$$(NUM$(G%), -1%) + ";"
        G1$ = CHR$(13%)
        G2$ = CVT%$(2100%)
        BQ$ = "Q BA" + BQ$ + ":="

        ! CHANGE THIS ACCOUNT TO YOURS
                  !
                  !
        !         V

2020     G3$ = "[1,238]COMBP2" + G1$ + G2$ + BQ$ + PROGRAM.NAME$ + ",CTL" + G1$ + G$
         V$ = SYS(CHR$(8%)+G3$)
         CHAIN "$QUE" 31000%
                       !>> QUEUE UP ,CTL

2100    !!!!!! BACK FROM ALL THE QUEUES

2150     BACK_FROM_QUEUE% = -1%

2200    GOSUB 25%
        GOTO 4950%

4950    PRINT $KB%, FNV$(1%,1%,N$,EOS%,V%); L$;
                FNV$(20%,2%,"END OF "+PB$,0%,V%);
        GOTO 32760                          !>>EXIT POINT
                                            !  SUBROUTINES FOLLOW

10000   DEF FNV$(X%,Y%,Z$,Z1%,V%)
        IF X%<1%!X3%=0% THEN
                FNV$=Z$
        ELSE
                Z9$=CHR$(13%)+CHR$(155%)+"Y"+CHR$(159%+Y%)
                Z8$=Z9$+CHR$(159%+X%)
                Z8$=Z8$+CHR$(155%)+"J" IF Z1%=-1%
                Z8$=Z8$+CHR$(155%)+"K" IF Z1%=-2%
                Z8$=Z8$+Z$
                Z8$=Z8$+STRING$(Z1%,V%-32%*(V%=0%))+Z9$+CHR$(159%+X%+LEN(Z$)) IF Z1%>0%
                FNV$=Z8$

10010   FNEND
                                !>>FNV$=CONSTRUCT CURSOR POSITIONED STRING
                                !  X%,Y%=SCREEN COORDINATES
                                !  Z$=BASE STRING
                                !  Z1%=LENGTH TO "CLEAR"
                                !  V%="CLEAR" CHARACTER
```


```
32000   !>> ERROR TRAPS

32700   PRINT L$;"?UNFORESEEN ERROR DETECTED IN "; P9$ &
        \ ON ERROR GO TO 0 &
                                        !  ERRORS END HERE
32760   CLOSE Z% FOR Z% = 1% TO 12% &
                                        !  EXIT THRU HERE &
                                        !  CLOSE EVERYTHING
32767   END
```

BASIC SIG

```
9320    !*********************************************************************
        !**     QUE PRINT FILE Z$ AND RETURN TO Z1$ AT LINE Z%              **
        !**     Z1% = NO. COPIES TO PRINT                                   **
        !**     Z2$ = LINE PRINTER                                          **
        !*********************************************************************

        DEF FN_QUE$ (Z$,Z1$,Z%,Z1%,Z2$)

        Z$ = SYS(CHR$ (8%)+Z1$+CHR$ (13%)+CVT%$ (Z%) &
             +"Q '+Z2$+'/PR:64 = '+Z$+'/DE/CO:'+EDIT$ (NUM$ (Z1%),2%))

        CHAIN "$QUE" LINE  31000%

9330            FNEND

10000   !*********************************************************************
        !**     KB: INPUT                                                   **
        !*********************************************************************

        DEF FN_INPUT$ (Z0$)
                FN_INPUT$ = N$
                ERR,OR% = 0%
                B_SLASH% = -1%
                                                !* B_SLASH% IS TURNED OFF
                                                !* UNLESS \ IS ENTERED

10010           PRINT #12%, Z0$;
                INPUT LINE #12%, Z1$
                Z1$ = EDIT$ (Z1$,E9%)
                GOTO 10040 IF Z1$ = "\"
                                                !* PROMPT FOR AND RETURN INPUT

10020           Z% = INSTR(1%,Z1$,CHR$ (8%))
                IF Z% = 0% THEN
                        FN_INPUT$ = Z1$
                ELSE
                        Z1$ = LEFT(Z1$,Z% -2%)+RIGHT(Z1$,Z%+1%)
                        GOTO 10020
                                                !* REMOVE BACKSPACES
10030           B_SLASH% = 0%
                                                !* \ WASN'T ENTERED
10040           FNEND


10090   !*********************************************************************
        !**     PARSE Z0$ INTO FN_SPLIT$ AND 2ND_HALF$                      **
        !*********************************************************************

        DEF FN_SPLIT$ (Z0$,Z1$)
```

```
            Z%,T1% = INSTR(1%,Z0$,Z1$)
            Z% = LEN(Z0$)+1% UNLESS Z%
            FN_SPLIT$ = LEFT(Z0$,Z% -1%)
            2ND_HALF$ = RIGHT(Z0$,Z%+LEN(Z1$))

10110       FNEND

10160   !*********************************************************************
        !**     PARSE COMMON CORE                                           **
        !**     ';' IS THE DELIMITER                                        **
        !*********************************************************************

        DEF FN_COMMOM%(Z$)

            08$,09$ = Z$
            Z1$,Z2$,Z3$ = N$
            Z5$ = ';'

10170       Z1$ = FN_SPLIT$ (SYS(CHR$ (7%)),Z5$)
            FN_COMMOM% = T1%
            IF T1% = 0% THEN
                    Z1$ = N$
            ELSE
                    Z2$ = FN_SPLIT$ (2ND_HALF$,Z5$)
                    Z3$ = FN_SPLIT$ (2ND_HALF$,Z5$)

10180       06$ = Z1$
            09$ = Z2$ IF LEN(Z2$)
            08$ = Z3$ IF LEN(Z3$)
            07$ = EDIT$ (2ND_HALF$,128%)

10190       FNEND

10200   !*********************************************************************
        !**     ROUNDING FUNCTION                                           **
        !*********************************************************************

        DEF FN_ROUND (Z,Z%) = FIX (ABS (Z) * 10. ** Z% +.5)/10. ** Z% * SGN(Z)


10210   !*********************************************************************
        !**     CALCULATES REMAINDER                                        **
        !*********************************************************************

        DEF FN_REMAINDER%(Z1%,Z2%) = Z1% - (Z1%/Z2%) * Z2%

10330   !*********************************************************************
        !**     STRING CENTERER                                             **
        !*********************************************************************
```


```
        DEF FN_CENTER$ (Z$,Z%)

                Z$ = LEFT(Z$,Z%) IF LEN(Z$) > Z%
                Z1% = (Z% - LEN(Z$))/2%

10340           FN_CENTER$ = SPACE$ (Z1%) + Z$ + SPACE$ (Z% - Z1% - LEN(Z$))

10350           FNEND

10360   !*********************************************************************
        !**     RETURN A RSTS DATE                                          **
        !*********************************************************************

        DEF FN_DATE%(Z$)

                ON ERROR GOTO 10405

                Z$ = DATE$ (0%) IF Z$ = "TODAY"
                Z$ = LEFT(Z$,2%) + "-" + MID(Z$,3%,2%) + "-" + RIGHT(Z$,5%) &
                        IF LEN(Z$) = 6% UNLESS INSTR(1%,Z$,"-")
                Z4% = INSTR(1%,Z$,"-")
                Z5% = INSTR(Z4%+1%,Z$,"-")
                ERR,OR% = 1%
                FN_DATE% = -1%

10365           IF Z$ = "00-XXX-00" THEN
                        Z8% = -1%
                        GOTO 10400

10370           GOTO 10410 IF Z4% * Z5% = 0%
                Z6% = VAL (RIGHT (Z$, Z5% + 1%)) - 70%
                GOTO 10410 IF Z6% < 0% OR Z6% > 29%
                Z6% = Z6%*1000%
                IF INSTR(1%,"SOND",EDIT$ (MID(Z$,Z4%+1%,1%),32%)) THEN
                        Z6% = Z6% + 243%
                END IF

10380           IF Z5% - Z4% < 4% THEN
                        Z7% = VAL (LEFT (Z$, Z4% - 1%))
                        GOTO 10410 IF Z7% < 1% OR Z7% > 12%
                        Z$ = MID (Z$, Z4% + 1%, Z5% - Z4%) &
                                + RIGHT (DATE$ ((Z7% - 1%) * 31% + 1%+Z6%),4%)
                        Z6% = Z6% + (Z7% - 1%) * 29%

10390           Z$ = EDIT$ (Z$,34%)
                Z$ = "0" + Z$ IF INSTR(1%,Z$,"-") = 2%
                GOTO 10400 IF Z$ = EDIT$ (DATE$ (Z8%),32%) &
                        FOR Z8% = Z6% + 1% TO Z6% + 366%
                GOTO 10410
```

BASIC SIG

```
10400           ERR,OR% = 0%
                FN_DATE% = Z8%
                GOTO 10410

10405   IF ERL> = 10360% AND ERL< = 10400% THEN RESUME 10410 &
                                        !* FN_DATE% ERROR
10410           FNEND

10420   !************************************************************************
        !**     NON-PRIV WAY OF GETTING TODAY'S JULIAN DATE                    **
        !************************************************************************

        DEF FN_DATE_1% = SWAP%(CVT$%(RIGHT(SYS(CHR$ (6%)+CHR$ (-3%)),11%))) &

10430   !************************************************************************
        !**     NON-PRIV WAY OF GETTING JOB NUMBER                             **
        !************************************************************************

        DEF FN_JOB% = ASCII(SYS(CHR$ (6%)+CHR$ (-3%)))/2% &

10500   !************************************************************************
        !**     FIND NUMBER OF DAYS BETWEEN 2 DATES                            **
        !************************************************************************

        DEF FN_DATE_3%(Z1%,Z2%)

                IF Z2% < Z1% THEN
                        FN_DATE_3% = - FN_DATE_3% (Z2%,Z1%)
                        GOTO 10530
                END IF
                                        !*  Z1% = 1ST DATE
                                        !*  Z2% = 2ND DATE
                                        !*  FN_DATE_3% > 0 IF Z1% < Z2%

10510           ERR,OR%,FN_DATE_3% = 0%
                Z3% = 1970% + Z1%/1000%
                Z4% = 1970% + Z2%/1000%
                Z7% = (Z4% - Z3%) * 365% + FN_REMAINDER% (Z2%,1000%) &
                                         - FN_REMAINDER% (Z1%,1000%)
10520           Z7% = Z7% + 1% &
                        IF EDIT$ (LEFT(DATE$ ((Z5% - 1970%) * 1000% &
                                                +60%),6%),32%) = '29-FEB' &
                        FOR Z5% = Z3% TO Z4% -1% &
                        IF Z3%<Z4%

                FN_DATE_3% = Z7%

10530           FNEND

10540   !************************************************************************
```

```
        !**     DATE ADDER                                                **
        !**************************************************************************

        DEF FN_DATE_2%(Z1%,Z2%)

                FN_DATE_2% = -1%
                ERR.OR% = 1%

10550           GOTO 10600 IF Z1% < 0%
                GOTO 10595 IF Z2% = 0%
                                        !* CHECK PARAMETER VALUE
10560           Z4% = Z1%/1000% -70%
                Z4% = Z4% - Z4%/4% * 4%
                IF Z4% = -3% AND Z2% < 0% THEN
                        Z4% = 0%
                ELSE
                        Z4% = -1% UNLESS Z4% = 0% AND Z2% > 0%

10565           Z5% = Z2%
                Z5% = SGN(Z2%) * (366%+Z4%) IF (SGN(Z2%)*Z5%) > 366%+Z4%
                Z2% = Z2% - Z5%

10570                                   !* Z4% = 0 IF A LEAP YEAR
                                        !* Z2% = AMOUNT TO ADD OR SUBTRACT
                                        !* (ONE YEAR AT A TIME)

10580           Z3% = Z1% + Z5%
                GOTO 10600 IF Z3% > 29365% OR Z3% < 0%
                Z6% = Z3% - Z3%/1000% * 1000%

10585           IF ((Z6% > 366% + Z4%) AND SGN(Z5%) > 0%) &
                                        OR((SGN(Z5%) < 0%) &
                                        AND(Z6% = 0% OR Z6%>366%+Z4%)) THEN
                        Z3% = Z3% + SGN(Z5%)*(634% - Z4%)

                                                !* ADD OR SUBTRACT THE DAYS
                                                !* TEST FOR YEAR OVERFLOW
10590           Z1% = Z3%
                GOTO 10560 IF Z2%               !* RESET START DATE
                                                !* BACK TO ADD OR SUBTRACT
                                                !* SOME MORE
10595           FN_DATE_2% = Z1%
                ERR.OR% = 0%                    !* FN_DATE_2% = RESULTANT DATE
                                                !* INTEGER
                                                !* NO ERRORS WERE MADE

10600           FNEND
```

BASIC SIG

```
11500   !************************************************************************
        !**     POSITION TO (X,Y)                                              **
        !**     CLEAR TO END OF LINE                                           **
        !**     PROMPT WITH Z$                                                 **
        !************************************************************************

        DEF FN_INPUT_3$ (X%,Y%,Z$)

                Z% = FN_SCREEN_1% (X%,Y%,-2%)
                FN_INPUT_3$ = FN_INPUT$ (Z$)
                X% = FN_SCREEN_3% (N$,0%) IF(E3% AND 512%)
                X% = FN_SCREEN_3% (N$,1%) IF(E3% AND 1024%)
                E3% = E3% AND 255%

11505           FNEND

11520   !************************************************************************
        !**     POSITION TO (X,Y)                                              **
        !**     WAIT FOR RESPONSE                                              **
        !**     TEST FOR A LENGTH OF < = Z9%                                   **
        !************************************************************************

        DEF FN_INPUT_4$ (X%,Y%,Z9%)

                Z$ = FN_INPUT_2$ (X%,Y%)
                IF LEN(Z$)>Z9% THEN
                        Z% = FN_SCREEN_3%  ('TOO LONG BY' + NUM$ (LEN(Z$)- Z9%)&
                                        + 'CHARS',1%)
                        GOTO 11520
                END IF

11525           FN_INPUT_4$ = Z$ + N$
                FNEND

11530   !************************************************************************
        !**     FN_SCREEN_1% - POSITION CURSOR ON VT100                        **
        !**     AND OPTIONALLY CLEAR SCREEN                                    **
        !**     Z% =    -1% TO CLEAR TO END OF SCREEN                          **
        !**             -2% TO CLEAR TO END OF LINE                            **
        !************************************************************************

        DEF FN_SCREEN_1% (X%, Y%, Z%)

        PRINT CHR$(155%) + '[' + NUM1$(Y%) + ';' + NUM1$(X%) + 'H'; &
                IF X% + Y% > 0%
        IF Z% = -1% THEN
                PRINT CHR$(155%) + '[0J';
        ELSE
                IF Z% = -2% THEN PRINT CHR$(155%) + '[0K';
11540    FNEND
```

```
11530   !*********************************************************************
        !**     FN_SCREEN_1% - POSITION CURSOR ON VT52                      **
        !**     AND OPTIONALLY CLEAR SCREEN                                 **
        !**     Z% =    -1% TO CLEAR TO END OF SCREEN                       **
        !**             -2% TO CLEAR TO END OF LINE                         **
        !*********************************************************************

        DEF FN_SCREEN_1%(X%,Y%,Z%)
                Z1$ = CHR$ (159%+Y%)+CHR$ (159%+X%)
                PRINT CHR$ (13%);CHR$ (155%);'Y';Z1$;
                Z1$ = CHR$ (155%)
                PRINT Z1$;'J';          IF Z% = -1%
                PRINT Z1$;'K';          IF Z% = -2%
                                                !* 0 = CLEAR TO EOS
                                                !* 2 = CLEAR TO EOL

11540           FNEND


11550   !*********************************************************************
        !**      FN_SCREEN_2% - DISPLAY OUTPUT                              **
        !**              Z%   - 0%       ATTIBUTES OFF                      **
        !**              1%      BOLD OR INCREASED INTENSITY                **
        !**              4%      UNDERSCORE                                 **
        !**              5%      BLINK                                      **
        !**              7%      NEGATIVE (REVERSE) IMAGE                   **
        !**              VT100                                              **
        !*********************************************************************

        DEF FN_SCREEN_2% (Z$, X%, Y%, Z%)

        T% = FN_SCREEN_1% (X%, Y%, 0%)
        PRINT CHR$(155%) + '[' + NUM1$(Z%) + 'm'; IF Z%
        PRINT Z$;
        PRINT CHR$(155%) + '[0m'; IF Z%

11555   FNEND


11550   !*********************************************************************
        !**     POSITION TO (X,Y)                                           **
        !**     AND PRINT Z$;                                               **
        !**     VT52                                                        **
        !*********************************************************************

        DEF FN_SCREEN_2%(X%,Y%,Z$)

                FN_SCREEN_2% = FN_SCREEN_1% (X%,Y%,0%)
                PRINT Z$;
```

```
11555           FNEND

11560   !*************************************************************************
        !**     CENTER Z$                                                       **
        !**     POSITION TO APPROPRIATE LINE                                    **
        !**     PRINT A BELL AND THE MESSAGE                                    **
        !*************************************************************************

        DEF FN_SCREEN_3%  (Z$,Z1%)

                Z% = 80%
                Z$ = LEFT (Z$,Z% -4%) IF LEN (Z$) > Z% -4%
                Z% = Z% - LEN(Z$) IF LEN(Z$)
                Z% = 2% IF Z% < 2%
                Z$ = CHR$ (7%) + Z$ IF LEN(Z$)
                E3% = E3% OR (512%*(Z1%+1%))
                Z1% = 23% + Z1%
                Z% = FN_SCREEN_1% (1%,Z1%,-2%) + &
                     FN_SCREEN_2% (Z%/2%, Z1%, Z$)
11570           FNEND
```

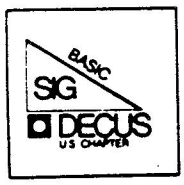
```
1       !       VTSPY.B2S
        !       A program to look at account action
100     !
        !       M a p   S t a t e m e n t s
        !               Map Name                Description
110     Map (DETAIL)    D.Job$ =   3%           !Job Number &
                ,       D.Sp1$ =   1%           !Filler &
                ,       D.Prj$ =   4%           !Project number &
                ,       D.Com$ =   1%           !Comma &
                ,       D.Prg$ =   4%           !Programmer &
                ,       D.Sp2$ =   1%           !Filler &
                ,       D.Whr$ =   5%           !Where (KB) &
                ,       D.Sp3$ =   1%           !Filler &
                ,       D.Wht$ =   6%           !Job Name &
                ,       D.Sp4$ =   1%           !Filler &
                ,       D.Chn$ =   2%           !Channel &
                ,       D.Sp5$ =   1%           !Filler &
                ,       D.Fil$ =  23%           !File Spec &
                ,       D.Sp6$ =   1%           !Filler &
                ,       D.Rec$ =   7%           !Record &
                ,       D.Sp7$ =   1%           !Filler &
                ,       D.Siz$ =   7%           !File Size &
                ,       D.Sp8$ =   1%           !Filler &
                ,       D.Bpm$ =   8%           !Flags
120     Map (DETAIL)    D.All$ =  79%           !Whole Line
130     Map (PLINE)  Line$(15%) = 79%
140     Map (PLINE)  Line.All$ = 1184%
150     Map (POLD)   Line.Old$(15%) = 79%
160     Map (POLD)   Line.Old.All$ = 1184%
200     !
        !       D i m e n s i o n   S t a t e m e n t s
210     Dim Mon.Tab%(30%), Wcb%(15%), Fcb%(15%), Word%(30%)
220     Dim Sizze(100%,12%)
1000    !
```

```
        !      G e t   P a r a m e t e r s   /   S e t u p
1010    On Error Goto 32000                        !Error Trapping
        Trap$ = Sys(Chr$(6%) + Chr$(-7%))
        Print If CCpos(0%)                         !Do <CR> <LF>
        Print 'VTSPY'; Chr$(9%); 'V7.06';          !Title Line &
              Chr$(9%); Fne$(0%); '   ';           !       More &
              Date$(0%); '   '; Time$(0%)          !       Done

1030    Print
        Input 'Interval.....<5>';Interval%
        Interval% = Abs(Interval%)
        Interval% = 5% If Interval% = 0%

1040    Print
        Input 'Size.........<0>';Sizze
        Sizze = Abs(Int(Sizze))

1050    Line.All$ = ''
        Line.Old.All$ = ''
        Begin% = -1%
        Sec = Time(0%)

2000    !
        !  Program Starts Here
        !
        !
        !

2005    Start% = -1%

2010    Open 'KB:' as file #2%, Mode 8%

2011    A$ = Sys(Chr$(3%) + Chr$(2%))

2012    Open 'KB:' for output as File #1%
2015    A% = FnMon.Tab%
        A% = FnDev.Name%
        Prn% = FnPrn%(FnJob.Num%)

2020    Scr% = Fnclear(1%,Term%)

2030    If Sizze <> Size.Old or Start% Then
              Top$ = 'Display of all Open Files Containing '  + &
                        Num1$(Sizze) + ' or More Segments'
              Dif% = Int((79 - Len(Top$)) / 2)
              Top$ = Space$(Dif%) + Top$ + Space$(Dif%)
              Scr% = Fnprint%(1%,Term%,1%,0%,Top$)
              Size.Old = Sizze
```



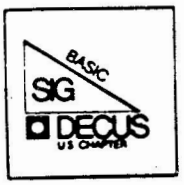
```
2035    If Priv.Mask% <> Priv.Old% Then
            Priv.Old% = Priv.Mask%

2040    If Start% Then
            Hdr$ = "Job    Who    Where  What  C#       File Spec" &
                 + "          Block       Of   Bl/Min"
            Scr% = FnPrint%(1%,Term%,4%,0%,Hdr$)
            Start% = 0%

2041    Scr% = FnPrint%(1%,Term%,3%,0%,"Working...")

2045    Cnt% = 0%

2050    For I% = Job% + 1% to Max.Jb%
            Goto 2100 Unless FnLogged.In%(I%)
            Goto 2100 If Priv.Mask% And (FnMatch%(FnPpn%(I%),Priv.Mask%) = 0%)
            If (Swap%(Ppn%) And 255%) <> 1% &
                And (Swap%(FnPpn%(I%)) And 255%) = 1%
                     Then Goto 2100

2060            D.All$ = Space$(80%)
                Iob%   = FnIob%(I%)
                Flag%  = -1%
                For J% = 1% to 12%
                    D.Rec$ = Space$(7%)
                    D.Siz$ = Space$(7%)
                    D.Bpm$ = Space$(8%)
                    Base% = Peek(Iob% + 2% * J%)
                    Goto 2090 Unless Base%
                    Idx% = Peek(Base%) And 255%
                    If Idx% Then
                       Goto 2090
                    Else
                       Gosub 10100
                       Size(I%,J%) = R.Size If Size(I%,J%) = 0 &
                                           Or R.Size < Size(I%,J%)

2070                    Goto 2090 If F.Size < Size
                        If Flag% Then
                               D.Job$ = String$(3%-Len(Num1$(I%)),32%)+Num1$(I%)
                               D.Com$ = ","
                               D.Ppn$ = FnPpn$(I%)
                               D.Wht$ = FnLow.Case$(FnJob.Name$(I%))
                               Ddb%   = Peek(Iob%)
                               Ttint% = Peek(Ddb% + 30%)
                               Dds%   = Peek(Ddb% + 2%)
                               D.Whr$ = "Det"
                               Flag% = 0%
                               If ((Dds% And 255%) = I% * 2%) &
```

```
                    And (Peek(Ddb% + 6%) And 8192% <> 0%) Then
                            D.Whr$ = Fnconsole.Kb$(I%)
                            D.Whr$ = FnLow.Case$(D.Whr$)
                            D.Whr$ = Cvt$$(D.Whr$,128%) + '*' &
                                If Ttint% And 16384%
                    End If

2075        Goto 2100 If Detach% And D.Whr$ = 'Det'
            Rset D.Chn$ = Num1$(J%)
            Bpm = (R.Size - Sizze(I%,J%)) * (60 / (Time(0%) - Sec)) &
                    Unless Begin%
            Temp$=Num1$(Int((Bpm-Int(Bpm)+.005)*100.))
            Temp$=Num1$(Int(Bpm))+'.'+Temp$+String$(2%-Len(Temp$),48%)
            D.Bpm$ = String$(8%-Len(Temp$),32%)+Temp$
            Cnt% = Cnt% + 1%
            Line$(Cnt%) = D.All$

2080        If Cnt% > 14% Then
                    J% = 12%
                    I% = Max.Jb%

2090        Next J%

2100    Next I%

2110    Begin% = 0% If Time(0%) - Sec > 0

3000    !
        !  Do Printout Here
        !
        !

3010    For I% = 0% to Cnt%
            If Line$(I%) <> Line.Old$(I%) Then
                    Scr% = Fnprint%(1%,Term%,I% + 4%,0%,Line$(I%))
                    Line.Old$(I%) = Line$(I%)

3011        Next I%

3012    I% = I% + 1%
        Goto 3040 If I% > 15%

3030    If Line.Old$(I%) <> '' Then
                Line.Old$(I%) = ''
                Scr% = Fncursor(1%,Term%,I% + 4%,1%) If Term%
                Print #1% If Term%
                I% = I% + 1%
                Goto 3030 Unless I% > 15%

3035    If Term% = 0% Then
```




```
                Scr% = Fncursor(1%,Term%,Cnt% + 5%,1%)
                Print #1%, Chr$(155%);"J"

3040    If Job% Then
                Mess$ = "**** Skipping First " + Num1$(Job%) + " Jobs ****"
                Y% = Int((80 - Len(Mess$)) / 2)
                Scr% = Fnprint%(1%,Term%,2%,Y%,Mess$)
                Sleep 4
                Scr% = Fnprint%(1%,Term%,2%,Y%,Space$(Len(Mess$)))

3050    Scr% = Fnprint%(1%,Term%,3%,0%,Space$(10%))
        Scr% = Fncursor(1%,Term%,3%,0%)

4000    !
        !  Wait For Input ..... If Any
        !
        !
        !

4010    Sleep Interval% - 1%

4020    Wait 1%
        Get #2%
        Gosub 10200

4030    Par$ = Cvt$$(Inp$,-1%)
        Goto 2020 If Par$ = "" And Start%
        Goto 2030 If Par$ = ""

4040    If Left(Par$,1%) = "I" Then
                Interval% = Val(Right(Par$,2%))
                Goto 4110

4050    If Left(Par$,1%) = "J" Then
                Job% = Val(Right(Par$,2%))
                Job% = Max.Jb% - 1% If Job% > Max.Jb% - 1%
                Goto 4110

4060    If Left(Par$,1%) = "S" Then
                Sizze = Abs(Int(Val(Right(Par$,2%))))
                Goto 4110

4070    If Par$ = "A" Then
                Detach% = -1%
                Goto 4110

4080    If Par$ = "D" Then
                Detach% = 0%
                Goto 4110
```

```
4090    If Left(Par$,1%) = 'A' Then
                Acct$ = Right(Par$,2%)
                Lbr%  = Instr(1%,Acct$,'(')
                Com%  = Instr(1%,Acct$,',')
                Rbr%  = Instr(1%,Acct$,')')
                Rbr%  = Len(Acct$) + 1% If Rbr% = 0%
                PrJ%  = Val(Seg$(Acct$,Lbr% + 1%,Com% - 1%))
                Prg$  = Seg$(Acct$,Com% + 1%,Rbr% - 1%)
                Priv.Mask% = Swap%(PrJ%) + 255% If Prg$ = '*'
                Priv.Mask% = Swap%(PrJ%) + Val(Prg$) If Prg$ <> '*'
                Goto 4110 Unless PrJ% = 1% And (Swap%(Ppn%) And 255%) <> 1%
                Er.$ = '[Not a Chance....]'
                Scr% = Fnprint%(1%,Term%,23%,1%,Er.$)
                Sleep 3
                Scr% = Fnprint%(1%,Term%,23%,1%,Space$(19%))
                Priv.Mask% = Priv.Old%
                Goto 2030

4095    If Instr(1%,'EXIT',Par$) = 1% Then
                Scr% = Fnclear (1%, Term%)
                Close 1,2
                Goto 32767
        End If
        If Instr(1%,'HELP',Par$) = 1% or Par$ = '?' Then
                Gosub 4200
                Sleep 20
                Start% = -1%
                Line.All$ = ''
                Line.Old.All$ =''
                Goto 4020

4100    Er.$ = '[Unrecognized Command - ' + Inp$ + '...]'
        Scr% = Fnprint%(1%,Term%,23%,1%,Er.$)
        Sleep 3
        Scr% = Fnprint%(1%,Term%,23%,1%,Space$(Len(Er.$)))
        Goto 2030

4110    Goto 2020 If Start%
        Goto 2030

4200    Scr% = Fnclear(1%,Term%)
        Print #1%,Tab(28%);'FILE STATUS VIDEO DISPLAY'
        Print #1%,Tab(28%);'-------------------------'
        Print #1%, For I% = 1% to 3%
        Print #1%, 'LEGAL INPUT:'
        Print #1%
        Print #1%, 'In          - CHANGES INTERVAL TO n SECONDS'
        Print #1%, 'Jn          - IGNORES FIRST n JOBS'
        Print #1%, 'Sn          - IGNORES FILES LESS THAN n BLOCKS'
        Print #1%, 'A           - IGNORES DETACHED JOBS'
```




```
        Print #1%, 'D           - REENABLES DETACHED JOBS'
        Print #1%, 'APPN        - DISPLAYS ONLY ACCOUNT #PPN'
        Print #1%, 'A0,0        - TURNS OFF ACCOUNT LOOKUP'
        Return

10000   !

        !       P r o g r a m m e r   D e f i n e d

        !       R o u t i n e s   /   F u n c t i o n s

10001!  Programmer Defined Subroutines
     !

10100!  What We Have Here Is A WCB That Points To A FCB Dept.
     !

10110   A% = FnWcb.Decode%(Base%)
        A% = FnFcb.Decode%(Wcb.Fcb% - 28%)
        R.Size = ((256. * 256.) * Wcb.Msb%) + (256. * Wcb.Nsb%) + Wcb.Lsb%
        Temp$ = Num1$(Int(R.Size))
        Temp$ = Left(Temp$,Len(Temp$)-3%)+','+Right(Temp$,Len(Temp$)-2%) &
                If Len(Temp$)>3%
        D.Rec$ = String$(7%-Len(Temp$),32%)+Temp$

10120   F.Size = ((256. * 256.) * Fcb.Msb%) + (256. * Fcb.Nsb%) + Fcb.Lsb%
        Temp$ = Num1$(Int(F.Size))
        Temp$ = Left(Temp$,Len(Temp$)-3%)+','+Right(Temp$,Len(Temp$)-2%) &
                If Len(Temp$)>3%
        D.Siz$ = String$(7%-Len(Temp$),32%)+Temp$

10140   Device$ = Mid(Dev.Nme$,(Fcb.Unt% * 3% + 1%),3%) + ':'
        Device$ = FnLow.Case$(Device$)
        D.Fil$ = Device$ + FnPpn1$(Fcb.Ppn%) + Fcb.Fil$
        Return

10200   !
        !  Field the Input
        !
        !
        !

10210   If Recount = 1% Then
                Field #2%, 1% as Term$
        Else
                Field #2%, Recount - 2% as Inp$ , 2% as Term$

10220   If Ascii(Term$) = 3% Then
                Scr% = Fnclear(1%,Term%)
                Close 1
```


```
            Goto 32767

10230   Return

15000!  Programmer Defined Functions
     !

15005!  Return An Error Message

15010   Def Fne$(E%) = Right(Sys(Chr$(6%) +     !Print an error &
                             Chr$(9%) +     !       message &
                             Chr$(E%)),3%)  !Done

15100!  Function To Setup Monitor Tables In Array Mon.Tab%
     !

15110   Def FnMon.Tab%                          !Here We Go
        Change Sys(Chr$(6%)+Chr$(-3%))  To Mon.Tab%
        Max.Kb% = Mon.Tab%(3%)
        Max.Jb% = Mon.Tab%(4%)
        Dev.Cn% = FnSwp%( 5%)
        Dev.Pt% = FnSwp%( 7%)
        Mem.Ls% = FnSwp%( 9%)
        Job.Tb% = FnSwp%(11%)
        Job.St% = FnSwp%(13%)
        Job.Wt% = FnSwp%(15%)
        Unt.Cl% = Fnswp%(17%)
        Unt.Cn% = FnSwp%(19%)
        Sat.Ct% = FnSwp%(21%)
        Jsb.Tb% = FnSwp%(23%)
        Sat.Cm% = FnSwp%(25%)

15120   Change Sys(Chr$(6%)+Chr$(-12%)) To Mon.Tab%
        Fre.Es% = FnSwp%( 3%)
        Dev.Nm% = FnSwp%( 5%)
        Csr.Tb% = FnSwp%( 7%)
        Dev.Kb% = FnSwp%( 9%)
        Tty.Hc% = FnSwp%(11%)
        Job.Ct% = FnSwp%(13%)
        Rts.Lt% = FnSwp%(15%)
        Erl.Ct% = FnSwp%(17%)
        Snd.Ls% = FnSwp%(19%)
        Log.Nm% = FnSwp%(21%)
        Dev.Sy% = FnSwp%(23%)
        Mem.Sz% = FnSwp%(25%)
        Ccl.Lt% = FnSwp%(27%)
        Fnend

15150!  Function To Do The Swap% For Above
     !
```

```
15160   Def FnSwp%(X%) = Mon.Tab%(X%) + Swap%(Mon.Tab%(X%+1%))

15200!  Function To Return Our Job Number
     !

15210   Def FnJob.Num% = (Peek(518%) And 255%) / 2%

15225!  Function To Return Address Of JDB For Job X%
     !

15230   Def FnJdb%(X%) = Peek(Job.Tb% + 2% * X%)

15350!  Function To Return Address Of JDB2 For Job X%
     !

15360   Def FnJdb2%(X%) = Peek(FnJdb%(X%) + 8%)

15375!  Function To Return PPN For Job X%
     !

15380   Def FnPpn%(X%) = Peek(FnJdb2%(X%) + 24%)

15390!  Function To Get A Job Name
     !

15395   Def FnJob.Name$(X%) = Rad$(Peek(FnJdb2%(X%)+12%)) + &
                              Rad$(Peek(FnJdb2%(X%)+14%))

15400!  Function To See If A Job Is Logged In This Slot
     !

15410   Def FnLogged.In%(X%) = (Peek(Job.Tb% + 2% * X%) <> 0%)

15425!  Function To Get Someone's IOB
     !

15430   Def FnIob%(X%) = Peek(FnJdb%(X%))

15440!  Function To Find Someone's Console KB
     !

15450   Def FnConsole.Kb%(X%) = (Swap%(Peek(Peek(FnIob%(X%)) + 2%)) And 255%)

15500!  Function To Format A Ppn
     !

15510   Def FnPpn$(X%)                          !Return $
            P% = FnPpn%(X%)
        Rset D.Prj$ = "[" + Num1$(Swap%(P%) And 255%)
```

```
        Lset D.Prg$ = Num1$(P% And 255%) + "]"
        Fnend

15520   Def FnPpn1$(P%) = "[" + Num1$(Swap%(P%) And 255%) &
                  + "," + Num1$(P% And 255%) + "]"

15525!  Function To Format A Console KB:
     !

15530   Def FnConsole.Kb$(X%)
        Kb% = FnConsole.Kb%(X%)
        If (Ttint% And 255%) = 8% Then
                FnConsole.Kb$ = "PK" + Num1$(Kb% - 1%)
        Else
                FnConsole.Kb$ = "KB" + Num1$(Kb%)

15531   Fnend

15630!  Function To Change Upper Case Name To Lower Case
     !

15640   Def FnLow.Case$(W$)
        Change W$ To Word%
                For Cha% = 2% To Word%(0%)
                   Word%(Cha%) = Word%(Cha%) + 32% &
                        If Word%(Cha%) > 64% &
                                And Word%(Cha%) < 91%
                Next Cha%
        Change Word% To W$
        FnLow.Case$ = W$
        Fnend

15700!  Function to Check on Match between Ppn of Current Job
     !  and Requested Ppn

15710   Def FnMatch%(X%,Y%)

15720   FnMatch% = 0%
        Goto 15740 If Target.Job% <> 0%
        Goto 15730 If Y% - Swap%(PrJ%) = 255%
        FnMatch% = -1% If X% = Y%
        Fnexit

15730   FnMatch% = -1% If (X% And Y%) = X% And (Swap%(X%) And 255%) = PrJ%
        Fnexit

15740   If (Swap%(Ppn%) And 255%) = (Swap%(X%) And 255%) &
        Or (Swap%(Ppn%) And 255%) = 1% Then
                FnMatch% = -1%
```


```
15750   Fnend

16000!  Functions For Large File Systems Department
     !

16010!  Decode A Window Control Block
     !

16020   Def FnWcb.Decode%(Wcb.Base%)
          Wcb%(K% / 2%) = Peek(Wcb.Base% + K%)  For K% = 0% To 30% Step 2%

16030   Wcb.Idx% = Wcb%(0%) And 255%
        Wcb.Sts% = Swap%(Wcb%(0%)) And 255%
        Wcb.Job% = Wcb%(1%) And 255%
        Wcb.Fls% = Swap%(Wcb%(1%)) And 255%
        Wcb.Msb% = Swap%(Wcb%(2%)) And 255%
        Wcb.Nsb% = Swap%(Wcb%(3%)) And 255%
        Wcb.Lsb% = Wcb%(3%) And 255%
        Wcb.Fcb% = Wcb%(4%)
        Wcb.Wcb% = Wcb%(6%)
        Wcb.Nxt% = Wcb%(7%)
        Fnend

16050!  Decode A File Control Block
     !

16060   Def FnFcb.Decode%(Fcb.Base%)
           Fcb%(K% / 2%) = Peek(Fcb.Base% + K%) For K% = 0% To 30% Step 2%

16070   Fcb.Nxt% = Fcb%(0%)
        Fcb.Prn% = Fcb%(2%)
        Fcb.Fil$ = FnLow.Case$(Rad$(Fcb%(3%))          + &
                               Rad$(Fcb%(4%)))         + &
                                ".".                   + &
                   FnLow.Case$(Rad$(Fcb%(5%)))
        Fcb.Stt% = Fcb%(6%) And 255%
        Fcb.Msb% = Swap%(Fcb%(12%)) And 255%
        Fcb.Unt% = Fcb%(12%) And 255%
        Fcb.Nsb% = Swap%(Fcb%(13%)) And 255%
        Fcb.Lsb% = Fcb%(13%) And 255%
        Fcb.Wcb% = Fcb%(15%)
        Fnend

16100!  Function To Build Device Name Tables
     !

16110   Def FnDev.Name%
        For M% = 0% To (Dev.Kb% - 2%) Step 2%
        Dev.Gen$ = Cvt%$(Swap%(Peek(M% + Dev.Nm%)))
        Dev.Nme$ = Dev.Nme$ + Dev.Gen$ + Chr$(48% + K%) &
```


```
                For K% = 0% To Peek(Dev.Cn% + M%)
        Next M%

16120   M% = Dev.Nm% + M%
        Dev.Ddb$ = "?"

16130   M% = M% + 2%
        K% = Peek(M%)
        If K% <> -1% Then
                Dev.Ddb$ = Dev.Ddb$ + Cvt%$(Swap%(K%))
                Goto 16130

16140   Fnend

16200!  Function To Find The Device For A WCB
     !

16210   Def FnDev$(Hnd,Idx%)
        If Hnd.Idx% = 0% Then
                Dev$ = Mid(Dev.Nme$,(Fcb.Unt% * 3% + 1%),3%) + ":"
        Else
                Dev$ = Mid(Dev.Ddb$,Hnd.Idx%,2%) + Num1$(Fcb.Unt%) + ":"

16220   Dev$ = "NL:" If Dev$ = "NL0:"
        FnDev$ = Dev$
        Fnend

17000   !
        !       Screen Functions For Paint Mode
        !

17010   !       Clear The Screen Department

17020   Def Fnclear (Channel%,Terminal%)
                        !I/O Channel
                        !Terminal Type:
                        !       0 For VTXX
                        !       1 For ADDS
                        !       2 For VT52
17030   Print #Channel%, Chr$(155%) ; "[0H" ; Chr$(155%) ; "[0J" ; If Terminal% = 0%
                        !VT100
        Print #Channel%, Chr$(12%) ; If Terminal% = 1%
                        !ADDS
        Print #Channel%, Chr$(155%) ; "H" ; Chr$(155%) ; "J" ; If Terminal% = 2%
                        !VT52


17040   Fnend
                        !End of definition
```

BASIC SIG

```
17050   !       Cursor Position Routine

17060   Def Fncursor(Channel%,Terminal%,Row%,Col%)
                        !I/O Channel
                        !Terminal Type
                        !       0 For VT100
                        !       1 For ADDS
                        !       2 For VT52
                        !Screen Row
                        !Screen Column

17070   Goto 17080 If Terminal% = 1%
                        !See if they have an ADDS
        If Terminal% = 2% Then
                Print #Channel%,        Chr$(155%)              ; &
                                        Chr$( 89%)              ; &
                                        Chr$( 31% + Row%)       ; &
                                        Chr$( 32% + Col%)       ;
                                !Cursor Addressing For VT52
        Else
                Print #Channel%,        Chr$(155%)              + &
                                        "[" + Num1$ (Col%)      + &
                                        ";" + Num1$ (Row%)      + &
                                        "H"                     ;
        End If

        Goto 17090
                        !End of VTXX Stuff

17080   Cur.ars1% = Int(Col% / 10%)
        Cur.ars2% = Col% - (10% * Cur.ars1%)
        Print #Channel%,        Chr$(139%)              ; &
                                Chr$( 63% + Row%)       ; &
                                Chr$(155%)              ; &
                                Chr$(  5%)              ; &
                                Chr$(Cur.ars1% + 128%)  ; &
                                Chr$(Cur.ars2% + 128%)  ;
                        !Cursor Leadin For ADDS 980

17090   Fnend
                        !Exit

17100   !       Print this string wherever

17110   Def Fnprint%(Channel%,Terminal%,Row%,Col%,This$)
                        !I/O Channel
                        !Terminal Type
                        !       0 For VTXX
                        !       1 For ADDS
                        !Screen Row
```


```
                    !Screen Column
                    !Print This

17120   Dummy% = Fncursor(Channel%,Terminal%,Row%,Col%)
                    !Call in the Cursor Routines
        Print #Channel%, This$ ;
                    !Work Done Here

17130   Fnend
                    !Exit

32000   !

        !       E r r o r   H a n d l i n g   C o d e

32010   If Err = 11 Then
                Resume 1030 If Erl = 1040
                Resume 32767 If Erl = 1020
                Resume 2020 If Start%
                Resume 2030

32020   If Err = 28 Then
                Scr% = Fnclear(1%,Term%)
                Close 1,2
                Resume 32767

32030   If Err = 15 Then
                Resume 2020 If Start%
                Resume 2030

32040   If Err = 52 Then
                Resume 4100

32099   On Error Goto 0

32767   End
```

BASIC SIG

This piece of 'Magic' is really a BASIC-PLUS-2 V2.2 bug, but it is interesting.

```
2000    R.NUMBER = 1599
        Z% = 24567.
        GET #1%, RECORD R.NUMBER
                ! This statement will consistantly (at SCALE 6)
                ! set record 1598...not 1599
```

Now for the old 'get rid of ERL references so COMPILE/NOLINE can be done" trick!

```
2000    AVG = TOTAL/ITEMS
        PRINT 'Average is! ';AVG
                .
                .
                .
19020   IF      ERL = 2000%
        THEN    PRINT 'Average unknown"
                RESUME 9000
```

******* Now the /NOLINE version *******

```
2000    MY.ERL% = 2000%
        AVG = TOTAL/ITEMS
        MY>ERL% = 0%
        PRINT 'Average is! ';AVG
                .
                .
                .
19020   IF      MY.ERL% = 2000%
        THEN    MY.ERL% = 0%
                PRINT "Average unknown"
                RESUME 9000
```

TO READ A LOG FILE WHILE THE BATCH JOB IS RUNNING ON VAX

EDIT/SOS/READ comfilename.LOG

In editor, answer:

P ^:*

(including the space)

This starts the file printing from the beginning.

If the file gets to line 65500 and stops, hit a lot of carriage returns and answer:

/READONLY/NONUMBERS

P ^:*

The rest of the file should start printing.

Hit Control C and then Control Z to get out of the editor.

```
        .TITLE DBØCAN
;
;       P R O G R A M   T I T L E   B L O C K
;
;       PROGRAM:        DBØCAN          VERSION 1A.ØØ
;       EDIT LEVEL:     ØØØØ            DATE 06-13-8Ø
;       AUTHOR(S):      WIT
;       OP SYS: RSX-11M V3.2
;
;       C A L L   F O R M A T
;
;       CALL DBØCAN
;
;       ARGUMENT NAME           DESCRIPTION
;       =============           ============================
;
        .PAGE
;       M O D I F I C A T I O N   H I S T O R Y   L O G
;
;       VER/ED  DATE    INITIAL REASON
;       ======  ====    ======= ======================
;
        .PAGE
;       P R O G R A M   D E S C R I P T I O N
;
;       THE PURPOSE OF THIS ROUTINE IS CANCEL THE CURRENT TASK
;       IT IS DESIGNED FOR USE WITH THE DBØCHN SUBROUTINE
;       WHICH SPAWNS MCR WITH A RUN COMMAND
        .PAGE
;       M A C R O   C A L L S
;
        .MCALL EXIT$S
        .PAGE
DBØCAN::
        EXIT$S          ; CALL EXIT NORMAL
        .END
```

```
1       SUB DB0CHN (PROGRAM.NAME$,TASK.ID$)                                 &
        !                                                                   &
        ! P R O G R A M   T I T L E   B L O C K                             &
        !                                                                   &
        ! PROGRAM:       DB0CHN          VERSION 1A.00                     &
        ! EDIT LEVEL:    0001            DATE 06-13-80                     &
        ! AUTHOR(S):     WIT                                               &
        ! OP SYS:        RSX-11M V3.2                                      &
        !                                                                   &
        \ ON ERROR GO TO 19000                                              &

1010    CALL DB0RAD("MCR...",MCR$,ERROR.CODE%)                              &
        !               CONVERT MCR... TO RAD50 IN MCR$                     &
        \ CALL DB0TTN(T%)                                                   &
        !               GET TERMINAL NUMBER                                 &
        \ T1$=NUM1$(T%)                                                     &
        !               CONVER TERMINAL NUMBER TO A STRING                  &
        \ T1$="0"+T1$                   IF LEN(T1$)<2%                      &
        !               PAD WITH LEADING ZERO IF NEEDED                     &
        \ T1$ = "T" IF                  IF LEN(T1$)<3%                      &
        !               ADD A T IN FRONT IF TERMINAL NUMBER IS NOT          &
        !               GREATER THAN 3                                      &
        \ TASK.ID$=EDIT$(TASK.ID$,128%)+T1$                                 &
        !               ADD TERMINAL NUMBER TO END TO TASKID                &
        \ TASK.ID$=LEFT(TASK.ID$,3%)                                        &
        !               THIS WILL CAUSE THE PROGRAM TO BE T01T01 IF         &
        !               NO NAME WAS PASSED                                  &
        \ CMD.LINE$="RUN "+PROGRAM.NAME$+"/TASK="+TASK.ID$+T1$              &
        !               BUILD THE COMMAND LINE                              &
        \ CMD.LINE.LEN%=LEN(CMD.LINE$)                                      &
        \ DUMMY1$=STRING$(4%,0%)                                            &
        \ DUMMY$ =SPACE$(16%)                                               &
        \ A%=RCTRLC                                                         &
        \ CALL DB0CLS           ! CLOSE ALL THE OPEN CHANNELS               &
        \ CALL SPAWN BY REF (   MCR$,1%,1%,0%,DUMMY$,DUMMY$,I%,             &
                                CMD.LINE$,CMD.LINE.LEN%,0%,DUMMY1$,         &
                                STAT% )                                     &
        \ IF STAT% <> 1% THEN PRINT "??Error in chain ";NUM1$(STAT%)        &

1020    CALL DB0CAN                                                         &
        \ GO TO 32000                                                       &

19000!  *                                                                   &
        ! S T A N D A R D   E R R O R   H A N D L I N G                     &
        !                                                                   &

19900   PRINT                                                               &
        \ ERR.MESSAGES$=ERT$(ERR)                                           &
        \ PRINT "??Error ";ERR.MESSAGES$                                    &
        \ PRINT "IN ";ERN$;" at line ";ERL                                  &
        \ ERR.MESSAGES$=""                                                  &
        \ ERR.IND%=-1%                                                      &
        \ RESUME 32767                                                      &

32000!  *                                                                   &
        ! E N D   O F   J O B   P R O C E S S I N G                         &
        !                                                                   &

32767   !                                                                   &
        !       E N D   O F   P R O G R A M                                 &
        !                                                                   &
        \ SUBEND                                                            &
```

```
1       SUB DB0CLS                                                      &
        !                                                               &
        ! P R O G R A M   T I T L E   B L O C K                         &
        !                                                               &
        ! PROGRAM:       DB0CLS          VERSION 1A.00                 &
        ! EDIT LEVEL:    0000            DATE 06-12-80                 &
        ! AUTHOR(S):     WIT                                           &
        ! OP SYS:        RSX-11M V3.2                                  &
        !                                                               &
        \ ON ERROR GO TO 19000                                          &

15      !                                                               &
        ! C A L L   F O R M A T                                         &
        !                                                               &
        ! CALL DB0CLS                                                  &
        !                                                               &
        ! ARGUMENT NAME         DESCRIPTION                             &
        ! =============         ============================            &
        !                                                               &

20!     *                                                               &
        ! M O D I F I C A T I O N   H I S T O R Y   L O G               &
        !                                                               &
        ! VER/ED        DATE    INITIAL REASON                          &
        ! ======        ====    ======= ======================          &
        !                                                               &

100!    *                                                               &
        ! P R O G R A M   D E S C R I P T I O N                         &
        !                                                               &
        !       CLOSE ALL CHANNELS                                      &

400!    *                                                               &
        ! V A R I A B L E S   A N D   A R R A Y S   U S E D             &
        !                                                               &
        ! NAME                  DESCRIPTION                             &
        ! ==========            ============================            &
        !                                                               &
        ! I%                    LOOP COUNTER                            &

1000!   *                                                               &
        !       M A I N   P R O G R A M   L O G I C                     &
        !                                                               &

1010    FOR I%=12% TO 1% STEP -1%                                       &
        \ IF BUFSIZ(I%)<>0% THEN CLOSE #I%                              &

1020    NEXT I%                                                         &
        \ GO TO 32000                                                   &

19000!  *                                                               &
        ! S T A N D A R D   E R R O R   H A N D L I N G                 &
        !                                                               &

19900   PRINT                                                           &
        \ ERR.MESSAGES$=ERT$(ERR)                                       &
        \ PRINT "??Error ";ERR.MESSAGES$                                &
        \ PRINT "IN ";ERN$;" at line ";ERL                              &
        \ ERR.MESSAGES$=""                                              &
        \ ERR.IND%=-1%                                                  &
        \ RESUME 32767                                                  &

32000!  *                                                               &
        ! E N D   O F   J O B   P R O C E S S I N G                     &
        !                                                               &
```

```
32767   !                                                              &
        !        E N D   O F   P R O G R A M                           &
        !                                                              &
        \ SUBEND                                                       &
```

```
1       SUB DBØTTN(TERM.NUM%)                                           &
        !                                                               &
        ! P R O G R A M   T I T L E   B L O C K                         &
        !                                                               &
        ! PROGRAM:       DBØTTN          VERSION 1A.ØØ                 &
        ! EDIT LEVEL:    ØØØØ            DATE 06-11-8Ø                 &
        ! AUTHOR(S):     WIT                                           &
        ! OP SYS:        RSX-11M V3.2                                  &
        !                                                               &
        \ ON ERROR GO TO 19ØØØ                                          &

15      !                                                               &
        ! C A L L   F O R M A T                                         &
        !                                                               &
        ! CALL DBØTTN (TERM.NUM%)                                       &
        !                                                               &
        ! ARGUMENT NAME          DESCRIPTION                            &
        ! =============          ==============================         &
        !                                                               &
        ! TERM.NUM%              TERMINAL NUMBER OF CURRENT TT:         &

2Ø!     *                                                               &
        ! M O D I F I C A T I O N   H I S T O R Y   L O G               &
        !                                                               &
        ! VER/ED         DATE    INITIAL REASON                         &
        ! ======         ====    ======= ======================         &
        !                                                               &

1ØØ!    *                                                               &
        ! P R O G R A M   D E S C R I P T I O N                         &
        !                                                               &
        !       GET YOUR TERMINAL NUMBER                                &

4ØØ!    *                                                               &
        ! V A R I A B L E S   A N D   A R R A Y S   U S E D             &
        !                                                               &
        ! NAME                   DESCRIPTION                            &
        ! ==========             ==============================         &
        !                                                               &
        ! TERM.NUM%              NUMBER OF THE CURRENT TT:              &
        ! THE.BUF$               WORK BUFFER USED IN CALL TO GETLUN      &

7ØØ!    *                                                               &
        ! S U B R O U T I N E S   U S E D                               &
        !                                                               &
        ! NAME/LINE #            DESCRIPTION                            &
        ! ===========            ==============================         &
        !                                                               &
        ! GETLUN                 RSX SYSTEM SUBROUTINE TO GET INFO ON   &
        !                                A LOGICAL CHANNEL              &

1ØØØ!   *                                                               &
        !       M A I N   P R O G R A M   L O G I C                     &
        !                                                               &
        \ THE.BUF$=SPACE$(12%)                                          &
        \ CALL GETLUN BY REF (13%, THE.BUF$ )                           &
        \ IF LEFT(THE.BUF$,2%)<>"TT" THEN TERM.NUM%=Ø%                  &
        \       GO TO 32ØØØ                                             &

1Ø2Ø    TERM.NUM%=(ASCII(RIGHT(THE.BUF$,3%)) AND 127%)                  &
        \ GO TO 32ØØØ                                                   &
```



```
19ØØØ!  *                                                               &
        ! S T A N D A R D   E R R O R   H A N D L I N G                 &
        !                                                               &
```

```
19900   PRINT                                                          &
        \ ERR.MESSAGES$=ERT$(ERR)                                      &
        \ PRINT "??Error ";ERR.MESSAGES$                               &
        \ PRINT "IN ";ERN$;" at line ";ERL                             &
        \ ERR.MESSAGES$=""                                             &
        \ ERR.IND%=-1%                                                 &
        \ RESUME 32767                                                 &

32000!  *                                                              &
        ! E N D   O F   J O B   P R O C E S S I N G                    &
        !                                                              &
        \ THE.BUF$=""                                                  &

32767   !                                                              &
        !       E N D   O F   P R O G R A M                            &
        !                                                              &
        \ SUBEND                                                       &
```

```
1       SUB DBØRAD(IN$,OUT$,ERR.IND%)                                    &
        !                                                                &
        ! P R O G R A M   T I T L E   B L O C K                          &
        !                                                                &
        ! PROGRAM:       DBØRAD          VERSION 1A.ØØ                  &
        ! EDIT LEVEL:    ØØØØ            DATE Ø6-11-8Ø                  &
        ! AUTHOR(S):     WIT                                            &
        ! OP SYS:        RSX-11M V3.2                                   &
        !                                                                &
        \ ON ERROR GO TO 19ØØØ                                           &

15      !                                                                &
        ! C A L L   F O R M A T                                          &
        !                                                                &
        ! CALL DBØRAD ( IN$, OUT$, ERR.IND%)                             &
        !                                                                &
        ! ARGUMENT NAME          DESCRIPTION                             &
        ! =============          ============================            &
        ! IN$                    INPUT STRING IN ASCII                   &
        ! OUT$                   RETURNED STRING IN RAD5Ø                &
        ! ERR.IND%       =       Ø% NO ERROR                             &
        !                        ERROR CODE                              &
        !                                                                &

2Ø!     *                                                                &
        ! M O D I F I C A T I O N   H I S T O R Y   L O G                &
        !                                                                &
        ! VER/ED         DATE    INITIAL REASON                          &
        ! ======         ====    ======= ======================          &
        !                                                                &

1ØØ!    *                                                                &
        ! P R O G R A M   D E S C R I P T I O N                          &
        !                                                                &
        !       CONVERT ASCII TO RAD5Ø                                   &

4ØØ!    *                                                                &
        ! V A R I A B L E S   A N D   A R R A Y S   U S E D              &
        !                                                                &
        ! NAME                   DESCRIPTION                             &
        ! ==========             ================================        &
        !                                                                &
        ! A$                     WORK VARIABLE CONTAINING THE RAD5Ø       &
        !                                INTERGER                        &
        ! ERR.IND%               ERROR CONDITION CODE INDICATOR          &
        ! ERR.MESSAGES$          STRING CONTAINING THE ERROR MESSAGE     &
        ! IN$                    THE INPUT STRING                        &
        ! J%                     LOOP COUNTER                            &
        ! J1%                    INTERGER VALUE OF CONVERTED STRING      &
        ! J2%                    RAD5Ø POSSITION OF THIS LETTER          &
        ! K%                     LOOP COUNTER                            &
        ! OUT$                   THE OUTPUT RAD5Ø STRING                 &
        ! ST$                    CONVERSION STRING                       &
        ! TEMP$                  TEMPORARY STRING CONTAINING THE STRING  &
        !                                TO BE CONVERTED                 &
        ! TEMP1$                 THE CURRENT 3 CHARACTER TO BE           &
        !                                TO BE CONVERTED                 &
        ! THE.LEN                TOTAL NUMBER OF 3 CHARACTER STRINGS     &
        !                                TO BE CONVERTED                 &
```



```
1ØØØ!   *                                                                &
        !       M A I N   P R O G R A M   L O G I C                      &
        !                                                                &
        \ ST$=" ABCDEFGHIJKLMNOPQRSTUVWXYZ$.?Ø123456789"                 &
        \ ERR.IND% = Ø%                                                  &
```

```
        \ THE.LEN = LEN(IN$) / 3.                                          &
        \ IF THE.LEN <> INT(THE.LEN) THEN THE.LEN=INT(THE.LEN+1.)          &

1020    TEMP$=SPACE$(THE.LEN*3%)                                           &
        \ LSET TEMP$=IN$                                                   &
        \ OUT$=""                                                          &

1030    FOR J% = 1% TO THE.LEN                                             &
        \ J1% = 0%                                                         &
        \ TEMP1$=MID(TEMP$,(J%-1%)*3% + 1% , 3%)                           &
        \ FOR K% = 1% TO 3%                                                &
        \ J2% = INSTR(1%,ST$,MID(TEMP1$,K%,1%))                            &
        \ GO TO 1040     IF J2% = 0%                                       &
        \ J1%=J1%*40%+(J2%-1%)                                             &
        \ NEXT K%                                                          &
        \ A$=CVT%$(J1%)                                                    &
        \ OUT$=OUT$+RIGHT(A$,2%)+LEFT(A$,1%)                               &
        \ NEXT J%                                                          &
        \ GO TO 1050                                                       &

1040    ERR.IND%=-1%                                                       &

1050    TEMP1$=""                                                          &
        \ TEMP$=""                                                         &
        \ GO TO 32000                                                      &

19000!  *                                                                  &
        ! S T A N D A R D   E R R O R   H A N D L I N G                    &
        !                                                                  &

19900   PRINT                                                              &
        \ ERR.MESSAGES$=ERT$(ERR)                                          &
        \ PRINT "??Error ";ERR.MESSAGES$                                   &
        \ PRINT "IN ";ERN$;" at line ";ERL                                 &
        \ ERR.MESSAGES$=""                                                 &
        \ ERR.IND%=-1%                                                     &
        \ RESUME 32767                                                     &

32000!  *                                                                  &
        ! E N D   O F   J O B   P R O C E S S I N G                        &
        !                                                                  &

32767   !                                                                  &
        !       E N D   O F   P R O G R A M                                &
        !                                                                  &
        \ SUBEND                                                           &
```