A Further Development of The

Massachusetts Institute of Technology

Computer Aided Design Executive System

by

LIEUTENANT COMMANDER DAVID LEE STONE, U.S. NAVY

B.S.E.E., Purdue University
(1973)

SUBMITTED TO THE DEPARTMENTS OF
OCEAN ENGINEERING AND MECHANICAL ENGINEERING
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREES OF

OCEAN ENGINEER
and
MASTER OF SCIENCE IN MECHANICAL ENGINEERING
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1982

A FURTHER DEVELOPMENT OF THE

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

COMPUTER AIDED DESIGN EXECUTIVE SYSTEM

by

DAVID LEE STONE

Submitted to the Department of Ocean Engineering
on May 07, 1982 in partial fulfillment
of the requirements for the Degrees of
Ocean Engineer
and
Master of Science in Mechanical Engineering

ABSTRACT

The MIT Design Executive System, which is referred to as DEX, is a design oriented, structured interactive computer system. DEX provides the designer with a flexible but controlled environment in which to interact through DEX with program modules, databases, and the operating system. The structured design environment is provided through the use of menus.

A further development of the MIT Computer Aided Design EXecutive system provides truly dynamic loading of program modules at execution time. A dynamic storage capability is added along with the capability to determine file status and dynamically allocate files during execution of DEX. In the event of an abnormal termination of execution, this condition is trapped and any open database is saved.

Improvements are added to the present DEX database system, with compression of the database after delete. Backward pointers are added, and a proposal for a dynamically extendible database is provided.

Thesis Supervisor: Chryssostomos Chryssostomidis
Title: Associate Professor of Naval Architecture

ACKNOWLEDGEMENTS

To my wife Nancy whose enduring patience and encouragement has kept me going through it all. A special appreciation is extended to my children, Michelle, Daniel, John, and Edward who always sent me off in the morning with "we love you Daddy". Also to my thesis supervisor Professor Chryssostomos Chryssostomidis whose encouragement, thought provoking questions, and recommendations opened the way to the insight neccessary to complete this project.

I would like to recognize the assistance provided by the staff of the Information Processing Center at the Massachussetts Institute of Technology. The assistance of many of these staff members in interpriting IBM documentation, and understanding system interaction is greatly appreciated.

I would like to extend a special note of appreciation to my good friend Bishop Melvin M. Scott, Jr. who has given me wise counsel and perspective, and who has often lifted burdens from my shoulders in time of crisis. Bishop Scott and his counselors Edward K. Abbott and Roger Brucks with whom I served in the Billerica Ward Bishopric of the Church of Jesus Christ of Latter Day Saints, have given me a great deal of support and strength in completing both my Church and Educational responsibilities. They are good friends whose support is greatly appreciated.

TABLE OF CONTENTS

List of Figures

CHAPTER 1

BACKGROUND AND THESIS ORGANIZATION

1.1 Background

In recent years the computer has come to the forefront of the design process, as numerous computer aided design systems have been developed. However, we as designers still do not realize the full potential of this powerful tool. There are several reasons for this. While the recent improvements in the user/ program interface have made the computer more straightforward to use, there are still several common problems which make the designer unwilling to accept the outlay of time required to change from one design program to another. Some of the common problems that these design programs, and the systems that contain them suffer from are:

1. Too often the programs are inflexible, and do not allow the user to deviate from the preconceived design process.

2. There is no consistent input/output procedure, and the output of one program cannot easily be used as input to another.

3. Because of the inconsistencies, the user must learn new procedures for each new program. This results in excessive training time requirements.

4. The program is often not forgiving, and after spending much time running the program an error can result in the loss of all output.

5. Inability of the programs to be transported to other facilities.

6. Often the programs are not user friendly. That is they were not designed with the ease of use as a major consideration in writing the program.

In 1974, researchers at the Massachusetts Institute of Technology and the University of Michigan joined together in a project to develop a structured computer aided design system that would eliminate or reduce the above characteristics. They named the system the Design EXecutive

System (DEX). The DEX system was written in a structured, top down manner, with each subroutine having a specific function. This was done to enhance transportability. If a given function was not available at a facility, then only the routine containing that function and the references to it would have to be changed. Also when transporting DEX, the few site dependent routines would be separate and could be easily changed without having to rewrite all of DEX. DEX can be adapted to almost any computer system that supports the Fortran programming language.

DEX provides an environment for running various application programs, called "modules". This provides ease of use and consistency by requiring all module programmers to interface with the user in the same manner. The DEX system was designed as a transactional manager that would insulate the user from both the module and the operating system, with DEX controlling all transactions. DEX consists of three levels or groups of programs:

1. DEX proper - The main body of DEX which provides the operating environment or "umbrella" of DEX, within which the user and the modules interact.

These routines provide the user with the consistent impression of the system.

2. EXtended DEX library - These are a collection of 45 utility subroutines and general functions which can be used by the module writer to facilitate the construction of the module. They are already set up to communicate with DEX properly. They include routines for reading, writing, editing data, unit conversion, and outputing messages and status indications.

3. Module - A module is a single subroutine or a complete set of subroutines written and executed together to perform a specific task. This is the actual application program which performs the calculations that the user is interested in.

The DEX system will be explained in more detail in Chapter 2. Celotto in his thesis [Celotto, 1981] gives the references for the earlier work on DEX, and gives an excellent example of how to write a module and how to run the DEX system. He also gives a detailed description of the Extended DEX library which was primarily his work.

1.2 Purpose

The purpose of this work is to make a further development in DEX proper. The first area of emphasis was in the DEX assembly language routines. While DEX is written primarily in Fortran for transportability, there are some system functions that were not available to DEX through Fortran at the Massachusetts Institute of Technology. It thus became neccessary to write a few functions as site dependent IBM/370 assembly language routines. These functional areas were:

- User identification.
- System time.
- Enhanced file management routines.
- Enhanced dynamic loading and unloading of modules.
- Enhanced dynamic allocation of memory.
- Enhanced error recovery with recovery from abnormal termination.

This was my major area of emphasis.

The second area of emphasis was in the DEX database editing routines. To the existing database management system in DEX the following capabilities were added:

- Enhanced delete function.
- Bi-directional pointers.
- Compression of the database (elimination of deleted entries) when the database is full or at the users request.
- Expansion of the database array buffer when it is full through the use of dynamic memory allocation.

1.3 Preview of Things to Come

Chapter 2 provides further detail on the DEX system. Chapter 3 will look at the assembly language utility routines, and the DEX philosophy of file management, dynamic loading, and dynamic memory allocation. Chapter 4 will give

further detail on the DEX database system, what it was, what it is now, and how it should be developed in the future. Chapter 5 will give conclusions and recommendations for the future. Appendix A is an assembly language programmers quide, written for the user who desires to write or modify DEX assembly language routines. Appendix B is a sample session with DEX's database editor. Appendix C is the program listings for the assembly language routines addressed in Chapter 3. And finally, Appendix D is the program listings for the database routines discussed in Chapter 4.

CHAPTER 2

THE DESIGN EXECUTIVE SYSTEM (DEX)

2.1 The DEX System

The DEX System is a highly structured interactive Design Executive program which manages all interactions between: 1.) the user, 2.) application program "modules" operating within DEX, 3.) DEX databases in memory, 4.) other sources and destinations of data, and 5.) the operating system. The design philosophy of the DEX System is discussed in the next section. DEX provides a major control structure utilizing menus to communicate with the user. When in the DEX environment the user is prompted to enter an item from one of the DEX menus shown in figure 2.1 indicating the action that the user desires to be performed. The menu DEX.MAIN is the major DEX control menu, and contains the major DEX commands. DEX begins its interaction with the user by prompting for an item from this menu, and returns to this menu after the completion of the requested action or actions.

The menu DEX.DISP is reached by entering the menu item DISPLAY from the menu DEX.MAIN, and is used to display at

```
 $ENTER AN ITEM FROM MENU - DEX.MAIN
display menu all
```

	MENU DEX.DISP	MENU DB-TYPES	MENU DBEDCMDS	MENU DEX.ALTR	MENU DXYES-NO	MENU DEX.MAIN
1	MENU	INTEGER	CREATE	TERSE	YES	LIBRARY
2	NEWS	REAL	STORE	VERBOSE	NO	HELP
3	MODE	ARRAY-RL	DELETE	KEYBOARD		DISPLAY
4			COMMENT	GRAPHIC		ALTER
5			EXPLAIN	ECHO-ON		TIDY
6			PRINT	ECHO-OFF		OPEN-DB
7			DUMP	DONE		EDIT-DB
8			SET-TITL			CLOSE-DB
9			GET-TITL			BEGIN
10			DONE			CONTINUE
11						SYSTEM
12						QUIT-DEX

Figure 2.1 The DEX Menus

the terminal a menu or list of menus, DEX news, or the current mode. The system can be operating in either the DEX mode or a module mode. After the item is displayed control is automatically is returned to the menu DEX.MAIN. The menu DEX.ALTR is reached by entering the menu item ALTER from the menu DEX.MAIN, and is used to alter the state of DEX. An example might be to change from verbose to terse messages. This menu unlike DEX.DISP maintains control until the user has entered all desired menu items. In this case control is not returned to DEX.MAIN until the user enters the menu item DONE.

The menu DBEDCMDS contains the commands used to edit a DEX database. The menu DBEDCMDS is reached by entering the menu item EDIT-DB from the menu DEX.MAIN. The database edit commands will be discussed in Chapter 4. The menu DB-TYPES is used by the database edit routines to obtain the type of variable to be entered in the database. The menu DXYES-NO can be used by any routine to obtain a yes/no answer to a question.

The DEX.MAIN menu item LIBRARY displays a list of the modules currently in the DEX library. The menu item HELP displays a help file written by the module author to explain

the operation of a given module. The menu item TIDY is used to clear the graphics screen. The menu items OPEN-DB, EDIT-DB, and CLOSE-DB are used for manipulating databases, and will be discussed in Chapter 4. The menu item BEGIN is used to begin execution of a module. Upon completion of the module control is returned to the DEX mode. The menu item CONTINUE is used to resume execution of a module after execution of the module was interrupted by a user request to temporarily return to the DEX mode. The menu item SYSTEM is used to temporarily exit to the operating system, and the system command RETURN is used to return to DEX. Finally the menu item QUIT-DEX is used to terminate the DEX session.

2.2 The Design Philosophy of the DEX System

The design philosophy of the DEX system was to provide the user with the maximum flexibility, while still maintaining consistency through control of all interactions. Celotto in his work [Celotto, 1981] pointed out that "there are five characteristics of DEX which reflect the design philosophy of the system:

1. The user is in the design loop.

2. The system allows the design process to be executed in more than one sequence.

3. The system talks with the user in plain English.

4. The system is forgiving.

5. The system has multiple capabilities for input and output."

Let us explore each of these important characteristics.

2.3 The User In the Design Loop

The design process is iterative in nature, and there is not neccessarily one correct flow through the procedure. Computer programs allow the relatively quick completion of complex and time consuming calculations, but often only in the sequence preconceived by the program writer. DEX allows the user to be in the design loop so the user can choose the sequence in which individual modules are executed. This is accomplished through dynamic loading and starting of modules. Additionally the user may choose to modify the

output of one module before it is input to another module or written to its destination. Thus the user can control the sequence of flow through the design spiral, and modify or insert additional data between steps.

2.4 Multiple Design Sequences

The flexibility offered by DEX is enhanced by the use of menus to allow the user to control the flow of execution. Thus the user has a wide choice of paths to follow through the design process. A menu is a list of options from which the user can choose the next step in the process, or the next variable to be defined. At present DEX allows a maximum of twelve items per menu, and a maximum of 25 menus. At any time the user can make a request to return to DEX from a module by typing the $ sign followed by the name of the desired menu and menu item. Menu names and menu items can be up to eight characters long with no embedded blanks.

If the user is in the DEX environment, then DEX will prompt to enter an item from a DEX menu. If in the module environment, that is if the user has told DEX to begin a module, then the user will be prompted to enter an item from

one of the module menus depending on the point of execution of the module. If the user doesn't know the items in the menu, the menu items can be displayed by typing:

$ display menu menuname

Where $ indicates a request to return to DEX, display is a menu item from menu DEX.MAIN, menu is a menu item from menu DEX.DISP, and menuname is the name of the menu that you desire to be displayed. After reviewing the menu the user types:

continue

which is an item from menu DEX.MAIN, and the user is returned to the prompting message for the menu item. The menu items are of three different types:

1. Information - such as data needed for a module's execution

2. Commands - such as BEGIN for begin calculation, or DONE indicating the desire to terminate execution of this module.

3. Other Menus - such as INPUT implying proceed to the menu INPUT which might have as its items the possible sources of input.

2.5 DEX Communicates with the user in plain English

The messages and prompts which DEX gives the user are written in English sentences, and are designed to be as clear as possible. The responses which the user gives are items from the various DEX or Module menus. These menu items are also English words designed to be a logical response to the queries from DEX. Also the user can string together a series of menu items into a sentence which will cause DEX to match each word with the appropriate menu, resulting in a sequence of actions. Thus the interaction is very much like an oral dialogue. This makes the DEX system easier to learn as the dialogue is consistent.

2.6 DEX is Forgiving

Because of the complexity of DEX, the occurrence of errors is inevitable. Whether it be simply typing something

that DEX does not expect, or the attempt to load a program that does not exist. The designers of DEX tried to anticipate as many errors as possible. And where possible, diagnostic messages in plain English were included in the DEX message pool. Where possible, DEX advises the user of the error and then allows another try at the same place. If this is not possible DEX will return control to a previous routine or menu or back to DEX itself. In the event of a fatal error this investigator has added the capability to intercept the abort to the operating system, returning control to a DEX fatal error handling routine. This routine announces the problem to the user, saves any open database, and gives the user the option of terminating execution (recommended since the status of the system is unknown), or continuing execution until things have deteriorated beyond recovery.

2.7 DEX has Many Sources/Destinations for Information

In Celotto's work [Celotto, 1981], he used the term "information" rather than input or output, because the terms "input", and "output" are too limiting in concept. He rather talked of "sources" of information, and "destinations" of

information. This writer will adopt the same convention. DEX enables the communication of information by the dynamic allocation of databases and files. DEX distinguishes between two types of files, a.) databases, and b.) disk files. In addition to these two types of files, DEX can communicate information to and from the terminal (or plotter) in the form of alphanumeric characters or graphics. The terminal is the destination for DEX messages, and the source for menu entries by the user. The DEX operating environment can be seen to have five sources of information and four destinations; they are:

1. DEX - created and edited databases which can be saved on or loaded from disk.

2. The user at the terminal using DEX utility routines to read or write alphanumerics.

3. The user at a graphics device using DEX graphics routines to read or create x-y co-ordinate plots, or other graphical data. (This capability has not yet been implemented in the present version of DEX at MIT.)

4. Sequential disk files.

5. Module default data (source of information only).

The reader is referred to the work by Celotto[[Celotto, 1981] for a more detailed description of how to run DEX, and of the Extended DEX library of utility routines. A description of the DEX database system will be given in Chapter 4.

CHAPTER 3

THE DEX ASSEMBLY LANGUAGE UTILITIES

3.1 General Description

As previously stated the philosophy of DEX is to provide the most flexibility to the designer using DEX, while still maintaining uniformity of dialogue and consistent information transfer. As a result DEX is a complex but structured program. Since the user is in the design loop, DEX has no way of knowing in advance the path that the user will follow through the design sequence. For this reason it cannot be determined in advance which files the user will need access to, or which modules the user will need loaded. Additionally the user may start using a database which might become full, and need to be expanded.

As a result, if all of the possible combinations were attempted to be provided for when DEX was compiled, the size and cost of DEX would be prohibitive. It would not be feasible to attempt such an undertaking, even if the resources were available. For this reason there are several functions which are essential to the dynamic nature of DEX. They are:

1. Dynamic file management.

2. Dynamic loading and unloading of modules.

3. Dynamic allocation of storage.

4. Error recovery.

Additionally there are several cosmetic or accounting capabilities that would enhance the DEX environment. These are:

1. Obtaining time from the system,

2. The obtaining of a users ID for communication and accounting of DEX use.

3. The ability to interrupt DEX, go to the system to perform some function, and then return to DEX at the point where you stopped.

4. The ability to shift the contents of a memory location either left or right.

However, these types of functions are generally not available to the IBM/370 Fortran programmer, and therefore it was necessary to develop them in assembly language at MIT. The description of each of these capabilities are given in the subsequent sections. The code listings for the assembly language routines are included in Appendix C.

3.2 Dynamic File Management

In the traditional program organization the input/output files must be allocated to the logical unit number either before the program is executed or in an "open" statement in the program. This results in a fixed file structure, with the unit number being allocated to one file and one file only for the entire execution of the program. In the DEX System, it is unknown which modules the user will invoke, and therefore what files will need to be allocated to which units. DEX requires the dynamic allocation of files at execution time, in order to provide the capability to change file allocation at any time. Also it is necessary for DEX to be able to check the existence of a file before attempting to allocate it. DEX needs to be able to CLOSE files after use and free the allocation, so that the unit can be

used for other files. Before allocating a file DEX must be able to determine if a file is already attached to the desired unit, and if so what file. The following is a list of the assembly language routines used for dynamic file management:

- ALLOC - Allocate a file for read or write. DEX allocates logical unit number (LUN) 18 as output device (OUTDEV), and LUN 19 as input device (INPDEV). Other assignments of LUN used by DEX are:

Messages device	MSGSDV=13
Usage device	USEDEV=14
Information device	INFODV=17
Database device	DBSDEV=12
News device	NEWSDV=15
Help device	HELPDV=16

- CKFILE - Check for the existence on the disk of a given file by its name.

- CLOSE - Close an open file.

- FREE - Free an allocated file.

- QUERY - Query what file if any is allocated to a given logical unit number (LUN).

The routines ALLOC and CKFILE were existing routines modified by this investigator. The routines CLOSE and FREE were existing routines which were not modified. The QUERY routine is a new routine developed by this investigator. The description of these routines is given in more detail below.

ALLOC The routine ALLOC is a combination of the previous routines ALLOCW and ALLOC. It is used to allocate a fortran I/O file to a given unit number. A third argument has been added, which is a WFLAG (write flag) which if .TRUE. tells the routine to allocate the file for write with the pointer at the end of the file. ALLOC is an Integer Function. The calling sequence is,

RET = ALLOC(LUN,FILENAME,WFLAG)

Where, RET is assigned the integer value zero if the file was successfully allocated, and a non zero return code if the file could not be allocated. The current version of the MIT IBM operating system returns the

value 24 for all the error conditions, so that they cannot be distinguished.

LUN is the Logical Unit Number which is the unit to which the module or DEX is writing or reading from.

FILENAME is a 24 byte string containing the CMS name of the file, segregated into 8 byte pieces. If the filename is given as a '*', then the terminal is allocated.

WFLAG is a flag which indicates that the file is to be allocated for write when it is .TRUE..

CKFILE The routine CKFILE is a logical function which checks the existence of a given file. This routine uses the FSSTATE macro which is a Conversational Monitor System (CMS) macro which checks the existence of a file. Error handling was added so the routine returns an RCODE to indicate the reason for a failure to find the given file. The calling sequence is,

```
RET = CKFILE(FILENAME,RCODE)
```

Where FILENAME is an 18 character string containing the full filename.

RET is a logical variable which is set .TRUE. if the file exists, and .FALSE. if the file does not exist.

RCODE is an error return code of integer type. The meaning of the return codes are,

RCODE = 0 No error

RCODE = 1 Invalid character in FILEID

RCODE = 2 Invalid FILEMODE

RCODE = 3 File not found

RCODE = 4 Disk not accessed

CLOSE The CLOSE routine was not modified by this investigator, but is included here for completeness. This routine closes a file for fortran to allow for file cleanup after use. The calling sequence is,

```
CALL CLOSE(LUN,&NOTGOOD)
```

Where LUN is the Logical Unit Number to which the file to be closed is attached (the LUN is in full word binary)

&NOTGOOD is for a fortran RETURN I convention. It is the label number of the line in the fortran routine which called CLOSE where control will be returned if the CLOSE is not successful. This is accomplished in assembly language by returning a 0 in register 15 if the normal return is to be taken, and a 4 in register 15 if the &NOTGOOD return is to be taken. For example;

```
          CALL CLOSE(18,&33333)
          RCODE = 0
          go to 99999
    C...     The file could not be closed
    33333 RCODE = 1
            .
            .
    99999 RETURN
          end
```

FREE The FREE routine was not modified by this investigator, but is included here for completeness. This routine frees

the file from the given LUN. The present version ignores the filename and frees whatever file is attached to that LUN. The calling sequence is,

CALL FREE(FILENUM<,FILENAME>,&NOTGOOD)

Where FILENUM is the LUN in character format

FILENAME is the name of the file to be freed; it is currently ignored.

&NOTGOOD is for the fortran RETURN I convention, and is as described in the CLOSE routine description above.

QUERY The QUERY routine was developed by this investigator. The QUERY routine makes use of the RDJFCB (Read Job File Control Block) system macro, and the IBM CMS (Conversational Monitor System) macros CMSCB and DEVTYPE [IBM, 3,4,5]. When passed a fortran I/O unit number (UNIT), this routine reads the system Job File Control Block (JFCB) for FTNNF001, where NN = unit number. The JFCB contains information about any file which is attached to that unit number. The CMSCB macro

(CMS Control Block) is then used to map the JFCB. The CMSCB macro is a series of labeled fields with the same length as the fields in the File Control Block. These labels can then be used to extract data from the JFCB. The FCBREAD field is checked to see if a file was attached. If none is then a .FALSE. is returned. If a file is attached a .TRUE. is returned, and the FILENAME, FILETYPE, and FILEMODE are extracted from the JFCB and returned. The CMS DEVTYPE macro is invoked to determine the device type, and this is returned. The calling sequence is,

```
TRUFLG = QUERY(UNIT,RCODE,NAME,TYPE,MODE,DEV)
```

Where UNIT is the fortran LUN for which information is desired.

RCODE is a return code which is not currently used. It was included as an argument in the event that later changes in the system control block structure necessitated a return code to identify the reason for failure of the query.

NAME is the filename returned.

TYPE is the filetype returned.

MODE is the filemode returned.

DEV is the device to which the unit is attached and is returned to the calling routine as an 8 character device. The possible device returns are,

PRINTER
READER
TERMINAL
TAPE
DISK
PUNCH
CRT

If no file is attached to the UNIT then NAME,TYPE,MODE, and DEV are set to zero.

3.3 Dynamic Loading and Unloading of Modules

The dynamic loading and unloading of modules is an important feature of DEX, and is an integral part of

the system, providing much of DEX's flexibility. In the previous version of DEX at MIT dynamic unloading was not implemented, and dynamic loading was simulated by reserving a very large storage area at the time DEX was initialized. Then when a module was loaded it would always be loaded at the start of this area. This allowed modules to be dynamically loaded, but caused several problems. First it was necessary to waste storage, because the area had to be sufficiently large to hold the largest routine in the DEX library. Additionally if two modules loaded were of different size and happened to use the same name for some of their routines, linkage conflicts could arise. This investigator developed the following assembly language routines to provide truly dynamic loading and unloading of modules. These routines make use of the OS/VS1 supervisor services macros. [IBM,6,7,8]

- LOAD - Dynamically load a module into memory and return the entry point.

- START - Dynamically start a loaded module.

- UNLOAD - Unload a previously loaded module.

These routines will now be described in greater detail.

LOAD The LOAD routine loads a module into main storage and returns the entry point address. It uses the OS/VS1 Supervisor 'Load' macro [IBM,7]. LOAD assumes the filetype is TEXT. LOAD calls the routine CKFILE with the filename to be loaded and a filetype of TEXT, to check that the file to be loaded exists. LOAD is a logical function, and if the load is unsuccessful a .FALSE. is returned. If the module is loaded, it's entry point is returned and LOAD is .TRUE.. This routine is used in Appendix A as an example. The calling sequence is,

```
TRUVAL = LOAD(FILENAME,ENTRY,RFCODE)
```

Where FILENAME is an 8 character filename, and a filetype of text is assumed.

ENTRY is the returned entry point address where the module was loaded.

RFCODE is a fatal return code which is set to one prior to entry. If no fatal error occurs the RFCODE is reset to zero. However, if a

fatal error occurs then execution terminates abnormally and RFCODE is still equal to one. This indicates to the DXABND routine that the error occurred in LOAD.

The concept of a fatal return code introduced above, was introduced by this investigator to identify the routine in which a fatal error occurred. The fatal return code (RFCODE) is passed in common to the DEX abnormal end routine (DXABND). The routine DXABND will only receive control if an abnormal end has occurred. The value of the RFCODE when DXABND receives control indicates the routine where the error occurred. Prior to calling any routine where a fatal error could occur RFCODE is set to the value assigned to that routine. Thus if a fatal error occurs control will not return normally but will go to DXABND with the RFCODE still set. If the routine terminates normally then the RFCODE can be reset to zero. Currently only four RFCODES have been assigned. It was decided that further assignments of RFCODEs could be made as experimentation with DEX indicated the need. The current assignments are:

RFCODE = 0 No Fatal Error

RFCODE = 1 Logical Function LOAD

RFCODE = 2 Subroutine START

RFCODE = 3 Logical Function FREEMN

RFCODE = 4 Module program

START The START routine when passed a module entry point (which was obtained using LOAD) passes control to that routine, and then returns control to the calling routine after the module has completed. In other words, it provides linkage between DEX and the module program. The calling sequence is,

```
CALL START(ENTRY,RFCODE)
```

Where ENTRY is the entry address for a module as obtained from the LOAD routine.

RFCODE is a fatal return code which is set to two prior to calling this routine. If no fatal error occurs then RFCODE is set to zero before returning to the calling routine. If a fatal

error occurs then execution is terminated abnormally, and the RFCODE is still equal to two. This then indicates to the DXABND routine that the error occurred in START.

UNLOAD The UNLOAD routine deletes a previously loaded module from main storage. The UNLOAD routine is a logical function and returns UNLOAD = .TRUE. if the unload is successful. If the indicated module can't be found then UNLOAD is returned as a .FALSE.. The UNLOAD routine uses the OS/VS1 Supervisor 'DELETE' macro [IBM,7]. The calling sequence is,

```
TRUVAL = UNLOAD(FILENAME)
```

Where FILENAME is an 8 character name of a module loaded using the LOAD routine.

No fatal return is necessary as there is no fatal error condition.

3.4 Dynamic Allocation of Storage

The traditional storage approach is that an array must be dimensioned statically to the worst case size. This is not only inflexible, but is wasteful of storage. In the DEX System the dynamic allocation of storage is essential to the very philosophy of the database, and it is expected that the allocation of storage should expand and contract with the data stored. This is important if DEX is to be a truly flexible design tool.

- FREEMN - Free previously allocated storage.

- GETMN - Get a block of main storage and return its entry point.

- READEC Read a block of memory at a location offset from the beginning of an existing memory area.

- WRITEC Write a block of data to memory at a location offset from the beginning of the memory area.

The routines FREEMN, and GETMN were developed by this investigator to provide the dynamic storage allocation capability in the MIT version of DEX in a manner that would facilitate the future development of a dynamic database. The previous version was locked at 2000 words of main storage for the database and there was no capability to free the storage and get an enlarged storage area. The routines READEC and WRITEC were existing routines based on the previous implementation which simulated the Control Data Corporation (CDC) ECS storage scheme, but with a fixed amount of storage. The basic structure is still the same, but in the current version the READEC and WRITEC routines were modified to take the pointer and length information describing the storage area as arguments. The routine GETMN is called to get a block of storage of any size up to the maximum size of 16 million bytes. However, the system has imposed a currently authorized limit of 1 million bytes on the DEX account. This limit can be raised but as it is raised the system is slowed, and limits are placed on the use of DEX. The pointer to the storage area is returned and placed in common. This information is then used to manage the storage area, allowing a new area to be obtained, the old copied, and then the old released using the FREEMN routine.

FREEMN The routine FREEMN frees a block of storage via the system FREEMAIN macro. The number of words of storage and the pointer to the area to be freed are passed as arguments. The calling sequence is,

```
FREFLG = FREEMN(NWORDS,POINTR,RFCODE)
```

Where NWORDS is the number of words of storage to be freed.

POINTR is an integer pointer to the start of the main storage area to be freed.

RFCODE is a fatal return code which is set to three prior to entry. If no fatal error occurs the RFCODE is reset to zero. However, if a fatal error occurs then execution terminates abnormally and RFCODE is still equal to three. This indicates to the DXABND routine that the error occurred in FREEMN.

FREFLG will be .TRUE. if the area is successfully freed. The only reason that it would fail is if the size exceeds the allowed storage

size. This would cause an abnormal end which is trapped for.

GETMN The routine GETMN acquires a block of storage via the system GETMAIN macro. The number of words of storage to get is passed as an argument. A pointer to the area is returned. The virtual storage area obtained by the GETMAIN macro begins on a doubleword boundary (see Appendix A-3.1), or a page boundary. The area is not cleared to zero when it is allocated. This routine does not need a fatal RFCODE because the GETMAIN macro is invoked with a conditional code. If the storage size requested exceeds the allowed limit then the GETMAIN is ignored, and GETMN is returned as a .FALSE.. The calling sequence is,

```
GETFLG = GETMN(NWORDS,POINTR)
```

Where NWORDS is the number of words of storage to be acquired.

POINTR is an integer pointer to the start of the main storage area.

READEC The routine READEC reads a block of storage from the simulated CDC Extended Core Storage (ECS) area obtained by the GETMN routine. The block of storage is read into a string (symbolic name) starting at the first byte. The block begins at a given offset from the start of the ECS area. This routine is used to read a stored array or comment from the ECS area. Comments are known to be of fixed length 64. Arrays are stored with the length of the array in the first location and the n elements of the array in the next n locations. The calling sequence is,

```
CALL READEC(STRING,OFFSET,LENWDS,ECSPTR,ECSLEN)
```

Where STRING is the symbolic name of the array where the words of data are stored as they are read.

OFFSET is the number of words that the desired block is offset from the beginning of the ECS area pointed to by ECSPTR.

LENWDS is the number of words of storage to be read (length in words).

ECSPTR is the pointer to the beginning of the ECS area. ECSPTR is returned by GETMN when it acquires the ECS area.

ECSLEN is the length of the ECS area and is used to check that the request is not an attempt to access beyond the area. If it is the request is ignored and a return code of 4 is returned in register zero. This check is included to make the routine general, but is presently redundant as DEX makes a check prior to calling READEC. The check is made in the routine DBAPTR that the NEXECS plus LENWDS does not exceed MAXECS. If it does DBAPTR is set .FALSE., and upon return to DBEDIT, the routine DBXECS is called to expand the ECS area.

The READEC routine reads a block of storage LENWDS long starting at ECSPTR plus OFFSET into STRING. The sequence of action when using READEC to read a database array from main storage would be; first the pointer to the array relative to the start of ECS would be obtained from the database. Then READEC would be called with that offset and a length of one. The value returned would be the length of the stored array.

READEC would again be called, but with an array as the string location, OFFSET+1, and the length of the array just read with the previous READEC call.

WRITEC The routine WRITEC writes a block of storage into the 'ECS' storage area obtained by the GETMN routine. A block of storage is taken from string and stored in the ECS area. The block is stored at the given offset from the start of the ECS area. This routine is used to write an array or comment into the ECS area. Comments are known to be of fixed length 64. Arrays are stored with the length of the array in the first location and the n elements of the array in the next n locations. The calling sequence is,

```
CALL WRITEC(STRING,OFFSET,LENWDS,ECSPTR,ECSLEN)
```

Where STRING is the symbolic name of the array whose contents are to be moved into the ECS area.

OFFSET is the number of words that the block is to be offset from the location pointed to by ECSPTR.

LENWDS is the number of words of storage to be written (length in words).

ECSPTR is the pointer to the beginning of the ECS area. ECSPTR is returned by GETMN when it acquires the ECS area.

ECSLEN is the length of the ECS area and is used to check that the request is not an attempt to access beyond the area. It is handled in the same manner as in READEC.

The WRITEC routine writes a block of storage LENWDS long from STRING into the ECS area starting at ECSPTR plus OFFSET. The sequence of action when using WRITEC to write a database array into main storage would be: First the next available ECS location would be determined from the variable NEXECS, then WRITEC would be called to write the length into the next available location. WRITEC would again be called to write a pointer to the previous block into the next location. A pointer back to the database entry would be written, and then WRITEC would be called with the array and LENWDS equal to the length of the array.

3.5 Error Recovery

As has been previously discussed, DEX is a very forgiving system. If the user enters a number in the wrong format, then DEX announces it and indicates which data items must be reentered. If the user is prompted for a menu item, and enters an incorrect response, DEX again indicates the error and allows the user to recover. However, there are some system functions which are not forgiving, and cause an abnormal termination, such as an error in a module which tries an illegal action. As a result a routine was developed to trap the abnormal end condition in the system and reroute it to a DEX fatal error handling routine.

- ABTRAP - Intercepts abnormal termination, and transfers control to the fortran routine DXABND (developed by this investigator). The routine DXABND announces to the user that an abnormal termination has occurred, and gives the user the option of saving any open database. It then gives the user the option of continuing or terminating the session.

The routine ABTRAP uses the CMS macro HNDSVC to set up a trap for a supervisor call SVC 13 which is the command used by the CMS operating system to branch to the system abend routine. The HNDSVC macro sets up the trap with an address where control is to be transferred anytime the SVC 13 command is issued. The address used for the trap is the label ABNORM in the ABTRAP routine. After the trap has been successfully set ABTRAP returns control to DEX and indicates successful completion by returning with ABTRAP .TRUE.. Then if at anytime later an abnormal condition arises, the operating system takes control and issues an SVC 13 to call the system abend routine. This condition is intercepted by the trap and rerouted to the location ABNORM in the routine ABTRAP. This section of assembly language code clears the trap to prevent a recursive abend sequence, and branches to the DEX routine DXABND. DXABND is a fortran routine which informs the user of the abnormal end condition, checks the RFCODE to determine which routine caused the error,provides the opportunity to save any open database, and then gives the user the option of terminating or continuing the DEX session.

The DXABND routine makes the least number of calls possible because of the uncertain status of DEX and the operating

system. The primary objective here was to save the database so that all would not be lost. However, if the user desires an attempt will be made to recover and continue operating. This action is not recommended because the user's virtual machine has been altered and the conditions will continue to deteriorate. The calling sequence is,

```
TRUFLG = ABTRAP(DUMMY)
```

DUMMY is a dummy argument to satisfy local fortran logical function convention that a function requires an argument.

3.6 Other Utility Routines

It would be beneficial for DEX to have several other capabilities. It would be useful to be able to identify the user's logon ID in order to keep an account of DEX users, and also to include the name in the welcome message.

Often it is convenient, when running a program, to return to the system to check something, or perform

some system function and then return to the program to resume execution.

Because of the large range of possible functions which could be performed under DEX, it would be useful to have the date and time indicated on the record of each DEX session so that the chronology of the design could be kept.

In assembly language coding it is sometimes necessary to convert from words of memory to bytes. This can be done by shifting the contents of the word to the left. Also some system macros return the results packed into a single register with an address portion and perhaps a length portion. This information can be more easily extracted by shifting the contents of the register to purge the undesired portion. Thus a routine which shifts a word of memory a given number of bits would be useful. The assembly language routines which provide these capabilities are:

- ISHIFT - To shift a word of memory left or right a given number of bits.

- JGVMID - To obtain the users Logon ID from the system.

- SYSTEM - To exit to the system and then be able to return to DEX after completing the desired function. It should be noted that ONLY transient system commands, and commands which are nucleus-resident are possible. They are:

The nucleus-resident CMS commands,

CP	GENMOD	START
DEBUG	INCLUDE	STATE
ERASE	LOAD	STATEW
FETCH	LOADMOD	

The transient system CMS commands,

ACCESS	HELP	RELEASE
ASSGN	LISTFILE	RENAME

COMPARE	MODMAP	SET
DISK	OPTION	SVCTRACE
DLBL	PRINT	SYNONYM
FILEDEF	PUNCH	TAPE
GENDIRT	QUERY	TYPE
GLOBAL	READCARD	

- TIME - To obtain the time of day and date from the system.

These routines were developed by Jeff Rice, except for JGVMID which was developed by J. Graham of the MIT Information Proccessing Service and are presented here for completeness. The calling sequence is,

```
CALL ISHIFT(WORD,NBITS)
```

Where WORD is the word to be shifted

NBITS is the number if bits to shift. If it is positive the shift is to the left if negative the shift is to the right.

CALL JGVMID(VMID)

VMID is the user ID who is currently logged on and using this program. This is the return argument from the system, and is 8 bytes in length.

CALL SYSTEM _ Branches to the CMS subset.

CALL TIME(STRING)

Where STRING is the symbolic name where the time and date are to be stored. The format is HH.MM.SS.TH YY/DDD. This routine uses the supervisor services macro TIME to get the system time, and then extracts the time/date information in the correct format.

CHAPTER 4

THE DEX DATABASE SYSTEM

4.1 General Description

The DEX Database System identifies entries in the database using the name of the variable or datum as a key. The variable's location in the database is not important. When the variable is placed in the database or later needs to be located for data manipulation, its address is calculated using a "hashing" algorithm. The hashing algorithm translates the name (sometimes called a key) into a number which is uniformly scattered or randomized. This number is then used to determine where the element is to be stored [Donovan, 1972; Martin, 1977]. This technique and the logical and physical database structure will be explained in section 4.2. The formal description of the overall database structure is called a "schema" [Martin, 1977].

4.1.1 The DEX Database Philosophy

The DEX Database System was designed to provide the user with the maximum flexibility in database manipulation. A new database can be constructed and used, or an existing database can be loaded and used, by either the module program, or the user using database manipulation menu items from the menus DEX.MAIN or DBEDCMDS. Opening a database means that it is opened in memory for manipulation. If it is a new database, then memory is initialized for it. If it is an existing database which is stored on disk, it will be loaded into memory if necessary. No database will be loaded if a copy of the database requested already exists in memory. After the database has been created and used, it will not necessarily be saved on disk. DEX will prompt the user to determine if the user desires that the database be saved. Databases can be manipulated by either a module program or the user in the DEX environment. The possible methods are:

1. A database can be manipulated from within a module program using the database management routines directly, or using the Extended DEX Library. The

Extended DEX Library was created to facilitate the module author in handling different sources and destinations of information including databases. The reader is referred to the work by Celotto [Celotto, 1981] which gives a description of the Extended DEX Library routines, and how to use them. The basic DEX database management routines are:

DBOPEN - Opens a database in memory. If the database is new memory is initialized for the new database. If the database already exists on disk it will be loaded into memory and used. DEX will not load a database if it is already in memory.

DBVINS - Insert a new variable name (key) into an open database in memory.

AGET - Retrieve real array values from the database in memory, and store them in the indicated array. (in all of the references to the database below the database is in memory)

IGET - Retrieve an integer value from the database for the given key, and store it in the indicated integer variable.

CMTGET - Retrieve the comment associated with a given key, and store it in the indicated variable. The length of the comment can be up to 64 characters.

RGET - Retrieve a real value from the database for the given key, and store it in the indicated real variable.

TGET - Retrieve the title of the open database, and store it in the indicated variable. The title can be up to 64 characters long.

APUT - Store the values of an array in the database

CMTPUT - Store the 64 character comment describing the datum (key).

IPUT - Store an integer value in the database for the given key.

RPUT - Store a real value in the database for the given key.

TPUT - Store a 64 character string into the database as its title.

DBVDEL - Removes a variable name(key) from an open database by making the NODE where it is stored unavailable to the user. The NODE is flagged for later compression of the database.

DBCLOS - Save the database on disk if the user desires, and close the database.

2. The user can manipulate a database, while in the DEX mode, using database commands from the menus DEX.MAIN and DBEDCMDS. The Database Edit Commands menu (DBEDCMDS) is reached by entering the DEX.MAIN menu item EDIT-DB. The menu items on the menu DEX.MAIN which involve databases, and the routine called are:

OPEN-DB - calls DBOPUT - The DB Open Utility routine checks to see if there is already an open database. If there is the user is given the option of saving it. If the user wants to use a different database a name is obtained, and the user is given the option of using a saved database. If a saved database is to be used it is loaded into memory. If a new database is to be created the database memory area is initialized.

EDIT-DB - calls DBEDIT - The routine DBEDIT prompts the user to enter an item from the menu DBEDCMDS. The Database Edit Commands are given below.

CLOSE-DB - calls DBCLUT - The DB Close Utility routine announces the name of the open database if one exists, and gives the user the option of saving the database with the same or a new name. If the database cannot be saved this is indicated, and the user prompted for a different name. The database is then closed.

The Database Edit Commands from the menu DBEDCMDS, and the routine called when that command is entered are:

CREATE - calls DBEDCR - Creates a node in the database for the given data name or key.

STORE - calls DBEDST - Obtains a value from the user for the given variable name or key, and puts it in the database at the previously created node. Valid types of data are integer, real, and one-dimensional real array.

DELETE - calls DBEDDL - Removes a variable name(key) from an open database by making the NODE where it is stored unavailable to the user. The NODE is flagged for later compression of the database.

COMMENT - calls DBEDCM - Puts a comment in the database describing the variable. For the present version of DEX where no provision has been made for units, it is desirable to include the

units in the comment. The comment can be up to 64 characters in length.

EXPLAIN - calls DBEDEX - Gets the variable's comment from the database, and displays it at the terminal.

PRINT - calls DBEDPR - Prints the value stored for the given data name.

DUMP - calls DBEDDM - Dumps the current contents of the database to the terminal.

SET-TITL - calls DBEDTS - Prompts the user to enter a database title of up to 64 characters in length, and stores it in the database.

GET-TITL - calls DBEDTG - Gets the title of the database from the database, and displays it at the terminal.

DONE - Returns control to the DEX major control routine DXMAJC which prompts the user to enter a menu item from the menu DEX.MAIN.

A very important feature of the EDIT-DB capability, is that the user can edit a database while in the DEX environment or mode, but outside of the module, so that the user can modify the results generated by a program, before proceeding to the next module used in the design sequence. The DBEDIT routines are a substructure of DEX proper, and they are reached from the DEX mode by entering the menu item EDIT-DB from the menu DEX.MAIN. If the database has not been opened, the user would first enter the menu item OPEN-DB from the menu DEX.MAIN to open the database. The menu item EDIT-DB is one of the DEX commands, and transfers control to the routine DBEDIT, which then prompts the user to enter a menu item from the menu DBEDCMDS, and asks for the name of the datum to be manipulated. The DBEDIT commands are listed above.

After the user has completed editing the database, and has returned to the DEX prompt "enter an item from menu DEX.MAIN" there are many paths that might be taken. If the user has finished using the database, then he would enter the menu item CLOS-DB from the menu DEX.MAIN. DEX will then announce the name of the open database, and ask if the user wants to save it. If the

answer is yes, the user will be given the option of changing the name. The database will then be written to the disk, and the file closed.

4.2 The DEX Database Organization

For the reader to understand the operation of the DEX database, and the further developments made by this investigator, it is necessary to present the formal description of the database structure. The overall view of the database is called its "schema". Martin discussed different levels of description of data: [Martin, 1977]

1. The Global Logical Database Description, or SCHEMA - This is the overall view of the logical layout of the data. This is the way that the data is normally perceived.

2. Physical Database Description - A description of the physical layout of the data, and how it is actually stored.

The DEX Database SCHEMA is represented in figure 4.1. This is the overall logical view of the data that would be seen at the terminal if the user issued the DBEDIT command dump, which would dump the contents of the database to the terminal. The physical representation of the data is shown in figures 4.2 - 4.7, and is described in the following sections.

4.2.1 The Physical Description of the DEX Database

The physical description of the DEX Database, or how the data is actually stored, will now be considered. The present version of DEX allows up to 200 variables to be stored in the database. These variables can be one of three types: integer, real, or real array. If the type is real array, then the number of elements in the array and the actual array values are stored in a separate ECS array storage area which is dynamically allocated, and initialized to 2000 words of memory. As the area fills it is expanded in increments of 2000 words. A pointer to the array's location is stored in the value field of the variable in the database.

```
$TITLE: **SAMPLE DATABASE

$  NAME     TYPE            VALUE  COMMENT

   NCREW     (I)               100  NUMBER OF CREW
   DIAMETER  (R)    **UNDEFINED**  HULL DIAMETER (SQUARE FEET)
   POWER     (A)     ARRAY(     7)
   SPEEDAR   (A)     ARRAY(     1)
   LENGTH    (R)      0.46000E+03

THE ARRAYS


$NAME: SPEEDAR , LENGTH=  3   (    1)
$   0.50000E+01   0.10000E+02   0.20000E+02
$NAME: POWER   , LENGTH=  3   (    7)
$   0.80000E+05   0.10000E+06   0.15000E+06
```

Figure 4.1 The Database SCHEMA, and Sample Data

Each individual array can contain up to 200 elements. However, once the array has been stored its length becomes fixed at the number stored, even if it is less than 200.

Figure 4.2 is a representation of the physical layout of the DEX Database.

The actual physical structure of the database is a series of arrays representing the fields of the schema. If the database is an already existing one then the values of the arrays represented in figure 4.2 are loaded from a disk file. If the database is new, then the values of the arrays are initialized in the routine DBINIT when the database is opened. The initial values will be explained below.

The KEY to the database (KEY is the field used to locate entries) is the variable name. The name is up to eight characters in length, and is stored in a two-dimensional array IDENT(I,J). The first four characters of the name are stored in IDENT(I,1), and the second four in IDENT(I,2). The value of the index I indicates the number of the node in the database. The IDENT array is initialized with blanks. The variable type is stored in the array NODTYP(I), which is initialized to all zeros. Next is the array DATUM(I), which will contain the value of the data element. If the variable is an array then the DATUM will contain a pointer to the location of the array in memory. The DATUM array is ini-

TITLE

NAVAIL	DELCNT	MAXECS	NEXECS	PRECS
1	0	2000	1	0

BUCKET	1	2	3	4	5	.....	31	32
	0	0	0	0	0	.....	0	0

NODE	IDENT1	IDENT2	NODTYP	DATUM	LINK	LINKF	CMTPTR
1			0	0	2	0	0
2			0	0	3	0	0
3			0	0	4	0	0
4			0	0	5	0	0
5			0	0	6	0	0
6			0	0	7	0	0
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
199			0	0	200	0	0
200			0	0	0	0	0

ARRAY STORAGE AREA

Figure 4.2 The physical Database Structure

tialized to zero. Next is the array LINK(I), it is initialized to the node number plus one. This is used initially to indicate the next available node in the database. It is later used to link together variables with the same hash value. This will be made clear in an example which will follow. Next is the array CMTPTR(I), which will contain a pointer to a 64 character comment which is stored in the array buffer. The arrays are stored in a separate area of main storage which is obtained dynamically at execution time. The database main structure then is made up of the arrays IDENT(I,1), IDENT(I,2), NODTYP(I), DATUM(I), LINK(I), and CMTPTR(I). The index I is dimensioned 200 in the present version of DEX. In addition to this main 200 node database body there is the auxiliary storage area where the comments and arrays are stored. Also there is a title array which allows for a 64 character title of the database to be stored. The addressing scheme uses a hashing function which converts the name of the datum into a number. The resultant hashed numbers are uniformly distributed between 1 and 32. For example, if the name of the datum was length, then the hashing function would yield the result;

HASH(LENGTH) = 32

The hashing function will be described in more detail in the next section. The result of the hash is used as a pointer to an element in a 32 element array BUCKET(I). The BUCKET is used to contain a pointer to the node in the database where LENGTH is stored. Length is stored in the next available node in the database which is indicated by the variable NAVAIL. NAVAIL is initially set to one. It is updated from the value of the LINK at the node which is currently the next available when the next datum is stored.

4.2.2 The DEX Hashing Function

The DEX hashing function uses the internal machine code values for each character in the KEY name, summing these codes up until the first blank is encountered. The resulting number is truncated using the modulo function to obtain a number between 1 and 32.

4.2.3 An Example of Editing the DEX Database

In this section, an example demonstrating the inserting of several nodes in the database will be given. An example

DEX database editing session is given in Appendix B, and shows the interaction with DEX which was required to enter the variables described in this example in the database. The sample session is given for information only, and will not be discussed as this example is a description of the physical structure of the database. The hashing values used in this example are not necessarily those that would actually result for these variables, but were chosen merely for discussion purposes. Figures 4.3, 4.4, and 4.5 will show how the database is changed as each entry is made. Figure 4.6 will show the effect of deleting an entry from the database. Initially the database is as shown in figure 4.2.

The first name to be entered is LENGTH. Its type is real. It is assumed for this example that for each datum entered DEX has already prompted the user for the datum name and type. This name is hashed and the routine DBHASH returns the value 32. The bucket of 32 is checked to see if there has been a previous entry. Since the value of BUCKET(32) is zero this is the first entry. Since NAVAIL is one, the next available NODE is NODE one. So LENGTH is entered in NODE one as follows: (see figure 4.3)

NODE=1

NAVAIL=LINK(1)=2

LINK(1)=BUCKET(32)=0

BUCKET(32)=NODE=1

IDENT(1,1)=LENG

IDENT(1,2)=TH

NODTYP=-2 The 2 indicates real type and the negative indicates that no data has been entered for this node.

LINKF=-32 The LINKF of the first or only entry in a chain is used to contain the negative value of the BUCKET as a BUCKET pointer. A negative number is used to distinguish the BUCKET pointer from a forward LINK to another NODE.

As a result LENGTH has been entered in NODE one of the database with a NODTYP of -2 to indicate real type with no value entered. The next available node has been set from the link of NODE one, and is NAVAIL=2. The LINK has been updated from the BUCKET to point to any other nodes with the same HASH value. In this case the LINK is zero indicating that there are no others. BUCKET(32) is now pointing to NODE one, and LINKF is pointing back to bucket 32.

TITLE

SAMPLE DATABASE

NAVAIL	DELCNT	MAXECS	NEXECS	PRECS
2	0	2000	1	0

BUCKET	1	2	3	4	5	.....	31	32
	0	0	0	0	0	.....	0	1

NODE	IDENT1	IDENT2	NODTYP	DATUM	LINK	LINKF	CMTPTR
1	LENG	TH	-2	0	0	-32	0
2			0	0	3	0	0
3			0	0	4	0	0
4			0	0	5	0	0
5			0	0	6	0	0
6			0	0	7	0	0
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
199			0	0	200	0	0
200			0	0	0	0	0

ARRAY STORAGE AREA

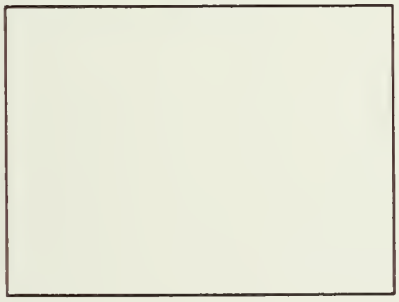

Figure 4.3 Length is entered in the database.

TITLE

SAMPLE DATABASE

NAVAIL	DELCNT	MAXECS	NEXECS	PRECS
3	0	2000	1	0

BUCKET	1	2	3	4	5	...	9	.....	31	32
	0	0	0	0	0	...	2	.....	0	1

NODE	IDENT1	IDENT2	NODTYP	DATUM	LINK	LINKF	CMTPTR
1	LENG	TH	-2	0	0	-32	0
2	DIAM	ETER	-2	0	0	-9	0
3			0	0	4	0	0
4			0	0	5	0	0
5			0	0	6	0	0
6			0	0	7	0	0
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
199			0	0	200	0	0
200			0	0	0	0	0

ARRAY STORAGE AREA

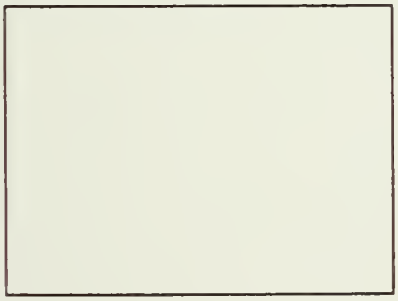

Figure 4.4 Diameter is entered in the database

TITLE

SAMPLE DATABASE

NAVAIL	DELCNT	MAXECS	NEXECS	PRECS
5	0	2000	1	0

BUCKET	1	2	3	4	5	...	9	.....	31	32
	0	4	0	0	0	...	3	.....	0	1

NODE	IDENT1	IDENT2	NODTYP	DATUM	LINK	LINKF	CMTPTR
1	LENG	TH	-2	0	0	-32	0
2	DIAM	ETER	-2	0	0	3	0
3	HEIG	HT	-2	0	2	-9	0
4	SPEE	DAR	-3	5	0	-2	0
5			0	0	6	0	0
6			0	0	7	0	0
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
199			0	0	200	0	0
200			0	0	0	0	0

ARRAY STORAGE AREA

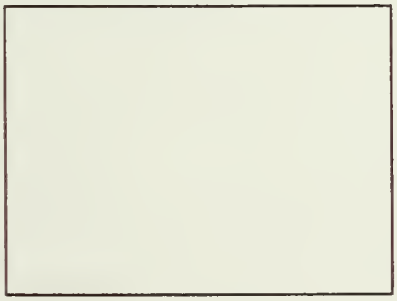

Figure 4.5 Speedar is entered in the database.

TITLE

SAMPLE DATABASE

NAVAIL	DELCNT	MAXECS	NEXECS	PRECS
5	1	2000	1	0

BUCKET	1	2	3	4	5	...	9	.....	31	32
	0	4	0	0	0	...	2	.....	0	1

NODE	IDENT1	IDENT2	NODTYP	DATUM	LINK	LINKF	CMTPTR
1	LENG	TH	-2	0	0	-32	0
2	DIAM	ETER	-2	0	0	-9	0
3	HEIG	HT	-9	0	2	-9	0
4	SPEE	DAR	-3	5	0	-2	0
5			0	0	6	0	0
6			0	0	7	0	0
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
199			0	0	200	0	0
200			0	0	0	0	0

ARRAY STORAGE AREA

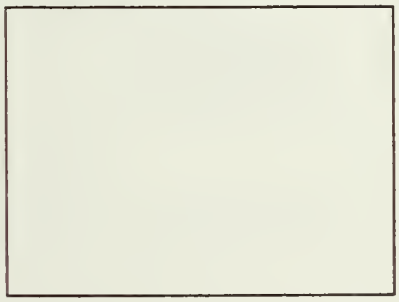

Figure 4.6 Height is deleted from the database.

TITLE

SAMPLE DATABASE

NAVAIL	DELCNT	MAXECS	NEXECS	PRECS
7	1	2000	13	1

BUCKET	1	2	3	4	5	...	9	.....	31	32
	0	4	0	0	6	...	2	.....	5	1

NODE	IDENT1	IDENT2	NODTYP	DATUM	LINK	LINKF	CMTPTR
1	LENG	TH	-2	0	0	-32	0
2	DIAM	ETER	-2	0	0	-9	0
3	HEIG	HT	-9	0	2	-9	0
4	SPEE	DAR	3	1	0	-2	0
5	NCRE	W	1	100	0	-31	0
6	POWE	R	3	7	0	-5	0
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
199			0	0	200	0	0
200			0	0	0	0	0

ARRAY STORAGE AREA

1	3
2	0
3	4
4	0.50000E+01
5	0.10000E+02
6	0.20000E+02
7	3
8	1
9	6
10	0.80000E+05
.	.
.	.

Figure 4.7 Array values are stored.

The next variable to be entered is DIAMETER, which the user has indicated is of real type. Diameter is hashed and the routine DBHASH returns a value of 9. Since NAVAIL=2, DIAMETER will be entered in NODE two as follows:

```
NODE=2
NAVAIL=LINK(2)=3
LINK(2)=BUCKET(9)=0
BUCKET(9)=NODE=2
IDENT(2,1)='DIAM'
IDENT(2,2)='ETER'
NODTYP=-2  Indicating real type.
LINKF(2)=-9
```

As a result DIAMETER has been placed in the database at NODE two. The next available node has been updated from the LINK of NODE two such that the NAVAIL now is three. The LINK(2) has been updated from the BUCKET(9), and is equal to zero. The NODTYP is -2 to indicate a real number will be entered. Since this is the only datum that has hashed to BUCKET(9) so far LINKF(2) is set to point to BUCKET(9).

The next variable entered is HEIGHT which is of real type. HEIGHT is hashed and the routine DBHASH returns a value of 9. When the BUCKET(9) is checked it is found that it contains a pointer to NODE two. This is a collision

since both items cannot occupy NODE two. Since NAVAIL=3, HEIGHT will be entered in NODE three, and will be connected to node two by the pointers LINK and LINKF as follows:

```
NODE=3
NAVAIL=LINK(3)=4
LINK(3)=BUCKET(9)=2
LINKF(2)=NODE=3
BUCKET(9)=NODE=3
IDENT(3,1)='HEIG'
IDENT(3,2)='HT  '
NODTYP=-2  Indicating real type.
LINKF(2)=-9
```

As a result HEIGHT is entered into the database at NODE three, and NAVAIL is set equal to four. This time when LINK(3) is set equal to BUCKET(9), the value 2 is placed in the LINK. A pointer to NODE three is now placed in BUCKET(9). Since both HEIGHT and DIAMETER hashed to BUCKET(9) they are linked together. Thus BUCKET(9) points to HEIGHT in NODE three, NODE three is linked to DIAMETER in NODE two by LINK(3)=2, and DIAMETER in NODE two is linked to HEIGHT in NODE three by LINKF(2)=3. HEIGHT is linked to the BUCKET(9), to which both were hashed, by LINKF(3)=-9.

The next variable to be entered is SPEEDAR, which is a real array with 5 elements. SPEEDAR is hashed and the routine DBHASH returns a value of 2. Since NAVAIL=4, SPEEDAR will be entered in NODE four as follows:

```
NODE=4

NAVAIL=LINK(4)=5

LINK(4)=BUCKET(2)=0

BUCKET(2)=NODE=4

IDENT(4,1)='SPEE'

IDENT(4,2)='DAR '

NODTYP=-3  Indicating real- array type.

LINKF(2)=-2

DATUM(4)=5   Indicating that 5 array elements will later
be stored.
```

As a result SPEEDAR is entered into the database at NODE four, and the NAVAIL is set to five. LINK(4) is updated from BUCKET(2) and becomes zero. BUCKET(2) is set to point to NODE four. The NODTYP is set equal to -3 which indicates that this entry is an array. The DATUM is set to the number of elements which the user expects to store in the array. When values are entered in the array, the actual number of elements that were entered will be stored followed by pointers to the previous stored array, and a pointer back to the

NODE in the database. Then the actual elements of the array will be stored following these.

At this point HEIGHT will be deleted from the database to show how this is done. The entry is deleted by setting its NODTYP=-9 The LINK of the deleted NODE replaces the LINK of the NODE pointed to by the LINKF of the deleted NODE. The LINKF of the deleted NODE replaces the LINKF of the NODE pointed to by the LINK. If the LINK is zero then this is the end of the chain. If the entry is the first entry after BUCKET, then the value of the deleted LINK is placed in BUCKET. The LINKF is handled as before except its value is the pointer back to the BUCKET. DEX knows that it is a pointer to the BUCKET and not to a NODE because it is negative.

In this example HEIGHT is deleted. So HEIGHT is hashed to BUCKET(9), which points to NODE three. The IDENT of NODE three is compared to the variable name, and since there is a match NODE 3 will be deleted. If there had been no match, each NODE in the chain would be searched for HEIGHT until a match occurred. The links are updated as previously described. The DATUM is set equal to zero, and the NODTYP is set to a minus nine to indicate the NODE has been

deleted. If the NODTYP is a +3 indicating that an array is stored, then the DATUM is set equal to -DATUM. Thus if the NODTYP=-9 and the DATUM is less than zero, then the absolute value of DATUM is the pointer to the array storage block which must be compressed. The array storage area compression capability has not been implemented yet, and is an area for future work. A CMTPTR is handled similarlly.

NODE=3

Since there is a LINK the LINKF of that NODE must be modified LINKF(2)=LINKF(3)=-9

BUCKET(9)=LINK(3)=2

DATUM(3)=0 If NODTYP=3 then DATUM(I)=-DATUM(I) so that the pointer to the array storage area is saved for later compression of the array storage area.

CMTPTR(3)=0 If CMTPTR is not 0, CMTPTR(I)=-CMTPTR(I)

LINKF(3)=0

NODTYP=-9 Indicating that this datum has been deleted.

It should be noted by the reader that at this point the variable names LENGTH, DIAMETER, SPEEDAR, and HEIGHT were created as NODEs in the database only, and then HEIGHT was deleted as a NODE in the database. That is to say that the NODEs are created, but no value has been stored for those datum names. To store a value, DEX prompts the user for the name, type, and value. The name is hashed to a BUCKET, and the NODE determined. If the NODE does not exist the user is

so informed. The NODE is checked to see if its IDENT matches the name, if it does then the value is stored in DATUM(NODE). If there is no match the next NODE in the chain is checked until a NODE is found with that name. Once the NODE containing the desired name has been located, its NODTYP is compared to the type indicated by the user. If the type does not match, then the user is informed and the user prompted for a new menu item.

Figure 4.7 shows the database at a later stage after two additional variables and data for several of the variables have been entered. NODE five contains NCREW with a NODTYP of one indicating an integer value. Since the NODTYP is positive a value has been entered for this NODE. NODE six contains a real array POWER. Values have been entered for the two arrays SPEEDAR and POWER. The DATUM for SPEEDAR points to location one in the array storage area, and the DATUM for POWER points to location seven in the array storage area. Looking at the array storage area, the reader will notice that location one contains a three indicating that three numbers are stored. The next entry is a pointer to the previous stored array. This value is zero in this case because this is the first stored array. The next number is a pointer to the NODE four, linking this array back

to the SPEEDAR entry in the database. This is followed by the three speeds which are the array entries. It should be noted that the user had originally said five numbers would be stored, but since only three were stored the size is now constrained to three, the number stored. The next number in the array storage is at location seven which is the beginning of the next array block. This number indicates that three values are stored in the array. Next is the pointer to the previous array which points to location one. This is followed by the pointer to NODE six to which this array belongs. The three array values follow in the next three locations (only one is shown). Looking at the figure 4.7 it is seen that DELCNT = 1 indicating one item has been deleted from the database. MAXECS = 2000 indicating that only one block of ECS storage has been obtained. NEXECS = 13 indicating that the next available location in the ECS area is at location 13. PRECS = 1 indicating that the previous array was stored beginning at location 1. The storage of comments is handled in a similar manner to the arrays.

4.3 Further Developments in the DEX Database

The DEX Database System as described in section 4.2 had the limitation of fixed size. It also lacked the capability

of compressing the database after delete or after some given number of deletes. The DEX Database System handled deletes by making the node available and setting the NODTYP = -9. This technique worked fine, except that it broke up the consistent upward flow of the LINKs, and if there were a large number of deletes would leave a lot of holes in the database. This would be detrimental to any future development of a dynamically extendible database. Another feature of the database is that it had pointers in one direction only. This made it more difficult to locate previous entries in the chain. The decision was made to modify the DEX Database System to be consistent with the possibility of future development in the flexibility of the Database System. To this end, it was decided to add the following capabilities to the already existing database.

1. Bi-directional Pointers - To provide the capability of easily reconnecting the chain of LINKs after a delete. Also to provide possible future flexibility in error recovery. This was done by adding to the physical database structure a forward link, LINKF(I).

2. Modified Delete Structure - The delete technique was modified so that the node is not made available after delete. The next node and previous node in the chain are connected using the bi-directional link structure, and the deleted node made unavailable by putting a -9 in the NODTYP. This provides a consistent link structure with the LINK always pointing backwards in the database, and the LINKF always pointing forward. This facilitates compressing the database.

3. Database Compression Feature - As the database fills up it will have holes left by the now unavailable deleted entries. Therefore, a routine (DBCMPR) was provided to compress the database when it is full or at the users request. This will also facilitate any future development in a dynamically extendible and contractable database.

4. Dynamic Expansion of the Array Storage Area - Utilizing the dynamic memory management routines GETMN and FREEMN, the auxiliary array storage area was made expandable. As the 2000 word storage area fills up, a larger block of storage is obtained.

The current array area is copied into it and the first area released.

4.3.1 The Pointer Structure

The DEX pointer structure has been modified to include the original backward LINK(I), with the addition of the LINKF(I). When collisions occur in the database, and several nodes with the same hash value are chained together, then the direction of the links is as follows. The node address of the last item entered is placed in the bucket, and its LINK points back to the previous node that was entered with that hash value. The LINKF points forward from the previous node to the current one. The links are propagated along the chain in the same manner. The LINKF of the last node entered contains the negative of the bucket address to which it hashed.

4.3.2 The Modified Delete Structure

The delete structure was modified so that it would preserve the clean structure of the links. When a node is

deleted it is not made available but is flagged with a -9 in NODTYP. This will be used by the DBCMPR (compression routine) routine. The next node in a chain and the previous node in the chain have their links joined together by using the LINK and LINKF of the deleted entry to tie them together.

4.3.3 The Database Compression Feature (DBCMPR)

The further capability to compress the database was developed so that as the database fills up, the database can be periodically compressed to open up any holes left by delete. The compress technique used was to search the database until a type of -9 was found. Then a count was updated to keep track of the number of deletes. The nodes were then moved up a number of positions equal to the count until the next -9 was encountered. As the Node is moved up its links were modified including any link to the bucket.

4.3.4 Dynamic Expansion of the Array Storage Area (DBXECS)

With the further development of the DEX dynamic storage allocation achieved by this investigation, it has become possible to begin the development of the routines to allow the database storage to grow with the data. This investigator developed a simple first step routine to expand the database array area as it becomes full. The routine was called DBXECS for Database Expand ECS area. The routine has the following calling sequence,

```
TRUVAL = DBXECS(IDUM)
```

The routine is a logical function with a dummy argument. The required variables are passed in the common DBDAIO which was modified to include the following variables:

MAXECS - is the maximum ECS storage size. It is initialized to 2000 words, and is incremented by 2000 each time the ECS area is expanded.

NEXECS - is a pointer to the next free location in the ECS storage area. (as previously used)

PRECS - is a pointer back to where the previous array is stored in the ECS area.

ECSPTR(2) - is an array containing the address of the old ECS area and the new ECS area.

ECSLEN(2) - is the array containing the length of the two ECS areas. (ECSLEN(1) = MAXECS - 2000, ECSLEN(2) = MAXECS)

The DBXECS routine is called as the array area is filled up. It acquires a new block of main storage whose size is now MAXECS+2000. It saves the current MAXECS and the previous MAXECS. It then copies from ECS area one to ECS area two, frees area one, and then saves the new pointer and maximums. It then returns to the calling routine.

The DBSAVE and DBLOAD routines have been modified so that the final MAXECS, the NEXECS, and the PRECS are saved in the database.

4.4 Future Developments in the DEX Database

The DEX Database System has been developed with the philosophy of the DEX System in mind. That is to provide the user the maximum flexibility while still maintaining consistency of structure, and uniformity of dialogue. The DEX Database System has been designed to use the usual DEX control structure and menu communication system. It has also been designed with its future growth in mind. There are two major areas that need to be considered by future investigators.

The first is to take the database the next step beyond the work of this investigator, and provide a dynamically extendible and contractable database that responds to the size of the database as it grows or items are deleted. This writer has taken a first step in this direction by providing the memory allocation routines, the capability to compress the database after delete (DBCMPR), the capability to expand the array storage area (DBXECS), bi-directional LINKs, and the enhanced delete capabilities described in the previous section. However, the current version still has a limit of 200 nodes in the main database structure. This limitation needs to be addressed, and a routine developed to make the

database completly expandible. Fagin in his work on "extendible hashing" for dynamic database management has suggested a technique[[Fagin, 1977] that provides a dynamic storage structure that grows and shrinks gracefully as the database grows and shrinks".

The technique chosen by this investigator was to expand the database by acquiring more memory. This does not depend on the use of direct access file management, but has a limitation imposed by the amount of memory required. In the current version of DEX the size of the array storage area is limited by the number of nodes in the database and the size of array, overhead, and comment for each node. This is not an insignificant amount of storage, but is less than the theoretically available sixteen million. If the number of NODES in the database is also made expandible in memory, then the required storage could become unmanageable. Another technique which was considered, and temporarily set aside because of its machine dependencies, is Virtual Storage Access Method (VSAM). This among other things allows direct access files on disk storage. With this procedure only the segments needed at any given time are in memory. Thus storage requirements would be kept at a manageable level. This would facilitate the use of Fagin's technique of hashing to

a segment of the memory and then to a node within the page. The database could then be made totally expandible and contractable without large storage requirements. Both techniques will be experimented with and perhaps a combination of the two will be the best approach. The second area is to provide an alerting capability which allows for a structure that chains dependent variables to other variables upon which it depends. Then alert the user when a change has occurred in an independent variable upon which this variable depends.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

The work of this investigation has provided a further enhancement of the already existing DEX capabilities, and has developed the structure necessary to achieve a truly dynamic database that can respond to the requirements of the complex design problem.

This work has made improvements in the area of dynamic file management which provide a fairly adequate file management structure. Additionally the area of dynamic loading and unloading of modules has been solved by this investigators development of the LOAD, START, and UNLOAD routines utilizing supervisor services macros. The dynamic storage allocation routines and database enhancements have taken the database capabilities a step closer to the concept of an extendible database system that can dynamically respond to the growth of the database.

There are several areas of DEX which still need development. First is the further development of the DEX database system to expand upon the work of this investigator to achieve a dynamically expandible database. Along this line,

the limit of 200 elements in the main body of the database must be addressed and the procedure developed to expand it. Future investigations will include experimentation with both dynamic memory allocation techniques to expand the database in memory, and Direct Access File Management techniques to expand the database in direct access files on disk, bringing only those segments needed into memory. It will be determined which of these is the best procedure, or if a combination of the procedures is the best approach to removing the database size limitations. The development of the dynamic storage allocation routines GETMN and FREEMN have provided the basic tools for this development, along with the modifications to the database system to make it upward compatible with the addition of a consistent bi-directional LINK structure. Finally with the database compress algorithm and the capability to expand the ECS storage area this work has taken the next step in the development of a truly dynamic database.

Second is the implementation of the graphics capabilities into the MIT DEX environment. This capability is the logical next step in the development of DEX as a design system. The decision has been made to pursue segmented graphics as the base for the MIT implementation of DEX graphics. This

will allow the use of segmented graphics routines already implemented in DEX at other installations.

In conclusion DEX then has been developed as a truly Design Executive with a strict adherhence to structured programming in its development. DEX follows the structured programming rule of one function for one routine with a consistent top down control structure. DEX allows the user to exercise as much or as little control over the design process as desired. With the dynamic file, memory, and module management capabilities coupled with the removal of restrictions from the DEX Database system and the future addition of a powerful graphics design capability will give the designer the maximum consistency and flexibility in the control of the flow of a design process.

REFERENCES

Celotto, Richard Charles, Lieutenant, U.S. Navy, An Investigation Into the Use of Databases in Computer-Aided Naval Ship Design, Thesis, Massachusetts Institute of Technology, Cambridge Mass., June 1981

Donovan, James J., System Programming, McGraw Hill Computer Science Series, McGraw Hill Book Co., New York, 1972

Fagin, Ronald, Jurg Nivergelt, Nicholas Pippenger, H. Raymond Strong, Extendible Hashing - A Fast Access Method For Dynamic Files, IBM Research Report RJ2305, July 1978

Herzog, Bertram, "A Transportable Fortran Based Executive System for Computer Aided Ship Design Education", Computer Applications in the Automation of Shipyard Operation and Ship Design II, Editors Jacobsen et al., North-Holland Publishing Company, Amsterdam, 1976, pp. 79-87

Herzog, B., Module Programmers Guide for the Interactive Computing System DEX at the University of Colorado, revised March 1978

IBM,1: IBM System/370 Principles of Operation, Form GA22-7000

IBM,2: OS/VS - DOS/VSE - VM/370 Assembler Language, Form GC33-4010

IBM,3: OS/VS1 Data Management for System Programmers, Form GC26-3837

IBM,4: OS/VS1 Data Management Macro Instructions, Form GC26-3872

IBM,5: OS/VS1 Data Management Services Guide, Form GC26-3874

IBM,6: IBM Virtual Machine/System Product: System Programmers Guide, Form SC19-6203

IBM,7: OS/VS1 Supervisor Services and Macro Instructions, Form GC24-5103

IBM,8: OS/VS1 Supervisor Logic, Form SY24-5155

IBM,9: System 370 Reference Summary Card, Form GX20-1850

Osborne, Adam, An Introduction to Microcomputers, Vol. I Basic Concepts, 2nd edition, Osborne/Mcgraw-Hill, Berkley Calif., 1980

Madnick, Stuart E., John J. Donovan, Operating Systems, McGraw Hill Computer Science Series, McGraw Hill Book Co., New York, 1974

Martin, James, Computer Data-Base Organization, Second Edition, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1977

APPENDIX A

ASSEMBLY LANGUAGE PROGRAMMING

A-1.0 Why Assembly Language

The assembly language is the symbolic language which is most like the actual machine language. Machine language is the actual numerical codes and addresses stored in memory which the machine hardware translates into action. Depending on the Computer, the machine language will be in Binary (base 2), Octal (base 8), or Hexadecimal (base 16). These numerical machine language codes provide control at the hardware level over machine functions, but are highly burdensome and time-consuming. The advantage of assembly language over machine language is that it allows symbolic representation of instructions and addresses while still allowing just as much control over specific instructions and machine functions.

The advantage of assembly language over high level languages such as Fortran, Cobol, Algol, or PL/I is that the assembly language provides: [Madnick 1974, IBM, 2]

1. Full control over all machine functions.

2. The form closest to machine language while still providing symbolic representation.

3. Ability to override system defaults assumed by the higher level language.

4. Ability to obtain system status information not accessible by higher level languages.

5. Ability to closely control your program down to the byte and even bit level.

These advantages are very important in an executive manager program such as DEX, where it is important to know the status of the system, program modules, attached files, memory management, database management, and DEX requests by the user, so that these elements can be integrated and closely controlled. In this Appendix I will provide background on the general knowledge neccessary to writing assembly language code on any machine. I will then address the specific machine structure for the IBM/370 on which this version of DEX is written. I will then address the structure of an

assembly language program on the 370 (the general structure can be adapted to other machines), and explain in greater detail an example assembly language program from the utility routines written in assembly language for DEX.

A-2.0 General Approach to Preparing for a New Computer

When preparing to write assembly language programs on a new computer, the programmer must gain an understanding of the machine's structure in order to control it. Madnick and Donovan [Madnick 1974] suggested a general approach to a new computer that could be adopted by a user wishing to understand the structure and become familiar with a new computer. The procedure consists of a series of questions that should be answered before attempting to write assembly language code on a new machine.

1. Memory

What is the memory's basic unit, size, and addressing scheme?

2. Registers

How many registers are there? What are their sizes, functions, and interrelationships?

3. Data

What type of data can be handled by the computer? Can it handle characters, numbers, logical data? How is this data stored?

4. Instructions

What are the classes of instructions on the machine? Are there arithmetic, logical, symbolic instructions? What are their formats? How are they stored in memory?

5. Special Hardware Features (pertinent to operating systems)

The answers to questions 1,2,3, and 4 will provide sufficient understanding of the machine structure for most users. However, the user who must under-

stand the operating system in order to interface with it, such as the DEX system programmer (not neccessary for the DEX module programmer), must understand additional hardware features.

a. Status of Program - How does the CPU keep track of which instruction to execute? Does the CPU differentiate between operating system state and user state?

b. Input/Output Processing - How is input and output handled? If separate I/O processors are used, how does the CPU communicate with them?

c. Interrupts - What kinds of interrupts exist? How are interrupts handled, and can they be intercepted? This is important in DEX for trapping abnormal termination, control information, and other status.

d. Masking - Can interrupts be ignored or suspended?

e. Protection - Is there hardware available that can prevent the reading, writing, or execution of parts

of memory? Is it possible to access system routines for DEX use?

f. Timers - Is there a clock on the system that can be accessed?

g. States - Does the CPU have different states? Does the CPU have privileged instructions that may be executed only in certain states?

To Madnick and Donovan's list I would add that the DEX system programmer must also gain an understanding of the operating system control block structure [IBM,3]. Where control blocks are areas of storage in which the system keeps information on data blocks, file status, and executable module status. Also I would recommend that the DEX system programmer become familiar with the system supervisor control program and how to communicate with it. The IBM/370 does this with the SVC (supervisor call) instruction [IBM,4].

For the 370 there are several references which the DEX system programmer should read [Madnick 1974, IBM,1,2,3,4,5].

A-3.0 IBM/370 Specific Machine Structure

In this section I will answer the questions of section A-2.0 as they apply to the IBM/370.

A-3.1 Memory

The IBM/370 is a 32 bit word machine. The basic unit of memory is a byte = 8 bits. A word is composed of 4 bytes and each byte is individually addressable. Therefore each addressable position in memory contains 8 bits of information, where a bit contains either a binary one or zero. Instructions may operate on the following groupings of bytes:

Memory Unit's Name	No. of Bytes	Length in Bits
Byte	1	8
Halfword	2	16
Word	4	32
Doubleword	8	64

The IBM/370 has a total of 2^{24} bytes of memory (about 16 million). The size is based on the addressing scheme used. The 370 uses a 24 bit address for referencing memory, and uses a base/offset addressing scheme. This scheme uses 12 bits of a 32 bit instruction for the offset and 4 more bits to indicate a register that will be used as the base register. Into this base register is placed a 24 bit base or starting address. Another 4 bits of the instruction are used to reference a register used as an index register. The address is then calculated by adding the offset to the contents of the base register. If the storage area is an array the index register can then be used to reference the proper element in the array.

Address = C(base register) + Offset + C(index register)

where C(register) indicates the contents of register

Any of the 4 bytes in a memory word can be individually addressed, but normally the low order byte is addressed. If the four bytes of the memory word had addresses 94,95,96, and 97 respectively, the memory word would be addressed as 94.

A-3.2 Registers

The 370 registers are of four general types:

- 16 General Purpose 32 bit registers
- 4 Floating Point 64 bit registers
- 16 Special Control 32 bit registers
- 1 Program Status Word 64 bit register

The general purpose registers can be used as base address registers or for arithmetic or logical operations. When used as a base register they contain a starting address, and care should be used not to alter its contents after the desired address is stored. In the case of arithmetic and logical operations these registers act as scratch pads in which calculation results are accumulated, manipulated or compared. The floating point registers are used in floating point calculations where greater accuracy is required. The Program Status Word (PSW) contains the address of the instruction to be executed, as well as protection information, and interrupt status.

The 16 control registers are assigned to special operating system facilities, such as paging or virtual memory handling. They are used, in a fashion similar to the general purpose registers, to act as scratch areas for the system facilities. They are used to hold information specifying if an operation can take place or to furnish and manipulate system information, such as the memory page size, for the control programs. The control registers are referenced by the numbers 0-15 in control instructions such as LOAD CONTROL 13,LOCATION.

There is no interrelationship on the 370 between the 4 types of registers. Whithin a given register group, such as the general purpose registers, the registers are interrelated by the instructions operating on them.

A-3.3 Data

All data and instructions are stored as binary ones and zeros. However, for convenience these numbers are written in the hexadecimal (base 16) number system. A hexadecimal digit is represented by four binary bits. The relation

between binary, hexadecimal and decimal numbers are given below:

Binary	Hexadecimal	Decimal
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	A	10
1011	B	11
1100	C	12
1101	D	13
1110	E	14
1111	F	15

So for example the binary number B'0000 0010 1111 1100' has the hexadecimal equivalent X' 0 2 F C '. The bits in a memory word are numbered from left to right, with the left most bit numbered zero and the right most bit numbered thirty one. The IBM/370 allows seven different data formats:

1. Short Form Fixed Point - This format is a 16 bit halfword with bit 0 used as a sign bit. With a

binary zero indicating a positive number, and a binary one indicating a two's compliment negative number. A knowledge of binary arithmetic is assumed for this discussion [Osborne,1980]. Bits 1-15 are then an integer number. Some instructions are unsigned and treat the data as a 16 bit unsigned number.

2. Long Form Fixed Point - Similar to the short form except that bits 1-31 contain the integer number.

3. Short Form Floating Point - Is a 32 bit representation, with bit 0 used as a sign bit, bits 1-7 used as an exponent, and bits 8-31 used as a fractional number.

4. Long Form Floating Point - Is a 64 bit representation, with bit 0 used for sign, bits 1-7 for exponent, and bits 8-63 used for the fraction.

5. Decimal-Number Formats - Decimal numbers may be represented in either of two formats, packed decimal or zoned (sometimes refered to as unpacked decimal) decimal. Either of these formats can have

from 1 to 16 bytes, with each byte consisting of two 4 bit codes. The packed decimal format is best suited for decimal arithmetic, while zoned decimal is best suited for input/output.

a. Packed Decimal - In this format each byte contains two 4 bit decimal digits (0-9) with binary representation 0000-1001, except for the right most byte which contains one binary decimal code and one 4 bit sign code in the right most 4 bits. The preferred sign code is 1100 (hex C) for positive, and 1101 (hex D) for negative. There are instructions which perform decimal arithmetic and return the results in the same format. There are also instructions which convert from packed decimal to zoned decimal (unpacked), or from packed decimal to binary.

b. Zoned Decimal - In this format each byte contains a decimal digit in the right 4 bits and a zone code in the left 4 bits. The zone code is a hex F (1111 binary) which can easily be masked out to extract the decimal information.

In this format the zone of the right most byte is treated as a sign code with the same representation used in packed decimal. Zoned decimal digits are part of a larger character set which represents alphanumeric and special characters as two hexadecimal numbers. Thus the zoned decimal numbers are represented by the hex number "F" followed by a decimal digit between 0-9. The zoned decimal numbers are therefore best suited for the input, editing, and output of human-readable numeric data.

6. Logical (Character) Format - In this representation, each character is represented by an 8 bit code [IBM, 9].

A-3.4 Instruction Format

The 370 has five types of instructions and five formats of instructions. The five types are:

1. General Instructions - These are the arithmetic, logical, comparison, branching, memory and register

manipulating, data conversion, and mask and test instructions.

2. Decimal Instructions - These are instructions which manipulate numbers which are in one of the decimal representations.

3. Floating Point Instructions - These are the instructions which normalize, manipulate, and perform floating point arithmetic.

4. Control Instructions - These are the instructions which are used to control and manipulate the supervisor state. These are privileged instructions and cannot be executed from the problem state (that's the user program).

5. Input/Output Instructions - These are the instructions which are used to control the input/output operations on the machine. Generally the program writer will not access these instructions, but will instead access system macros designed to facilitate I/O.

The basic form of all of the different formats of instructions is an 8 bit Op Code, and two operands. In assembly language the 8 bit Op Code is represented with a mnemonic code, such as A for add or L for load. The Op Code operates on the two operands. Generally the flow is from the second operand into the first operand, except in store instructions where the data indicated by the first operand is stored in the location indicated by the second operand. An example of the first type of flow would be:

```
A 1,2
```

which would add the contents of register 2 to register 1 and leave the result in register 1. An example of the second type of flow would be:

```
ST 5,TEMP
```

which would store the contents of register 5 into the memory location which has the label TEMP. The five formats of instruction are:

1. RR Format (2 bytes in length) - A register-register instruction which has an op code which operates on the contents of two registers.

2. RX Format (4 bytes in length) - A register-indexed memory instruction which moves the contents of a memory location into a register, stores the contents of the register in memory, or performs an arithmetic or logical operation on the contents of the register and the memory location. In this instruction the memory address is found by adding the offset, base, and index.

3. RS Format (4 bytes in length) - A register-memory instruction which is similar to the RX instruction except that it has two register operands and a base/offset memory operand.

4. SI Format (4 bytes in length) - An immediate operand - memory instruction which performs the operation on the memory location and the immediate 8 bit value contained in the instruction.

5. SS Format (6 bytes in length) - A memory-memory instruction which performs the operation on the two locations in memory.

A-3.5 Special Hardware Features

STATUS OF A PROGRAM The program status word (PSW) is the internal register which is the basic control register for the 370. It contains information on interrupt status, what interrupts are masked out of use, whether the machine is in supervisor (privileged) state or problem state (working on a user problem), and the address of the next instruction.

64 bits

System Mask	Key	CMWP	Interrupt code	ILC	CC	Program Mask	Instruction Address
8 bits	4	4	16	2	2	4	24

bit 0 bit 63

The SYSTEM MASK indicates whether or not interrupts will be accepted from a given channel. Bits 0-5 correspond to channels 0-5 and must be on (binary one) if interrupts are to be accepted. Bit 6 set on indicates that interrupts will be accepted from all channels above 6. Bit 7 on indicates that external interrupts are allowed. If the I/O channel is masked (off) then the hardware will hold the interrupt until the interrupt is again enabled.

The KEY is for protection of memory. If an area of memory is referenced its key must match the key in the PSW allowed for the executing program. If the keys do not match a protection exception occurs.

The CMWP field is a collection of 4 flags (C=extended control mode, M=machine check mode, W=wait mode, P=problem state mode). C and M flags are used by the operating system for system control testing, and machine hardware failure checking. They will not be discussed as they are not applicable to the general assembly language programmer. The W bit indicates that the CPU is running (W=0) or waiting (W=1). The P flag indicates the state of the machine (P=0 implies the supervisory state, P=1 implies the user problem is running). The INTERRUPT CODE tells the CPU the type of interrupt last received and thus where to go in memory to handle it. The ILC field is the Instruction Length Code and tells you the length of the previous instruction executed. The CC field is the Condition Code and contains the current value of the condition set by certain instructions. The CC field is tested by the conditional branch instructions, and is set to flag the results of arithmetic and comparison instructions. It might be set to indicate that the result of an operation was positive, negative, or zero. This con-

dition code can then be checked by a conditional branch instruction such as branch on zero (BZ).

The PROGRAM MASK allows the masking of other non I/O interrupts such as underflow and overflow in arithmetic calculations. The INSTRUCTION ADDRESS contains the address of the next instruction to be executed. This field, the condition code, and the previous instruction length code ILC are very useful for debugging programs. The load address of the program can be subtracted (in hex arithmetic) from the address in the PSW at the time of an error to get a relative address of the error.

Of the special hardware features mentioned, the status of program, masking, and protection were covered in the above discussion of the program status word. Input/Output processing and Interupts will now be discussed.

Input/Output processing involves the transfer of information between the main memory of the computer and an I/O device such as a disk drive. On the IBM 370, I/O devices are attached to I/O processors called CHANNELS, which control all data transfer. The CPU communicates with the I/O channels through the I/O instructions:

Test Channel

Clear Channel

Store Channel ID

Clear I/O

Halt Device

Halt I/O

Start I/O

Start I/O Fast Relief

Test I/O

The first 3 instructions reference a channel only. The last 6 instructions address a channel and a device on that channel. The I/O address is a 16 bit address consisting of two parts. The leftmost eight bits is the channel address, and the rightmost 8 bits is the device address. There are up to 256 channels numbered from 0-255, and their type and assignments are system dependent. Each channel has the facility for addressing up to 256 devices. The CPU uses the START I/O command to identify a channel and tell it to start operation. The channel then loads the Channel Address Word (CAW) from a hardware fixed location in memory. The CAW contains the address in memory of the I/O program the channel is to execute.

The programmer could write an assembly language program to control individual channels and devices. However, the system provides Data Management Macro Instructions which perform I/O operations, and provide the facilities to open, close, and manipulate files [IBM,4]. The reader is refered to the literature for a more detailed description of these facilities. [Madnick, 1974, Donovan, 1982, IBM, 1,2,3,4,5]

Interrupts are hardware signals indicating that some condition requires the immediate attention of the CPU. On the 370 there are five conditions or interrupt types:

Operating Exception - tried to execute an illegal instruction.

Machine Checking - Hardware Failure Detection

External - such as an alarm

Input/Output - channel is finished or waiting to transfer to CPU

Supervisor Call (SVC) - request to the operating system.

An interrupt signal causes a hardware automatic response which notes where execution is when the interrupt occurs, and starts an interrupt action. When the interrupt occurs, the hardware first saves the current PSW in a predefined location in memory. Then the hardware loads the PSW from a specific reserved memory location, which depends on the type of interrupt. Then the CPU executes the interrupt handler program. When the interrupt handler completes its function it restores the old PSW from the fixed location. The CPU resumes execution of the interrupted program at the place where it was interrupted.

A-4.0 Assembly Language Structure

An assembly language program is made up of statements that are either instructions or comments. The assembly language instructions can be divided into three types:

1. Machine Instructions (machine-op)

   Which are the symbolic representation of the machine language instruction. An example of a machine instruction would be the add instruction, in which

the CPU is instructed to add the contents of two registers. For example the machine instruction

```
ADD 1,2
```

would add the contents of register 1 to the contents of register 2. This type of instruction will be explained more fully in the example program.

2. Assembler Instructions (psuedo-op)

An assembler instruction or psuedo-op is a note to the assembler which is not executable at execution time, but rather tells the assembler how to set up the program while it is assembling it. Examples of this type of instruction would be DC, and DS which define constants or storage. Assembler instructions can also be used to define entry points and other references.

3. Macro Instructions

A macro instruction allows the assembler to substitute a predefined block of code for that macro sym-

bol when it is encountered in the program. This allows the user to define a block of code which is used often in a program as a macro. The name of the macro is then used in the program and the assembler expands it at assembly time. Macros can take arguments as well, and also allows for conditional expansion based on the arguments. The system provides several utility macro definitions for Input/Output handling, data management, and communication with supervisor operations.

There are three types of comments in an assembly language program. In a macro definition comments must begin with a period in column one followed by an *. In the main part of the assembly language program comments on a line by themselves must begin with an * in column one and must not go beyond column 71. Comments may be included on the same line as instructions, but must be after the last field of the instruction and must be separated from the last instruction field by at least one blank.

A-4.1 General Structure of Program

The general structure of an assembly language program depends on the conventions of the particular machine you are using. The basic ideas will be the same as for the IBM/370 but the reader will need to investigate these conventions at his computer facility. Generally psuedo-ops will be used to set up the entry and external linkage to the routine, as well as the starting reference and base register convention. Additionally psuedo-ops will be used to define storage areas. The general convention is that it is the responsibility of the assembly language routine to save all registers at entry, and restore them before returning control to the calling routine. After this setup is accomplished the routine performs its calculations, stores its results, and returns to the calling routine. This structure will be explained in the example, and is summarized below:

- Define program START, ENTRY points, control sectioning (CSECT,DSECT), and external references (EXTRN)

- Identify the base register or registers the program will be USING.

- Save the general purpose registers using the store multiple(SM) instruction into the save area indicated by register 13. The store multiple instruction stores the contents of all the registers from the register indicated by the first operand to that indicated by the second operand into the memory area indicated by the third operand.

- Pick up the arguments if any. The convention for passing arguments will be explained in the section on coding conventions.

- Perform the required calculations and operations.

- Save the results in the arguments.

- Restore the registers by using the load multiple instruction (LM). This is neccessary because calculations in your routine have changed the contents of the registers, and this could destroy the results obtained by the calling routine. Also if the registers aren't

restored the return path to the calling routine could be lost.

- Return control to the calling program by branching to the address contained in register 14. (BR 14).

- Use the END psuedo-op to end assembly.

A-4.2 Coding Conventions

The statement field in an assembly language program begins in column 1 and ends in column 71. Column 72 is used to indicate a continuation line follows. If a line is continued then the continuation line begins in column 16. The format of an instruction statement consists of four parts 1) the label, 2) the operation code, 3)the operand entry, and 4) an optional remark. The order is important, and must be from 1-4. The entries must be separated by one or more blanks. If used the label must begin in column 1, and the operation code must start at least one column to the right of column 1. Generally for ease of reading the op-code is placed in column 10, and the operands start in column 16

with the optional remark in any convenient place to the right of the operands.

A-4.3 Subroutine Calling and Return Convention

When calling an assembly language subroutine, the convention is that the addresses of the arguments are loaded consecutively into a parameter list (PLIST), with the address of argument one stored in the first word of storage, and the address of argument two stored in the second word and so on. Then the address of the parameter list is loaded into register one. For example assuming the two arguments NAME and RCODE:

```
        LA    3,NAME        Load address of Name into R3
        ST    3,PLIST       Store it in Plist
        LA    3,RCODE       Load address of Rcode into R3
        ST    3,PLIST+4     Store it in the second word of
PLIST
        LA    1,PLIST       Load the Address of PLIST into
R1
```

The convention for passing control is to load the address of the routine you want to branch to into register 15, and the address where you want control to return into register 14. You then branch to the address in register 15. This can be accomplished nicely by use of the BALR (branch and link) instruction, which loads the address of the next sequential instruction into the first operand, and branches to the address contained in the second operand register. For example:

```
L     15,CKADDR
BALR  14,15
CL    0,ZERO
```

In this example the address of the CKFILE routine was previously loaded into the location with the symbol CKADDR. The first instruction loads register 15 from the contents of location CKADDR. The BALR instruction loads the address of the compare logical (CL) instruction which follows it into register 14, and then branches to the address in register 15. Now control has been passed to the CKFILE routine. When the subroutine has completed it's operation it does an unconditional branch to the address contained in register 14 (BR 14) which returns control back to the CL instruction.

This CL instruction then compares the return code in register 0 to zero.

Another convention often used is that register 13 contains the address of a save area for saving register contents.

As the writer of an assembly language subroutine you then can expect the following register conventions when your subroutine receives control.

- Register 15 will contain the address at which your routine has been loaded (since it was used to contain the address of the subroutine in the BALR 14,15 instruction), and thus can be used as the base or reference address register.

- Register one will contain the address of a parameter list where the entries in the parameter list are pointers to the arguments. There are two techniques for obtaining this data. Assuming that there are three arguments, the first technique would be to load multiple into registers 2,3,and 4 from the location pointed to by register 1.

```
LM 2,4,0(1)
```

Would result in registers 2,3, and 4 being loaded consecutively from the address pointed to by register 1, C(1)+4, and C(1)+8. Each of these three registers would then be pointing at one of the arguments. To get the argument it is then neccessary to load from the address pointed to by the register. For example:

```
L 5,0(2)
```

This would load register 5 from the address contained in register 2 plus an offset of zero. The address contained in register two is the address of the first argument, which was loaded into register two by the load multiple instruction above. The other technique is to individually load each register with the address of an argument and then use that register to load another register with the actual argument. For example,

```
L  2,0(1)  load register 2 with the address pointed
           to by register 1, which is the first address in the
           parameter list.  As a result register 2 will be
           pointing at the first argument.
```

L 5,0(2) load register 5 from the location pointed to by register 2. Places the first argument in register 5.

L 3,4(1) load register 3 with the address pointed to by register 1 plus 4. In other words the pointer to the second argument.

L 6,0(3) load register 6 from the location pointed to by register 3.

L 4,8(1) again load register 4 from the location pointed to by register 1 plus 8. Register 4 will contain the address found by adding an offset of 8 bytes to the address of the PLIST which is the address that was loaded into register 1.

L 7,0(4) now register 4 points to the third argument, and this instruction loads it into register 7.

- Register 13 will contain the address of a save area.

- Register 14 will contain the return address in the calling routine.

A-4.4

Macro Organization

Macro definitions are composed of the MACRO psuedo-op indicating that a macro definition follows, a macro name which can take arguments, a body of executable assembly language instructions and conditional assembly macro instructions, and a MEND (Macro END) psuedo-op indicating that the macro definition has ended. Macro definitions must be either at the begining of a program or in a macro library. Arguments in the macro definition are indicated by & followed by a name. Wherever this occurs in the macro definition the corresponding argument will be substituted for it. Macros can be used for three basic functions.

1. For Text Insertion - This is the most common use of macros. In this fashion macros are used to insert commonly used assembly language instructions into the source program at assembly time. This allows the programmer to not have to continuously write

repetitious code in each program. An example would be a macro to set up the linkage convention between subroutines, such as the DEX IN and OUT macros.

2. For Text Modification - Modify the statements in the macro definition depending on how they are to be used.

3. For Text Manipulation - Utilizing conditional assembly change the way a macro is expanded based on the arguments so that a different block of code is substituted in different situations.

A-4.5 Example Assembly Language Routine

In this section I will consider a specific assembly language subroutine from the DEX assembly language utility routines. The detailed analysis of this routine will hopefully bring the readers understanding of assembly language programming into focus The routine that I will consider is the dynamic loading routine LOAD. This routine is a logical function. However, the passing of arguments will be the same as for a sub-

routine. The only difference is that in this case the truth value will be returned in register 0, with R0 = 0 indicating .false., and R0 = 1 indicating .true.. The first thing you will encounter when looking at the LOAD routine in figure A-1, is the block of comments explaining its purpose and calling sequence. Comments are important in any programming language to insure that the understanding of the program is passed on, and to facilitate future maintenance and modification. In assembly language comments are absolutely essential in order to obtain any continuity of what was originally done and intended.

After the comment section is the actual program code. The first line of which is the line - LOAD IN CSECT,etc.. This line is a macro definition, and the reader is referred to figure A-2 to see the expansion of the macro. The expanded code is indicated by a + sign in column 1. This is an excellent example of one of the reasons for using macros. All of the DEX assembly language routines have to save the registers and set up the base-using convention. This was done once when the IN macro was written and now is just invoked with each new subroutine. This macro and the OUT macro

which is further down in the code both take arguments which allow for modifying the use of the macro depending on the situation.

Looking at the expanded macro code in figure A-2, it can be seen that the first expanded instruction is the line - LOAD CSECT. In this line is the label LOAD and the psuedo-op CSECT. The CSECT defines to the assembler the beginning of this program as a control section which means it is executable and can be relocated. The routine is given the label LOAD. Another psuedo-op used to indicate a control section is the START psuedo-op, which is used at the start of the first routine in a group of programs to be loaded into memory. All subsequent routines use the CSECT. The reason START was not used is that psuedo-op indicates the first control section in a block of source code. Since this routine is a logical function in DEX it is not the first routine in the source file, the CSECT was used to indicate this is a control section other than the first. A control section is the smallest block of code that can be relocated as a unit.

The next instruction encountered in the macro expansion is a branch to a location which is offset 10 from the contents of register 15. Since register 15 will contain the address of this routine when execution is transfered here the address is 10 (hex A) from the beginning of the program. This is in effect a branch to the STM instruction at statement number 34. This is neccessary because the macro expansion uses data constants (DC) at statements 32 and 33, and the CPU can't execute these since they are not instructions. The next instruction is the STM or store multiple which is used to save the registers in the location passed in register 13 + an offset of 12. The order in which the registers are stored in this case is register 14, 15, and then 1-12. After this the LR (load register) instruction is used to copy the contents of register 15 into register 12 so that register 15 can be used for calling other routines. Thus register 12 now becomes the base register, and this is indicated by the USING psuedo-op which is a note to the assembler that it can use register 15 for the base register and it can assume that the address of LOAD will be loaded into it at execution time.

At assembly time the assembler calculates its symbolic addresses relative to the address indicated by the USING psuedo-op. Thus looking at the excerpt from the listing (figure A-3) it can be seen that the address of the symbol NAME at statement 92 is 104 hex relative to LOAD. This then becomes the offset for the symbol NAME. Looking at the reference to NAME in statement number 42, the MVC (move) instruction:

```
MVC   NAME(8),0(2)
```

The instruction says to move 8 bytes of the contents pointed to by register 2 into storage location NAME. Looking to the left of the instruction at the object code column you will find a 12 digit hexadecimal number which is the machine code translation of this instruction:

```
D207 C104 2000
```

The D2 is the op-code, the 07 is the length to be moved less one, the C104 is the first operand address, and the 2000 is the second operand address. Translating the two numbers using the IBM system/370 reference Sum-

mary card [IBM, 6], in the first operand the C is the base register (hex C = decimal 12) and the 104 is the offset. In the second operand the register is number 2 and the offset is 000. Therefore all symbolic references are translated to a base register and the offset from the address in the USING instruction.

Now continuing with the macro expansion of the IN macro, the next group of instructions loads the address of this routine's save area and then chains it to the one passed. The LA instruction at statement 37 loads register 15 with the address of the SAVEAREA argument. The next instruction at statement 38 stores the contents of register 13 into the location determined by adding 4 to the contents of register 15. This loads the address of the save area passed from the routine which called LOAD into LOAD's SAVEAREA+4. The next instruction at statement 39 saves the address of the old save area in SAVEAREA which is pointed to by register 15. Finally the instruction at statement 40 loads register 13 with the address of the new SAVEAREA. The last thing to discuss regarding to IN macro is the argument STORAGE=0. This argument indicates to the IN macro that no dynamic storage is to be allocated. If

desired the programmer could indicate by this argument that a block of main storage is to be allocated for use by this routine at execution of this routine. This completes the group of instructions resulting from the expansion of the macro IN (these are the instructions flagged by the +).

The next instruction at line 41 picks up the first argument in the parameter list by loading register 2 with the pointer in register 1. Then the MVC instruction which was explained in the paragraph on address translation above, moves the contents of the location pointed to by register 2 into the location NAME. The next instruction at line 43 loads the pointer to the third argument (rfcode) into register 3. Then the next MVC instruction moves a litteral "1" into the argument. This indicates that if a fatal error occurs it was in the LOAD routine. If no fatal error occurs this code will be reset to zero before returning.

The next group of instructions moves the address of NAME and RCODE into the parameter list and loads the address of the parameter list into register 1. This was discussed previously. Then register 15 is loaded

with the contents of location indicated by the label CKADDR. Looking at statement 100, the label CKADDR is defined using a DC (define constant) psuedo-op, and into this location is placed a V type constant. The V type constant is an external reference which will be resolved at execution time. the V constant tells the assembler to set up an external symbol table with the entry CKFILE. At execution time the address of CKFILE will be known and the address will be placed in this location. Thus the load instruction at line 51 is loading register 15 with the address of the subroutine CKFILE. This and the next two instructions were previously discussed. The BALR 14,15 branches to the subroutine CKFILE because that will be the address loaded into register 15, and then links by placing the return address in register 14. Upon return register 0 will contain a zero if CKFILE is .false. and a one if CKFILE is true. The CL (compare logical) instruction compares the contents of register 0 with the 0 stored in location ZERO. If the comparison is true the CC flag in the PSW is set, and the next instruction BE BOMB is a branch on equal. The BE tests the PSW and if the two were equal it branches to BOMB. This would be the case if the file which we are attempting to load

does not exist. If execution goes to BOMB, the fatal return code in argument 3 is reset to zero by use of the load and move instructions. Then the OUT macro is invoked with the arguments for a return register of zero and a return code of zero indicating a .false. condition. Thus if the routine calling load was an assembly language program, it will test register 0 for the return code. If the calling routine was a fortran routine, then this logical function LOAD will be .false. and the truthflg to which it is equated will be set .false..

If CKFILE was successful in finding the file to be loaded, then the next instruction encountered is the name of a system macro. This macro is called LOAD and takes the argument eploc=NAME. Which passes to the macro the name of the module being loaded. The system LOAD macro is expanded to three instructions. A load address (LA) of the NAME into register 0, a subtract (SR) register 1 from itself to zero it out(this indicates that we are not using a DCB or data control block), and SVC 8. The SVC instruction is a call to the supervisor program in the Operating System which indicates by the code 8 that the module name contained

in register 0 is to be loaded into memory. Upon return from the supervisor call the register 0 will contain the entry point location of the module in memory (the address). The contents of register 0 are then stored in the location ENTRY, and the rfcode is reset to zero by the same load argument address and move sequence. Also the contents of ENTRY is moved into argument 2. Then the OUT macro is invoked with a return code equal to one indicating that the LOAD was .true.. The OUT macro expansion allows for restoring the registers to the state they were upon entry, and then a BR 14 returns to the calling routine. Where BR 14 is branch to the location in register 14, which is the next instruction in the calling routine. Also in the OUT macro is a LTORG psuedo-op which defines the place to put literal constants. A literal constant is of the form =F'1'. A literal can be used in an instruction as one of the operands, without having to define a storage location yourself. Literals are placed either at the end of the program, or at the location defined by the LTORG instruction.

The last section of the program is the data constant area. In this area there are two types of psuedo-ops

used to reserve storage. The first is the define storage (DS) which reserves the indicated length of storage but does not set it to any initial value. An example of this is the SAVEAREA DS 18F, which saves 18 fullwords of storage with the first labeled SAVEAREA. The other type is the define constant (DC), which sets the storage to the value indicated initially. Valid types are F for fullword H for halfword, X for hexadecimal, V for external address value, C for character, B for binary, and A for address.

```
*****************************************************************************
*
*         LOAD -
*
*         SUBROUTINE TO LOAD A MODULE
*
*         CALLING SEQUENCE -
*
*               TRUTHVALUE = LOAD(FILENAME,ENTRY,RFCODE)
*
*         WHERE FILENAME IS THE FILENAME OF MODULE TO BE LOADED.
*               THE FILETYPE IS ASSUMED TO BE TEXT SINCE THAT IS THE
*               ONLY TYPE WHICH CAN BE EXECUTED.
*         WHERE ENTRY IS AN INTEGER ENTRY POINT ADDR OF THE LOADED
*               MODULE TO BE RETURNED TO THE CALLING PROGRAM
*         WHERE RFCODE IS A FATAL RETURN CODE. ABNORMAL TERMINATION
*               IS ASSUMED AND RFCODE IS SET EQUAL TO ONE UPON ENTRY
*               TO THIS ROUTINE.  IF ABNORMAL TERMINATION OCCURS
*               RFCODE = 1 INDICATES THE FAILURE OCCURED HERE.  IF THE
*               TERMINATION IS NORMAL RFCODE IS SET TO ZERO PRIOR TO
*               RETURN TO THE CALLING ROUTINE.
*
*         TRUTHVALUE IS TRUE IF LOAD WAS SUCCESSFUL AND FALSE IF
*               LOAD WAS UNSUCCESSFUL
*
*         USES THE OS/VS1 SUPERVISOR 'LOAD' MACRO.  FOR FURTHER INFO
*           SEE 'OS/VS1 SUPERVISOR SERVICES AND MACRO INSTRUCTIONS',
*              GC24-5103
*
*****************************************************************************
LOAD      IN    CSECT,STORAGE=0,SAVE=SAVEAREA
          L     2,0(1)          LOAD THE ADDR OF ARG1 INTO R2
          MVC   NAME(8),0(2)    GET LOAD MODULE NAME
          L     3,8(1)          LOAD THE ADDR OF ARG3 INTO R3
          MVC   0(4,3),=F'1'    SET RFCODE = 1 LOAD CAUSED ANY FATAL ERR
          ST    1,TEMP          SAVE R1 UNTIL AFTER LOAD
          LA    3,NAME          LOAD THE ADDRESS OF NAME INTO R3 AND
          ST    3,PLIST         STORE IT IN THE PARAMETER LIST
          LA    3,RCODE         LOAD THE ADDRESS OF THE RETURN CODE IN R3
          ST    3,PLIST+4       AND STORE IT IN THE PARAMETER LIST
          LA    1,PLIST         LOAD R1 WITH PARAMETER LIST ADDRESS
          L     15,CKADDR       LOAD THE ENTRY ADDRESS FOR CKFILE
          BALR  14,15           BRANCH TO ENTERY POINT
          CL    0,ZERO          COMPARE RETURN CODE IN R0 TO ZERO
          BE    BOMB            IF RETURN CODE = 0 LOAD MODULE NOT FOUND
***
          LOAD EPLOC=NAME       LOAD MODULE (EPLOC = ENTRY POINT LOCATION)
          ST    0,ENTRY         STORE THE ENTRY POINT ADDR IN ENTRY
```

```
         L     1,TEMP          RESTORE R1
         L     2,4(1)          USE R2 FOR THE ADDR OF ARG2
         L     3,8(1)          USE R3 FOR THE ADDR OF ARG3
         MVC   0(4,2),ENTRY    STORE THE ENTRY ADDR IN ARG2
         MVC   0(4,3),ZERO     RESET RFCODE = 0, NORMAL TERMINATION
***
         OUT   STORAGE=0,SAVE=SAVEAREA,RETC=1,RETREG=0  RETURN 'TRUE'
***
**************************** TEXT FILE DID NOT EXIST RETURN FALSE
***
BOMB     L     1,TEMP          RESTORE R1
         L     3,8(1)          USE R3 FOR THE ADDR OF ARG3
         MVC   0(4,3),ZERO     RESET RFCODE = 0, NORMAL TERMINATION
         OUT   STORAGE=0,SAVE=SAVEAREA,RETC=0,RETREG=0  RETURN  'FALSE'
*******************************************************************************
***  DATA CONSTANT AREA
SAVEAREA DS     18F           AREA FOR SAVING THE REGISTERS
TEMP     DS     F             TEMPORARY ONE REGISTER SAVE AREA
NAME     DC     CL8' '        FILENAME
         DC     CL8'TEXT'     FILETYPE
         DC     CL2' '        FILEMODE
         DS     0F            ALLIGN ON FULL WORD BOUNDARY
RCODE    DS     F             RETURN CODE FROM CKFILE STORED HERE
ENTRY    DS     A             STORAGE AREA FOR MODULE ENTRY ADDRESS
ZERO     DC     4X'00'        ZERO FULL WORD FOR COMPARISON
PLIST    DS     2A            PARAMETER LIST
CKADDR   DC     V(CKFILE)     ENTRY POINT ADDRESS FOR CKFILE ROUTINE
         END
```

figure A-1

```
 1 ********************************************************************************
 2 *
 3 *         LOAD -
 4 *
 5 *         SUBROUTINE TO LOAD A MODULE
 6 *
 7 *         CALLING SEQUENCE -
 8 *
 9 *               TRUTHVALUE = LOAD(FILENAME,ENTRY,RFCODE)
 0 *
 1 *         WHERE FILENAME IS THE FILENAME OF MODULE TO BE LOADED.
 2 *         WHERE ENTRY IS AN INTEGER ENTRY POINT ADDR OF THE LOADED
 3 *               MODULE TO BE RETURNED TO THE CALLING PROGRAM
 4 *         WHERE RFCODE IS A FATAL RETURN CODE. ABNORMAL TERMINATION
 5 *               IS ASSUMED AND RFCODE IS SET EQUAL TO ONE UPON ENTRY
16 *               TO THIS ROUTINE.  IF ABNORMAL TERMINATION OCCURS
17 *               RFCODE = 1 INDICATES THE FAILURE OCCURED HERE.  IF THE
18 *               TERMINATION IS NORMAL RFCODE IS SET TO ZERO PRIOR TO
19 *               RETURN TO THE CALLING ROUTINE.
20 *
21 *         TRUTHVALUE IS TRUE IF LOAD WAS SUCCESSFUL AND FALSE IF
22 *               LOAD WAS UNSUCCESSFUL
23 *
24 *         USES THE OS/VS1 SUPERVISOR 'LOAD' MACRO.  FOR FURTHER INFO
25 *           SEE 'OS/VS1 SUPERVISOR SERVICES AND MACRO INSTRUCTIONS',
26 *              GC24-5103
27 *
28 ********************************************************************************
29 LOAD     IN    CSECT,STORAGE=0,SAVE=SAVEAREA
30+LOAD      CSECT
31+         B     10(0,15)                                BRANCH AROUND ID
32+         DC    AL1(4)                                  LENGTH OF IDENTIFIER
33+         DC    CL4'LOAD'                               IDENTIFIER
34+         STM   14,12,12(13)                            SAVE REGISTERS
35+          LR    12,15
36+          USING LOAD,12
37+         LA    15,SAVEAREA    STATIC SAVE AREA
38+         ST    13,4(15)       CHAIN OLD AREA IN
39+         ST    15,8(13)
40+         LR    13,15
41          L     2,0(1)         LOAD THE ADDR OF ARG1 INTO R2
42          MVC   NAME(8),0(2)   GET LOAD MODULE NAME
43          L     3,8(1)         LOAD THE ADDR OF ARG3 INTO R3
44          MVC   0(4,3),=F'1'   SET RFCODE = 1 LOAD CAUSED ANY FATAL ERR
45          ST    1,TEMP         SAVE R1 UNTIL AFTER LOAD
46          LA    3,NAME         LOAD THE ADDRESS OF NAME INTO R3 AND
47          ST    3,PLIST        STORE IT IN THE PARAMETER LIST
48          LA    3,RCODE        LOAD THE ADDRESS OF THE RETURN CODE IN R3
49          ST    3,PLIST+4      AND STORE IT IN THE PARAMETER LIST
```

```
50          LA    1,PLIST       LOAD R1 WITH PARAMETER LIST ADDRESS
51          L     15,CKADDR     LOAD THE ENTRY ADDRESS FOR CKFILE
52          BALR  14,15         BRANCH TO ENTERY POINT
53          CL    0,ZERO        COMPARE RETURN CODE IN R0 TO ZERO
54          BE    BOMB          IF RETURN CODE = 0 LOAD MODULE NOT FOUND
55 ***
56          LOAD EPLOC=NAME     LOAD MODULE (EPLOC = ENTRY POINT LOCATION
57+         LA    0,NAME              LOAD PARAMETER INTO REGISTER ZERO
58+         SR    1,1                 SHOW NO DCB PRESENT
59+         SVC   8                   ISSUE LOAD SVC
60          ST    0,ENTRY       STORE THE ENTRY POINT ADDR IN ENTRY
61          L     1,TEMP        RESTORE R1
62          L     2,4(1)        USE R2 FOR THE ADDR OF ARG2
63          L     3,8(1)        USE R3 FOR THE ADDR OF ARG3
64          MVC   0(4,2),ENTRY  STORE THE ENTRY ADDR IN ARG2
65          MVC   0(4,3),ZERO   RESET RFCODE = 0, NORMAL TERMINATION
66 ***
67          OUT   STORAGE=0,SAVE=SAVEAREA,RETC=1,RETREG=0  RETURN 'TRUE'
68+         L     13,4(13)       RETURN TO ORIGINAL SAVE AREA
69+         LA    0,1
70+         ST    0,20(13)
71+         LM    14,12,12(13)
72+         BR    14
73+         LTORG
74                =F'1'
75 ***
76 ************************** TEXT FILE DID NOT EXIST RETURN FALSE
77 ***
78 BOMB      L     1,TEMP        RESTORE R1
79          L     3,8(1)        USE R3 FOR THE ADDR OF ARG3
80          MVC   0(4,3),ZERO   RESET RFCODE = 0, NORMAL TERMINATION
81          OUT   STORAGE=0,SAVE=SAVEAREA,RETC=0,RETREG=0  RETURN  'FALSE
82+         L     13,4(13)       RETURN TO ORIGINAL SAVE AREA
83+         LA    0,0
84+         ST    0,20(13)
85+         LM    14,12,12(13)
86+         BR    14
87+         LTORG
88 ********************************************************************************
89 ***  DATA CONSTANT AREA
90 SAVEAREA DS     18F          AREA FOR SAVING THE REGISTERS
91 TEMP     DS     F            TEMPORARY ONE REGISTER SAVE AREA
92 NAME     DC    CL8' '        FILENAME
93          DC    CL8'TEXT'     FILETYPE
94          DC    CL2' '        FILEMODE
95          DS    0F            ALLIGN ON FULL WORD BOUNDARY
96 RCODE    DS    F             RETURN CODE FROM CKFILE STORED HERE
97 ENTRY    DS    A             STORAGE AREA FOR MODULE ENTRY ADDRESS
98 ZERO     DC    4X'00'        ZERO FULL WORD FOR COMPARISON
```

```
 99 PLIST    DS    2A            PARAMETER LIST
100 CKADDR   DC    V(CKFILE)     ENTRY POINT ADDRESS FOR CKFILE ROUTINE
101          END
```

figure A-2

EXCERPT FROM LISTING

```
                EXTERNAL SYMBOL DICTIONARY  PAGE 1
-SYMBOL   TYPE  ID   ADDR  LENGTH LDID
 LOAD      SD  0001 000000 000130  CKFILE    ER  0002

                        PAGE 2
-  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT
                                      29 LOAD     IN    CSECT,STORAGE=0,SAVE=SAVEAR
 000000                               30+LOAD      CSECT
 000000 47F0 F00A      0000A          31+          B     10(0,15)
 000004 04                            32+          DC    AL1(4)
 000005 D3D6C1C4                      33+          DC    CL4'LOAD'
 000009 00
 00000A 90EC D00C      0000C          34+          STM   14,12,12(13)
 00000E 18CF                          35+           LR    12,15
                              00000   36+           USING LOAD,12
 000010 41F0 C0B8      000B8          37+          LA    15,SAVEAREA
 000014 50DF 0004      00004          38+          ST    13,4(15)
 000018 50FD 0008      00008          39+          ST    15,8(13)
 00001C 18DF                          40+          LR    13,15
 00001E 5821 0000      00000          41           L     2,0(1)
 000022 D207 C104 2000 00104 00000    42           MVC   NAME(8),0(2)
 000028 5831 0008      00008          43           L     3,8(1)
 00002C D203 3000 C090 00000 00090    44           MVC   0(4,3),=F'1'
 000032 5010 C100      00100          45           ST    1,TEMP
 000036 4130 C104      00104          46           LA    3,NAME
 00003A 5030 C124      00124          47           ST    3,PLIST
 00003E 4130 C118      00118          48           LA    3,RCODE
 000042 5030 C128      00128          49           ST    3,PLIST+4
 000046 4110 C124      00124          50           LA    1,PLIST
 00004A 58F0 C12C      0012C          51           L     15,CKADDR
 00004E 05EF                          52           BALR  14,15
 000050 5500 C120      00120          53           CL    0,ZERO
 000054 4780 C094      00094          54           BE    BOMB

                        PAGE 3
-  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT
0                                     55 ***
                                      56           LOAD EPLOC=NAME
 000058 4100 C104      00104          57+          LA    0,NAME
 00005C 1B11                          58+          SR    1,1
 00005E 0A08                          59+          SVC   8
 000060 5000 C11C      0011C          60           ST    0,ENTRY
 000064 5810 C100      00100          61           L     1,TEMP
 000068 5821 0004      00004          62           L     2,4(1)
 00006C 5831 0008      00008          63           L     3,8(1)
                       150
```

```
000070 D203 2000 C11C 00000 0011C    64           MVC   0(4,2),ENTRY
000076 D203 3000 C120 00000 00120    65           MVC   0(4,3),ZERO
                                     66 ***
67           OUT   STORAGE=0,SAVE=SAVEAREA,RETC=1,RETREG=0  RETURN 'TRUE'
00007C 58DD 0004      00004          68+          L     13,4(13)
000080 4100 0001      00001          69+          LA    0,1
000084 500D 0014      00014          70+          ST    0,20(13)
000088 98EC D00C      0000C          71+          LM    14,12,12(13)
00008C 07FE                          72+          BR    14
000090                               73+          LTORG
000090 00000001                      74                =F'1'
                                     75 ***
                                     76 ***  TEXT FILE DID NOT EXIST
                                        ***  RETURN FALSE
                                     77 ***
000094 5810 C100      00100          78 BOMB      L     1,TEMP
000098 5831 0008      00008          79           L     3,8(1)
00009C D203 3000 C120 00000 00120    80           MVC   0(4,3),ZERO
81           OUT   STORAGE=0,SAVE=SAVEAREA,RETC=0,RETREG=0 RETURN 'FALSE'
0000A2 58DD 0004      00004          82+          L     13,4(13)
0000A6 4100 0000      00000          83+          LA    0,0
0000AA 500D 0014      00014          84+          ST    0,20(13)
0000AE 98EC D00C      0000C          85+          LM    14,12,12(13)
0000B2 07FE                          86+          BR    14
0000B8                               87+          LTORG
                                     88 ******************************
                                     89 ***  DATA CONSTANT AREA
0000B8                               90 SAVEAREA DS     18F
000100                               91 TEMP     DS     F
000104 4040404040404040              92 NAME     DC     CL8' '
00010C E3C5E7E340404040              93          DC     CL8'TEXT'
000114 4040                          94          DC     CL2' '
000118                               95          DS     0F
000118                               96 RCODE    DS     F
00011C                               97 ENTRY    DS     A
000120 00000000                      98 ZERO     DC     4X'00'
000124                               99 PLIST    DS     2A
00012C 00000000                     100 CKADDR   DC     V(CKFILE)
                                    101          END

      RELOCATION DICTIONARY      PAGE    4
-POS.ID   REL.ID   FLAGS   ADDRESS
0 0001     0002      1C     00012C

      CROSS-REFERENCE            PAGE    5
-SYMBOL    LEN    VALUE   DEFN    REFERENCES
 BOMB      00004 00000094 00078  00054
 CKADDR    00004 0000012C 00100  00051
 ENTRY     00004 0000011C 00097  00060 00064
```

```
 LOAD      00001 00000000 00030  00036
 NAME      00008 00000104 00092  00042 00046 00057
 PLIST     00004 00000124 00099  00047 00049 00050
 RCODE     00004 00000118 00096  00048
 SAVEAREA  00004 000000B8 00090  00037
 TEMP      00004 00000100 00091  00045 00061 00078
 ZERO      00001 00000120 00098  00053 00065 00080

       LITERAL CROSS-REFERENCE    PAGE    6
-SYMBOL    LEN   VALUE   DEFN    REFERENCES
0=F'1'    00004 00000090 00074  00044
```

figure A-3

APPENDIX B
A Sample DEX Database Editing Session

```
start main
EXECUTION BEGINS...
 DEX VERSION 82.1(35 98883)
 $DEX AT MIT WELCOMES USER CHRYSDX
 $ON 82/138
 $AT 14.39.27.00
 $THIS  IS THE MAY 1982 DEX VERSION
 $ENTER AN ITEM FROM MENU - DEX.MAIN
open-db
 $ENTER THE NAME OF THE DATA-BASE
 $FILE NAME
 $ENTER UP TO 24 CHARACTERS
example database<
 $DO YOU WISH TO USE A PREVIOUSLY CREATED A DATABASE?
 $ENTER AN ITEM FROM MENU - DXYES-NO
no
 $ENTER AN ITEM FROM MENU - DEX.MAIN
edit-db
 $THE OPEN DATABASE IS NEW
 FILE:EXAMPLE DATABASE
 $*** NO TITLE ***
 $ENTER A DATABASE COMMAND
 $ENTER AN ITEM FROM MENU - DBEDCMDS
set-titl
 $PLEASE ENTER DATABASE TITLE
 $ENTER UP TO 64 CHARACTERS
sample database<
 $ENTER A DATABASE COMMAND
 $ENTER AN ITEM FROM MENU - DBEDCMDS
create
 $ENTER TYPE
 $ENTER AN ITEM FROM MENU - DB-TYPES
real
 $ENTER NAME
length
 $ENTER A DATABASE COMMAND
 $ENTER AN ITEM FROM MENU - DBEDCMDS
create real diameter
 $ENTER A DATABASE COMMAND
 $ENTER AN ITEM FROM MENU - DBEDCMDS
create real height
 $ENTER A DATABASE COMMAND
 $ENTER AN ITEM FROM MENU - DBEDCMDS
create array-rl speedar
 $ERROR IN READING AN INTEGER VALUE
```

```
 $BAD CHARACTER IS FLAGGED. PRIOR ITEMS ACCEPTED.
 $THE BAD VALUE AND SUBSEQUENT ITEMS MUST BE REENTERED
 $CREATE ARRAY-RL SPEEDAR
 $                  $ENTER UP TO   1 INTEGER NUMBERS
5
 $ENTER NAME
speedar
 $ENTER A DATABASE COMMAND
 $ENTER AN ITEM FROM MENU - DBEDCMDS
create integer ncrew
 $ENTER A DATABASE COMMAND
 $ENTER AN ITEM FROM MENU - DBEDCMDS
create
 $ENTER TYPE
 $ENTER AN ITEM FROM MENU - DB-TYPES
array-rl
 $ENTER ARRAY SIZE
 $ENTER UP TO   1 INTEGER NUMBERS
9
 $ENTER NAME
power
 $ENTER A DATABASE COMMAND
 $ENTER AN ITEM FROM MENU - DBEDCMDS
store
 $ENTER TYPE
 $ENTER AN ITEM FROM MENU - DB-TYPES
real
 $ENTER NAME
length
 $ENTER UP TO   1 REAL NUMBERS
460.
*
 $ENTER A DATABASE COMMAND
 $ENTER AN ITEM FROM MENU - DBEDCMDS
delete
 $ENTER NAME
height
 $ENTER A DATABASE COMMAND
 $ENTER AN ITEM FROM MENU - DBEDCMDS
store
 $ENTER TYPE
 $ENTER AN ITEM FROM MENU - DB-TYPES
real
 $ENTER NAME
ncrew
 $WRONG TYPE
 $ENTER A DATABASE COMMAND
 $ENTER AN ITEM FROM MENU - DBEDCMDS
store integer ncrew
```

```
 $ENTER UP TO   1 INTEGER NUMBERS
100
 $ENTER A DATABASE COMMAND
 $ENTER AN ITEM FROM MENU - DBEDCMDS
store
 $ENTER TYPE
 $ENTER AN ITEM FROM MENU - DB-TYPES
array-rl
 $ENTER ARRAY SIZE
 $ENTER UP TO   1 INTEGER NUMBERS
3
 $ENTER NAME
speedar
 $ENTER UP TO   3 REAL NUMBERS
5. 10.
 $ENTER UP TO   1 ADDITIONAL REAL NUMBERS
20.
 $ENTER A DATABASE COMMAND
 $ENTER AN ITEM FROM MENU - DBEDCMDS
store array-rl 3 power
 $ENTER UP TO   3 REAL NUMBERS
80000. 100000. 150000.
 $ENTER A DATABASE COMMAND
 $ENTER AN ITEM FROM MENU - DBEDCMDS
print diameter
 $DIAMETER
 $=UNDEFINED
 $ENTER A DATABASE COMMAND
 $ENTER AN ITEM FROM MENU - DBEDCMDS
print length
 $LENGTH
 $    0.460000E+03
 $ENTER A DATABASE COMMAND
 $ENTER AN ITEM FROM MENU - DBEDCMDS
dump
 $TITLE: **SAMPLE DATABASE


 $  NAME     TYPE            VALUE  COMMENT

    NCREW     (I)              100
    DIAMETER (R)    **UNDEFINED**
    POWER     (A)     ARRAY(    7)
    SPEEDAR   (A)     ARRAY(    1)
    LENGTH    (R)      0.46000E+03


 THE ARRAYS
```

```
 $NAME: SPEEDAR , LENGTH=  3   (    1)

 $   0.50000E+01   0.10000E+02   0.20000E+02

 $NAME: POWER   , LENGTH=  3   (    7)

 $   0.80000E+05   0.10000E+06   0.15000E+06
 $ENTER A DATABASE COMMAND
 $ENTER AN ITEM FROM MENU - DBEDCMDS
done
 $ENTER AN ITEM FROM MENU - DEX.MAIN
close-db
 $THE OPEN DATABASE IS
 FILE:EXAMPLE DATABASE
 $DO YOU WISH TO SAVE DATABASE?
 $ENTER AN ITEM FROM MENU - DXYES-NO
no
 $ENTER AN ITEM FROM MENU - DEX.MAIN
quit-dex
```

APPENDIX C

DEX ASSEMBLY LANGUAGE ROUTINES

```
         TITLE 'ABTRAP - TRAP AN ABNORMAL END CONOITION'                 ABT00010
*****************************************************************      ABT00020
*                                                                       ABT00030
*        ABTRAP -                                                       ABT00040
*                                                                       ABT00050
*                  TRAPS AN ABENO CONDITION BY INTERCEPTING             ABT00060
*                  THE ISSUEING OF AN SVC 13 (SUPERVISOR CALL 13)       ABT00070
*                  WHICH IS USEO BY THE SYSTEM AS AN ABNORMAL           ABT00080
*                  TERMINATION INTERUPT.                                ABT00090
*                                                                       ABT00100
*                  THIS ROUTINE SHOULD BE CALLED ONLY ONCE              ABT00110
*                  OURING THE INITIALIZATION OF DEX.  IT SETS UP        ABT00120
*                  THE TRAP CONDITION ANO IDENTIFIES TO THE SYSTEM      ABT00130
*                  THE NAME OF THE OEX SUBROUTINE THAT WILL HANDLE      ABT00140
*                  ANY ABENDS.                                          ABT00150
*                                                                       ABT00160
*        CALLING SEQUENCE :                                             ABT00170
*                                                                       ABT00180
*        TRUTHFLG=ABTRAP(DUMMY)                                         ABT00190
*                                                                       ABT00200
*            WHERE OUMMY IS A OUMMY ARGUMENT TO SATISFY FORTRAN         ABT00210
*            LOGICAL FUNCTION CONVENTION.                               ABT00220
*                                                                       ABT00230
*            TRUTHFLG IS 'TRUE' IF THE ABEND TRAP IS PROPERLY SET UP    ABT00240
*            TRUTHFLG IS 'FALSE' IF THE SVC TRAP COULD NOT BE SET UP    ABT00250
*                                                                       ABT00260
*            USES THE HNOSVC MACRO, SEE THE CMS COMMAND AND MACRO       ABT00270
*            REFERENCE GUIDE.                                           ABT00280
*                                                                       ABT00290
*****************************************************************      ABT00300
ABTRAP   IN    CSECT,STORAGE=O,SAVE=SAVE                               ABT00310
***                                                                    ABT00320
         HNOSVC SET,(13,ABNORM),ERROR=NOPE      CMS MACRO               ABT00330
***                                                                    ABT00340
OUT      OUT   STORAGE=O,SAVE=SAVE,RETC=1,RETREG=O    RETURN 'TRUE'     ABT00350
NOPE     OUT   STORAGE=O,SAVE=SAVE,RETC=O,RETREG=O    RETURN 'FALSE'    ABT00360
*****************************************************************      ABT00370
***      OATA CONSTANT AREA                                            ABT00380
***                                                                    ABT00390
SAVE     OS    18F                                                     ABT00400
*****************************************************************      ABT00410
*                                                                       ABT00420
*        ABNORM IS THE RETURN POINT IF AN ABEND IS TRAPPED BY THE       ABT00430
*        SYSTEM.  IT PROVIOES LINKAGE TO THE DEX ABNORMAL END ROUTINE   ABT00440
*        OXABND.                                                        ABT00450
*                                                                       ABT00460
*****************************************************************      ABT00470
```

```
ABNORM   STM    14,12,12(13)       SAVE REGISTERS IN AREA PASSED IN R13  ABT00480
         USING ABNORM,12           USE R12 FOR LINKAGE PER HNOSVC DOC.   ABT00490
         HNOSVC CLR,13             CLEAR SVC 13 TRAP                     ABT00500
         L      15,ADCON           LOAD R15 WITH POINTER TO DXABNO       ABT00510
         BALR   14,15              BRANCH TO DXABNO ROUTINE              ABT00520
         LM     14,12,12(13)       RESTORE REGISTERS                     ABT00530
         BR     14                 RETURN TO CONTROL PROGRAM             ABT00540
*************************************************************************ABT00550
***      ADDRESS CONSTANT AREA                                           ABT00560
***                                                                      ABT00570
ADCON    DC     V(DXABNO)          EXTERNAL ROUTINE ADDR CONSTANT        ABT00580
         END                                                             ABT00590
```

```
         TITLE 'ALLOC - ALLOCATE A FORTRAN FILE'                        ALL00010
*********************************************************************** ALL00020
*                                                                       ALL00030
*        ALLOC - SUBROUTINE TO ALLOCATE A FORTRAN FILE WITH OR          ALL00040
*                WITHOUT A POINTER TO THE END OF THE FILE               ALL00050
*                USES THE SUPERVISOR CALL (SVC 202) TO INVOKE           ALL00060
*                THE CMS FILEDEF COMMAND                                ALL00070
*                                                                       ALL00080
*        CALLING SEQUENCE -                                             ALL00090
*                                                                       ALL00100
*        RET =  ALLOC(LUN,FILENAME,WFLAG)                               ALL00110
*                                                                       ALL00120
*        WHERE LUN IS THE LUN, (IN CHARS), AND FILENAME IS A 24 BYTE    ALL00130
*        STRING CONTAINING THE FILE NAME, SEGREGATED INTO 8 BYTE PIECES ALL00140
*                                                                       ALL00150
*        WFLAG IS A FLAG INDICATING WHETHER THE FLAG IS TO BE ALLOCATED ALL00160
*        FOR READ OR WRITE.  IF WFLAG IS TRUE THE FILE IS ALLOCATED WIT ALL00170
*        A POINTER TO THE END OF THE FILE                               ALL00180
*                                                                       ALL00190
*        IT CAN ALSO BE A '*', IN WHICH CASE THE TERMINAL IS ALLOCATED  ALL00200
*                                                                       ALL00210
*        RET IS THE RETURN FROM THE FILEDEF COMMAND                     ALL00220
*                                                                       ALL00230
*        FOR FURTHER INFO, SEE THE CMS COMMAND AND MACRO REFERENCE      ALL00240
*                                                                       ALL00250
*********************************************************************** ALL00260
         EJECT                                                          ALL00270
ALLOC    IN    CSECT,STORAGE=D                                          ALL00280
         LM    2,4,D(1)          PICK UP PARMS                          ALL00290
         L     6,D(2)            PICK UP LUN                            ALL00300
         CLI   D(3),C'*'         IS IT THE TERMINAL?                    ALL00310
         BNE   CKFLAG            NO, MUST BE A DISK FILE                ALL00320
         ST    6,TERMPARM+8      STORE LUN                              ALL00330
         LA    1,TERMPARM        LOAD ADDR OF PARM LIST                 ALL00340
         B     SVC               AND GO SVC                             ALL00350
CKFLAG   CLI   D(4),=F'D'        IS THE WFLAG FALSE                     ALL00360
         BNE   FILEW             NO, THEN ALLOCATE FOR WRITE            ALL00370
*********************************************************************** ALL00380
***   ALLOCATE THE FILE USING PLIST                                     ALL00390
***                                                                     ALL00400
FILE     ST    6,FILENUM         STORE IN PARM LIST                     ALL00410
         MVC   FILENAME(24),D(3)   FILENAME TO BE ALLOCATED             ALL00420
         LA    1,PLIST           COMMAND POINTER                        ALL00430
         B     SVC               AND GO SVC                             ALL00440
*********************************************************************** ALL00450
***   ALLOCATE THE FILE FOR WRITE USING PLISTW                          ALL00460
***                                                                     ALL00470
```

```
FILEW    ST    6,FILENUMW      STORE IN PARM LIST                        ALL00480
         MVC   FILENAMW(24),0(3)   FILENAME TO BE ALLOCATED              ALL00490
         LA    1,PLISTW        COMMAND POINTER                           ALL00500
*****************************************************************      ALL00510
***   THE SUPERVISOR CALL FOR SENDING A COMMAND TO THE SYSTEM           ALL00520
***                                                                     ALL00530
SVC      SVC   202             EXECUTE COMMAND                           ALL00540
         DC    AL4(BOMB)       ERROR RETURN ADDRESS                      ALL00550
*****************************************************************      ALL00560
***   SET UP THE RETURN                                                 ALL00570
***                                                                     ALL00580
         OUT   STORAGE=0,RETC=0,RETREG=0                                 ALL00590
BOMB     LA    10,15                                                     ALL00600
         OUT   STORAGE=0,RETC=0(10),RETREG=0 RETURN WITH CODE            ALL00610
*****************************************************************      ALL00620
***   DATA CONSTANT AREA                                                ALL00630
***                                                                     ALL00640
PLIST    DS    0D                                                        ALL00650
         DC    CL8'FILEDEF'    COMMAND                                   ALL00660
FILENUM  DC    CL8' '          UNIT NUMBER                               ALL00670
         DC    CL8'DISK'                                                 ALL00680
FILENAME DC    3CL8' '         3 PARTS OF FILE NAME                      ALL00690
         DC    CL8'('          INDICATE START OF OPTION LIST             ALL00700
         DC    CL8'PERM'       SAVE UNTIL FREED                          ALL00710
         DC    8X'FF'          END OF STRING INDICATOR                   ALL00720
*                                                                        ALL00730
*-----------------------------------------------------------------*      ALL00740
*                                                                        ALL00750
PLISTW   DS    0D                                                        ALL00760
         DC    CL8'FILEDEF'    COMMAND                                   ALL00770
FILENUMW DC    CL8' '          UNIT NUMBER                               ALL00780
         DC    CL8'DISK'                                                 ALL00790
FILENAMW DC    3CL8' '         3 PARTS OF FILE NAME                      ALL00800
         DC    CL8'('          INDICATE START OF OPTION LIST             ALL00810
         DC    CL8'DISP'                                                 ALL00820
         DC    CL8'MOD'                                                  ALL00830
         DC    CL8'PERM'       SAVE UNTIL FREED                          ALL00840
         DC    8X'FF'          END OF STRING INDICATOR                   ALL00850
*                                                                        ALL00860
*-----------------------------------------------------------------*      ALL00870
*                                                                        ALL00880
TERMPARM DC    CL8'FILEDEF',CL8' ',CL8'TERM',CL8'(',CL8'PERM'            ALL00890
         DC    CL8'RECFM'                                                ALL00900
         DC    CL8'F',CL8'BLKSIZE',CL8'140'                              ALL00910
         DC    CL8'LRECL',CL8'140',8X'FF'                                ALL00920
         END                                                             ALL00930
```

```
         TITLE 'CKFILE - VERIFY THE EXISTANCE OF A FILE'                CKF00010
*********************************************************************** CKF00020
*                                                                       CKF00030
*        CKFILE -                                                       CKF00040
*                                                                       CKF00050
*              VERIFIES A FILE'S EXISTANCE BY FILENAME                  CKF00060
*                                                                       CKF00070
*                                                                       CKF00080
*        CALLING SEQUENCE :                                             CKF00090
*                                                                       CKF00100
*        RET = CKFILE(FILENAME,RCODE)                                   CKF00110
*                                                                       CKF00120
*              WHERE FILENAME IS AN 18 CHARACTER STRING CONTAINING      CKF00130
*              THE FILENAME AND RET IS 'TRUE' IS THE FILE EXISTS        CKF00140
*              OR 'FALSE' IF IT DOES NOT.                               CKF00150
*                                                                       CKF00160
*              RCODE IS AN ERROR RETURN CODE.                           CKF00170
*              RCODE IS OF INTEGER TYPE.                                CKF00180
*                                                                       CKF00190
*                   RCODE = 0     NO ERROR                            CKF00200
*                   RCODE = 1     INVALID CHARACTER IN FILEID         CKF00210
*                   RCODE = 2     INVALID FILEMODE                    CKF00220
*                   RCODE = 3     FILE NOT FOUND                      CKF00230
*                   RCODE = 4     DISK NOT ACCESSED                   CKF00240
*                                                                       CKF00250
*             USES THE FSSTATE MACRO.  FOR FURTHER INFO SEE THE         CKF00260
*             'CMS COMMAND AND MACRO REFERENCE GUIDE', SC19-6209        CKF00270
*                                                                       CKF00280
*********************************************************************** CKF00290
CKFILE   IN    CSECT,STORAGE=D,SAVE=SAVE                               CKF00300
         LM    2,3,0(1)       PICK UP THE FILENAME FROM PARAMETER LIST CKF00310
         ST    13,TEMP        SAVE R13 UNTIL AFTER FSSTATE MACRO       CKF00320
         FSSTATE (2),ERROR=NOPE   CMS MACRO TO CHECK STATUS OF A FILE  CKF00330
         L     13,TEMP        RESTORE R13                              CKF00340
         SR    4,4            ZERO OUT R4 TO RETURN A ZERO RCODE       CKF00350
         ST    4,RCODE        SET RCODE = 0 NO ERROR                   CKF00360
         MVC   0(4,3),RCODE   MOVE RCODE INTO ARG LIST                 CKF00370
         OUT   STORAGE=D,SAVE=SAVE,RETC=1,RETREG=0     RETURN 'TRUE'   CKF00380
*********************************************************************** CKF00390
***      ERROR SECTION:  CHECK THE RETURN CODES IN R15 FOR ERROR TYPE   CKF00400
***                                                                     CKF00410
NOPE     CL    15,=F'20'      CHECK FOR INVALID CHARACTER IN FILEID    CKF00420
         BE    RC01                                                    CKF00430
         CL    15,=F'24'      CHECK FOR INVALID FILEMODE               CKF00440
         BE    RC02                                                    CKF00450
         CL    15,=F'28'      CHECK FOR FILE NOT FOUND                 CKF00460
         BE    RC03                                                    CKF00470
```

```
         CL    15,=F'36'     CHECK FDR DISK NDT ACCESSED                CKF00480
         BE    RCD4                                                     CKF00490
         B     ERROUT                                                   CKF00500
RCD1     L     4,=F'1'       SET RCDDE = 1                              CKF00510
         ST    4,RCDDE                                                  CKF00520
         B     ERRDUT                                                   CKF00530
RCD2     L     4,=F'2'       SET RCODE = 2                              CKF00540
         ST    4,RCDDE                                                  CKF00550
         B     ERRDUT                                                   CKF00560
RCD3     L     4,=F'3'       SET RCODE = 3                              CKF00570
         ST    4,RCDDE                                                  CKF00580
         B     ERRDUT                                                   CKF00590
RCD4     L     4,=F'4'       SET RCDDE = 4                              CKF00600
         ST    4,RCODE                                                  CKF00610
*************************************************************************CKF00620
ERROUT   MVC   O(4,3),RCODE  MDVE RCODE INTO ARG LIST                   CKF00630
         DUT   STDRAGE=O,SAVE=SAVE,RETC=O,RETREG=O     RETURN 'FALSE'   CKF00640
*************************************************************************CKF00650
***      DATA STORAGE AREA                                              CKF00660
***                                                                     CKF00670
SAVE     DS    18F           REGISTER SAVE AREA                         CKF00680
RCDDE    DS    F             RETURN CODE                                CKF00690
TEMP     DS    F             TEMPORARY WDRD DF STORAGE                  CKF00700
         END                                                            CKF00710
```

```
         TITLE 'CLOSE - CLOSE A FORTRAN FILE AND CLEAN UP'                  CL000010
*********************************************************************** CL000020
*                                                                       CL000030
*        CLOSE -                                                        CL000040
*                                                                       CL000050
*        CLOSE A FILE FOR FORTRAN, TO ALLOW A CLEANUP TO OCCUR,         CL000060
*              FOR PURPOSES OF DYNAMIC FILE ALLOCATION                  CL000070
*                                                                       CL000080
*        CALLING SEQUENCE                                               CL000090
*                                                                       CL000100
*        CALL CLOSE(LUN,&NOTGOOD)                                       CL000110
*                                                                       CL000120
*        WHERE LUN IS THE LUN(FULL-WORD BINARY), AND THE &NOTGOOD IS    CL000130
*        A SECONDARY RETURN LABEL (FORTRAN RETURN I CONVENTION) IN THE  CL000140
*        FORTRAN ROUTINE WHICH CALLED CLOSE. THIS RETURN IS TAKEN       CL000150
*        IF THE FILE , FOR WHATEVER REASON, COULD NOT BE CLOSED.        CL000160
*                                                                       CL000170
*        MAKES USE OF INTIMATE KNOWLEDGE OF THE FORTRAN OBJECT TIME     CL000180
*        LIBRARY ROUTINES, DOCUMENTED IN THE 'IBM FORTRAN IV            CL000190
*        LIBRARY PLM'                                                   CL000200
*                                                                       CL000210
*        FORTRAN KEEPS A TABLE OF DCB ADDRS                             CL000220
*              CALLED IHNUATBL IN ONE VERSION OF THE LIBRARY, IHDUATBL  CL000230
*              IN THE OTHER.  IT CONTAINS A FOUR WORD ENTRY FOR EACH    CL000240
*              LUN, WHICH CONTAINS THE DCB ADDRESS FOR THE FILE, OR A   CL000250
*              VALUE OF 1 TO INDICATE AN UNUSED ENTRY                   CL000260
*                                                                       CL000270
*********************************************************************** CL000280
         EJECT                                                          CL000290
CLOSE    IN    CSECT,STORAGE=D,SAVE=SAVEAREA                           CL000300
         L     2,0(1)          PICK UP PARAMETER                       CL000310
         L     11,MODIIADD      MOD II LIBRARY UNIT TABLE              CL000320
         LTR   11,11           WAS IT THERE?                           CL000330
         BNZ   GOAHEAD         YES, GO AHEAD                           CL000340
         L     11,MODIADD      NO? THEN TRY MOD I LIBRARY              CL000350
         LTR   11,11           IS THIS ONE HERE                        CL000360
         BNZ   NOCANDO         NEITHER? THEN I CAN'T DO ANYTHING       CL000370
GOAHEAD  DS    0H              CONTINUE                                CL000380
         USING UATBL,11                                                CL000390
         L     2,0(,2)         GET LUN                                 CL000400
         LTR   2,2             TEST FOR VALIDITY                       CL000410
         BNH   NOCANDO         NEGATIVE OR ZERO                        CL000420
         BCTR  2,0             DECREMENT TO CORRECT OFFSET             CL000430
         SLL   2,4             MULTIPLY BY 16 FOR OFFSET               CL000440
         LH    3,UATABLEN      LOAD THE LENGTH                         CL000450
         CR    2,3             IS THE LUN TOO BIG??                    CL000460
         BH    NOCANDO                                                 CL000470
```

```
         LA    10,8(2,11)      ADDRESS OF UNIT ENTRY                      CL000480
         USING UNITNTRY,10     ESTABLISH ADDRESSABLIITY                   CL000490
         L     9,UABLKAD       ADDRESS OF UNIT BLOCK                      CL000500
         LA    5,1             GET A ONE FOR PROCESSING USE               CL000510
         CR    9,5             IS THE UNIT IN USE?                        CL000520
         BE    DONE            NO, THEN WE DON'T HAVE TO CLOSE IT         CL000530
         LA    8,76(,9)        POINT TO DCB IN UNIT BLOCK                 CL000540
         CLOSE ((8))           AND CLOSE IT                               CL000550
         FREEMAIN R,LV=164,A=(9) AND RETURN THE STORAGE                   CL000560
         ST    5,UABLKAD       AND RESET UNIT ENTRY TO SHOW 'UNUSED'      CL000570
DONE     OUT   STORAGE=0,RETC=0,SAVE=SAVEAREA OK, WE'RE SUCESSFUL         CL000580
NOCANDO  OUT   STORAGE=0,RETC=4,SAVE=SAVEAREA OOPS  CAN'T DO IT          CL000590
MODIADD  DC    V(IHNUATBL)                                                CL000600
MODIIADD DC    V(IHDUATBL)                                                CL000610
SAVEAREA DS    72C                                                        CL000620
UATBL    DSECT                                                            CL000630
UAPREFIX DS    DD                                                         CL000640
UACURRNT DS    H               CURRENT LUN IN PROGRESS??                  CL000650
UATABLEN DS    H               NUMBER OF ENTRIES IN TABLE                 CL000660
DEFDSRNS DS    4X            DSRNS FOR DEFAULT UNITS(MSG,READ,PRINT,PUN)  CL000670
UNITNTRY EQU   *                                                          CL000680
UABLKAD  DS    A               ADDRESS OF UNIT BLOCK, OR '1' IF UNUSED    CL000690
DSA      DSECT                                                            CL000700
         END                                                              CL000710
```

```
         TITLE 'FREE - FREE A FORTRAN FILE'                                    FRE00010
*****************************************************************            FRE00020
*                                                                              FRE00030
*        FREE -                                                                FRE00040
*                                                                              FRE00050
*        SUBROUTINE TO FREE A FORTRAN FILE                                     FRE00060
*                                                                              FRE00070
*        CALLING SEQUENCE -                                                    FRE00080
*                                                                              FRE00090
*              CALL FREE(FILENUM<,FILENAME>,&NOGOOD)                           FRE00100
*                                                                              FRE00110
*        WHERE FILENUM IS THE TWO CHARACTERS REPRESENTING THE                  FRE00120
*              LUN (E. G. 2H10), AND FILENAME                                  FRE00130
*              IS THE NAME OF THE FILE TO BE FREED (DEFAULTS TO                FRE00140
*              WHATEVER IS ALLOCATED TO FTNNF001), AND                         FRE00150
*              &NOGOOD IS A STATEMENT TO RETURN TO UPON                        FRE00160
*              NON COMPLETION                                                  FRE00170
*                                                                              FRE00180
*        NOTE: SHOULD BE CHANGED  TO ALLOW ONE CHARACTER NUMBERS               FRE00190
*        THIS VERSION (JULY 1978) DOES TRY TO CLEAN UP AND CLOSE               FRE00200
*              THE FILE, BY CALLING THE SUBROUTINE CLOSE                       FRE00210
*****************************************************************            FRE00220
FREE     IN    CSECT,STORAGE=0,SAVE=SAVEAREA                                   FRE00230
         LM    2,3,0(1)                                                        FRE00240
         PACK  PACKED(8),0(2,2)                                                FRE00250
         CVB   5,PACKED       TO BINARY FOR CLOSE                              FRE00260
         ST    5,NUM                                                           FRE00270
         LA    1,CLOSPARM     PARM LIST FOR CLOSE                              FRE00280
         CALL  CLOSE                                                           FRE00290
         MVC   FILENUM(2),0(2)     PICK UP FILENUM                             FRE00300
*        NOTE - IGNORES FILENAME TOTALLY                                       FRE00310
         LA    1,PLIST                                                         FRE00320
         SVC   202                                                             FRE00330
         DC    AL4(BOMB)                                                       FRE00340
         OUT   STORAGE=0,RETC=0,SAVE=SAVEAREA                                  FRE00350
BOMB     OUT   STORAGE=0,RETC=4,SAVE=SAVEAREA,RETREG=15                        FRE00360
PLIST    DS    0D                                                              FRE00370
         DC    CL8'FILEDEF'                                                    FRE00380
FILENUM  DC    CL8'01'                                                         FRE00390
         DC    CL8'CLEAR'                                                      FRE00400
         DC    8X'FF'                                                          FRE00410
PACKED   DS    D                                                               FRE00420
NUM      DS    F                                                               FRE00430
CLOSPARM DC    X'80',AL3(NUM)                                                  FRE00440
SAVEAREA DS    72C                                                             FRE00450
DSA      DSECT                                                                 FRE00460
         END                                                                   FRE00470
```

```
         TITLE 'FREEMN - FREES MAIN STORAGE (SIMULATED ECS)'            FRE00010
*************************************************************************FRE00020
*                                                                       FRE00030
*                                                                       FRE00040
*        SUBROUTINE FREEMN                                              FRE00050
*                                                                       FRE00060
*        CALL FREEMN(NWORDS,POINTR,RFCODE)                              FRE00070
*               ARGUMENTS NWORDS, POINTR, AND RFCODE ARE INTEGER TYPE   FRE00080
*                                                                       FRE00090
*        WHERE NWORDS IS THE NUMBER OF WORDS OF STORAGE TO FREE         FRE00100
*                                                                       FRE00110
*        WHERE POINTR IS THE POINTER TO THE STORAGE AREA TO BE FREED    FRE00120
*                                                                       FRE00130
*        WHERE RFCODE IS A FATAL RETURN CODE. ABNORMAL TERMINATION      FRE00140
*              IS ASSUMED AND RFCODE IS SET EQUAL TO THREE UPON ENTRY   FRE00150
*              TO THIS ROUTINE.  IF ABNORMAL TERMINATION OCCURS         FRE00160
*              RFCODE = 3 INDICATES THE FAILURE OCCURED HERE.  IF THE   FRE00170
*              TERMINATION IS NORMAL RFCODE IS SET TO ZERO PRIOR TO     FRE00180
*              RETURN TO THE CALLING ROUTINE.                           FRE00190
*                                                                       FRE00200
*        FREES NWORDS WORDS OF STORAGE, VIA FREEMAIN, STARTING AT THE   FRE00210
*        ADDRESS POINTED TO BY POINTR                                   FRE00220
*                                                                       FRE00230
*        USES THE SUPERVISOR SERVICES MACRO INSTRUCTION FREEMAIN, FOR   FRE00240
*        FURTHER INFORMATION SEE 'OS/VS1 SUPERVISOR SERVICES AND MACRO  FRE00250
*        INSTRUCTIONS', IBM GC24-5103                                   FRE00260
*                                                                       FRE00270
*************************************************************************FRE00280
FREEMN   IN    CSECT,STORAGE=0,SAVE=SAVEAREA                           FRE00290
         LM    2,4,0(1)              PICK UP ARG1, ARG2, AND ARG3 ADDR FRE00300
         L     2,0(2)                AMT (IN WORDS) OF STORAGE TO FREE FRE00310
         L     3,0(3)                POINTER TO AREA TO BE FREED       FRE00320
         MVC   0(4,4),=F'3'  SET RFCODE = 3 FREEMN CAUSED ANY FATAL ERR FRE00330
         ST    3,POINTR              SAVE IN POINTR                    FRE00340
         SLL   2,2                   CHANGE NWORDS INTO BYTES          FRE00350
         FREEMAIN E,LV=(2),A=POINTR  FREE STORAGE                      FRE00360
         MVC   0(4,3),ZERO   RESET RFCODE = 0, NORMAL TERMINATION      FRE00370
         OUT   STORAGE=0,SAVE=SAVEAREA      AND RETURN                 FRE00380
*************************************************************************FRE00390
***      DATA CONSTANT AREA                                             FRE00400
***                                                                     FRE00410
SAVEAREA DS    18F            AREA IN WHICH TO SAVE REGISTERS          FRE00420
POINTR   DS    A              POINTER TO THE AREA TO BE RELEASED       FRE00430
ZERO     DC    4X'00'         ZERO FULL WORD FOR COMPARISON            FRE00440
         END                                                           FRE00450
```

```
         TITLE 'GETMN - GETS A BLOCK OF MAIN STORAGE (SIMULATED ECS)'   GET00010
*************************************************************************GET00020
*                                                                       GET00030
*                                                                       GET00040
*        SUBROUTINE GETMN                                               GET00050
*                                                                       GET00060
*        TRUTHVALUE = GETMN(NWOROS,POINTR)                              GET00070
*                                                                       GET00080
*                ARGUMENTS NWOROS, AND POINTR ARE OF INTEGER TYPE       GET00090
*                                                                       GET00100
*        WHERE NWOROS IS THE NUMBER OF WOROS OF STORAGE TO FREE         GET00110
*                                                                       GET00120
*        WHERE POINTR IS THE POINTER TO THE STORAGE AREA OBTAINED       GET00130
*                                                                       GET00140
*        ACQUIRES NWORDS WORDS OF STORAGE, VIA GETMAIN, AND RETURNS     GET00150
*        A POINTER TO THE ADDRESS OF THE AREA                           GET00160
*                                                                       GET00170
*        USES THE SUPERVISOR SERVICES MACRO INSTRUCTION GETMAIN, FOR    GET00180
*        FURTHER INFORMATION SEE 'OS/VS1 SUPERVISOR SERVICES AND MACRO  GET00190
*        INSTRUCTIONS', IBM GC24-5103                                   GET00200
*                                                                       GET00210
*                                                                       GET00220
*************************************************************************GET00230
GETMN    IN    CSECT,STORAGE=0,SAVE=SAVEAREA                            GET00240
         LM    2,3,0(1)                PICK UP ARG1, AND ARG2 ADDR      GET00250
         L     2,0(2)                  AMT (IN WOROS) OF STORAGE TO GET GET00260
         SLL   2,2                     INTO BYTES                       GET00270
         GETMAIN EC,LV=(2),A=POINTR    GET STORAGE                      GET00280
         CL    15,=F'4'                CHECK FOR ERROR RETURN           GET00290
         BE    NOSTOR                  NOT ENOUGH STORAGE               GET00300
         MVC   0(4,3),POINTR           RETURN ADDRESS IN ARG2=POINTER   GET00310
***                                                                      GET00320
         OUT   STORAGE=0,SAVE=SAVEAREA,RETC=1,RETREG=0  RETURN 'TRUE'   GET00330
***                                                                      GET00340
NOSTOR   OUT   STORAGE=0,SAVE=SAVEAREA,RETC=0,RETREG=0  RETURN 'FALSE'  GET00350
*************************************************************************GET00360
***      DATA CONSTANT AREA                                              GET00370
***                                                                      GET00380
SAVEAREA DS    18F             AREA IN WHICH TO SAVE REGISTERS          GET00390
POINTR   DS    A               POINTER TO THE ALLOCATED STORAGE AREA    GET00400
         END                                                            GET00410
```

```
         TITLE 'ISHIFT - SHIFT A WORD AROUND'                                   ISH00010
*****************************************************************************  ISH00020
*                                                                              ISH00030
*        ISHIFT                                                                ISH00040
*                                                                              ISH00050
*        SHIFT A WORD A SPECIFIED NUMBER OF BITS                               ISH00060
*                                                                              ISH00070
*        CALLING SEQUENCE                                                      ISH00080
*                                                                              ISH00090
*        CALL ISHIFT(WORD,NBITS)                                               ISH00100
*                                                                              ISH00110
*        SHIFTS THE WORD 'WORD' NBITS BITS TO THE LEFT                         ISH00120
*        (IF NBITS IS NEGATIVE, IT WILL SHIFT TO THE RIGHT)                    ISH00130
*                                                                              ISH00140
*****************************************************************************  ISH00150
ISHIFT   IN    CSECT,STORAGE=0                                                 ISH00160
***                                                                            ISH00170
         REGISTER                                                              ISH00180
         LM    R3,R4,0(R1)          PICK UP THE ARGUMENT LIST                  ISH00190
         LTR   R4,R4                LOAD AND TEST R4 FOR NEGATIVE              ISH00200
         BNM   LEFT                 IF NBITS NOT MINUS GO TO LEFT              ISH00210
***                                                                            ISH00220
         SRA   R3,0(R4)             OTHERWISE SHIFT THE WORD RIGHT             ISH00230
         B     OUT                  AND BRANCH OUT                             ISH00240
***                                                                            ISH00250
LEFT     LCR   R4,R4                                                           ISH00260
         SLA   R3,0(R4)             SHIFT THE WORD LEFT                        ISH00270
***                                                                            ISH00280
OUT      ST    R3,0(R1)             PUT THE SHIFTED WORD BACK IN ARG1          ISH00290
         OUT   STORAGE=0                                                       ISH00300
         END                                                                   ISH00310
```

```
         TITLE 'JGVMID - OBTAIN THE VM USERID FOR THE PERSON USING DEX'
*********************************************************************
*                                                                   *
*    ROUTINE TO RETURN THE VM USERIO                                
*        CALL JGVMID(VMID)                                          
*    WHERE VMID WILL BE 8 BYTES, EBCOIC                             
*        J. GRAHAM / USER SERVICES                                  
*        DATE:  02/28/78                                            
*********************************************************************
JGVMID   START 0                                                    
         STM   14,12,12(13)  SAVE REGISTERS                         
         BALR  12,0          ESTABLISH BASE                         
         USING *,12             REGISTER                            
*                                                                   
         L     2,0(1)         MOVE 1ST ARG ADDRESS TO R2            
         LA    6,RETAREA                                            
         LA    10,CMMOL       (MAX) NO. OF BYTES TO BE STORED       
         DC    X'83',X'6A',XL2'0000'                                
         MVC   0(8,2),USERID  MOVE ID INTO CALL LIST                 
*                                                                   
         LM    2,12,28(13)    RESTORE REGISTERS                     
         MVI   12(13),X'FF'   INDICATE CONTROL RETURNED TO CALLING PROG
         BCR   15,14          RETURN TO CALLING PROGRAM             
*********************************************************************
***      DATA STORAGE AREA                                          
***                                                                 
         DS    0D                                                   
SAVEAREA DS    18F                                                  
RETAREA  DS    30                                                   
CMMDL    EQU   *-RETAREA                                            
BUFFER   ORG   RETAREA                                              
DUMMY    DS    XL16                                                 
USERIO   DS    CL8                                                  
         END                                                        
```

```
         TITLE 'LOAO - OYNAMICALLY LDAO A MDOULE'                          LOA00010
*********************************************************************** LOA00020
*                                                                       LOA00030
*        LOAO -                                                         LOA00040
*                                                                       LOA00050
*        SUBRDUTINE TO LOAD A MDOULE                                    LOA00060
*                                                                       LOA00070
*        CALLING SEQUENCE -                                             LOA00080
*                                                                       LOA00090
*              TRUTHVALUE = LOAO(FILENAME,ENTRY,RFCODE)                 LOA00100
*                                                                       LOA00110
*        WHERE FILENAME IS THE FILENAME OF MODULE TO BE LOADEO.         LOA00120
*              THE FILETYPE IS ASSUMEO TD BE TEXT SINCE THAT IS         LOA00130
*              THE ONLY TYPE WHICH CAN BE EXECUTEO.                     LOA00140
*        WHERE ENTRY IS AN INTEGER ENTRY PDINT ADDR OF THE LOAOEO       LOA00150
*              MDOULE TO BE RETURNEO TO THE CALLING PROGRAM             LOA00160
*        WHERE RFCODE IS A FATAL RETURN COOE. ABNORMAL TERMINATION      LOA00170
*              IS ASSUMEO ANO RFCDDE IS SET EQUAL TO ONE UPON ENTRY     LOA00180
*              TO THIS ROUTINE.  IF ABNDRMAL TERMINATION DCCURS         LDA00190
*              RFCODE = 1 INOICATES THE FAILURE OCCUREO HERE.  IF THE   LOA00200
*              TERMINATION IS NDRMAL RFCDOE IS SET TO ZERO PRIDR TO     LDA00210
*              RETURN TO THE CALLING ROUTINE.                           LDA00220
*                                                                       LDA00230
*        TRUTHVALUE IS TRUE IF LDAD WAS SUCCESSFUL ANO FALSE IF         LOA00240
*              LOAO WAS UNSUCCESSFUL                                    LOA00250
*                                                                       LOA00260
*        USES THE DS/VS1 SUPERVISOR 'LOAD' MACRO.  FDR FURTHER INFO     LDA00270
*          SEE 'OS/VS1 SUPERVISOR SERVICES AND MACRO INSTRUCTIONS',     LOA00280
*             GC24-5103                                                 LDA00290
*                                                                       LOA00300
*********************************************************************** LDA00310
LOAD     IN    CSECT,STORAGE=O,SAVE=SAVEAREA                            LOA00320
         L     2,O(1)          LDAD THE AOOR OF ARG1 INTO R2            LDA00330
         MVC   NAME(8),O(2)    GET LOAO MDDULE NAME                     LDA00340
         L     3,8(1)          LDAD THE AOOR OF ARG3 INTO R3            LOA00350
         MVC   O(4,3),=F'1'    SET RFCDOE = 1 LOAO CAUSEO ANY FATAL ERR LOA00360
         ST    1,TEMP          SAVE R1 UNTIL AFER LOAO                  LOA00370
         LA    3,NAME          LOAD THE AOORESS OF NAME INTO R3 ANO     LOA00380
         ST    3,PLIST         STORE IT IN THE PARAMETER LIST           LOA00390
         LA    3,RCDOE         LOAO THE AOORESS OF THE RETURN COOE IN R3 LOA00400
         ST    3,PLIST+4       ANO STORE IT IN THE PARAMETER LIST       LOA00410
         LA    1,PLIST         LOAO R1 WITH PARAMETER LIST ADORESS      LOA00420
         L     15,CKAODR       LDAO THE ENTRY AOORESS FOR CKFILE        LOA00430
         BALR  14,15           BRANCH TO ENTERY PDINT                   LOA00440
         CL    O,ZERO          COMPARE RETURN COOE IN RO TO ZERO        LOA00450
         BE    BDMB            IF RETURN COOE = O LOAO MODULE NOT FOUND LOA00460
***                                                                      LOA00470
```

```
         LOAO EPLOC=NAME     LOAO MOOULE (EPLOC = ENTRY POINT LOCATION) LOA00480
         ST    O,ENTRY       STORE THE ENTRY POINT AOOR IN ENTRY        LOA00490
         L     1,TEMP        RESTORE R1                                 LOA00500
         L     2,4(1)        USE R2 FOR THE AOOR OF ARG2                LOA00510
         L     3,8(1)        USE R3 FOR THE AOOR OF ARG3                LOA00520
         MVC   O(4,2),ENTRY  STORE THE ENTRY AOOR IN ARG2               LOA00530
         MVC   O(4,3),ZERO   RESET RFCOOE = O, NORMAL TERMINATION       LOA00540
***                                                                     LOA00550
         OUT   STORAGE=O,SAVE=SAVEAREA,RETC=1,RETREG=O  RETURN 'TRUE'   LOA00560
***                                                                     LOA00570
**************************  TEXT FILE OIO NOT EXIST RETURN FALSE        LOA00580
***                                                                     LOA00590
BOMB     L     1,TEMP        RESTORE R1                                 LOA00600
         L     3,8(1)        USE R3 FOR THE AOOR OF ARG3                LOA00610
         MVC   O(4,3),ZERO   RESET RFCOOE = O, NORMAL TERMINATION       LOA00620
         OUT   STORAGE=O,SAVE=SAVEAREA,RETC=O,RETREG=O  RETURN  'FALSE' LOA00630
************************************************************************ LOA00640
***  OATA CONSTANT AREA                                                 LOA00650
SAVEAREA OS     18F          AREA FOR SAVING THE REGISTERS              LOA00660
TEMP     OS     F            TEMPORARY ONE REGISTER SAVE AREA           LOA00670
NAME     OC     CL8' '       FILENAME                                   LOA00680
         OC     CL8'TEXT'    FILETYPE                                   LOA00690
         OC     CL2' '       FILEMOOE                                   LOA00700
         OS     OF           ALLIGN ON FULL WORO BOUNOARY               LOA00710
RCOOE    OS     F            RETURN COOE FROM CKFILE STOREO HERE        LOA00720
ENTRY    OS     A            STORAGE AREA FOR MOOULE ENTRY AOORESS      LOA00730
ZERO     OC     4X'OO'       ZERO FULL WORO FOR COMPARISON              LOA00740
PLIST    OS     2A           PARAMETER LIST                             LOA00750
CKAOOR   OC     V(CKFILE)    ENTRY POINT AOORESS FOR CKFILE ROUTINE     LOA00760
         ENO                                                            LOA00770
```

```
         TITLE 'QUERY - QUERY THE FILEOEF FOR A GIVEN LUN'                 QUE00010
*********************************************************************** QUE00020
***                                                                     QUE00030
***             LOGICAL FUNCTION QUERY                                  QUE00040
***                                                                     QUE00050
***                 CALLING SEQUENCE:                                   QUE00060
***                                                                     QUE00070
***                 TRUTHFLG = QUERY(UNIT,RCODE,NAME,TYPE,MOOE,OEV)     QUE00080
***                                                                     QUE00090
***                 WHEN PASSED A FORTRAN I/O UNIT NUMBER (UNIT)        QUE00100
***                 THIS SUBROUTINE INTERCEPTS THE SYSTEM JOB FILE      QUE00110
***                 CONTROL BLOCK (JFCB) FOR OONAME = FTNNFOO1 WHERE    QUE00120
***                 NN = UNIT NUMBER.  THE FCBREAD FIELD IN THE JFCB IS QUE00130
***                 COMPARED TO ZERO, IF FCBREAO = O THEN NO FILEOEF    QUE00140
***                 EXISTS ANO QUERY2 = 'FALSE' IS RETURNEO (NAME,TYPE, QUE00150
***                 ANO MOOE ARE ZEROEO OUT).  IF THE FILEOEF EXISTS THE QUE00160
***                 INFORMATION ABOUT THE FILE IS EXTRACTED FROM THE    QUE00170
***                 JFCB BY MAPPING IT BY A OUMMY SECTION CMSCB MACRO   QUE00180
***                 ANO THE FOLLOWING ARE RETURNEO:                     QUE00190
***                                                                     QUE00200
***                     QUERY2 = 'TRUE'                                 QUE00210
***                     RCOOE = O                                       QUE00220
***                     NAME = FILENAME                                 QUE00230
***                     TYPE = FILETYPE                                 QUE00240
***                     MODE = FILEMODE                                 QUE00250
***                     OEV  = OEVICE                                   QUE00260
***                                                                     QUE00270
***                 USES THE ROJFCB SYSTEM MACRO.  FOR FURTHER INFO     QUE00280
***                 SEE THE 'OS/VS1 OATA MANAGEMENT FOR SYSTEM          QUE00290
***                 PROGRAMERS', GC26-3837-2                            QUE00300
***                                                                     QUE00310
***                                                                     QUE00320
*********************************************************************** QUE00330
*********************************************************************** QUE00340
***                                                                     QUE00350
QUERY    IN    CSECT,STORAGE=O,SAVE=SAVEAREA                            QUE00360
*********************************************************************** QUE00370
***  CONVERT UNIT NO. TO FTOOFOO1 FORMAT ANO INSERT IN OONAME FIELD     QUE00380
***  OF THE DATA CONTROL BLOCK (OCBOONAM)                               QUE00390
***                                                                     QUE00400
         LA    1O,FILEOEF            USE REGISTER 1O FOR MAPING OCB   QUE00410
         USING IHADCB,1O             IHAOCB IS BEGINING OF DCB OSECT  QUE00420
         L     2,O(1)                LOAO THE AOORESS OF ARG1         QUE00430
         L     3,O(2)                MOVE ARG1=UNIT INTO R3           QUE00440
         CVO   3,OECIMAL             CONVERT BINARY TO PACKEO DECIMAL QUE00450
         UNPK  OECIMAL(8),OECIMAL(8) UNPACK OECIMAL TO MAKE ZONEO     QUE00460
         OI    OECIMAL+7,X'FO'       CONVERT SIGN ZONE TO F ZONE      QUE00470
```

```
         L     3,UNIT                  LDAD FTOO INTO R3                QUE00480
         ICM   3,B'0011',DECIMAL+6     INSERT THE LAST 2 ZONED DIGITS   QUE00490
         ST    3,DCBOONAM              MDDIFY DCB NAME FDR RDJFCB       QUE00500
         L     5,UNIT+4                                                 QUE00510
         ST    5,DCBOONAM+4                                             QUE00520
*********************************************************************** QUE00530
***  READ JFCB AND CHECK FDR EXISTANCE DF THE FILEDEF                   QUE00540
***                                                                     QUE00550
         LA    11,JFCBAREA             USE REGISTER 11 FOR MAPING JFCB  QUE00560
         USING IHAJFCB,11              FCBSECT IS START OF CMSCB OSECT  QUE00570
         ST    1,TEMP                  SAVE R1 UNTIL AFTER RDJFCB MACRD QUE00580
         LA    1,MSGDFF                PREPARE TD ISSUE CP CDMMAND TD   QUE00590
         SVC   202                     TURN ERRDR MSG DFF AT TERMINAL   QUE00600
         DC    AL4(*+4)                IF ERRDR CONTINUE WITH NEXT INST QUE00610
         RDJFCB MF=(E,EX1)             SYS MACRD TD READ DATA CDNTRL BLK QUE00620
         CLC   FCBREAD(4),ZERD         CHECK FOR NO CDNTRDL BLDCK READ  QUE00630
         BE    NOFILE                  ND MATCH MEANS ND FILEDEF        QUE00640
*********************************************************************** QUE00650
***  NO ERROR, RETURN QUERY2 = 'TRUE' AND THE FILEDEF INFO              QUE00660
***                                                                     QUE00670
         DEVTYPE DCBDDNAM,DTYPE                                         QUE00680
         L     1,TEMP                  RESTDRE R1 AFTER RDJFCB MACRD    QUE00690
         L     3,4(1)                  USE R3 FDR ADDR DF ARG2          QUE00700
         L     4,8(1)                  USE R4 FDR ADDR DF ARG3          QUE00710
         L     5,12(1)                 USE R5 FDR ADDR DF ARG4          QUE00720
         L     6,16(1)                 USE R6 FDR ADDR DF ARG5          QUE00730
         L     7,20(1)                 USE R7 FDR ADDR DF ARG6          QUE00740
         MVC   0(8,3),=F'0'            PUT A 0 IN ARG2, ND FATAL RCDDE  QUE00750
         MVC   0(8,4),FCBDSNAM         STDRE THE DATA SET NAME IN ARG3  QUE00760
         MVC   0(8,5),FCBDSTYP         STDRE THE DATA SET TYPE IN ARG4  QUE00770
         MVC   0(2,6),FCBDSMD          STDRE THE DATA SET MDDE IN ARG5  QUE00780
***                                                                     QUE00790
         L     8,DTYPE                                                  QUE00800
         N     8,MASK                                                   QUE00810
         ST    8,DTYPE                                                  QUE00820
         CLC   DTYPE(4),DEV1          CHECK IF DEVICE CDDE IS A PRINTER QUE00830
         BNE   NEXT1                   IF ND MATCH CHECK NEXT TYPE     QUE00840
         MVC   0(8,7),PRINTR           IF MATCH STDRE PRINTER IN ARG6   QUE00850
         B     DUT                     AND RETURN                       QUE00860
***                                                                     QUE00870
NEXT1    CLC   DTYPE(4),DEV2          CHECK IF DEVICE CDDE IS A READER   QUE00880
         BNE   NEXT2                   IF ND MATCH CHECK NEXT TYPE     QUE00890
         MVC   0(8,7),READER           IF MATCH STDRE READER IN ARG6    QUE00900
         B     DUT                     AND RETURN                       QUE00910
***                                                                     QUE00920
NEXT2    CLC   DTYPE(4),DEV3          CHECK IF DEVICE CDDE IS A TERMINL  QUE00930
         BNE   NEXT3                   IF NO MATCH CHECK NEXT TYPE     QUE00940
```

```
         MVC   0(8,7),TERMNL          IF MATCH STORE TERMINAL IN ARG6   QUE00950
         B     OUT                    AND RETURN                        QUE00960
***                                                                     QUE00970
NEXT3    CLC   OTYPE(4),DEV4         CHECK IF OEVICE CODE IS A TAPE     QUE00980
         BNE   NEXT4                  IF NO MATCH CHECK NEXT TYPE       QUE00990
         MVC   0(8,7),TAPE            IF MATCH STORE TAPE IN ARG6       QUE01000
         B     OUT                    AND RETURN                        QUE01010
***                                                                     QUE01020
NEXT4    CLC   DTYPE(4),DEV5         CHECK IF OEVICE CODE IS A DISK     QUE01030
         BNE   NEXT5                  IF NO MATCH CHECK NEXT TYPE       QUE01040
         MVC   0(8,7),DISK            IF MATCH STORE DISK IN ARG6       QUE01050
         B     OUT                    AND RETURN                        QUE01060
***                                                                     QUE01070
NEXT5    CLC   DTYPE(4),DEV6         CHECK IF DEVICE CODE IS A PUNCH    QUE01080
         BNE   NEXT6                  IF NO MATCH CHEXK NEXT TYPE       QUE01090
         MVC   0(8,7),PUNCH           IF MATCH STORE PUNCH IN ARG6      QUE01100
         B     DUT                    AND RETURN                        QUE01110
***                                                                     QUE01120
NEXT6    CLC   DTYPE(4),OEV7         CHECK IF OEVICE CODE IS A CRT      QUE01130
         BNE   DUT                    IF NO MATCH DONE                  QUE01140
         MVC   0(8,7),CRT             IF MATCH STORE CRT IN ARG6        QUE01150
         B     OUT                    AND RETURN                        QUE01160
*********************************************************************** QUE01170
***  NO FILEOEF, RETURN FLAG = 0 AND BLANK FILE INFORMATION             QUE01180
***                                                                     QUE01190
NOFILE   SR    4,4                    UNIT NOT ASSIGNED RETURN ARG2=0   QUE01200
         L     1,TEMP                 RESTORE R1 AFTER ROJFCB MACRO     QUE01210
         L     3,4(1)                 USE R3 FOR THE ADOR OF ARG2       QUE01220
         L     5,8(1)                 USE R5 FOR THE ADDR OF ARG3       QUE01230
         L     6,12(1)                USE R6 FOR THE ADDR OF ARG4       QUE01240
         L     7,16(1)                USE R7 FOR THE ADOR OF ARG5       QUE01250
         L     8,20(1)                USE R8 FOR THE ADDR OF ARG6       QUE01260
         ST    4,0(3)                 STORE 0 IN RFCODE -> NOT FATAL    QUE01270
         ST    4,0(5)                 ZERO OUT FILE NAME                QUE01280
         ST    4,4(5)                                                   QUE01290
         ST    4,0(6)                 ZERO OUT FILE TYPE                QUE01300
         ST    4,4(6)                                                   QUE01310
         ST    4,0(7)                 ZERO OUT FILE MODE                QUE01320
         ST    4,0(8)                 ZERO OUT DEVICE                   QUE01330
         ST    4,4(8)                                                   QUE01340
***                                                                     QUE01350
         LA    1,MSGDN                PREPARE TO ISSUE CP COMMAND TO    QUE01360
         SVC   202                    RESTORE EMSG AT TERMINAL TO TEXT  QUE01370
         OC    AL4(*+4)               IF ERROR CONTINUE WITH NEXT INST  QUE01380
         OUT   STORAGE=0,SAVE=SAVEAREA,RETC=0,RETREG=0  RETURN 'FALSE'  QUE01390
*********************************************************************** QUE01400
DUT      LA    1,MSGDN                PREPARE TO ISSUE CP COMMAND TO    QUE01410
```

```
         SVC   202                      RESTORE EMSG AT TERMINAL TO TEXT  QUEO1420
         OC    AL4(*+4)                 IF ERROR CONTINUE WITH NEXT INST  QUEO1430
         OUT   STORAGE=O,SAVE=SAVEAREA,RETC=1,RETREG=O  RETURN 'TRUE'    QUEO1440
*************************************************************************** QUEO1450
EX1      ROJFCB (FILEOEF,),MF=L                                            QUEO1460
*************************************************************************** QUEO1470
***  DATA CONSTANT AREA                                                   QUEO1480
SAVEAREA OS    18F                      SAVE AREA FOR REGISTERS           QUEO1490
OTYPE    OS    2F                       RETURN AREA FOR OEVTYPE MACRO     QUEO1500
TEMP     OS    F                        TEMPORARY STORAGE AREA            QUEO1510
OCOOE    OS    F                        DEVICE COOE                       QUEO1520
OECIMAL  OS    D                        TEMP STORAGE FOR OECIMAL CONVERT  QUEO1530
MASK     OC    X'OOOOFFFF'              MASK FOR EXTRACTING OEVICE CODE   QUEO1540
UNIT     OC    C'FTOOFOO1'              AREA FOR CONSTRUCTING UNIT NO.    QUEO1550
ZERO     OC    X'OOOOOOOO'              ZERO WORO FOR COMPARISON          QUEO1560
***                                                                       QUEO1570
DEVLST   DS    OF                       LIST OF DEVICES TO RETURN IN OEV  QUEO1580
PRINTR   OC    CL8'PRINTER'                                               QUEO1590
READER   OC    CL8'REAOER'                                                QUEO1600
TERMNL   OC    CL8'TERMINAL'                                              QUEO1610
TAPE     OC    CL8'TAPE'                                                  QUEO1620
DISK     OC    CL8'OISK'                                                  QUEO1630
PUNCH    OC    CL8'PUNCH'                                                 QUEO1640
CRT      DC    CL8'CRT'                                                   QUEO1650
***                                                                       QUEO1660
OEV1     OC    X'OOOOO808'              DEVICE COOE FOR A PRINTER         QUEO1670
OEV2     OC    X'OOOOO801'              DEVICE COOE FOR A REAOER          QUEO1680
OEV3     OC    X'OOOOO820'              DEVICE CODE FOR A TERMINAL        QUEO1690
OEV4     OC    X'OOOO8001'              DEVICE COOE FOR A TAPE            QUEO1700
OEV5     OC    X'OOOO2008'              DEVICE COOE FOR A DISK            QUEO1710
OEV6     OC    X'OOOOO802'              OEVICE COOE FOR A PUNCH           QUEO1720
OEV7     OC    X'OOOO1C'                OEVICE COOE FOR A CRT             QUEO1730
***                                                                       QUEO1740
MSGOFF   OS    OO                       STORAGE FOR CP COMMANO TO TURN    QUEO1750
         OC    CL8'CP'                  ERROR MESSAGE OFF AT TERMINAL     QUEO1760
         OC    CL8'SET'                                                   QUEO1770
         OC    CL8'EMSG'                                                  QUEO1780
         OC    CL8'OFF'                                                   QUEO1790
         DC    8X'FF'                                                     QUEO1800
***                                                                       QUEO1810
MSGON    OS    OO                       STORAGE FOR CP COMMANO TO RESTORE QUEO1820
         DC    CL8'CP'                  ERROR MESSAGE TO TEXT AT TERMINAL QUEO1830
         OC    CL8'SET'                                                   QUEO1840
         OC    CL8'EMSG'                                                  QUEO1850
         OC    CL8'TEXT'                                                  QUEO1860
         OC    8X'FF'                                                     QUEO1870
*************************************************************************** QUEO1880
```

```
***  OATA CONTROL BLOCK                                                    QUEO1890
FILEDEF  OCB    OSORG=OA,EXLST=LST,MACRF=(RI),OONAME=DON                  QUEO1900
*************************************************************************** QUEO1910
***  AREA INTO WHICH RDJFCB MACRO OUMPS SYSTEM OATA CONTROL BLOCK          QUEO1920
LST      OS     OF                                                        QUEO1930
         OC     X'07'                                                     QUEO1940
         OC     AL3(JFCBAREA)                                             QUEO1950
JFCBAREA OS     OF,176C                                                   QUEO1960
*************************************************************************** QUEO1970
         CMSCB                          MACRO FOR OSECT TO MAP JFCB       QUEO1980
*************************************************************************** QUEO1990
         OCBD   OSORG=PS                MACRO FOR OSECT TO MAP OCB        QUEO2000
         ENO                                                              QUEO2010
```

```
         TITLE 'READEC AND WRITEC ROUTINES TO SIMULATE CDC ECS STORAGE' REA00010
*************************************************************************** REA00020
*                                                                         REA00030
*        READEC,WRITEC                                                    REA00040
*                                                                         REA00050
*        SUBROUTINES TO SIMULATE CDC ECS                                  REA00060
*                                                                         REA00070
*                                                                         REA00080
*        CALL READEC(STRING,OFFSET,LENWDS,ECSPTR,ECSLEN)                  REA00090
*                                                                         REA00100
*        'READS' LENWDS WORDS OF 'ECS', STARTING AT OFFSETFROM ECSPTR     REA00110
*              INTO STRING, STARTING AT BYTE ONE                          REA00120
*                                                                         REA00130
*        ECSPTR IS THE POINTER TO THE BEGINING OF THE ECS AREA            REA00140
*                                                                         REA00150
*        ECSLEN IS THE LENGTH OF THE ECS AREA                             REA00160
*                                                                         REA00170
*        CALL WRITEC(STRING,OFFSET,LENWDS,ECSPTR,ECSLEN)                  REA00180
*                                                                         REA00190
*       OPPOSITE OF READEC, SAME ARGS, BUT MOVE IS FROM STRING TO 'ECS'   REA00200
*                                                                         REA00210
*************************************************************************** REA00220
***                                                                       REA00230
READEC   IN    ENTRY,STORAGE=O                                           REA00240
         LM    2,6,O(1)        PICK UP ADDRESS OF ARGS                   REA00250
*                                                                         REA00260
*                                R2 = STRING                              REA00270
*                                R3 = OFFSET                              REA00280
*                                R4 = LENWDS                              REA00290
*                                R5 = ECSPTR                              REA00300
*                                R6 = ECSLEN                              REA00310
*                                                                         REA00320
         MVC   ECSPTR(4),O(5)     LOAD ECSPTR                             REA00330
         MVC   ECSLEN(4),O(6)     LOAD ECSLEN                             REA00340
         L     3,O(3)          CGET OFFSET INTO ECS                      REA00350
         SLL   3,2             ... IN BYTES                              REA00360
         C     3,ECSLEN        CHECK IF VALID                            REA00370
         BH    RNOGOOD                                                   REA00380
         L     5,ECSPTR                                                  REA00390
         L     4,O(4)          STRING LENGTH                             REA00400
         SLL   4,2             ... IN BYTES                              REA00410
         BCTR  4,O             MINUS ONE FOR HARDWARE                    REA00420
         LA    5,O(3,5)        ADJUST ECSADD WITH OFFSET                 REA00430
         EX    4,RMVC          MOVE INTO STRING                          REA00440
*------------------------------------------------------------------------* REA00450
*                                                                         REA00460
         OUT   STORAGE=O                                                 REA00470
```

```
*                                                                     REA00480
RMVC     MVC   0(0,2),0(5)                                            REA00490
*                                                                     REA00500
RNOGOOO  OUT   STORAGE=O,RETC=4    BAO RETURN                         REA00510
***                                                                   REA00520
*********************************************************************** REA00530
***                                                                   REA00540
***            ROUTINE WRITEC ENTRY POINT                             REA00550
***                                                                   REA00560
WRITEC   IN    ENTRY,STORAGE=O                                        REA00570
         LM    2,6,0(1)       PICK UP ADDRESS OF ARGS                 REA00580
*                                                                     REA00590
*                                  R2 = STRING                        REA00600
*                                  R3 = OFFSET                        REA00610
*                                  R4 = LENWOS                        REA00620
*                                  R5 = ECSPTR                        REA00630
*                                  R6 = ECSLEN                        REA00640
*                                                                     REA00650
         MVC   ECSPTR(4),0(5)     LOAO ECSPTR                        REA00660
         MVC   ECSLEN(4),0(6)     LOAO ECSLEN                        REA00670
         L     3,0(3)          GET OFFSET                            REA00680
         SLL   3,2             ...IN BYTES                           REA00690
         C     3,ECSLEN        CHECK FOR TOO BIG                     REA00700
         BH    WNOGOOD                                               REA00710
         L     5,ECSPTR        BASE FOR ECS                          REA00720
         L     4,0(4)          GET LENGTH                            REA00730
         SLL   4,2             ...IN BYTES                           REA00740
         BCTR  4,0             MINUS ONE FOR HARDWARE                REA00750
         LA    5,0(3,5)        AOJUST ECS AOOR FOR OFFSET            REA00760
         EX    4,WMVC          STORE STRING INTO ECS                 REA00770
*                                                                     REA00780
*--------------------------------------------------------------------* REA00790
*                                                                     REA00800
         OUT   STORAGE=O                                             REA00810
*                                                                     REA00820
WNOGOOO  OUT   STORAGE=O,RETC=4    BAO RETURN                        REA00830
*                                                                     REA00840
WMVC     MVC   0(0,5),0(2)                                           REA00850
*********************************************************************** REA00860
***                                                                   REA00870
***            OATA CONSTANT AREA                                     REA00880
***                                                                   REA00890
ECSPTR   OS    A              POINTER TO BEGINNING OF ECS STORAGE AREA REA00900
ECSLEN   OC    A(0)           LENGTH OF THE ECS STORAGE AREA          REA00910
         ENO                                                         REA00920
```

```
         TITLE 'START - START MODULE EXECUTION'                          STA00010
*********************************************************************** STA00020
*                                                                       STA00030
*          START -                                                      STA00040
*                                                                       STA00050
*          SUBROUTINE TO START A MODULE AT THE ENTRY POINT GIVEN        STA00060
*                                                                       STA00070
*              CALLING SEQUENCE                                         STA00080
*                                                                       STA00090
*              CALL START(ENTRY,RFCODE)                                 STA00100
*                                                                       STA00110
*              WHERE ENTRY IS THE ENTRY POINT ADDRESS OF THE MODULE     STA00120
*              WHERE RFCODE IS A FATAL RETURN CODE.  ABNORMAL           STA00130
*                  TERMINATION IS ASSUMED AND THE RFCODE IS SET         STA00140
*                  EQUAL TO TWO UPON ENTRY.  IF AN ABNORMAL END         STA00150
*                  OCCURS RFCODE = 2 IMPLIES THE ERROR OCCURED IN       STA00160
*                  THIS ROUTINE.  IF TERMINATION IS NORMAL RFCODE       STA00170
*                  IS RESET TO ZERO PRIOR TO RETURNING TO THE           STA00180
*                  CALLING ROUTINE.                                     STA00190
*                                                                       STA00200
*              ENTRY AND RFCODE ARE INTEGER ARGUMENTS                   STA00210
*                                                                       STA00220
*********************************************************************** STA00230
*                                                                       STA00240
START    IN    CSECT,STORAGE=0,SAVE=SAVEAREA                            STA00250
         L     2,0(1)       PICK UP PARM LIST                           STA00260
         L     3,4(1)       USE R3 FOR ADDR OF RFCODE                   STA00270
         MVC   0(4,3),=F'2' SET RFCODE = 2, IF ABEND IT WAS IN START    STA00280
***  PUT THE MODULE ENTRY ADDRESS IN R15 AND BRANCH TO IT               STA00290
         L    15,0(2)       GET THE LOAD MODULE ENTRY POINT             STA00300
         BALR 14,15                                                     STA00310
         L     3,4(1)       USE R3 FOR ADDR OF RFCODE                   STA00320
         MVC   0(4,3),=F'0' SET RFCODE = 0, NORMAL TERMINATION          STA00330
         OUT   STORAGE=0,SAVE=SAVEAREA                                  STA00340
*********************************************************************** STA00350
*                                                                       STA00360
*** DATA CONSTANT AREA                                                  STA00370
*                                                                       STA00380
SAVEAREA DS    18F           REGISTER SAVE AREA                         STA00390
         END                                                            STA00400
```

```
         TITLE 'SYSTEM - EXIT TO CMS SUBSET'                                SYS00010
*************************************************************************** SYS00020
*                                                                           SYS00030
*        SYSTEM -                                                           SYS00040
*                                                                           SYS00050
*              SUBROUTINE TO EXIT TO THE CMS SUBSET (OPERATING SYSTEM)      SYS00060
*                                                                           SYS00070
*        CALLING SEQUENCE,                                                  SYS00080
*                                                                           SYS00090
*              CALL SYSTEM                                                  SYS00100
*                                                                           SYS00110
*        EXITS TO THE CMS OPERATING SYSTEM SUBSET TO ALLOW TRANSIENT        SYS00120
*        CMS COMMANDS.  TO RETURN TO DEX TYPE RETURN.                       SYS00130
*                                                                           SYS00140
*************************************************************************** SYS00150
*                                                                           SYS00160
SYSTEM   IN    CSECT,STORAGE=0,SAVE=SAVEAREA                               SYS00170
         LA    1,PARMS       SET UP FOR THE SVC CALL                       SYS00180
         CNDP  2,4                                                         SYS00190
         SVC   202           USE THE SVC TO INVOKE THE COMMAND IN PARMS    SYS00200
         DC    A(DUT)                                                      SYS00210
DUT      DUT   STORAGE=0,SAVE=SAVEAREA                                     SYS00220
*************************************************************************** SYS00230
***      DATA STORAGE AREA                                                  SYS00240
***                                                                         SYS00250
         DS    0D            START STORAGE ON DOUBLEWORD BOUNDARY          SYS00260
PARMS    DC    CL8'SUBSET'   COMMAND FOR CMS SUBSET                        SYS00270
         DC    8X'FF'        ENDS PARAMETERS                               SYS00280
SAVEAREA DS    18F           REGISTER SAVE AREA                            SYS00290
         END                                                               SYS00300
```

```
         TITLE 'TIME - GETS TIME AND DATE FROM SYSTEM'                    TIM00010
*************************************************************************** TIM00020
*                                                                         TIM00030
*        TIME                                                             TIM00040
*                                                                         TIM00050
*        GETS THE TIME AND DATE FROM THE SYSTEM                           TIM00060
*                                                                         TIM00070
*        CALLING SEQUENCE                                                 TIM00080
*                                                                         TIM00090
*         CALL TIME(STRING)                                               TIM00100
*                                                                         TIM00110
*       RETURN TIME AND DATE IN STRING IN THE FORM 'HH.MM.SS.TH YY/DDD'   TIM00120
*                                                                         TIM00130
*        USES TIME SVC - FOR MORE INFO, SEE 'SUPERVISOR SERVICES AND      TIM00140
*        MACRO INSTRUCTIONS', IBM GC24-5103                               TIM00150
*                                                                         TIM00160
*************************************************************************** TIM00170
TIME     IN    CSECT,STORAGE=128                                          TIM00180
         L     11,0(1)                                                    TIM00190
         USING MSG,11                                                     TIM00200
         MVC   HH(16),CHARS                                               TIM00210
         TIME                                                             TIM00220
         ST    1,DATE                                                     TIM00230
         ST    0,TIMER                                                    TIM00240
         UNPK  UNDATE(5),DATE(4)                                          TIM00250
         UNPK  UNTIME(9),TIMER(5)                                         TIM00260
         MVC   HH(2),UNTIME                                               TIM00270
         MVC   MM(2),UNTIME+2                                             TIM00280
         MVC   SS(2),UNTIME+4                                             TIM00290
         MVC   TH(2),UNTIME+6                                             TIM00300
         MVC   YY(2),UNDATE                                               TIM00310
         MVC   DDD(3),UNDATE+2                                            TIM00320
         OUT   STORAGE=128                                                TIM00330
CHARS    DC    C'HH.MM.SS.TH YY/DDD'                                      TIM00340
DSA      DSECT                                                            TIM00350
SAVEAREA DS    72C                                                        TIM00360
DATE     DS    F                                                          TIM00370
TIMER    DS    F                                                          TIM00380
UNDATE   DS    D                                                          TIM00390
UNTIME   DS    D                                                          TIM00400
MSG      DSECT                                                            TIM00410
HH       DS    2C                                                         TIM00420
         DS    C                                                          TIM00430
MM       DS    2C                                                         TIM00440
         DS    C                                                          TIM00450
SS       DS    2C                                                         TIM00460
         DS    C                                                          TIM00470
```

```
TH      DS    2C                                          TIM00480
        DS    C                                           TIM00490
YY      DS    2C                                          TIM00500
        DS    C                                           TIM00510
DDD     DS    3C                                          TIM00520
        END   TIME                                        TIM00530
```

```
         TITLE 'UNLDAO - DELETE A PREVIDUSLY LDADED MOOULE'                UNL00010
*************************************************************************** UNL00020
*                                                                           UNL00030
*        UNLOAD -                                                           UNL00040
*                                                                           UNL00050
*        SUBROUTINE TD DELETE A MODULE PREVIDUSLY LOADED                    UNL00060
*                                                                           UNL00070
*        CALLING SEQUENCE -                                                 UNL00080
*                                                                           UNL00090
*              TRUTHVALUE = UNLOAD(FILENAME)                                UNL00100
*                                                                           UNL00110
*        WHERE FILENAME IS THE FILENAME DF MODULE TD BE DELETED.            UNL00120
*              IT IS UP TD EIGHT CHARACTERS IN LENGTH                       UNL00130
*                                                                           UNL00140
*              TRUTHVALUE IS TRUE IF DELETE WAS SUCCESSFUL AND FALSE IF     UNL00150
*              OELETE WAS UNSUCCESSFUL                                      UNL00160
*                                                                           UNL00170
*        USES THE OS/VS1 SUPERVISOR 'DELETE' MACRD.  FDR FURTHER INFD,      UNL00180
*             SEE THE 'DS/VS1 SUPERVISOR SERVICES AND MACRD                 UNL00190
*             INSTRUCTIDN', GC24-5103                                       UNL00200
*                                                                           UNL00210
*************************************************************************** UNL00220
UNLOAD   IN    CSECT,STDRAGE=O,SAVE=SAVEAREA                                UNL00230
         L     1,O(1)          PICK UP PARM LIST                            UNL00240
         MVC   NAME(8),O(1)    GET LOAD MODULE NAME                         UNL00250
         DELETE EPLDC=NAME     UNLDAO MODULE (EPLDC = ENTRY PDINT LOC)      UNL00260
         CL    15,=F'4'        CDMPARE RETURN CDDE TD 4 (CDULDN'T FINO)     UNL00270
         BE    BOMB                                                         UNL00280
         DUT   STORAGE=O,SAVE=SAVEAREA,RETC=1,RETREG=O  RETURN 'TRUE'       UNL00290
BDMB     DUT   STORAGE=O,SAVE=SAVEAREA,RETC=O,RETREG=O  RETURN 'FALSE'      UNL00300
*************************************************************************** UNL00310
***  DATA CONSTANT AREA                                                     UNL00320
SAVEAREA OS     18F                                                         UNL00330
NAME     DC    CL8' '                                                       UNL00340
         OC    8X'FF'                                                       UNL00350
         END                                                                UNL00360
```

```
*************************************************************************** CMS00010
*                                                                         CMS00020
*        CMSCB MACRO - SIMULATED DS CONTRDL BLOCK                         CMS00030
*                                                                         CMS00040
*              USED TO MAP A FILE CONTRDL BLOCK                           CMS00050
*                                                                         CMS00060
*************************************************************************** CMS00070
         MACRO                                                            CMS00080
         CMSCB                                                            CMS00090
         PUSH  PRINT                                                      CMS00100
         AIF   ('&SYSPARM' NE 'SUP').ACCO1                                CMS00110
         PRINT DFF,NOGEN                                                  CMS00120
 .ACCO1  ANOP                                                             CMS00130
*                                                                         CMS00140
*   SIMULATED OS CONTROL BLOCKS                                           CMS00150
*                                                                         CMS00160
FCBSECT  OSECT                                                            CMS00170
FCBINIT  OS    OX -            INTERESTING TIOBITS                        CMS00180
FCBDPCB  EQU   X'08' -         OPEN ACQUIRED THIS CMS BLOCK               CMS00190
FCBPERM  EQU   X'04' -         PERMANENT CONTRDL BLOCK                    CMS00200
FCBBATCH EQU   X'02' -         SPECIAL BATCH OATA SET                     CMS00210
FCBCATML EQU   X'01' -         CONCATENATED MACLIB OATA SET               CMS00220
FCBOS    EQU   X'10'           FCB FOR OS FORMATTED DISK                  CMS00230
FCBDOSL  EQU   X'20'           CONCATENATED DDSLIB OATA SET               CMS00240
FCBNEXT  DS    A -             AL3(NEXT CMSCB)                            CMS00250
FCBPROC  DS    A -             A(SPECIAL PROCESSING ROUTINE)              CMS00260
FCBDD    OS    CL8 -            DATA DEFINITION NAME                      CMS00270
FCBDP    OS    CL8 -           CMS OPERATION                              CMS00280
IHAJFCB  OS    OD -            *** JOB FILE CONTROL BLOCK ***             CMS00290
JFCBOSNM DS    OX -            44 BYTES, DATA SET NAME                    CMS00300
FCBTAPID OS    OX -            TAPE IOENTIFICATION                        CMS00310
FCBDSNAM OS    CL8 -           DATA SET NAME                              CMS00320
FCBDSTYP DS    CL8 -           DATA SET TYPE                              CMS00330
FCBPRPU  EQU   FCBOSTYP+4 -    PRINTER/PUNCH COMMAND LIST                 CMS00340
FCBTBSP  DS    OX                  2 BYTES, TAPE BACKSPCE COUNT @VA04853 CMS00350
FCBDSMD  DS    CL2 -           DATA SET MODE                              CMS00360
FCBITEM  DS    H -             ITEM IOENTIFICATION NUMBER                 CMS00370
FCBBUFF  DS    F -             A(INPUT-OUTPUT BUFFER)                     CMS00380
FCBBYTE  OS    F -             DATA COUNT                                 CMS00390
FCBFORM  OS    CL2 -           FILE FORMAT: FIXED/VARIABLE RECORDS        CMS00400
FCBCOUT  OS    H -             RECORDS PER CMS PHYSICAL BLOCK             CMS00410
FCBREAD  OS    F -             N'BYTES ACTUALLY READ                      CMS00420
FCBOEV   DS    X -             DEVICE TYPE CODE                           CMS00430
FCBDUM   EQU   0 -             DUMMY DEVICE                               CMS00440
FCBPTR   EQU   4 -             PRINTER                                    CMS00450
FCBROR   EQU   8 -             READER                                     CMS00460
FCBCON   EQU   12 -            CONSOLE TERMINAL                           CMS00470
```

```
FCBTAP    EQU   16 -          TAPE                                  CMS00480
FCBDSK    EQU   20 -          DISK                                  CMS00490
FCBPCH    EQU   24 -          PUNCH                                 CMS00500
FCBCRT    EQU   28 -          CRT                                   CMS00510
FCBMODE   DS    X -           MODE: 1,2,3,4,5                       CMS00520
FCBXTENT  DS    H -           NUMBER OF ITEMS IN EXTENT             CMS00530
FCBRECL   DS    H -           DCB LRECL AT DPEN TIME                CMS00540
IOBIOFLG  DS    X -           I/D FLAGS                             CMS00550
FCBDCBCT  DS    X -           ND. DF OCB'S USING THIS FCB           CMS00560
FCBMEMBR  DS    2F            DS PDS MEMBER NAME                    CMS00570
FCBDSFST  DS    F             POINTER TD DS FST                     CMS00580
FCBOSOSN  DS    F             PDINTER TD DS DSNAME BLOCK            CMS00590
FCBR13    DS    F -           SAVEAREA VECTOR R13                   CMS00600
FCBKEYS   DS    A -           A(DDS IN'CORE KEY TABLE)              CMS00610
FCBPDS    OS    A -           A(PDS IN-CDRE OIRECTDRY)              CMS00620
JFCBMASK  DS    8X -          VARIDUS MASK BITS                     CMS00630
JFCBCROT  DS    3C -          DATA SET CREATIDN DATE (YDD)          CMS00640
JFCBXPOT  DS    3C -          DATA SET EXPIRATIDN DATE (YDD)        CMS00650
JFCBIND1  DS    X -           INOICATOR DNE                         CMS00660
JFCBIND2  DS    X -           INDICATDR TWD                         CMS00670
JFCBUFND  DS    X -           NUMBER DF BUFFERS                     CMS00680
JFCBFTEK  DS    OX -          BUFFERING TECHNIQUE                   CMS00690
JFCBFALN  DS    X -           BUFFER ALIGNMENT                      CMS00700
JFCBUFL   DS    H -           BUFFER LENGTH                         CMS00710
JFCEROPT  DS    X -           ERRDR OPTION                          CMS00720
JFCKEYLE  DS    X -           KEYLENGTH                             CMS00730
          DS    X -           ---NDT USED---                        CMS00740
JFCLIMCT  DS    3X -          BDAM SEARCH LIMIT                     CMS00750
FCBOSDRG  OS    OX -          OATA SET DRGANIZATIDN                 CMS00760
JFCOSDRG  OS    2X -                                                CMS00770
FCBRECFM  DS    OX -          RECDRD FDRMAT                         CMS00780
JFCRECFM  OS    X -                                                 CMS00790
JFCOPTCO  DS    X -           DPTION CDOES                          CMS00800
FCBBLKSZ  DS    OH -          BLDCK SIZE                            CMS00810
JFCBLKSI  DS    H -                                                 CMS00820
FCBLRECL  DS    OH -          LDGICAL RECDRO LENGTH                 CMS00830
JFCLRECL  DS    H -                                                 CMS00840
FCBIDSW   DS    X -           I/D DPERATIDN INOICATOR               CMS00850
FCBCLOSE  EQU   X'80' -       DURING "CLOSE"                        CMS00860
FCBCLEAV  EQU   X'40' -       DISP = LEAVE OURING CLDSE             CMS00870
FCBPRDCC  EQU   X'20' -       GOTD FCBPRDC OURING CLOSE             CMS00880
FCBPRDCD  EQU   X'10' -       GDTD FCBPRDC OURING DPEN              CMS00890
FCBCASE   EQU   X'08' -       ON=LDWER CASE CDNSDLE I/D             CMS00900
FCBPVMB   EQU   X'04' -       PUT-MDVE-VAR-BLK                      CMS00910
FCBIOWR   EQU   X'02' -       WRITE/PUT                             CMS00920
FCBIDRD   EQU   X'01' -       REAO/GET                              CMS00930
FCBIDSW2  DS    1X -          I/D DPERATIDN INDICATDRS              CMS00940
```

```
FCBMVPDS EQU   X'01' -        SW FDR MDVEFILE WITH PDS DPTIDN            CMS00950
FCBMMV   EQU   X'02'          MDVE PDS SWITCH FDR FIND         @VA03059  CMS00960
FCBMVFIL EQU   X'08'          MDVE FILE IS ACTIVE                        CMS00970
DEBLNGTH DS    OX -           L'DEB IN DBLW WORDS                        CMS00980
FCBTCLOS EQU   X'40'              A CLDSE TYPE T WAS DDNE       @VA08024  CMS00990
         DS    F -            ---NOT USED---                             CMS01000
IHADEB   DS    OD -           *** DATA EXTENT BLDCK ***                  CMS01010
DEBTCBAD DS    A -            A(MOVE-MDDE USER BUFFER)                   CMS01020
SEBSAV   DS    F              DYNAMIC SAVE FDR RET ADDR FDR    @VM02691  CMS01030
*                             SEB (DS I/D SIM)                           CMS01040
DEBDFLGS DS    4X -           DATA SET STAUS FLAGS                       CMS01050
DEBOPATB DS    4X -           DPEN/CLDSE DPTION BYTE                     CMS01060
IDBFLG   DS    OX -           (START DF IDBPREFIX FOR NORMAL SCH)        CMS01070
IOBBFLG  EQU   O -            DISPLACEMENT OF IOB FLAG IN IOB            CMS01080
IOBDUT   EQU   X'40' -        "WRITE,PUT" IN PRDGRESS                    CMS01090
IOBIN    EQU   X'20' -        "READ,GET" IN PROGRESS                     CMS01100
IDBUPD   EQU   X'10' -        "QSAM PUTX" IN PRDCESS                     CMS01110
IDBNXTAD DS    A -            A(NEXT BUFFER TD BE USED)                  CMS01120
IDBECB   DS    F -            ECB FOR QSAM NORMAL SCHEDULING             CMS01130
IHAIOB   DS    OF -           *** INPUT/OUTPUT BLOCK ***                 CMS01140
DEBDEBID DS    OX -           DEB IDENTIFICATIDN                         CMS01150
DEBDCBAD DS    A -            A(DATA CONTRDL BLDCK)                      CMS01160
IOBECBCC DS    OX -           ECB COMPLETIDN CDDE                        CMS01170
IOBBECBC EQU   12 -           DISPLACEMENT OF ECB CODE IN IDB            CMS01180
IOBBECBP EQU   12 -           DISPLACEMENT OF ECB PTR IN IDB             CMS01190
IDBECBPT DS    A  -           A(EVENT CDNTRDL BLDCK)                     CMS01200
IDBFLAG3 DS    OX -           I/O ERROR FLAG                             CMS01210
IDBBCSW  EQU   16 -           DISPLACEMENT DF CSW IN IDB                 CMS01220
IDBCSW   DS    8X -           LAST CCW STDRED(I.E., RESIDUAL CDUNT)      CMS01230
IDBSTART DS    A -            X'ID-NEXT BUFFER',AL3(INITIAL BUFFER)      CMS01240
IDBDCBPT DS    A -            A(DATA CONTRDL BLDCK)                      CMS01250
IDBEND   DS    OX -           END-DF-INPUT/OUTPUT BLDCK                  CMS01260
FCBEND   DS    OD -           END-DF FCB,JFCB,DEB,IDB BLDCKS             CMS01270
FCBENSIZ EQU   (*-FCBSECT)/8 - SIZE OF FCB ENTRY, DOUBLEWORDS            CMS01280
         SPACE 3                                                         CMS01290
         DRG   FCBDSTYP+4                                                CMS01300
FCBIOOUT DS    CL8 -          SPECIAL I/D CDMMAND LIST                   CMS01310
FCBIDBUF DS    A -            A(DATA BUFFER)                             CMS01320
FCBCONCR DS    C -            CDNSDLE COLOR CDDE                         CMS01330
FCBCDNMS DS    X -            CONSOLE MISCELLANEDUS INFO                 CMS01340
FCBIDCNT DS    H -            L'DATA BUFFER                              CMS01350
         SPACE 3                                                         CMS01360
*                                                                        CMS01370
*   DATA EVENT CONTRDL BLDCK                                             CMS01380
*                                                                        CMS01390
IHADECB  DSECT                                                           CMS01400
DECSDECB DS    F -            EVENT CONTRDL BLDCK                        CMS01410
```

```
DECTYPE  DS    H -             TYPE OF I/D REQUEST                    CMSO1420
DECBRD   EQU   X'80' -         READ SF                                CMSO1430
DECBWR   EQU   X'20' -         WRITE SF                               CMSO1440
DECLNGTH DS    H -             LENGTH OF KEY & DATA                   CMSO1450
DECDCBAD DS    A -             V(DATA CONTROL BLOCK)                  CMSO1460
DECAREA  DS    A -             V(KEY & DATA, BUFFER)                  CMSO1470
DECIDBPT DS    A -             V(IOB)                                 CMSO1480
*              BDAM EXTENSION                                         CMSO1490
DECKYADR DS    A -             V(KEY)                                 CMSO1500
DECRECPT DS    A -             V(BLOCK REFERENCE FIELD)               CMSO1510
         SPACE 3                                                      CMSO1520
*                                                                     CMSO1530
*   SDME FREQUENTLY USED EQUATES                                      CMSO1540
*                                                                     CMSO1550
DDNAM    EQU   FCBDSTYP -      FILETYPE = DATA SET NAME               CMSO1560
BLK      EQU   X'10' -         RECFM=BLOCKED RECORDS                  CMSO1570
BS       EQU   X'20' -         MACRF=BSAM                             CMSO1580
DA       EQU   X'20' -         DSORG=DIRECT ACCESS                    CMSO1590
FXD      EQU   X'80' -         RECFM=FIXED LENGTH RECORDS             CMSO1600
IS       EQU   X'80' -         DSORG=INDEXED SEQUENTIAL               CMSO1610
LDC      EQU   X'08' -         MACRF=LOCATE MODE                      CMSO1620
MOV      EQU   X'10' -         MACRF=MOVE MODE                        CMSO1630
PS       EQU   X'40' -         DSORG=PHYSICAL SEQUENTIAL              CMSO1640
PDU      EQU   X'03' -         DSORG=PARTITIONED UNMOVEABLE           CMSO1650
PD       EQU   X'02' -         DSORG=PARTITIONED ORGANIZATION         CMSO1660
PREVIOUS EQU   X'80' -         OFLGS=PREVIOUS I/O OPERATION           CMSO1670
QS       EQU   X'40' -         MACRF=QSAM                             CMSO1680
UND      EQU   X'C0' -         RECFM=UNDEFIN FORMAT RECORDS           CMSO1690
VAR      EQU   X'40' -         RECFM=VARIABLE LENGTH RECORDS          CMSO1700
         EJECT                                                        CMSO1710
         PDP   PRINT                                                  CMSO1720
         MEND                                                         CMSO1730
```

```
*********************************************************************** IN 00010
*                                                                     IN 00020
*         IN MACRO                                                    IN 00030
*                                                                     IN 00040
*               USED TO SET UP THE ENTRY CONVENTION FOR AN            IN 00050
*               ASSEMBLY LANGUAGE PROGRAM                             IN 00060
*                                                                     IN 00070
*********************************************************************** IN 00080
          MACRO                                                       IN 00090
&LABEL    IN    &CSECT,&STORAGE=1024,&DSECT=DSA,&SAVE=                IN 00100
          AIF   ('&CSECT' EQ 'ENTRY').ENTRY                           IN 00110
          AIF   ('&CSECT' EQ '').NOCSECT                              IN 00120
&LABEL    CSECT                                                       IN 00130
          SAVE  (14,12),,&LABEL                                       IN 00140
          AGO   .AWAY                                                 IN 00150
.ENTRY    ANOP                                                        IN 00160
          ENTRY &LABEL                                                IN 00170
.NOCSECT  ANOP                                                        IN 00180
&LABEL    SAVE  (14,12),,&LABEL                                       IN 00190
.AWAY     ANOP                                                        IN 00200
          LR    12,15                                                 IN 00210
          USING &LABEL,12                                             IN 00220
          AIF   ('&STORAGE' EQ '0').NODSA                             IN 00230
          GETMAIN R,LV=&STORAGE                                       IN 00240
          ST    1,8(13)                                               IN 00250
          ST    13,4(1)                                               IN 00260
          LR    1,13                                                  IN 00270
          L     13,8(13)                                              IN 00280
          USING &DSECT,13                                             IN 00290
          L     0,20(1)                                               IN 00300
          L     14,12(1)                                              IN 00310
          L     15,16(1)                                              IN 00320
          L     1,24(1)                                               IN 00330
          MEXIT                                                       IN 00340
.NODSA    ANOP                                                        IN 00350
          AIF  ('&SAVE' EQ '').NOSAVE                                 IN 00360
          LA    15,&SAVE        STATIC SAVE AREA                      IN 00370
          ST    13,4(15)        CHAIN OLD AREA IN                     IN 00380
          ST    15,8(13)                                              IN 00390
          LR    13,15                                                 IN 00400
.NOSAVE   ANOP                                                        IN 00410
          MEND                                                        IN 00420
```

```
*************************************************************************** OUT00010
*                                                                         OUT00020
*        OUT MACRO                                                        OUT00030
*                                                                         OUT00040
*              USED TO SET UP THE RETURN CONVENTION FOR AN                OUT00050
*              ASSEMBLY LANGUAGE PROGRAM                                  OUT00060
*                                                                         OUT00070
*************************************************************************** OUT00080
         MACRO                                                           OUT00090
&LABEL   OUT   &STORAGE=1024,&RETC=0,&RETREG=15,&SAVE=                   OUT00100
         GBLB  &REGINIT                                                  OUT00110
         LCLC  &LBL,&SAVEREG                                             OUT00120
         LCLA  &OFFSET                                                   OUT00130
&LBL     SETC  '&LABEL'                                                  OUT00140
         AIF   ('&STORAGE' EQ '0').NODSA                                 OUT00150
&LBL     LR    1,13                                                      OUT00160
         L     13,4(13)                                                  OUT00170
         FREEMAIN R,LV=&STORAGE,A=(1)                                    OUT00180
&LBL     SETC  ''                                                        OUT00190
         AGO   .RETURN                                                   OUT00200
.NODSA   ANOP                                                            OUT00210
         AIF   ('&SAVE' EQ '').RETURN                                    OUT00220
&LBL     L     13,4(13)       RETURN TO ORIGINAL SAVE AREA               OUT00230
&LBL     SETC  ''                                                        OUT00240
.RETURN  ANOP                                                            OUT00250
&LBL     LA    &RETREG,&RETC                                             OUT00260
         AIF   ('&REGINIT' EQ 'SET').REGSIN                              OUT00270
&OFFSET  SETA  (&RETREG+2)*4+12                                          OUT00280
         AIF   (&RETREG LT 13).LOW                                       OUT00290
&OFFSET  SETA  (&RETREG-13)*4+8                                          OUT00300
.LOW     ANOP                                                            OUT00310
&SAVEREG SETC  '&OFFSET'                                                 OUT00320
         AGO   .ON                                                       OUT00330
.REGSIN  ANOP                                                            OUT00340
&SAVEREG SETC  'SR'.'&RETREG'                                            OUT00350
.ON      ANOP                                                            OUT00360
&SAVEREG SETC  '&SAVEREG'.'(13)'                                         OUT00370
         ST    &RETREG,&SAVEREG                                          OUT00380
         LM    14,12,12(13)                                              OUT00390
         BR    14                                                        OUT00400
         LTORG                                                           OUT00410
         MEND                                                            OUT00420
```

APPENDIX D

DEX DATABASE EDITING ROUTINES

```
C                                                                               DBC00010
C*******************************************************************************DBC00020
      LOGICAL FUNCTION DBCMPR(RCDDE)                                            DBC00030
C  +-------------------------------------------------------------------+        DBC00040
C                                                                               DBC00050
C         LDGICAL FUNCTION DBCMPR CAN BE CALLED TD CDMPRESS THE                 DBC00060
C         DATABASE AFTER A NUMBER DF DELETES.                                   DBC00070
C                                                                               DBC00080
C         THE ARGUMENT RCODE IS RETURNED TO INDICATE THE REASON FDR             DBC00090
C         THE LDGICAL VALUE DF DBCMPR UPDN RETURN.                              DBC00100
C                                                                               DBC00110
C              RCDDE = O --> THE COMPRESS WAS SUCCESSFUL.                       DBC00120
C              RCODE = 1 --> THE COMPRESS FAILED BECAUSE THERE WERE             DBC00130
C                            ND DELETED NDDES.                                  DBC00140
C                                                                               DBC00150
C   +-------------------------------------------------------------------+       DBC00160
C...   DATABASE COMMON                                                          DBC00170
       CDMMDN /DBCDM/                                                           DBC00180
      1                TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,       DBC00190
      1                CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,       DBC00200
      2                INTTYP,RELTYP,ARRTYP,                                   DBC00210
      3                DBCLSD,DBACTV,FILEDB                                    DBC00220
       INTEGER                                                                  DBC00230
      1                TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,       DBC00240
      1                CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,       DBC00250
      2                INTTYP,RELTYP,ARRTYP                                    DBC00260
       LOGICAL                                                                  DBC00270
      3                DBCLSD,DBACTV,FILEDB                                    DBC00280
       DIMENSION       TITLE(16),BUCKET(32),IDENT(200,2),NODTYP(200),          DBC00290
      1                DATUM(200),LINK(200),LINKF(200),CMTPTR(200)             DBC00300
       INTEGER         DELNDD                                                  DBC00310
C                                                                               DBC00320
C              +++ INITIALIZE CDNSTANTS                                         DBC00330
C                                                                               DBC00340
       CDUNT=O                                                                  DBC00350
C                                                                               DBC00360
C...     SEQUENCE THRDUGH THE DATABASE LDOKING FDR DELETED ENTRIES              DBC00370
C...     DELETED ENTRIES HAVE NODTYP = -9, WHEN A DELETED NDDE IS               DBC00380
C...     FDUND COMPRESS UNTILL THE NEXT DELETED NDDE, THEN INCREMENT            DBC00390
C...     CDUNT AND CDNTINUE CDMPRESSING BY CDUNT.  ADJUST PDINTERS              DBC00400
C...     ACCDRDINGLY.                                                           DBC00410
C                                                                               DBC00420
C              +++ BEGIN SEARCH LDOP                                            DBC00430
       DD 1000 I=1,MAXNDD                                                       DBC00440
C              +++ IF NDDTYP(I)=O THE REST DF THE DB NDDES ARE EMPTY            DBC00450
         IF (NODTYP(I) .EQ. O) GD TD 6000                                       DBC00460
         IF (NDDTYP(I) .NE. -9) GD TO 1000                                      DBC00470
```

```
C             +++ A DELETEO ENTRY WAS FOUND, MOVE THE REST UP           OBC00480
           OELNOD=I                                                     OBC00490
           COUNT=COUNT+1                                                DBC00500
           GO TO 2000                                                   OBC00510
 1000 CONTINUE                                                          OBC00520
C             +++ ENO OF SEARCH LOOP                                    OBC00530
C             +++ IF YOU REACH THIS POINT THERE WERE NO DELETED NOOES   DBC00540
      GO TO 77777                                                       OBC00550
 2000 CONTINUE                                                          OBC00560
C             +++ BEGIN COMPRESS LOOP                                   DBC00570
         OO 3000 J=OELNOO,MAXNOO                                        OBC00580
           IF (NOOTYP(J) .EQ. O) GO TO 6000                             OBC00590
           IF (J .EQ. MAXNOO) GO TO 5000                                OBC00600
           IDENT(J,1)=IDENT(J+COUNT,1)                                  OBC00610
           IOENT(J,2)=IOENT(J+COUNT,2)                                  OBC00620
           NOOTYP(J)=NOOTYP(J+COUNT)                                    DBC00630
C             +++                                                       OBC00640
C             +++ LINK ANO LINKF ARE MOVEO UP BY COUNT.  IF LINK(J)=O   OBC00650
C             +++ THEN THIS IS THE LAST NOOE OR ONLY NOOE IN A CHAIN.   DBC00660
C             +++ IF LINK(J) IS NOT EQUAL TO ZERO THEN THE NODE TO WHICHDBC00670
C             +++ LINK(J) POINTS MUST HAVE ITS LINKF(LINK(J)) REOUCED BYDBC00680
C             +++ COUNT.                                                DBC00690
C             +++                                                       OBC00700
C             +++ IF LINKF(J) IS NEGATIVE THEN IT POINTS TO THE BUCKET  OBC00710
C             +++ WHICH IS POINTING TO THIS NODE.  SO THE BUCKET ENTRY  DBC00720
C             +++ MUST BE ADJUSTED TO POINT TO THE NEW NOOE POSITION J. DBC00730
C             +++                                                       DBC00740
           LINK(J)=LINK(J+COUNT)                                        DBC00750
           LINKF(J)=LINKF(J+COUNT)                                      DBC00760
           IF (LINK(J) .NE. O) LINKF(LINK(J))=LINKF(LINK(J))-COUNT      DBC00770
           IF (LINKF(J) .LT. O) BUCKET(IABS(LINKF(J)))=J                OBC00780
           CMTPTR(J)=CMTPTR(J+COUNT)                                    OBC00790
           DATUM(J)=OATUM(J+COUNT)                                      OBC00800
C                                                                       DBC00810
C             +++                                                       OBC00820
C             +++ IF THE CMTPTR IS NOT ZERO THERE IS A COMMENT, SO      OBC00830
C             +++ UPOATE THE POINTER BACK TO THE NOOE FROM THE STORAGE  DBC00840
C             +++ AREA.  IF THE NODTYP IS EQUAL TO 3 THEN THERE IS      OBC00850
C             +++ AN ARRAY STORED, SO UPOATE THE POINTER BACK TO THE    OBC00860
C             +++ NOOE FROM THE ARRAY STORAGE AREA.                     OBC00870
C             +++                                                       DBC00880
C                                                                       DBC00890
           IF (CMTPTR(J) .EQ. O) GO TO 2400                             DBC00900
              IPOINT = CMTPTR(J)                                        DBC00910
              CALL WRITEC(J,IPOINT+2,1,ECSPTR(1),ECSLEN(1))             OBC00920
 2400      IF (NOOTYP(J) .NE. 3) GO TO 2500                             DBC00930
              IPOINT = OATUM(J)                                         OBC00940
```

```
              CALL WRITEC(J,IPDINT+2,1,ECSPTR(1),ECSLEN(1))                DBC00950
C                                                                          DBC00960
C             +++                                                          DBC00970
C             +++ IF NODTYP(J)=-9 AFTER COMPRESS THEN THE NEXT DELETED     DBC00980
C             +++ ENTRY HAS BEEN FOUND.  UPDATE THE NODE NUMBER OF THE     DBC00990
C             +++ DELNOD AND EXIT THE LOOP.  COUNT WILL BE INCREMENTED      DBC01000
C             +++ AND THE COMPRESS LOOP ENTERED WITH THE NEW DELNOD,       DBC01010
C             +++ AND COUNT.                                               DBC01020
C             +++                                                          DBC01030
C                                                                          DBC01040
 2500       IF (NODTYP(J) .EQ. -9) DELNOD=J                              DBC01050
            IF (NDDTYP(J) .EQ. -9) GO TO 4000                            DBC01060
 3000     CONTINUE                                                        DBC01070
C             +++ END OF COMPRESS LOOP                                     DBC01080
 4000 CONTINUE                                                           DBC01090
      COUNT=COUNT+1                                                      DBC01100
      GO TO 2000                                                         DBC01110
 5000 CONTINUE                                                           DBC01120
C             +++ MOVE ZEROES INTO THE LAST DATABASE NODE                  DBC01130
         IDENT(MAXNOD,1)=BLANK                                           DBC01140
         IDENT(MAXNOD,2)=BLANK                                           DBC01150
         NODTYP(MAXNOD)=0                                                DBC01160
         DATUM(MAXNOD)=0                                                 DBC01170
         LINK(MAXNOD)=0                                                  DBC01180
         LINKF(MAXNOD)=0                                                 DBC01190
         CMTPTR(MAXNOD)=0                                                DBC01200
 6000 CONTINUE                                                           DBC01210
C             +++ COMPRESS WAS SUCCESSFUL RCODE = 0                        DBC01220
      DBCMPR=.TRUE.                                                      DBC01230
      DELCNT=0                                                           DBC01240
      NAVAIL=NAVAIL-COUNT                                                DBC01250
      RCODE=0                                                            DBC01260
      GO TO 99999                                                        DBC01270
77777 CONTINUE                                                           DBC01280
C             +++ THERE WERE NO DELETED NODES RETURN DBCMPR=.FALSE.        DBC01290
C             +++ AND RCODE = 1                                            DBC01300
      DBCMPR=.FALSE.                                                     DBC01310
      RCODE=1                                                            DBC01320
      GO TO 99999                                                        DBC01330
99999 CONTINUE                                                           DBC01340
      RETURN                                                             DBC01350
      END                                                                DBC01360
```

```
C                                                                       DBE00010
C ***********************************************************************DBE00020
      SUBRDUTINE DBEDIT                                                 DBE00030
C  +--------------------------------------------------------------------+DBE00040
C                                                                       DBE00050
C        DBEDIT -                                                       DBE00060
C                                                                       DBE00070
C        SUBROUTINE DBEDIT IS CALLED FRDM DXMAJC WHEN THE MENU          DBE00080
C        ITEM EDIT-DB IS ENTERED.                                       DBE00090
C                                                                       DBE00100
C        DISPLAYS THE MENU DBEDCMDS AND ALLDWS THE USER TD EDIT         DBE00110
C        THE DATABASE.  THE MENU CHDICES ARE;                           DBE00120
C                                                                       DBE00130
C             CREATE   - CREATE A NDDE IN THE DATABASE.                 DBE00140
C             STDRE    - STDRE A VALUE IN A CREATED NODE.               DBE00150
C             DELETE   - DELETE A NDDE FROM THE DATABASE BY MAKING      DBE00160
C                        IT UNAVAILABLE.                                DBE00170
C             CDMMENT  - STDRE A COMMENT IN THE DATABASE FDR THE DATUM. DBE00180
C             EXPLAIN  - EXPLAIN A DATUM BY PRINTING ITS CDMMENT.       DBE00190
C             PRINT    - PRINT THE VALUE DF A GIVEN DATUM.              DBE00200
C             DUMP     - DISPLAY THE CDNTENTS DF THE DATABASE AT THE    DBE00210
C                        TERMINAL.                                      DBE00220
C             SET-TITL - PUT A TITLE IN THE DATABASE.                   DBE00230
C             GET-TITL - DISPLAY THE TITLE DF THE DATABASE.             DBE00240
C             DDNE     - FINISHED WITH THIS MENU, RETURN                DBE00250
C                                                                       DBE00260
C  +--------------------------------------------------------------------+DBE00270
C                                                                       DBE00280
C...                 *THE COMMON CDNTROL PARAMETERS,CHARACTERS, AND    DBE00290
C...                 *GRAPHIC DISPLAY CDNTRDL PARAMETERS               DBE00300
      CDMMON /DEXCDM/                                                   DBE00310
     1               DXMDDE,LOADED,MDPEND,DXREQS,DLREQS,KEYBRD,        DBE00320
     2               TERSE ,ECHDMD,DXSC  ,CPWDXS,MDSC  ,CURMOD,        DBE00330
     3               NULMSG,DXNCPW,NAMLEN,                             DBE00340
     4               CLRMOD,MUNPST,ERASE,UPDATE,UPAINT,GCNTRL          DBE00350
      INTEGER                                                           DBE00360
     1               DXSC  ,CPWDXS,MDSC  ,CURMOD,NULMSG,DXNCPW,        DBE00370
     2               NAMLEN,CLRMDD,MUNPST,ERASE,UPDATE,UPAINT          DBE00380
      LOGICAL                                                           DBE00390
     1               DXMODE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,        DBE00400
     2               TERSE ,ECHOMD,GCNTRL                              DBE00410
      DIMENSIDN      DXSC(1),CURMDD(2)  ,NULMSG(1)                     DBE00420
C...                 *I/D DEVICES AND FILES                            DBE00430
      CDMMON /DEXFIL/                                                   DBE00440
     1               MSGSDV,USEDEV,INFDDV,DBSDEV,NEWSDV,HELPDV,        DBE00450
     2               DUTDEV,INPDEV,NFWDS ,                             DBE00460
     3               MSGSFL,USEFIL,INFDFL,DBSFIL,NEWSFL,HELPFL,        DBE00470
```

```
     4                DUTFIL,INPFIL,LDADFL,NDT1FL,NDT2FL                    DBE00480
      INTEGER                                                               DBE00490
     1                MSGSDV,USEDEV,INFDDV,DBSDEV,NEWSDV,HELPDV,            DBE00500
     2                DUTDEV,INPDEV,                                        DBE00510
     3                MSGSFL,USEFIL,INFDFL,DBSFIL,NEWSFL,HELPFL,            DBE00520
     4                DUTFIL,INPFIL,LDADFL,NDT1FL,NDT2FL,NFWDS              DBE00530
C...  FILE NAME DIMENSIDNING ADJUSTED FDR CDC AND CU DNLY.....              DBE00540
      DIMENSIDN        MSGSFL(11)  ,USEFIL(11)  ,INFDFL(11)  ,             DBE00550
     1                 DBSFIL(11)  ,NEWSFL(11)  ,HELPFL(11)  ,             DBE00560
     3                 DUTFIL(11)  ,INPFIL(11)  ,                          DBE00570
     2                 LDADFL(11)  ,NDT1FL(11)  ,NDT2FL(11)                DBE00580
      CDMMDN /DBCDM/                                                        DBE00590
     1                TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,     DBE00600
     1                CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,     DBE00610
     2                INTTYP,RELTYP,ARRTYP,                                 DBE00620
     3                DBCLSD,DBACTV,FILEDB                                  DBE00630
      INTEGER                                                               DBE00640
     1                TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,     DBE00650
     1                CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,     DBE00660
     2                INTTYP,RELTYP,ARRTYP                                  DBE00670
      LDGICAL                                                               DBE00680
     3                DBCLSD,DBACTV,FILEDB                                  DBE00690
      DIMENSIDN        TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),       DBE00700
     1                 DATUM(200),LINK(200),LINKF(200),CMTPTR(200)          DBE00710
      INTEGER  TYPE,DXMENU,CDMMAN,ARSIZE,DNE,ISPAC1,ISPAC2                  DBE00720
      INTEGER  WDSPA1,WDSPA2,WDSPA3                                         DBE00730
      LDGICAL LDG,DXNBRI,DXGNAM                                             DBE00740
      DIMENSIDN NAME(2),MITEM1(20),MITEM2(6)                                DBE00750
      DIMENSIDN MENNM1(2),MENNM2(2)                                         DBE00760
      DIMENSIDN ISPAC1(16),ISPAC2(16),RSPAC3(200)                           DBE00770
      DATA  WDSPA1 /16/, WDSPA2 /16/, WDSPA3 /200/                          DBE00780
      DATA NAMSUB /4HEDIT/                                                  DBE00790
      DATA DNE /1/                                                          DBE00800
      DATA MENNM1 /4HDBED,4HCMDS/                                           DBE00810
      DATA NITEM1 /10/                                                      DBE00820
      DATA MITEM1 /4HCREA,4HTE  ,4HSTDR,4HE   ,4HDELE,4HTE  ,               DBE00830
     +             4HCDMM,4HENT ,4HEXPL,4HAIN ,4HPRIN,4HT   ,              DBE00840
     +             4HDUMP,4H    ,4HSET-,4HTITL,4HGET-,4HTITL,              DBE00850
     +             4HDDNE,4H    /                                          DBE00860
      DATA MENNM2 /4HDB-T,4HYPES/                                           DBE00870
      DATA NITEM2 /3/                                                       DBE00880
      DATA MITEM2 /4HINTE,4HGER ,4HREAL,4H    ,4HARRA,4HY-RL/               DBE00890
      IF (DBCLSD) GD TD 11111                                               DBE00900
         IF (FILEDB       ) CALL DXMSGZ(NAMSUB,5)                           DBE00910
         IF (.NDT.(FILEDB)) CALL DXMSGZ(NAMSUB,6)                           DBE00920
         CALL DXFNPR(DBSFIL)                                                DBE00930
         CALL DBEDTG(ISPAC1,WDSPA1)                                         DBE00940
```

```
C...                                                                    DBE00950
1000  CONTINUE                                                          DBE00960
C...       +++ REQUEST A EDIT-DB COMMAND                                DBE00970
         CALL DXMSGF(NAMSUB,1,ISPAC1,WDSPA1)                            DBE00980
         COMMAN=DXMENU(MENNM1,NITEM1,MITEM1,ISPAC1)                     DBE00990
         IF (DXREQS) GO TO 99999                                        DBE01000
            IF (COMMAN.GT.6) GO TO 3000                                 DBE01010
               IF (COMMAN.GT.2) GO TO 2000                              DBE01020
C...         +++ REQUEST TYPE OF DATUM                                  DBE01030
                  CALL DXMSGF(NAMSUB,2,ISPAC1,WDSPA1)                   DBE01040
                  TYPE=DXMENU(MENNM2,NITEM2,MITEM2,ISPAC1)              DBE01050
                  IF (DXREQS) GO TO 99999                               DBE01060
                     IF (TYPE.LT.3) GO TO 2000                          DBE01070
C...                +++ FOR ARRAYS REQUEST SIZE                         DBE01080
                        CALL DXMSGF(NAMSUB,3,ISPAC1,WDSPA1)             DBE01090
                        LOG=DXNBRI(ONE,ISPAC2,RSPAC3,NEEDUM,ISPAC1,1)   DBE01100
                        IF (DXREQS) GO TO 99999                         DBE01110
                        IF (.NOT.LOG) GO TO 99999                       DBE01120
                        ARSIZE=ISPAC2(1)                                DBE01130
2000              CONTINUE                                              DBE01140
C...            +++ REQUEST NAME OF THE DATUM                           DBE01150
                  LOG=DXGNAM(NAME)                                      DBE01160
                  IF (DXREQS) GO TO 99999                               DBE01170
3000           CONTINUE                                                 DBE01180
                  GO TO (3010,3020,3030,3040,3050,                      DBE01190
     +                    3060,3070,3080,3090,3100),COMMAN              DBE01200
3010  CONTINUE                                                          DBE01210
C...            +++ CREATE A NODE IN THE DATABASE                       DBE01220
         CALL DBEDCR(NAME,TYPE,ARSIZE)                                  DBE01230
         GO TO 1000                                                     DBE01240
3020  CONTINUE                                                          DBE01250
C...            +++ STORE A DATA VALUE IN THE NODE                      DBE01260
         CALL DBEDST(NAME,TYPE,ARSIZE,RSPAC3,WDSPA3)                    DBE01270
         IF (DXREQS) GO TO 99999                                        DBE01280
            GO TO 1000                                                  DBE01290
3030  CONTINUE                                                          DBE01300
C...            +++ DELETE A NODE                                       DBE01310
         CALL DBEDDL(NAME)                                              DBE01320
         GO TO 1000                                                     DBE01330
3040  CONTINUE                                                          DBE01340
C...           +++ PUT A COMMENT FOR THE NODE INTO THE DATABASE         DBE01350
         CALL DBEDCM(NAME,ISPAC1,WDSPA1,ISPAC2,WDSPA2,NCHCMT)           DBE01360
         IF (DXREQS) GO TO 99999                                        DBE01370
            GO TO 1000                                                  DBE01380
3050  CONTINUE                                                          DBE01390
C...            +++ EXPLAIN THE NODE BY DISPLAYING THE COMMENT          DBE01400
         CALL DBEDEX(NAME,ISPAC1,WDSPA1,NCHCMT)                         DBE01410
```

```
         GO TO 1000                                                    DBEO1420
3060  CONTINUE                                                         DBEO1430
C...           +++ PRINT A SPECIFIED DATUM VALUE                       DBEO1440
         CALL DBEDPR(NAME,ARSIZE,RSPAC3,WDSPA3)                        DBEO1450
         GO TO 1000                                                    DBEO1460
3070  CONTINUE                                                         DBEO1470
C...           +++ DUMP THE WHOLE DATABASE                             DBEO1480
         CALL DBEDDM(ISPAC1,WDSPA1,RSPAC3,WDSPA3)                      DBEO1490
         IF (DXREQS) GO TO 99999                                       DBEO1500
            GO TO 1000                                                 DBEO1510
3080  CONTINUE                                                         DBEO1520
C...           +++ STORE TITLE IN DATABASE                             DBEO1530
         CALL DBEDTS(ISPAC1,WDSPA1,ISPAC2,WDSPA2,NCHTIT)               DBEO1540
         IF (DXREQS) GO TO 99999                                       DBEO1550
            GO TO 1000                                                 DBEO1560
3090  CONTINUE                                                         DBEO1570
C...           +++ GET THE TITLE FROM THE DATABASE                     DBEO1580
         CALL DBEDTG(ISPAC1,WDSPA1)                                    DBEO1590
         GO TO 1000                                                    DBEO1600
3100  CONTINUE                                                         DBEO1610
C...           +++ DONE WITH THIS MENU, SO RETURN                      DBEO1620
         GO TO 99999                                                   DBEO1630
C...                                                                   DBEO1640
11111 CONTINUE                                                         DBEO1650
         CALL DXMSGZ(NAMSUB,4)                                         DBEO1660
         GO TO 99999                                                   DBEO1670
99999 CONTINUE                                                         DBEO1680
         RETURN                                                        DBEO1690
      END                                                              DBEO1700
```

```
C                                                                       DBE01710
C ***********************************************************************DBE01720
      SUBROUTINE DBEDCR(NAME,TYPE,ARSIZE)                               DBE01730
C  +-------------------------------------------------------------------+DBE01740
C                                                                       DBE01750
C        DBEDCR -                                                       DBE01760
C                                                                       DBE01770
C             CREATE A NODE IN THE DATABASE FOR THE GIVEN VARIABLE      DBE01780
C             NAME.  TYPE IS 1,2, DR 3 FDR INTEGER, REAL, REAL-ARRAY.   DBE01790
C             IF TYPE = 3 THEN ARSIZE IS THE SIZE DF THE ARRAY.         DBE01800
C                                                                       DBE01810
C  +-------------------------------------------------------------------+DBE01820
C...                 *THE COMMON CDNTRDL PARAMETERS,CHARACTERS, AND     DBE01830
C...                 *GRAPHIC DISPLAY CONTROL PARAMETERS                DBE01840
      COMMDN /DEXCDM/                                                   DBE01850
     1               DXMODE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,         DBE01860
     2               TERSE ,ECHOMD,DXSC   ,CPWDXS,MDSC   ,CURMDD,       DBE01870
     3               NULMSG,DXNCPW,NAMLEN,                              DBE01880
     4               CLRMOD,MUNPST,ERASE,UPDATE,UPAINT,GCNTRL           DBE01890
      INTEGER                                                           DBE01900
     1               DXSC   ,CPWDXS,MDSC   ,CURMOD,NULMSG,DXNCPW,       DBE01910
     2               NAMLEN,CLRMOD,MUNPST,ERASE,UPDATE,UPAINT           DBE01920
      LDGICAL                                                           DBE01930
     1               DXMDDE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,         DBE01940
     2               TERSE ,ECHOMD,GCNTRL                               DBE01950
      DIMENSION      DXSC(1),CURMOD(2)   ,NULMSG(1)                     DBE01960
      INTEGER   TYPE,RCODE,ARSIZE                                       DBE01970
      LDGICAL  DBVINS,DBCMPR                                            DBE01980
      DIMENSION NAME(2)                                                 DBE01990
      DATA NAMSUB /4HEDIT/                                              DBE02000
C...          +++ INSERT A NODE IN THE DATABASE WITH L.F. DBVINS        DBE02010
 1000 IF (DBVINS(NAME,TYPE,ARSIZE,RCDDE)) GD TD 99999                   DBE02020
C...          +++ ERROR OCCURRED - ANNDUNCE IT.                         DBE02030
C...          +++ NODE ALREADY EXISTS                                   DBE02040
        IF (RCDDE.EQ.2) CALL DXMSGC(NAMSUB,7,NAME,NAMLEN,DXNCPW)        DBE02050
C...          +++ NO MORE SPACE FDR NDDES SD CDMPRESS THE DATABASE      DBE02060
        IF (RCDDE.EQ.3) GD TO 11111                                     DBE02070
         GD TO 99999                                                    DBE02080
11111   IF (DBCMPR(RCODE)) GD TD 1000                                   DBE02090
        CALL DXMSGZ(NAMSUB,8)                                           DBE02100
         GD TD 99999                                                    DBE02110
99999 CONTINUE                                                          DBE02120
        RETURN                                                          DBE02130
      END                                                               DBE02140
```

```
C                                                                       DBEO2150
C ***********************************************************************DBEO2160
      SUBROUTINE DBEDST(NAME,TYPE,ARSIZE,RARRAY,WDARSP)                 DBEO2170
C  +--------------------------------------------------------------------+DBEO2180
C                                                                       DBEO2190
C        DBEDST -                                                       DBEO2200
C                                                                       DBEO2210
C             STORE A VALUE OF GIVEN TYPE IN THE NODE PREVIOUSLY        DBEO2220
C             CREATED FOR THE VARIABLE NAME.                            DBEO2230
C                                                                       DBEO2240
C             ARSIZE IS THE SIZE OF THE ARRAY TO BE STORED.             DBEO2250
C                                                                       DBEO2260
C             RARRAY IS A TEMPORARY IN WHICH THE ARRAY WILL BE STORED   DBEO2270
C             UNTIL THE ELEMENTS ARE ENTERED INTO THE ECS STORAGE AREA. DBEO2280
C                                                                       DBEO2290
C             WDARSP IS THE DIMENSION OF RARRAY.                        DBEO2300
C                                                                       DBEO2310
C  +--------------------------------------------------------------------+DBEO2320
C...                  *THE COMMON CONTROL PARAMETERS,CHARACTERS, AND    DBEO2330
C...                  *GRAPHIC DISPLAY CONTROL PARAMETERS               DBEO2340
       COMMON /DEXCOM/                                                  DBEO2350
      1                DXMODE,LOADED,MDPEND,DXREQS,DLREQS,KEYBRD,       DBEO2360
      2                TERSE ,ECHOMD,DXSC  ,CPWDXS,MDSC  ,CURMOD,       DBEO2370
      3                NULMSG,DXNCPW,NAMLEN,                            DBEO2380
      4                CLRMOD,MUNPST,ERASE,UPDATE,UPAINT,GCNTRL         DBEO2390
       INTEGER                                                          DBEO2400
      1                DXSC  ,CPWDXS,MDSC  ,CURMOD,NULMSG,DXNCPW,       DBEO2410
      2                NAMLEN,CLRMOD,MUNPST,ERASE,UPDATE,UPAINT         DBEO2420
       LOGICAL                                                          DBEO2430
      1                DXMODE,LOADED,MDPEND,DXREQS,DLREQS,KEYBRD,       DBEO2440
      2                TERSE ,ECHOMD,GCNTRL                             DBEO2450
       DIMENSION       DXSC(1),CURMOD(2)  ,NULMSG(1)                    DBEO2460
       COMMON /DBCOM/                                                   DBEO2470
      1               TITLE ,BUCKET,IDENT ,NODTYP,DATUM ,LINK  ,LINKF ,DBEO2480
      1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,DBEO2490
      2               INTTYP,RELTYP,ARRTYP,                             DBEO2500
      3               DBCLSD,DBACTV,FILEDB                              DBEO2510
       INTEGER                                                          DBEO2520
      1               TITLE ,BUCKET,IDENT ,NODTYP,DATUM ,LINK  ,LINKF ,DBEO2530
      1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,DBEO2540
      2               INTTYP,RELTYP,ARRTYP                              DBEO2550
       LOGICAL                                                          DBEO2560
      3               DBCLSD,DBACTV,FILEDB                              DBEO2570
       DIMENSION      TITLE(16),BUCKET(32),IDENT(200,2),NODTYP(200),    DBEO2580
      1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)       DBEO2590
       INTEGER   TYPE,STYPE,ONE,TWO,RCODE,ARSIZE,DBNSKR,WDARSP          DBEO2600
       LOGICAL   DXNBRI,RPUT,IPUT,APUT,LOG,DBXECS                       DBEO2610
```

```
      DIMENSION NAME(2),RARRAY(WDARSP),IVALUE(1),RVALUE(1),IDUM(1),      DBE02620
     +          DUM(1)                                                   DBE02630
      EQUIVALENCE (IVALUE(1),RVALUE(1))                                   DBE02640
      DATA NAMSUB /4HEDIT/                                                DBE02650
      DATA     ONE /1/,TWO/2/                                             DBE02660
C                                                                         DBE02670
C...          +++ SEARCH FOR THE NODE USING THE HASHED VALUE AS AN INDEXDBE02680
C                                                                         DBE02690
      NODE=DBNSKR(NAME,STYPE,IDUM,RCODE)                                  DBE02700
      IF (NODE.EQ.0) GO TO 11111                                          DBE02710
         IF (TYPE.NE.IABS(STYPE)) GO TO 22222                             DBE02720
            GO TO (1000,2000,3000),TYPE                                   DBE02730
C...                                                                      DBE02740
1000  CONTINUE                                                            DBE02750
C...          +++ STORE AN INTEGER VALUE                                  DBE02760
            LOG=DXNBRI(ONE,IVALUE,DUM,NEED,NULMSG,ONE)                    DBE02770
            IF (DXREQS) GO TO 99999                                       DBE02780
            IF(.NOT.LOG) GO TO 99999                                      DBE02790
               LOG=IPUT(NAME,IVALUE,RCODE)                                DBE02800
               GO TO 99999                                                DBE02810
2000  CONTINUE                                                            DBE02820
C...          +++ STORE A REAL VALUE                                      DBE02830
            LOG=DXNBRI(ONE,IDUM,RVALUE,NEED,NULMSG,TWO)                   DBE02840
            IF (DXREQS) GO TO 99999                                       DBE02850
            IF(.NOT.LOG) GO TO 99999                                      DBE02860
               LOG=RPUT(NAME,RVALUE,RCODE)                                DBE02870
               GO TO 99999                                                DBE02880
3000  CONTINUE                                                            DBE02890
C...          +++ STORE A REAL ARRAY                                      DBE02900
            N=ARSIZE                                                      DBE02910
            IF (N.GT.MAXARR) GO TO 33333                                  DBE02920
               LOG=DXNBRI(N,IDUM,RARRAY,NEED,NULMSG,TWO)                  DBE02930
               IF (DXREQS) GO TO 99999                                    DBE02940
               IF(NEED.EQ.N) GO TO 99999                                  DBE02950
               N=N-NEED                                                   DBE02960
                  LOG=APUT(NAME,RARRAY,N,NSTORD,RCODE)                    DBE02970
                  IF(RCODE.EQ.4.OR.RCODE.EQ.5) GO TO 44444                DBE02980
                  GO TO 99999                                             DBE02990
C...                                                                      DBE03000
11111 CONTINUE                                                            DBE03010
C...          +++ NODE DOES NOT EXIST                                     DBE03020
            CALL DXMSGC(NAMSUB,9,NAME,NAMLEN,DXNCPW)                      DBE03030
            GO TO 99999                                                   DBE03040
22222 CONTINUE                                                            DBE03050
C...          +++ WRONG TYPE                                              DBE03060
            CALL DXMSGZ(NAMSUB,10)                                        DBE03070
            GO TO 99999                                                   DBE03080
```

```
33333 CONTINUE                                                          DBE03090
C...          +++ ARRAY TOO BIG TO BE STORED                            DBE03100
         CALL DXMSGZ(NAMSUB,11)                                         DBE03110
         GO TO 99999                                                    DBE03120
44444 CONTINUE                                                          DBE03130
C...          +++ ARRAY PUT DID NOT TAKE PLACE SO EXPAND THE ECS AREA   DBE03140
         LOG=DBXECS(IDUM)                                               DBE03150
         IF (LOG) GO TO 3000                                            DBE03160
C...          +++ COULD NOT EXPAND SO ANNOUNCE AND RETURN               DBE03170
         CALL DXMSGI(NAMSUB,17,NSTORD)                                  DBE03180
         GO TO 99999                                                    DBE03190
99999 CONTINUE                                                          DBE03200
         RETURN                                                         DBE03210
      END                                                               DBE03220
```

```
C                                                                       DBE03230
C ***********************************************************************DBE03240
      SUBROUTINE DBEDDL(NAME)                                           DBE03250
C  +-------------------------------------------------------------------+DBE03260
C                                                                       DBE03270
C        DBEDDL -                                                       DBE03280
C                                                                       DBE03290
C             DELETE A NDDE FRDM THE DATABASE BY MAKING IT UNAVAILABLE. DBE03300
C             THIS IS DDNE BY FLAGGING NDDTYP AS -9                     DBE03310
C                                                                       DBE03320
C  +-------------------------------------------------------------------+DBE03330
C                                                                       DBE03340
C...                 *THE COMMDN CDNTRDL PARAMETERS,CHARACTERS, AND     DBE03350
C...                 *GRAPHIC DISPLAY CONTRDL PARAMETERS                DBE03360
      CDMMON /DEXCOM/                                                   DBE03370
     1               DXMDDE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,         DBE03380
     2               TERSE ,ECHOMD,DXSC  ,CPWDXS,MDSC  ,CURMDD,         DBE03390
     3               NULMSG,DXNCPW,NAMLEN,                              DBE03400
     4               CLRMOD,MUNPST,ERASE,UPDATE,UPAINT,GCNTRL           DBE03410
      INTEGER                                                           DBE03420
     1               DXSC  ,CPWDXS,MDSC  ,CURMDD,NULMSG,DXNCPW,         DBE03430
     2               NAMLEN,CLRMDD,MUNPST,ERASE,UPDATE,UPAINT           DBE03440
      LDGICAL                                                           DBE03450
     1               DXMODE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,         DBE03460
     2               TERSE ,ECHOMD,GCNTRL                               DBE03470
      DIMENSIDN      DXSC(1),CURMDD(2)  ,NULMSG(1)                      DBE03480
      INTEGER RCDDE                                                     DBE03490
      LDGICAL  DBVDEL                                                   DBE03500
      DIMENSIDN NAME(2)                                                 DBE03510
      DATA NAMSUB /4HEDIT/                                              DBE03520
C...          +++ INVDKE THE LDGICAL FUNCTION DBVDEL TD DELETE THE NODE DBE03530
      IF (DBVDEL(NAME,RCDDE)) GD TD 99999                               DBE03540
C...          +++ UNABLE TD FIND NAME TD DELETE, ANNDUNCE.              DBE03550
         CALL DXMSGC(NAMSUB,9,NAME,NAMLEN,DXNCPW)                       DBE03560
         GD TO 99999                                                    DBE03570
99999 CDNTINUE                                                          DBE03580
         RETURN                                                         DBE03590
      END                                                               DBE03600
```

```
C                                                                       DBE03610
C ***********************************************************************DBE03620
      SUBROUTINE DBEDCM(NAME,MSG,MSGWDS,CMTBUF,WDSBUF,NCHCMT)           DBE03630
C  +-------------------------------------------------------------------+DBE03640
C                                                                       DBE03650
C        DBEDCM -                                                       DBE03660
C                                                                       DBE03670
C             PUT A COMMENT IN THE DATABASE FOR A PREVIDUSLY ENTERED    DBE03680
C             VARIABLE NAME.                                            DBE03690
C                                                                       DBE03700
C             NAME IS THE NAME DF THE DATUM.                            DBE03710
C                                                                       DBE03720
C             MSG IS A MESSAGE ARRAY, AND MSGWDS IS ITS DIMENSIDN.      DBE03730
C                                                                       DBE03740
C             CMTBUF IS THE CDMMENT ARRAY, WDSBUF IS ITS DIMENSIDN,     DBE03750
C             AND NCHCMT IS THE LENGTH OF THE CDMMENT TD BE ENTERED.    DBE03760
C                                                                       DBE03770
C  +-------------------------------------------------------------------+DBE03780
C                                                                       DBE03790
C...                  *THE CDMMDN CONTRDL PARAMETERS,CHARACTERS, AND    DBE03800
C...                  *GRAPHIC DISPLAY CONTRDL PARAMETERS               DBE03810
      CDMMON /DEXCDM/                                                   DBE03820
     1                DXMODE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,        DBE03830
     2                TERSE ,ECHDMD,DXSC  ,CPWDXS,MDSC  ,CURMDD,        DBE03840
     3                NULMSG,DXNCPW,NAMLEN,                             DBE03850
     4                CLRMDD,MUNPST,ERASE,UPDATE,UPAINT,GCNTRL          DBE03860
      INTEGER                                                           DBE03870
     1                DXSC  ,CPWDXS,MDSC  ,CURMDD,NULMSG,DXNCPW,        DBE03880
     2                NAMLEN,CLRMOD,MUNPST,ERASE,UPDATE,UPAINT          DBE03890
      LDGICAL                                                           DBE03900
     1                DXMDDE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,        DBE03910
     2                TERSE ,ECHDMD,GCNTRL                              DBE03920
      DIMENSIDN       DXSC(1),CURMOD(2)  ,NULMSG(1)                     DBE03930
      INTEGER   WDSBUF,CMTBUF,DBNSKR,RCODE,STYPE,CPWBUF                 DBE03940
      LDGICAL  LDG,DXCHRI,CMTPUT,DXPRMT,DBXECS                          DBE03950
      DIMENSIDN CMTBUF(WDSBUF),NAME(2),MSG(MSGWDS)                      DBE03960
      DATA NAMSUB /4HEDIT/                                              DBE03970
C...          +++ SEARCH THE DATABASE FDR THE NDDE                      DBE03980
      NODE=DBNSKR(NAME,STYPE,IDUM,RCDDE)                                DBE03990
      IF (NODE.EQ.O) GD TO 11111                                        DBE04000
         CALL DXMSGF(NAMSUB,12,MSG,MSGWDS)                              DBE04010
C...          +++ GET THE COMMENT FRDM THE USER                         DBE04020
         LDG=DXCHRI(NCHCMT,CMTBUF,WDSBUF,DXNCPW,NEED,MSG)               DBE04030
         IF (DXREQS) GO TD 99999                                        DBE04040
C...          +++ PUT THE CDMMENT IN THE DATABASE                       DBE04050
 1000       LDG=CMTPUT(NAME,CMTBUF,RCDDE)                              DBE04060
            IF (RCDDE.EQ.3) GD TO 22222                                 DBE04070
```

```
               GO TO 99999                                              DBE04080
C...                                                                    DBE04090
11111 CONTINUE                                                          DBE04100
C...           +++ NODE DOES NOT EXIST                                  DBE04110
         CALL DXMSGC(NAMSUB,9,NAME,NAMLEN,DXNCPW)                       DBE04120
         GO TO 99999                                                    DBE04130
22222 CONTINUE                                                          DBE04140
C...           +++ NO MORE SPACE FOR COMMENTS, EXPAND THE ECS AREA      DBE04150
         LOG=DBXECS(IDUM)                                               DBE04160
         IF (LOG) GO TO 1000                                            DBE04170
C...           +++ COULD NOT EXPAND ANNOUNCE AND RETURN                 DBE04180
         CALL DXMSGZ(NAMSUB,13)                                         DBE04190
         GO TO 99999                                                    DBE04200
99999 CONTINUE                                                          DBE04210
         RETURN                                                         DBE04220
      END                                                               DBE04230
```

```
C                                                                       OBE04240
C ***********************************************************************OBE04250
      SUBRDUTINE OBEOEX(NAME,CMTBUF,WOSBUF,NCHCMT)                      OBE04260
C  +--------------------------------------------------------------------+OBE04270
C                                                                       OBE04280
C        OBEOEX -                                                       OBE04290
C                                                                       OBE04300
C             EXPLAIN THE OATABASE ENTRY BY OISPLAYING ITS COMMENT.     OBE04310
C                                                                       OBE04320
C             NAME IS THE NAME DF THE OATUM.                            DBE04330
C                                                                       OBE04340
C             CMTBUF IS THE ARRAY BUFFER WHICH THE CDMMENT IS PLACED IN OBE04350
C             NCHCMT IS THE NUMBER OF CHARACTERS IN THE COMMENT.        OBE04360
C                                                                       OBE04370
C  +--------------------------------------------------------------------+DBE04380
C                                                                       OBE04390
C...                  *THE CDMMDN CONTROL PARAMETERS,CHARACTERS, ANO    DBE04400
C...                  *GRAPHIC DISPLAY CONTROL PARAMETERS               DBE04410
      COMMON /OEXCOM/                                                   DBE04420
     1                OXMDOE,LOAOEO,MOPEND,OXREQS,DLREQS,KEYBRO,        OBE04430
     2                TERSE ,ECHOMD,DXSC  ,CPWOXS,MDSC  ,CURMOO,        DBE04440
     3                NULMSG,OXNCPW,NAMLEN,                             OBE04450
     4                CLRMDO,MUNPST,ERASE,UPOATE,UPAINT,GCNTRL          OBE04460
      INTEGER                                                           OBE04470
     1                OXSC  ,CPWDXS,MDSC  ,CURMOO,NULMSG,OXNCPW,        DBE04480
     2                NAMLEN,CLRMDO,MUNPST,ERASE,UPOATE,UPAINT          DBE04490
      LOGICAL                                                           DBE04500
     1                OXMODE,LOADED,MOPENO,OXREQS,OLREQS,KEYBRO,        OBE04510
     2                TERSE ,ECHDMO,GCNTRL                              OBE04520
      DIMENSION       OXSC(1),CURMOD(2)  ,NULMSG(1)                     OBE04530
      INTEGER CMTBUF,RCODE,WOSBUF                                       DBE04540
      LOGICAL  LOG,CMTGET                                               DBE04550
      OIMENSION CMTBUF(WOSBUF),NAME(2)                                  DBE04560
      OATA NAMSUB /4HEDIT/                                              DBE04570
C...          +++ GET THE CDMMENT FROM THE OATABASE.                    DBE04580
      LOG=CMTGET(NAME,CMTBUF,RCOOE)                                     OBE04590
      IF (RCOOE.EQ.2) GO TO 11111                                       DBE04600
         IF (RCDOE.EQ.3) GO TO 22222                                    OBE04610
            CALL OXMSGN(CMTBUF,NCHCMT)                                  OBE04620
            GO TO 99999                                                 DBE04630
11111 CDNTINUE                                                          OBE04640
C...          +++ NOOE OOES NOT EXIST                                   OBE04650
         CALL OXMSGC(NAMSUB,9,NAME,NAMLEN,OXNCPW)                       OBE04660
         GO TO 99999                                                    OBE04670
22222 CONTINUE                                                          OBE04680
C...          +++ NO COMMENT STOREO                                     OBE04690
         CALL DXMSGZ(NAMSUB,14)                                         OBE04700
```

```
         GO TO 99999                                                    DBE04710
99999 CONTINUE                                                          DBE04720
         RETURN                                                         DBE04730
      END                                                               DBE04740
```

```
C                                                                       DBE04750
C ***********************************************************************DBE04760
      SUBRDUTINE DBEDPR(NAME,ARSIZE,ARSPAC,WDARSP)                      DBE04770
C  +--------------------------------------------------------------------+DBE04780
C                                                                       DBE04790
C        DBEDPR -                                                       DBE04800
C                                                                       DBE04810
C             PRINT THE VALUE DF A DATABASE ENTRY AT THE TERMINAL.      DBE04820
C                                                                       DBE04830
C             NAME IS THE NAME DF THE DATUM.                            DBE04840
C                                                                       DBE04850
C             ARSIZE IS THE SIZE DF THE ARRAY, ARSPAC IS A TEMPDRARY    DBE04860
C             ARRAY IN WHICH TD STDRE THE ARRAY, AND WDARSP IS THE      DBE04870
C             DIMENSIDN OF ARSPAC.                                      DBE04880
C                                                                       DBE04890
C  +--------------------------------------------------------------------+DBE04900
C                                                                       DBE04910
C...                  *THE COMMON CDNTROL PARAMETERS,CHARACTERS, AND    DBE04920
C...                  *GRAPHIC DISPLAY CDNTRDL PARAMETERS               DBE04930
       CDMMON /DEXCDM/                                                  DBE04940
      1                DXMODE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,       DBE04950
      2                TERSE ,ECHDMD,DXSC  ,CPWDXS,MDSC  ,CURMOD,       DBE04960
      3                NULMSG,DXNCPW,NAMLEN,                            DBE04970
      4                CLRMOD,MUNPST,ERASE,UPDATE,UPAINT,GCNTRL         DBE04980
       INTEGER                                                          DBE04990
      1                DXSC  ,CPWDXS,MDSC  ,CURMOD,NULMSG,DXNCPW,       DBE05000
      2                NAMLEN,CLRMOD,MUNPST,ERASE,UPDATE,UPAINT         DBE05010
       LDGICAL                                                          DBE05020
      1                DXMDDE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,       DBE05030
      2                TERSE ,ECHOMD,GCNTRL                             DBE05040
       DIMENSIDN       DXSC(1),CURMDD(2)  ,NULMSG(1)                    DBE05050
       CDMMON /DBCOM/                                                   DBE05060
      1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,DBE05070
      1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,DBE05080
      2               INTTYP,RELTYP,ARRTYP,                             DBE05090
      3               DBCLSD,DBACTV,FILEDB                              DBE05100
       INTEGER                                                          DBE05110
      1               TITLE ,BUCKET,IDENT ,NODTYP,DATUM ,LINK  ,LINKF ,DBE05120
      1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,DBE05130
      2               INTTYP,RELTYP,ARRTYP                              DBE05140
       LDGICAL                                                          DBE05150
      3               DBCLSD,DBACTV,FILEDB                              DBE05160
       DIMENSIDN      TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),    DBE05170
      1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)       DBE05180
       REAL     RDATUM                                                  DBE05190
       INTEGER  STYPE,RCODE,IDATUM,DBNSKR,UNDEF,UNDLEN,UNDCPW           DBE05200
       INTEGER ARSIZE,WDARSP                                            DBE05210
```

```
      DIMENSION NAME(2),UNDEF(3),ARSPAC(WDARSP)                          DBE05220
      EQUIVALENCE (IDATUM,RDATUM)                                        DBE05230
      DATA NAMSUB /4HEDIT/                                               DBE05240
      DATA UNDEF /4HUNDE,4HFINE,4HD    /                                 DBE05250
      DATA UNDLEN /10/                                                   DBE05260
C...          +++ SEARCH THE DATABASE FOR THE NODE                       DBE05270
      NODE=DBNSKR(NAME,STYPE,IDUM,RCODE)                                 DBE05280
      IF (NODE.EQ.0) GO TO 11111                                         DBE05290
C...          +++ EXTRACT THE VALUE                                      DBE05300
         IDATUM=DATUM(NODE)                                              DBE05310
         CALL DXMSGC(NAMSUB,15,NAME,NAMLEN,DXNCPW)                       DBE05320
         IF (STYPE.LT.0) GO TO 22222                                     DBE05330
C...          +++ DETERMINE THE TYPE                                     DBE05340
            GO TO (1000,2000,3000),STYPE                                 DBE05350
1000        CONTINUE                                                     DBE05360
C...          +++ PRINT INTEGER DATUM                                    DBE05370
               CALL DXMSGI(NAMSUB,15,IDATUM)                             DBE05380
               GO TO 99999                                               DBE05390
2000        CONTINUE                                                     DBE05400
C...          +++ PRINT REAL DATUM                                       DBE05410
               CALL DXMSGR(NAMSUB,15,RDATUM)                             DBE05420
               GO TO 99999                                               DBE05430
3000        CONTINUE                                                     DBE05440
C...          +++ PRINT REAL-ARRAY                                       DBE05450
               CALL DBEDAP(NAME,IDATUM,WDARSP,ARSPAC,WDARSP)             DBE05460
               GO TO 99999                                               DBE05470
C...                                                                     DBE05480
11111 CONTINUE                                                           DBE05490
C...          +++ NAMED DATUM DOES NOT EXIST                             DBE05500
         CALL DXMSGC(NAMSUB,9,NAME,NAMLEN,DXNCPW)                        DBE05510
         GO TO 99999                                                     DBE05520
22222 CONTINUE                                                           DBE05530
C...          +++ DATUM IS UNDEFINED                                     DBE05540
         CALL DXMSGC(NAMSUB,16,UNDEF,UNDLEN,DXNCPW)                      DBE05550
         GO TO 99999                                                     DBE05560
99999 CONTINUE                                                           DBE05570
         RETURN                                                          DBE05580
      END                                                                DBE05590
```

```
C                                                                       DBE05600
C ***********************************************************************DBE05610
      SUBROUTINE DBEDAP(NAME,NARRAY,NGET,RARRAY,WDARSP)                 DBE05620
C  +-------------------------------------------------------------------+DBE05630
C                                                                       DBE05640
C        DBEDAP -                                                       DBE05650
C                                                                       DBE05660
C             PRINT THE CONTENTS OF AN ARRAY FROM THE DATABASE.         DBE05670
C                                                                       DBE05680
C             NAME IS THE NAME OF THE DATUM.                            DBE05690
C                                                                       DBE05700
C             NARRAY IS THE POINTER TO THE LOCATION OF THE ARRAY IN     DBE05710
C             THE ECS STORAGE AREA.  NGET IS THE NUMBER TO GET.         DBE05720
C                                                                       DBE05730
C             RARRAY IS THE ARRAY IN WHICH TO PUT THE VALUES, AND       DBE05740
C             WDARSP IS THE DIMENSION OF RARRAY.                        DBE05750
C                                                                       DBE05760
C  +-------------------------------------------------------------------+DBE05770
C                                                                       DBE05780
C...                 *THE COMMON CONTROL PARAMETERS,CHARACTERS, AND     DBE05790
C...                 *GRAPHIC DISPLAY CONTROL PARAMETERS                DBE05800
      COMMON /DEXCOM/                                                   DBE05810
     1               DXMODE,LOADED,MOPEND,DXREQS,DLREQS,KEYBRD,         DBE05820
     2               TERSE ,ECHOMD,DXSC   ,CPWDXS,MDSC   ,CURMOD,       DBE05830
     3               NULMSG,DXNCPW,NAMLEN,                              DBE05840
     4               CLRMOD,MUNPST,ERASE,UPDATE,UPAINT,GCNTRL           DBE05850
      INTEGER                                                           DBE05860
     1               DXSC   ,CPWDXS,MDSC   ,CURMOD,NULMSG,DXNCPW,       DBE05870
     2               NAMLEN,CLRMOD,MUNPST,ERASE,UPDATE,UPAINT           DBE05880
      LOGICAL                                                           DBE05890
     1               DXMODE,LOADED,MOPEND,DXREQS,DLREQS,KEYBRD,         DBE05900
     2               TERSE ,ECHOMD,GCNTRL                               DBE05910
      DIMENSION      DXSC(1),CURMOD(2)   ,NULMSG(1)                     DBE05920
C...                 *I/O DEVICES AND FILES                             DBE05930
      COMMON /DEXFIL/                                                   DBE05940
     1               MSGSDV,USEDEV,INFODV,DBSDEV,NEWSDV,HELPDV,         DBE05950
     2               OUTDEV,INPDEV,NFWDS ,                              DBE05960
     3               MSGSFL,USEFIL,INFOFL,DBSFIL,NEWSFL,HELPFL,         DBE05970
     4               OUTFIL,INPFIL,LOADFL,NOT1FL,NOT2FL                 DBE05980
      INTEGER                                                           DBE05990
     1               MSGSDV,USEDEV,INFODV,DBSDEV,NEWSDV,HELPDV,         DBE06000
     2               OUTDEV,INPDEV,                                     DBE06010
     3               MSGSFL,USEFIL,INFOFL,DBSFIL,NEWSFL,HELPFL,         DBE06020
     4               OUTFIL,INPFIL,LOADFL,NOT1FL,NOT2FL,NFWDS           DBE06030
C...  FILE NAME DIMENSIONING ADJUSTED FOR CDC AND CU ONLY.....          DBE06040
      DIMENSION      MSGSFL(11)  ,USEFIL(11)  ,INFOFL(11)  ,            DBE06050
     1               DBSFIL(11)  ,NEWSFL(11)  ,HELPFL(11)  ,            DBE06060
```

```
     3                DUTFIL(11)  ,INPFIL(11)  ,                          DBE06070
     2                LDADFL(11)  ,NDT1FL(11)  ,NDT2FL(11)               DBE06080
C...                 *THE MESSAGE HANDLING STORE AND PARAMETERS          DBE06090
      COMMON /DEXMSG/                                                    DBE06100
     1                DXCHAR,MDCHAR,TRIMCH,MAXMSG,MAXWDS,MXPODL,         DBE06110
     2                SUBCNT,SUBCOD,MSGPTR,PVERB ,LVERB ,PTERSE,         DBE06120
     3                LTERSE,LSTRNG,VPOSIT,TPOSIT,MSPOOL                 DBE06130
      INTEGER                                                            DBE06140
     1                DXCHAR,MDCHAR,TRIMCH,MAXMSG,MAXWDS,MXPDOL,         DBE06150
     2                SUBCNT,SUBCOD,MSGPTR,PVERB ,LVERB ,PTERSE,         DBE06160
     3                LTERSE,LSTRNG,VPOSIT,TPOSIT,MSPDOL                 DBE06170
      DIMENSION       SUBCOD( 40),MSGPTR(102),PVERB (102),               DBE06180
     5                LVERB (102),PTERSE(102),LTERSE(102),               DBE06190
     6                LSTRNG(102),VPOSIT(102),TPOSIT(102),               DBE06200
     7                MSPOOL(950),TRIMCH(1)                              DBE06210
      INTEGER START,WDARSP                                               DBE06220
      LOGICAL DBNARD,LDG                                                 DBE06230
      DIMENSION NAME(2),RARRAY(WDARSP)                                   DBE06240
      DATA START /1/                                                     DBE06250
C...          +++ READ THE ARRAY FROM ECS STORAGE                       DBE06260
      LDG=DBNARD(NARRAY,RARRAY,START,NGET,NSTORD)                        DBE06270
      CALL DXGMSG                                                        DBE06280
      WRITE (OUTDEV,00001) DXCHAR,NAME,NSTORD,NARRAY                     DBE06290
      JSTEP=5                                                            DBE06300
      ISTEP=40*JSTEP                                                     DBE06310
      DO 2000 I=1,NSTORD,ISTEP                                           DBE06320
         JBEG=1                                                          DBE06330
         JEND=MINO(I+ISTEP-1,NSTORD)                                     DBE06340
         DO 1000 J=JBEG,JEND,JSTEP                                       DBE06350
            KBEG=J                                                       DBE06360
            KEND=MINO(J+JSTEP-1,JEND)                                    DBE06370
            CALL DXGMSG                                                  DBE06380
            WRITE (OUTDEV,00002) DXCHAR, (RARRAY(K),K=KBEG,KEND)        DBE06390
C...          +++ UPDATE GRAPHICS                                        DBE06400
1000        CONTINUE                                                     DBE06410
            CALL DXWAIT                                                  DBE06420
            CALL DXUGRF(ERASE)                                           DBE06430
2000  CONTINUE                                                           DBE06440
      CALL DXUGRF(UPDATE)                                                DBE06450
      RETURN                                                             DBE06460
00001 FORMAT (1H /1H ,A1,6HNAME: ,2A4,9H, LENGTH=,I3,3X,1H(,I5,1H)/)     DBE06470
00002 FORMAT (1H ,A1,5E14.5)                                             DBE06480
      END                                                                DBE06490
```

```
C                                                                       DBE06500
C ***********************************************************************DBE06510
      SUBRDUTINE DBEDDM(CMTBUF,CMTWDS,RARRAY,RARWDS)                    DBE06520
C  +--------------------------------------------------------------------+DBE06530
C                                                                       DBE06540
C        DBEDDM -                                                       DBE06550
C                                                                       DBE06560
C             DUMP THE CONTENTS OF THE DATABASE TO THE TERMINAL         DBE06570
C             TO BE DISPLAYED.                                          DBE06580
C                                                                       DBE06590
C             CMTBUF IS AN ARRAY IN WHICH TO STORE COMMENTS TEMPORARILY DBE06600
C             CMTWDS IS THE NUMBER OF COMMENT WORDS.                    DBE06610
C                                                                       DBE06620
C             RARRAY IS AN ARRAY IN WHICH TO STORE THE ARRAY TEMPORARILYDBE06630
C             RARWDS IS THE NUMBER OF WORDS IN THE ARRAY.               DBE06640
C                                                                       DBE06650
C  +--------------------------------------------------------------------+DBE06660
C                                                                       DBE06670
C...                  *THE COMMON CONTROL PARAMETERS,CHARACTERS, AND    DBE06680
C...                  *GRAPHIC DISPLAY CONTROL PARAMETERS               DBE06690
      COMMON /EXCOM/                                                    DBE06700
     1                DXMODE,LOADED,MOPEND,DXREQS,DLREQS,KEYBRD,       DBE06710
     2                TERSE ,ECHOMD,DXSC  ,CPWDXS,MDSC  ,CURMOD,       DBE06720
     3                NULMSG,DXNCPW,NAMLEN,                            DBE06730
     4                CLRMOD,MUNPST,ERASE,UPDATE,UPAINT,GCNTRL         DBE06740
      INTEGER                                                           DBE06750
     1                DXSC  ,CPWDXS,MDSC  ,CURMOD,NULMSG,DXNCPW,       DBE06760
     2                NAMLEN,CLRMOD,MUNPST,ERASE,UPDATE,UPAINT         DBE06770
      LOGICAL                                                           DBE06780
     1                DXMODE,LOADED,MOPEND,DXREQS,DLREQS,KEYBRD,       DBE06790
     2                TERSE ,ECHOMD,GCNTRL                             DBE06800
      DIMENSION       DXSC(1),CURMOD(2)  ,NULMSG(1)                     DBE06810
C...                  *I/O DEVICES AND FILES                            DBE06820
      COMMON /EXFIL/                                                    DBE06830
     1                MSGSDV,USEDEV,INFODV,DBSDEV,NEWSDV,HELPDV,       DBE06840
     2                OUTDEV,INPDEV,NFWDS ,                            DBE06850
     3                MSGSFL,USEFIL,INFOFL,DBSFIL,NEWSFL,HELPFL,       DBE06860
     4                OUTFIL,INPFIL,LOADFL,NOT1FL,NOT2FL               DBE06870
      INTEGER                                                           DBE06880
     1                MSGSDV,USEDEV,INFODV,DBSDEV,NEWSDV,HELPDV,       DBE06890
     2                OUTDEV,INPDEV,                                   DBE06900
     3                MSGSFL,USEFIL,INFOFL,DBSFIL,NEWSFL,HELPFL,       DBE06910
     4                OUTFIL,INPFIL,LOADFL,NOT1FL,NOT2FL,NFWDS         DBE06920
C...  FILE NAME DIMENSIONING ADJUSTED FOR CDC AND CU ONLY.....          DBE06930
      DIMENSION       MSGSFL(11)  ,USEFIL(11)  ,INFOFL(11)  ,          DBE06940
     1                DBSFIL(11)  ,NEWSFL(11)  ,HELPFL(11)  ,          DBE06950
     3                OUTFIL(11)  ,INPFIL(11)  ,                       DBE06960
```

```
     2                LDADFL(11)   ,NDT1FL(11)   ,NDT2FL(11)                DBE06970
C...                 *THE MESSAGE HANDLING STORE AND PARAMETERS              DBE06980
      COMMON /DEXMSG/                                                        DBE06990
     1                DXCHAR,MDCHAR,TRIMCH,MAXMSG,MAXWDS,MXPDDL,             DBE07000
     2                SUBCNT,SUBCDD,MSGPTR,PVERB ,LVERB ,PTERSE,             DBE07010
     3                LTERSE,LSTRNG,VPDSIT,TPDSIT,MSPDDL                     DBE07020
      INTEGER                                                                DBE07030
     1                DXCHAR,MDCHAR,TRIMCH,MAXMSG,MAXWDS,MXPDDL,             DBE07040
     2                SUBCNT,SUBCDD,MSGPTR,PVERB ,LVERB ,PTERSE,             DBE07050
     3                LTERSE,LSTRNG,VPDSIT,TPDSIT,MSPDDL                     DBE07060
      DIMENSION        SUBCDD( 40),MSGPTR(102),PVERB (102),                   DBE07070
     5                LVERB (102),PTERSE(102),LTERSE(102),                   DBE07080
     6                LSTRNG(102),VPDSIT(102),TPDSIT(102),                   DBE07090
     7                MSPDDL(950),TRIMCH(1)                                  DBE07100
      COMMON /DBCOM/                                                         DBE07110
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,       DBE07120
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,       DBE07130
     2               INTTYP,RELTYP,ARRTYP,                                   DBE07140
     3               DBCLSD,DBACTV,FILEDB                                    DBE07150
      INTEGER                                                                DBE07160
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,       DBE07170
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,       DBE07180
     2               INTTYP,RELTYP,ARRTYP                                    DBE07190
      LOGICAL                                                                DBE07200
     3               DBCLSD,DBACTV,FILEDB                                    DBE07210
      DIMENSION       TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),         DBE07220
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)             DBE07230
      INTEGER   BKTPTR,TYPE,CMTBUF,CMTWDS,RARWDS,UTYPE,CLASS                 DBE07240
      LOGICAL   NDCOMM,UNDEFD,DBNARD,DBNCRD,LOG                              DBE07250
      DIMENSION NAME(2),CMTBUF(CMTWDS),RARRAY(RARWDS),CLASS(3)               DBE07260
      EQUIVALENCE (IDATUM,RDATUM)                                            DBE07270
      DATA CLASS /4H (I),4H (R),4H (A)/                                      DBE07280
      CALL DXUGRF(ERASE)                                                     DBE07290
      CALL DBEDTG(CMTBUF,CMTWDS)                                             DBE07300
      CALL DXGMSG                                                            DBE07310
      WRITE (OUTDEV,00001) DXCHAR                                            DBE07320
C...            +++ SCAN THE BUCKET FOR NODE ENTRIES                         DBE07330
      DO 7000 BKTPTR=1,32                                                    DBE07340
         NODE=BUCKET(BKTPTR)                                                 DBE07350
1000     CONTINUE                                                            DBE07360
            IF (NODE.EQ.0) GO TO 6000                                        DBE07370
C...          +++ IF A NODE IS NOT ZERO DUMP IT                              DBE07380
              NAME(1)=IDENT(NODE,1)                                          DBE07390
              NAME(2)=IDENT(NODE,2)                                          DBE07400
              NDCOMM=(CMTPTR(NODE).EQ.0)                                     DBE07410
              IF (.NOT.NDCOMM) LOG=DBNCRD(CMTPTR(NODE),CMTBUF)               DBE07420
              TYPE=NODDTYP(NODE)                                             DBE07430
```

```
                UNDEFD=(TYPE.LT.1)                                      DBE07440
                IF (UNDEFD) GO TO 3000                                  DBE07450
                   IDATUM=DATUM(NODE)                                   DBE07460
                   GO TO (2100,2200,2300),TYPE                          DBE07470
2100               CONTINUE                                             DBE07480
C...                  +++ DUMP THE INTEGER DATUM                        DBE07490
                      IF (NDCOMM) GO TO 2110                            DBE07500
                         WRITE (OUTDEV,00002)                           DBE07510
     +                         NAME,CLASS(TYPE),IDATUM,CMTBUF           DBE07520
                         GO TO 4000                                     DBE07530
2110                  CONTINUE                                          DBE07540
                         WRITE (OUTDEV,00002) NAME,CLASS(TYPE),IDATUM   DBE07550
                         GO TO 4000                                     DBE07560
2200               CONTINUE                                             DBE07570
C...                  +++ DUMP THE REAL DATUM                           DBE07580
                      IF (NDCOMM) GO TO 2210                            DBE07590
                         WRITE (OUTDEV,00003)                           DBE07600
     +                         NAME,CLASS(TYPE),RDATUM,CMTBUF           DBE07610
                         GO TO 4000                                     DBE07620
2210                  CONTINUE                                          DBE07630
                         WRITE (OUTDEV,00003) NAME,CLASS(TYPE),RDATUM   DBE07640
                         GO TO 4000                                     DBE07650
2300               CONTINUE                                             DBE07660
C...                  +++ INDICATE AN ARRAY POINTER                     DBE07670
                      IF (NDCOMM) GO TO 2310                            DBE07680
                         WRITE (OUTDEV,00004)                           DBE07690
     +                         NAME,CLASS(TYPE),IDATUM,CMTBUF           DBE07700
                         GO TO 4000                                     DBE07710
2310                  CONTINUE                                          DBE07720
                         WRITE (OUTDEV,00004) NAME,CLASS(TYPE),IDATUM   DBE07730
                         GO TO 4000                                     DBE07740
3000            CONTINUE                                                DBE07750
C...               +++ INDICATE THE UNDEFINED ONES                      DBE07760
                   UTYPE=IABS(TYPE)                                     DBE07770
                   IF (NOCOMM) GO TO 3010                               DBE07780
                      WRITE (OUTDEV,00005) NAME,CLASS(UTYPE),CMTBUF     DBE07790
                      GO TO 4000                                        DBE07800
3010                  CONTINUE                                          DBE07810
                         WRITE (OUTDEV,00005) NAME,CLASS(UTYPE)         DBE07820
                         GO TO 4000                                     DBE07830
4000        CONTINUE                                                    DBE07840
               NODE=LINK(NODE)                                          DBE07850
               GO TO 1000                                               DBE07860
6000       CONTINUE                                                     DBE07870
7000    CONTINUE                                                        DBE07880
        WRITE (OUTDEV,00006)                                            DBE07890
C...               +++ DUMP THE ARRAYS                                  DBE07900
```

```
      DO 9000 NODE=1,MAXNOD                                              DBE07910
         IF (NODTYP(NODE).NE.3) GO TO 8000                               DBE07920
            NAME(1)=IDENT(NODE,1)                                        DBE07930
            NAME(2)=IDENT(NODE,2)                                        DBE07940
            NARRAY=DATUM(NODE)                                           DBE07950
            NGET=RARWDS                                                  DBE07960
            CALL DBEDAP(NAME,NARRAY,NGET,RARRAY,RARWDS)                  DBE07970
8000     CONTINUE                                                        DBE07980
9000  CONTINUE                                                           DBE07990
      CALL DXUGRF(UPDATE)                                                DBE08000
      CALL DXWAIT                                                        DBE08010
99999 CONTINUE                                                           DBE08020
         RETURN                                                          DBE08030
C...                                                                     DBE08040
00001 FORMAT (//1H ,A1,2X,4HNAME,4X,4HTYPE,11X,                          DBE08050
     +  5HVALUE,2X,7HCOMMENT/)                                           DBE08060
00002 FORMAT (1H ,3X,3A4,I16,2X,16A4)                                    DBE08070
00003 FORMAT (1H ,3X,3A4,E16.5,2X,16A4)                                  DBE08080
00004 FORMAT (1H ,3X,3A4,4X,6HARRAY(,I5,1H),2X,16A4)                     DBE08090
00005 FORMAT (1H ,3X,3A4,3X,13H**UNDEFINED**,2X,16A4)                    DBE08100
00006 FORMAT (1H //11H THE ARRAYS/)                                      DBE08110
      END                                                                DBE08120
```

```
C                                                                       DBE08130
C ***********************************************************************DBE08140
      SUBRDUTINE DBEDTS(TITLE,WDSTIT,MESSAG,MSGWDS,NCHTIT)              DBE08150
C  +-------------------------------------------------------------------+DBE08160
C                                                                       DBE08170
C        DBEDTS -                                                       DBE08180
C                                                                       DBE08190
C             STDRE A TITLE IN THE DATABASE.                            DBE08200
C                                                                       DBE08210
C             TITLE IS THE ARRAY WHERE THE DATABASE TITLE IS STDRED     DBE08220
C             TEMPORARILY, WDSTIT IS ITS DIMENSIDN.                     DBE08230
C                                                                       DBE08240
C             MESSAG IS AN ARRAY IN WHICH TD STDRE A MESSAGE, AND       DBE08250
C             MSGWDS IS ITS DIMENSION.                                  DBE08260
C                                                                       DBE08270
C             NCHTIT IS THE NUMBER DF CHARACTERS IN THE TITLE.          DBE08280
C                                                                       DBE08290
C  +-------------------------------------------------------------------+DBE08300
C                                                                       DBE08310
C...                 *THE COMMON CONTRDL PARAMETERS,CHARACTERS, AND     DBE08320
C...                 *GRAPHIC DISPLAY CONTROL PARAMETERS                DBE08330
      COMMON /DEXCOM/                                                   DBE08340
     1               DXMODE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,         DBE08350
     2               TERSE ,ECHDMD,DXSC  ,CPWDXS,MDSC  ,CURMDD,         DBE08360
     3               NULMSG,DXNCPW,NAMLEN,                              DBE08370
     4               CLRMOD,MUNPST,ERASE,UPDATE,UPAINT,GCNTRL           DBE08380
      INTEGER                                                           DBE08390
     1               DXSC  ,CPWDXS,MDSC  ,CURMOD,NULMSG,DXNCPW,         DBE08400
     2               NAMLEN,CLRMDD,MUNPST,ERASE,UPDATE,UPAINT           DBE08410
      LDGICAL                                                           DBE08420
     1               DXMDDE,LOADED,MDPEND,DXREQS,DLREQS,KEYBRD,         DBE08430
     2               TERSE ,ECHOMD,GCNTRL                               DBE08440
      DIMENSIDN      DXSC(1),CURMDD(2)  ,NULMSG(1)                      DBE08450
      INTEGER   TITLE,CPWTIT,WDSTIT,MESSAG,MSGWDS                       DBE08460
      LOGICAL   DXCHRI,LOG,DXPRMT,TPUT                                  DBE08470
      DIMENSIDN TITLE(WDSTIT),MESSAG(MSGWDS)                            DBE08480
      DATA      NAMSUB /4HEDTS/                                         DBE08490
      CALL DXMSGF(NAMSUB,1,MESSAG,MSGWDS)                               DBE08500
C...          +++ GET THE TITLE FRDM THE USER                           DBE08510
      LDG=DXCHRI(NCHTIT,TITLE,WDSTIT,DXNCPW,NEED,MESSAG)                DBE08520
      IF (DXREQS) GD TD 99999                                           DBE08530
C...          +++ PUT THE TITLE IN THE DATABASE                         DBE08540
         LDG=TPUT(TITLE)                                                DBE08550
99999 CDNTINUE                                                          DBE08560
         RETURN                                                         DBE08570
      END                                                               DBE08580
```

```
C                                                                       DBE08590
C ***********************************************************************DBE08600
      SUBROUTINE DBEDTG(TITLE,WDSTIT)                                   DBE08610
C  +--------------------------------------------------------------------+DBE08620
C                                                                       DBE08630
C        DBEDTG -                                                       DBE08640
C                                                                       DBE08650
C             GET THE TITLE DF THE DATABASE.                            DBE08660
C                                                                       DBE08670
C             TITLE IS AN ARRAY IN WHICH TO STDRE THE TITLE.            DBE08680
C             WDSTIT IS THE NUMBER DF WORDS IN THE TITLE.               DBE08690
C                                                                       DBE08700
C  +--------------------------------------------------------------------+DBE08710
C                                                                       DBE08720
C...                  *I/D DEVICES AND FILES                            DBE08730
      COMMDN /DEXFIL/                                                   DBE08740
     1                 MSGSDV,USEDEV,INFDDV,DBSDEV,NEWSDV,HELPDV,      DBE08750
     2                 OUTDEV,INPDEV,NFWDS ,                           DBE08760
     3                 MSGSFL,USEFIL,INFDFL,DBSFIL,NEWSFL,HELPFL,      DBE08770
     4                 DUTFIL,INPFIL,LDADFL,NDT1FL,NDT2FL              DBE08780
      INTEGER                                                           DBE08790
     1                 MSGSDV,USEDEV,INFDDV,DBSDEV,NEWSDV,HELPDV,      DBE08800
     2                 DUTDEV,INPDEV,                                  DBE08810
     3                 MSGSFL,USEFIL,INFOFL,DBSFIL,NEWSFL,HELPFL,      DBE08820
     4                 DUTFIL,INPFIL,LDADFL,NDT1FL,NDT2FL,NFWDS        DBE08830
C...  FILE NAME DIMENSIDNING ADJUSTED FDR CDC AND CU DNLY.....          DBE08840
      DIMENSIDN        MSGSFL(11)  ,USEFIL(11)  ,INFDFL(11)  ,         DBE08850
     1                 DBSFIL(11)  ,NEWSFL(11)  ,HELPFL(11)  ,         DBE08860
     3                 DUTFIL(11)  ,INPFIL(11)  ,                      DBE08870
     2                 LDADFL(11)  ,NDT1FL(11)  ,NDT2FL(11)            DBE08880
C...                  *THE MESSAGE HANDLING STDRE AND PARAMETERS        DBE08890
      COMMDN /DEXMSG/                                                   DBE08900
     1                 DXCHAR,MDCHAR,TRIMCH,MAXMSG,MAXWDS,MXPODL,      DBE08910
     2                 SUBCNT,SUBCOD,MSGPTR,PVERB ,LVERB ,PTERSE,      DBE08920
     3                 LTERSE,LSTRNG,VPOSIT,TPDSIT,MSPODL              DBE08930
      INTEGER                                                           DBE08940
     1                 DXCHAR,MDCHAR,TRIMCH,MAXMSG,MAXWDS,MXPDDL,      DBE08950
     2                 SUBCNT,SUBCDD,MSGPTR,PVERB ,LVERB ,PTERSE,      DBE08960
     3                 LTERSE,LSTRNG,VPDSIT,TPDSIT,MSPDOL              DBE08970
      DIMENSION        SUBCDD( 40),MSGPTR(102),PVERB (102),            DBE08980
     5                 LVERB (102),PTERSE(102),LTERSE(102),            DBE08990
     6                 LSTRNG(102),VPDSIT(102),TPDSIT(102),            DBE09000
     7                 MSPDDL(950),TRIMCH(1)                           DBE09010
      INTEGER TITLE,WDSTIT                                              DBE09020
      LOGICAL TGET                                                      DBE09030
      DIMENSIDN TITLE(WDSTIT)                                           DBE09040
      DATA NAMSUB /4HEDTG/                                              DBE09050
```

```
C...           +++ GET THE TITLE FRDM THE DATABASE USING L.F. TGET        DBE09060
      IF (.NDT.(TGET(TITLE))) GD TD 11111                                 DBE09070
         CALL DXGMSG                                                      DBE09080
C...           +++ DISPLAY THE TITLE AT THE TERMINAL                      DBE09090
         WRITE (OUTDEV,00001) DXCHAR,TITLE                                DBE09100
         GO TD 99999                                                      DBE09110
11111 CONTINUE                                                            DBE09120
C...           +++ NO TITLE STDRED                                        DBE09130
         CALL DXMSGZ(NAMSUB,1)                                            DBE09140
         GO TD 99999                                                      DBE09150
99999 CDNTINUE                                                            DBE09160
         RETURN                                                           DBE09170
00001 FDRMAT (1H ,A1,9HTITLE: **,16A4,2H**)                               DBE09180
      END                                                                 DBE09190
```

```
C                                                                       DBE09200
C ***********************************************************************DBE09210
      SUBROUTINE DBINIT                                                 DBE09220
C  +--------------------------------------------------------------------+DBE09230
C                                                                       DBE09240
C        DBINIT -                                                       DBE09250
C                                                                       DBE09260
C             INITIALIZE THE DATABASE IN MEMDRY.  CALLED FDR EACH       DBE09270
C             NEW DATABASE.                                             DBE09280
C                                                                       DBE09290
C  +--------------------------------------------------------------------+DBE09300
C                                                                       DBE09310
      COMMON /DBCDM/                                                    DBE09320
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBE09330
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBE09340
     2               INTTYP,RELTYP,ARRTYP,                              DBE09350
     3               DBCLSD,DBACTV,FILEDB                               DBE09360
      INTEGER                                                           DBE09370
     1               TITLE ,BUCKET,IDENT ,NODTYP,DATUM ,LINK  ,LINKF ,  DBE09380
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBE09390
     2               INTTYP,RELTYP,ARRTYP                               DBE09400
      LOGICAL                                                           DBE09410
     3               DBCLSD,DBACTV,FILEDB                               DBE09420
      DIMENSION      TITLE(16),BUCKET(32),IDENT(200,2),NODTYP(200),     DBE09430
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)        DBE09440
C...                 *BUFFER AND PARSER ARRAYS AND PARAMETERS           DBE09450
      CDMMDN /DEXPRS/                                                   DBE09460
     1                INBUFR,BUFLEN,CPWIBU,FLAGST,FLAGLN,FLAGCH,       DBE09470
     1                LENBUF,CURPDS,NXTPDS,BEGPDS,ENDPDS,INPEND,       DBE09480
     1                DELCHR,DELNUM,CPWDEL,STRDEL,CPWAQU,DELETE,       DBE09490
     1                BLANK,                                           DBE09500
     1                INTG,REAL,LINTG,LREAL,MENU,MOXY,XY               DBE09510
      INTEGER                                                           DBE09520
     1                INBUFR,BUFLEN,CPWIBU,FLAGST,FLAGLN,FLAGCH,       DBE09530
     1                LENBUF,CURPOS,NXTPDS,BEGPDS,ENDPDS,              DBE09540
     1                DELCHR,DELNUM,CPWDEL,STRDEL,CPWAQU,DELETE,       DBE09550
     1                BLANK,                                           DBE09560
     1                INTG,REAL,LINTG,LREAL,MENU,MDXY,XY               DBE09570
      LDGICAL         INPEND                                            DBE09580
      DIMENSIDN       INBUFR(133),FLAGST(133),FLAGCH(1),DELCHR(1),      DBE09590
     1                STRDEL(1),DELETE(1)                              DBE09600
      COMMON /DBDAID/                                                   DBE09610
     1               MAXECS,NEXECS,PRECS,ECSPTR,ECSLEN                  DBE09620
      INTEGER                                                           DBE09630
     1               PRECS,ECSPTR,ECSLEN                                DBE09640
      DIMENSIDN                                                         DBE09650
     1               ECSPTR(2),ECSLEN(2)                                DBE09660
```

```
      LOGICAL        GETMN,GETFLG                                                  OBE09670
      DATA   MAXL /2000/                                                           OBE09680
      DATA   NAMSUB /4HINDB/                                                       OBE09690
C...          +++ INITIALIZE THE MAX SIZES                                        OBE09700
      MAXNOD=200                                                                   OBE09710
      MAXARR=200                                                                   OBE09720
      NCHTIT=64                                                                    OBE09730
      NCHCMT=64                                                                    OBE09740
C...          +++ INITIALIZE THE TYPE CODES                                       OBE09750
      INTTYP=1                                                                     OBE09760
      RELTYP=2                                                                     OBE09770
      ARRTYP=3                                                                     OBE09780
C...          +++ INITIALIZE THE DATABASE VALUES                                  OBE09790
         NAVAIL = 1                                                                OBE09800
         OELCNT = 0                                                                OBE09810
         DO 1000 I = 1,MAXNOD                                                      OBE09820
            IDENT(I,1)=BLANK                                                       OBE09830
            IDENT(I,2)=BLANK                                                       OBE09840
            NODTYP(I)=0                                                            OBE09850
            LINK(I) = I+1                                                          OBE09860
            LINKF(I) = 0                                                           OBE09870
            CMTPTR(I) = 0                                                          OBE09880
1000     CONTINUE                                                                  OBE09890
         LINK(MAXNOD) = 0                                                          OBE09900
         DO 2000 I=1,32                                                            OBE09910
            BUCKET(I) = 0                                                          OBE09920
2000     CONTINUE                                                                  OBE09930
         DO 3000 I=1,16                                                            OBE09940
            TITLE(I)=0                                                             OBE09950
3000     CONTINUE                                                                  OBE09960
      NEXECS=1                                                                     OBE09970
      PRECS=0                                                                      OBE09980
      MAXECS=MAXL                                                                  OBE09990
C...          +++ GET MAIN STORAGE TO SIMULATE COC ECS                            OBE10000
      GETFLG=GETMN(MAXECS,ECSPTR(1))                                               OBE10010
      IF (.NOT. GETFLG) GO TO 88888                                                OBE10020
      ECSLEN(1)=MAXECS                                                             OBE10030
      GO TO 99999                                                                  OBE10040
88888 CONTINUE                                                                     OBE10050
C...          +++ NOT ENOUGH STORAGE LEFT IN THE USERS VIRTUAL MACHINE.           OBE10060
      CALL OXMSGZ (NAMSUB,1)                                                       OBE10070
99999 CONTINUE                                                                     OBE10080
      RETURN                                                                       OBE10090
      END                                                                          OBE10100
```

```
C                                                                              DBE10110
C ******************************************************************************DBE10120
      INTEGER FUNCTION DBGETN(RCODE)                                           DBE10130
C  +--------------------------------------------------------------------+      DBE10140
C                                                                              DBE10150
C        DBGETN -                                                              DBE10160
C                                                                              DBE10170
C             GET THE NUMBER OF THE NEXT AVAILABLE NODE.                       DBE10180
C                                                                              DBE10190
C             RCODE = 0 THERE ARE NODES STILL AVAILABLE                        DBE10200
C             RCODE = 1 NO MORE NODES AVAILABLE                                DBE10210
C                                                                              DBE10220
C  +--------------------------------------------------------------------+      DBE10230
C                                                                              DBE10240
      COMMON /DBCOM/                                                           DBE10250
     1               TITLE ,BUCKET,IDENT ,NODTYP,DATUM ,LINK  ,LINKF ,        DBE10260
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,        DBE10270
     2               INTTYP,RELTYP,ARRTYP,                                    DBE10280
     3               DBCLSD,DBACTV,FILEDB                                     DBE10290
      INTEGER                                                                  DBE10300
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,        DBE10310
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,        DBE10320
     2               INTTYP,RELTYP,ARRTYP                                     DBE10330
      LOGICAL                                                                  DBE10340
     3               DBCLSD,DBACTV,FILEDB                                     DBE10350
      DIMENSION      TITLE(16),BUCKET(32),IDENT(200,2),NODTYP(200),           DBE10360
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)              DBE10370
      INTEGER RCODE                                                            DBE10380
      DBGETN=NAVAIL                                                            DBE10390
C...          +++ NAVAIL = 0 INDICATES NO MORE NODES RCODE = 1                 DBE10400
      IF (DBGETN.EQ.0) RCODE=1                                                 DBE10410
C...          +++ NAVAIL NOT EQUAL 0 INDICATES STILL NODES RCODE = 0           DBE10420
      IF (DBGETN.NE.0) RCODE=0                                                 DBE10430
         IF (RCODE.EQ.0) NAVAIL=LINK(DBGETN)                                   DBE10440
         GO TO 99999                                                           DBE10450
99999 CONTINUE                                                                 DBE10460
         RETURN                                                                DBE10470
      END                                                                      DBE10480
```

```
C                                                                       DBE10490
C ***********************************************************************DBE10500
      LDGICAL FUNCTIDN DBNDIN(DNAME,TYPE,IDATUM,RCDDE)                  DBE10510
C  +--------------------------------------------------------------------+DBE10520
C                                                                       DBE10530
C        DBNDIN -                                                       DBE10540
C                                                                       DBE10550
C             ENTER THE DATUM AND TYPE FDR A NDDE INTD THE DATABASE.    DBE10560
C                                                                       DBE10570
C             DNAME IS THE NAME DF THE DATUM.                           DBE10580
C                                                                       DBE10590
C             RCDDE IS A RETURN CDDE                                    DBE10600
C                                                                       DBE10610
C                  RCDDE = O DATUM SUCCESSFULLY ENTERED                 DBE10620
C                  RCDDE = 1 ND DPEN DATABASE                           DBE10630
C                  RCDDE = 2 NAMED DATUM DDES NDT EXIST                 DBE10640
C                  RCDDE = 3 TYPE DISAGREEMENT                          DBE10650
C                                                                       DBE10660
C  +--------------------------------------------------------------------+DBE10670
C                                                                       DBE10680
      CDMMDN /DBCDM/                                                    DBE10690
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBE10700
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBE10710
     2               INTTYP,RELTYP,ARRTYP,                              DBE10720
     3               DBCLSD,DBACTV,FILEDB                               DBE10730
      INTEGER                                                           DBE10740
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBE10750
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBE10760
     2               INTTYP,RELTYP,ARRTYP                               DBE10770
      LDGICAL                                                           DBE10780
     3               DBCLSD,DBACTV,FILEDB                               DBE10790
      DIMENSIDN      TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),     DBE10800
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)        DBE10810
      INTEGER   DNAME,TYPE,IDATUM,RCDDE,DBNSKR                          DBE10820
      INTEGER   PRNDDE,STYPE                                            DBE10830
      DIMENSIDN DNAME(2)                                                DBE10840
      IF (DBCLSD) GD TD 11111                                           DBE10850
C...          +++ SEARCH FDR THE NDDE                                   DBE10860
         NDDE=DBNSKR(DNAME,STYPE,PRNDDE,RCDDE)                         DBE10870
         DBNDIN=(NDDE.NE.O)                                             DBE10880
         IF (.NDT.DBNDIN) GD TD 22222                                   DBE10890
            IF (IABS(STYPE).NE.TYPE) GD TD 33333                        DBE10900
C...          +++ IF FDUND ENTER THE DATUM AND TYPE                     DBE10910
              NDDTYP(NDDE)=TYPE                                         DBE10920
              DATUM(NDDE)=IDATUM                                        DBE10930
              RCDDE=O                                                   DBE10940
              GD TD 99999                                               DBE10950
```

```
11111 CONTINUE                                                          DBE10960
C...          +++ NO OPEN DATA BASE                                     DBE10970
         RCODE=1                                                        DBE10980
         GO TO 99999                                                    DBE10990
22222 CONTINUE                                                          DBE11000
C...          +++ NAMED DATUM DOES NOT EXIST                            DBE11010
         RCODE=2                                                        DBE11020
         GO TO 99999                                                    DBE11030
33333 CONTINUE                                                          DBE11040
C...          +++ TYPE DISAGREEMENT                                     DBE11050
         RCODE=3                                                        DBE11060
         GO TO 99999                                                    DBE11070
99999 CONTINUE                                                          DBE11080
         RETURN                                                         DBE11090
      END                                                               DBE11100
```

```
C                                                                       DBE11110
C ***********************************************************************DBE11120
      LDGICAL FUNCTION DBNDRD(DNAME,TYPE,IDATUM,RCDDE)                  DBE11130
C  +--------------------------------------------------------------------+DBE11140
C                                                                       DBE11150
C        DBNDRD -                                                       DBE11160
C                                                                       DBE11170
C             READ THE DATUM FDR A NDDE IN THE DATABASE.                DBE11180
C                                                                       DBE11190
C             DNAME IS THE DATUM NAME, AND TYPE IS 1, 2, DR 3 FOR       DBE11200
C             INTEGER, REAL, DR REAL-ARRAY.                             DBE11210
C                                                                       DBE11220
C             THE DATUM IS PUT IN IDATUM.                               DBE11230
C                                                                       DBE11240
C             RCDDE IS A RETURN CODE.                                   DBE11250
C                                                                       DBE11260
C                  RCDDE = 0 THE READ WAS SUCCESSFUL                    DBE11270
C                  RCDDE = 1 ND DPEN DATABASE                           DBE11280
C                  RCDDE = 2 DNAME NOT IN DATABASE                      DBE11290
C                  RCDDE = 3 WRONG TYPE                                 DBE11300
C                  RCODE = 4 ND DATUM STDRED YET                        DBE11310
C                                                                       DBE11320
C  +--------------------------------------------------------------------+DBE11330
C                                                                       DBE11340
      COMMON /DBCOM/                                                    DBE11350
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBE11360
     1               BKTLNK,CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,  DBE11370
     1               DELCNT,                                            DBE11380
     2               INTTYP,RELTYP,ARRTYP,                              DBE11390
     3               DBCLSD,DBACTV,FILEDB                               DBE11400
      INTEGER                                                           DBE11410
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBE11420
     1               BKTLNK,CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,  DBE11430
     1               DELCNT,                                            DBE11440
     2               INTTYP,RELTYP,ARRTYP                               DBE11450
      LDGICAL                                                           DBE11460
     3               DBCLSD,DBACTV,FILEDB                               DBE11470
      DIMENSIDN      TITLE(16),BUCKET(32),IDENT(200,2),NODTYP(200),     DBE11480
     1               DATUM(200),LINK(200),LINKF(200),BKTLNK(200),       DBE11490
     1               CMTPTR(200)                                        DBE11500
      INTEGER   DNAME,TYPE,IDATUM,RCDDE,DBNSKR                          DBE11510
      INTEGER   PRNODE,STYPE                                            DBE11520
      DIMENSIDN DNAME(2)                                                DBE11530
      IF (DBCLSD) GD TD 11111                                           DBE11540
         IDATUM=0                                                       DBE11550
C...           +++ SEARCH THE DATABASE FDR THE NODE                     DBE11560
         NDDE=DBNSKR(DNAME,STYPE,PRNODE,RCDDE)                          DBE11570
```

```
         OBNORO=(NOOE.NE.O)                                          OBE11580
         IF (.NOT.DBNDRO) GO TO 22222                                OBE11590
            IF (IABS(STYPE).NE.TYPE) GO TO 33333                     OBE11600
C...          +++ READ THE DATUM INTO IDATUM                         OBE11610
               IOATUM=OATUM(NOOE)                                    OBE11620
               IF (STYPE.LT.O) GO TO 44444                           OBE11630
                  RCOOE=O                                            OBE11640
                  GO TO 99999                                        DBE11650
11111 CONTINUE                                                       OBE11660
C...          +++ NO OPEN DATA BASE                                  DBE11670
      OBNORO=.FALSE.                                                 DBE11680
         RCOOE=1                                                     DBE11690
         GO TO 99999                                                 OBE11700
22222 CONTINUE                                                       DBE11710
C...          +++ NOT IN OATA BASE                                   OBE11720
         RCOOE=2                                                     OBE11730
         GO TO 99999                                                 DBE11740
33333 CONTINUE                                                       OBE11750
C...          +++ WRONG TYPE                                         OBE11760
         RCODE=3                                                     OBE11770
         GO TO 99999                                                 DBE11780
44444 CONTINUE                                                       DBE11790
C...          +++ NO OATUM STOREO                                    OBE11800
         RCOOE=4                                                     OBE11810
         GO TO 99999                                                 OBE11820
99999 CONTINUE                                                       OBE11830
         RETURN                                                      OBE11840
      END                                                            OBE11850
```

```
C                                                                       DBE11860
C ***********************************************************************DBE11870
      INTEGER FUNCTION DBNSKR(DNAME,TYPE,PRNDDE,RCDDE)                  DBE11880
C  +---------------------------------------------------------------------+DBE11890
C                                                                        DBE11900
C        DBNSKR -                                                        DBE11910
C                                                                        DBE11920
C             GIVEN THE VARIABLE NAME AS IN DNAME, IT SEARCHES THE       DBE11930
C             DATA-BASE TABLE USING THE HASHING FUNCTION FDR ITS INDEX.  DBE11940
C                                                                        DBE11950
C             DBNSKR TAKES ON THE VALUE DF THE NODE.                     DBE11960
C                                                                        DBE11970
C             DNAME IS THE NAME DF THE DATUM, AND TYPE IS ITS TYPE       DBE11980
C             PRNODE IS THE PREVEDUS NDDE.                               DBE11990
C                                                                        DBE12000
C             RCODE IS A RETURN CDDE;                                    DBE12010
C                                                                        DBE12020
C                  NODE NOT EQUAL TD ZERD                                DBE12030
C                       RCDDE = O DATUM FDUND IN A SUBSEQUENT NDDE IN    DBE12040
C                                 THE CHAIN.                             DBE12050
C                       RCDDE = 1 DATUM FDUND IN THE FIRST NDDE IN A     DBE12060
C                                 CHAIN.                                 DBE12070
C                  IF NDDE EQUALS O                                      DBE12080
C                       RCDDE = 1 DATUM NOT FDUND.                       DBE12090
C  +---------------------------------------------------------------------+DBE12100
C                                                                        DBE12110
      COMMON /DBCDM/                                                     DBE12120
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,   DBE12130
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,   DBE12140
     2               INTTYP,RELTYP,ARRTYP,                               DBE12150
     3               DBCLSD,DBACTV,FILEDB                                DBE12160
      INTEGER                                                            DBE12170
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,   DBE12180
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,   DBE12190
     2               INTTYP,RELTYP,ARRTYP                                DBE12200
      LDGICAL                                                            DBE12210
     3               DBCLSD,DBACTV,FILEDB                                DBE12220
      DIMENSION      TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),      DBE12230
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)         DBE12240
      INTEGER DNAME,TYPE,PRNODE,RCDDE                                    DBE12250
      INTEGER HASH,DBHASH                                                DBE12260
      DIMENSION DNAME(2)                                                 DBE12270
      HASH=DBHASH(DNAME)                                                 DBE12280
C...     +++ SEARCH FDR THE NDDE                                         DBE12290
      PRNODE=HASH                                                        DBE12300
      NDDE=BUCKET(HASH)                                                  DBE12310
      RCDDE=1                                                            DBE12320
```

```
1000  CONTINUE                                                          DBE12330
C...            +++ IF NODE = 0 THE DATUM WASN'T FOUND                  DBE12340
          IF (NODE.EQ.0) GO TO 3000                                     DBE12350
C...            +++ CHECK FOR A MATCH BETWEEN NAME AND IDENT            DBE12360
             IF ((IDENT(NODE,1).EQ.DNAME(1))                            DBE12370
     +       .AND.(IDENT(NODE,2).EQ.DNAME(2))) GO TO 2000               DBE12380
                PRNODE=NODE                                             DBE12390
                RCODE=0                                                 DBE12400
                NODE=LINK(PRNODE)                                       DBE12410
                GO TO 1000                                              DBE12420
2000        CONTINUE                                                    DBE12430
C...            +++ DATUM WAS FOUND                                     DBE12440
             TYPE = NODTYP(NODE)                                        DBE12450
             GO TO 11111                                                DBE12460
3000        CONTINUE                                                    DBE12470
C...            +++ DATUM WAS NOT FOUND                                 DBE12480
             PRNODE=HASH                                                DBE12490
             RCODE=1                                                    DBE12500
             GO TO 11111                                                DBE12510
11111 CONTINUE                                                          DBE12520
          DBNSKR=NODE                                                   DBE12530
          GO TO 99999                                                   DBE12540
99999 CONTINUE                                                          DBE12550
          RETURN                                                        DBE12560
        END                                                             DBE12570
```

```
C                                                                       DBE12580
C ***********************************************************************OBE12590
      SUBROUTINE OBOPUT(NAMEOF,NEWBAS,FILNAM,RCOOE)                     OBE12600
C  +--------------------------------------------------------------------+OBE12610
C                                                                       OBE12620
C        OBOPUT -                                                       OBE12630
C                                                                       OBE12640
C             OATABASE OPEN UTILITY - ROUTINE FOR OPENING A OATABASE    OBE12650
C             IN MEMORY.                                                OBE12660
C                                                                       OBE12670
C             RCOOE IS A RETURN COOE;                                   OBE12680
C                                                                       OBE12690
C                  RCOOE = O SUCCESSFUL                                 OBE12700
C                  RCOOE = 1 AN OPEN OATABASE ALREAOY EXISTS            OBE12710
C                  RCOOE = 2 FAILEO TO LOAO THE OATABASE                OBE12720
C                                                                       OBE12730
C                                                                       OBE12740
C  +--------------------------------------------------------------------+OBE12750
C                                                                       OBE12760
C...                 *THE COMMON CONTROL PARAMETERS,CHARACTERS, ANO     OBE12770
C...                 *GRAPHIC OISPLAY CONTROL PARAMETERS                OBE12780
      COMMON /OEXCOM/                                                   OBE12790
     1               OXMOOE,LOAOEO,MOPENO,OXREQS,OLREQS,KEYBRD,         OBE12800
     2               TERSE ,ECHOMO,OXSC  ,CPWOXS,MOSC  ,CURMOD,         OBE12810
     3               NULMSG,OXNCPW,NAMLEN,                              OBE12820
     4               CLRMOO,MUNPST,ERASE,UPOATE,UPAINT,GCNTRL           OBE12830
      INTEGER                                                           OBE12840
     1               OXSC  ,CPWOXS,MOSC  ,CURMOO,NULMSG,OXNCPW,         OBE12850
     2               NAMLEN,CLRMOO,MUNPST,ERASE,UPOATE,UPAINT           OBE12860
      LOGICAL                                                           OBE12870
     1               OXMOOE,LOAOEO,MOPENO,OXREQS,OLREQS,KEYBRO,         OBE12880
     2               TERSE ,ECHOMO,GCNTRL                               OBE12890
      OIMENSION      OXSC(1),CURMOO(2)  ,NULMSG(1)                      OBE12900
C...                 *I/O OEVICES ANO FILES                             OBE12910
      COMMON /OEXFIL/                                                   OBE12920
     1               MSGSOV,USEOEV,INFODV,OBSDEV,NEWSOV,HELPOV,         OBE12930
     2               OUTOEV,INPOEV,NFWOS ,                              OBE12940
     3               MSGSFL,USEFIL,INFOFL,OBSFIL,NEWSFL,HELPFL,         OBE12950
     4               OUTFIL,INPFIL,LOAOFL,NOT1FL,NOT2FL                 OBE12960
      INTEGER                                                           OBE12970
     1               MSGSOV,USEOEV,INFOOV,OBSOEV,NEWSOV,HELPOV,         OBE12980
     2               OUTOEV,INPOEV,                                     OBE12990
     3               MSGSFL,USEFIL,INFOFL,OBSFIL,NEWSFL,HELPFL,         OBE13000
     4               OUTFIL,INPFIL,LOAOFL,NOT1FL,NOT2FL,NFWOS           OBE13010
C...  FILE NAME OIMENSIONING AOJUSTEO FOR COC ANO CU ONLY.....          OBE13020
      OIMENSION      MSGSFL(11)  ,USEFIL(11)  ,INFOFL(11)  ,           OBE13030
     1               OBSFIL(11)  ,NEWSFL(11)  ,HELPFL(11)  ,           OBE13040
```

```
     3                DUTFIL(11)  ,INPFIL(11)  ,                          DBE13050
     2                LDAOFL(11)  ,NDT1FL(11)  ,NDT2FL(11)               DBE13060
      CDMMON /OBCOM/                                                     DBE13070
     1               TITLE ,BUCKET,IDENT ,NODTYP,DATUM ,LINK  ,LINKF ,   DBE13080
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,   DBE13090
     2               INTTYP,RELTYP,ARRTYP,                               DBE13100
     3               DBCLSD,DBACTV,FILEOB                                DBE13110
      INTEGER                                                            DBE13120
     1               TITLE ,BUCKET,IDENT ,NODTYP,DATUM ,LINK  ,LINKF ,   DBE13130
     1               CMTPTR,NAVAIL,MAXNDO,MAXARR,NCHTIT,NCHCMT,DELCNT,   DBE13140
     2               INTTYP,RELTYP,ARRTYP                                DBE13150
      LOGICAL                                                            DBE13160
     3               OBCLSD,DBACTV,FILEOB                                DBE13170
      DIMENSION       TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),     DBE13180
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)         DBE13190
      INTEGER   DXMENU,FILNAM,RCDDE                                      DBE13200
      INTEGER CCDBFN(20),XCDDE,TEMP(20)                                  DBE13210
      LDGICAL   OBLOAD,DXCKDV,NAMEDF,NEWBAS,MAKNEW,DXFNEQ,DBSAME,DXPRMT DBE13220
      DIMENSION MSG(16),MENNAM(2),ITEMS(4)                               DBE13230
      DATA      MENNAM /4HDXYE,4HS-ND/                                   DBE13240
      DATA      NITEMS /2/                                               DBE13250
      DATA      ITEMS /4HYES ,4H    ,4HND  ,4H    /                      DBE13260
      DATA      MSGWDS/16/,NAMSUB/4HDPUT/                                DBE13270
C                                                                        DBE13280
      IF (NAMEDF) GD TD 600                                              DBE13290
      IF (.NDT.DBCLSD) GD TD 700                                         DBE13300
      GD TD 900                                                          DBE13310
  600 CDNTINUE                                                           DBE13320
      IF (DBCLSD) GD TD 3000                                             DBE13330
      GD TD 11111                                                        DBE13340
  700 CDNTINUE                                                           DBE13350
C...          +++  ANNDUNCE THE NAME DF THE DPEN DATABASE               DBE13360
      WRITE(DUTDEV,00001)                                                DBE13370
00001 FORMAT(' THE DPEN DATABASE IS')                                    DBE13380
      CALL DXFNPR(DBSFIL)                                                DBE13390
      CALL DXFNMV(DBSFIL,TEMP)                                           DBE13400
C...          +++ ASK IF DBSFIL SHALL BE USED                           DBE13410
      CALL DXMSGF(NAMSUB,2,MSG,MSGWDS)                                   DBE13420
      ITEM=DXMENU(MENNAM,NITEMS,ITEMS,MSG)                               DBE13430
      IF (DXREQS) GD TD 99999                                            DBE13440
      IF (ITEM.EQ.1) GD TD 750                                           DBE13450
      IF (ITEM.EQ.2) GD TD 800                                           DBE13460
  750 CDNTINUE                                                           DBE13470
C...          +++ ASK IF WE WANT TD SAVE BEFDRE WE USE                  DBE13480
      CALL STRPAK(MSG,MSGWDS,4H<   ,51HDD YDU WANT TD SAVE DPEN DATABASEDBE13490
     1 BEFDRE USING IT?<)                                                DBE13500
      ITEM=DXMENU(MENNAM,NITEMS,ITEMS,MSG)                               DBE13510
```

```
      IF (DXREQS) GO TO 99999                                          DBE13520
      IF (ITEM.EQ.2) GO TO 99999                                       DBE13530
      CALL DBCLUT(.FALSE.,CCDBFN,XCODE)                                DBE13540
      IF (DXREQS) GO TO 99999                                          DBE13550
       IF (XCODE.GT.O) GO TO 11111                                     DBE13560
       GO TO 7000                                                      DBE13570
  800  CONTINUE                                                         DBE13580
      CALL DBCLUT(.FALSE.,CCDBFN,XCDDE)                                DBE13590
      IF (DXREQS) GO TO 99999                                          DBE13600
      IF (XCODE .GT.O) GO TO 11111                                     DBE13610
      IF (NAMEDF) GO TO 3000                                           DBE13620
      GO TO 2000                                                       DBE13630
  900 CONTINUE                                                         DBE13640
         IF (NAMEDF) GO TO 3000                                        DBE13650
1000        CONTINUE                                                   DBE13660
               IF (.NOT.(DBACTV)) GO TO 2000                         DBE13670
C...           +++ ANNOUNCE THE NAME OF THE ACTIVE DATABASE          DBE13680
                   CALL DXMSGZ(NAMSUB,1)                             DBE13690
                   CALL DXFNPR(DBSFIL)                               DBE13700
                   CALL DXFNMV(DBSFIL,TEMP)                          DBE13710
C...               +++ ASK IF DBSFIL SHALL BE USED                   DBE13720
                   CALL DXMSGF(NAMSUB,2,MSG,MSGWDS)                  DBE13730
                   ITEM=DXMENU(MENNAM,NITEMS,ITEMS,MSG)              DBE13740
                   IF (DXREQS) GO TO 99999                           DBE13750
                      IF (ITEM.EQ.1) GO TO 7000                      DBE13760
2000            CONTINUE                                             DBE13770
C...            +++ ASK FOR A NAME                                   DBE13780
                   IF (DXPRMT(O)) CALL DXMSGZ(NAMSUB,3)              DBE13790
                   CALL DXGFNM(DBSFIL)                               DBE13800
      IF (.NOT.DXREQS) GO TO 2100                                      DBE13810
      IF (DBACTV) CALL DXFNMV(TEMP,DBSFIL)                            DBE13820
      GO TO 99999                                                      DBE13830
 2100 CONTINUE                                                         DBE13840
                   IF (NEWBAS) GO TO 5000                            DBE13850
C...                +++ IS A SAVED BASE TO BE USED                   DBE13860
                       CALL DXMSGF(NAMSUB,4,MSG,MSGWDS)              DBE13870
                       ITEM=DXMENU(MENNAM,NITEMS,ITEMS,MSG)          DBE13880
                       IF (.NOT.DXREQS) GO TO 2200                   DBE13890
      IF (DBACTV) CALL DXFNMV(TEMP,DBSFIL)                            DBE13900
      GO TO 99999                                                      DBE13910
 2200 CONTINUE                                                         DBE13920
                         GO TO (6000,5000),ITEM                      DBE13930
3000     CONTINUE                                                     DBE13940
            IF (.NDT.(NEWBAS)) GO TO 4000                            DBE13950
               CALL DXFNMV(FILNAM,DBSFIL)                            DBE13960
               GO TO 5000                                            DBE13970
4000        CONTINUE                                                 DBE13980
```

```
               IF ((DBACTV).AND.(DXFNEQ(FILNAM,DBSFIL))) GD TD 7000      DBE13990
                  CALL DXFNMV(FILNAM,DBSFIL)                             DBE14000
                  GO TO 6000                                             DBE14010
5000           CONTINUE                                                  DBE14020
                  MAKNEW=DXCKDV(DBSDEV,DBSFIL)                           DBE14030
                  IF (.NDT.MAKNEW) GD TO 22222                           DBE14040
                  CALL DBINIT                                            DBE14050
                  FILEDB=.FALSE.                                         DBE14060
                  GD TD 7000                                             DBE14070
6000              CONTINUE                                               DBE14080
                     FILEDB=DBLDAD(RCDDE)                                DBE14090
                     IF (FILEDB) CALL DXMSGZ(NAMSUB,7)                   DBE14100
                     IF (FILEDB) GO TD 7000                              DBE14110
                        IF (NAMEDF) GD TD 22222                          DBE14120
C...                   +++ WE ARE ABLE TO TRY AGAIN, ANNOUNCE            DBE14130
C...                   +++ FAILURE                                       DBE14140
                           CALL DXMSGZ(NAMSUB,5)                         DBE14150
                           CALL DXFNPR(DBSFIL)                           DBE14160
C...                   +++ ASK IF TD TRY AGAIN                           DBE14170
                           CALL DXMSGF(NAMSUB,6,MSG,MSGWDS)              DBE14180
                           ITEM=DXMENU(MENNAM,NITEMS,ITEMS,MSG)          DBE14190
                           IF (.NDT.DXREQS) GD TD 6300                   DBE14200
      IF (DBACTV) CALL DXFNMV(TEMP,DBSFIL)                               DBE14210
      GO TD 99999                                                        DBE14220
 6300 CONTINUE                                                           DBE14230
                              GD TD (2000,22222),ITEM                    DBE14240
C...                                                                     DBE14250
7000  CDNTINUE                                                           DBE14260
C...           +++ SUCCESS                                               DBE14270
         DBCLSD=.FALSE.                                                  DBE14280
         DBACTV=.TRUE.                                                   DBE14290
         RCODE =O                                                        DBE14300
         GO TD 99999                                                     DBE14310
11111 CDNTINUE                                                           DBE14320
C...           +++ AN DPEN DATABASE EXISTS, CAN NDT OPEN ANDTHER         DBE14330
         RCODE=1                                                         DBE14340
         IF (NAMEDF) GO TD 99999                                         DBE14350
            CALL DXFNPR(DBSFIL)                                          DBE14360
            CALL DXMSGZ(NAMSUB,8)                                        DBE14370
            GD TO 99999                                                  DBE14380
22222 CONTINUE                                                           DBE14390
C...           +++ LOADING FAILURE                                       DBE14400
         DBACTV=.FALSE.                                                  DBE14410
         RCODE =2                                                        DBE14420
         GD TD 99999                                                     DBE14430
99999 CONTINUE                                                           DBE14440
```

```
        RETURN                                                          DBE14450
      END                                                               DBE14460
C                                                                       DBE14470
C ***********************************************************************DBE14480
      SUBRDUTINE DBCLUT(NAMEDF,FILNAM,RCDDE)                            DBE14490
C  +-------------------------------------------------------------------+DBE14500
C                                                                       DBE14510
C         DBCLUT -                                                      DBE14520
C                                                                       DBE14530
C              DATABASE CLDSE UTILITY - RDUTINE FDR CLDSING AN OPEN     DBE14540
C              DATABASE.                                                DBE14550
C                                                                       DBE14560
C              RCDDE IS A RETURN CODE;                                  DBE14570
C                                                                       DBE14580
C                   RCDDE = O DATABASE WAS SUCCESSFULLY CLDSED.         DBE14590
C                   RCDDE = 1 ND ACTIVE DATABASE.                       DBE14600
C                   RCDDE = 2 DATABASE CDULD NDT BE SAVED WITH THE      DBE14610
C                             GIVEN NAME.                               DBE14620
C                                                                       DBE14630
C  +-------------------------------------------------------------------+DBE14640
C                                                                       DBE14650
C...                 *THE COMMON CONTROL PARAMETERS,CHARACTERS, AND     DBE14660
C...                 *GRAPHIC DISPLAY CONTRDL PARAMETERS                DBE14670
      COMMON /DEXCDM/                                                   DBE14680
     1                DXMDDE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,        DBE14690
     2                TERSE ,ECHDMD,DXSC  ,CPWDXS,MDSC  ,CURMDD,        DBE14700
     3                NULMSG,DXNCPW,NAMLEN,                             DBE14710
     4                CLRMDD,MUNPST,ERASE,UPDATE,UPAINT,GCNTRL          DBE14720
      INTEGER                                                           DBE14730
     1                DXSC  ,CPWDXS,MDSC  ,CURMOD,NULMSG,DXNCPW,        DBE14740
     2                NAMLEN,CLRMDD,MUNPST,ERASE,UPDATE,UPAINT          DBE14750
      LOGICAL                                                           DBE14760
     1                DXMDDE,LOADED,MDPEND,DXREQS,DLREQS,KEYBRD,        DBE14770
     2                TERSE ,ECHOMD,GCNTRL                              DBE14780
      DIMENSIDN       DXSC(1),CURMOD(2)  ,NULMSG(1)                     DBE14790
C...                 *I/D DEVICES AND FILES                             DBE14800
      CDMMDN /DEXFIL/                                                   DBE14810
     1                MSGSDV,USEDEV,INFDDV,DBSDEV,NEWSDV,HELPDV,        DBE14820
     2                DUTDEV,INPDEV,NFWDS ,                             DBE14830
     3                MSGSFL,USEFIL,INFDFL,DBSFIL,NEWSFL,HELPFL,        DBE14840
     4                OUTFIL,INPFIL,LOADFL,NDT1FL,NOT2FL                DBE14850
      INTEGER                                                           DBE14860
     1                MSGSDV,USEDEV,INFDDV,DBSDEV,NEWSDV,HELPDV,        DBE14870
     2                DUTDEV,INPDEV,                                    DBE14880
     3                MSGSFL,USEFIL,INFDFL,DBSFIL,NEWSFL,HELPFL,        DBE14890
     4                DUTFIL,INPFIL,LOADFL,NOT1FL,NDT2FL,NFWDS          DBE14900
C...  FILE NAME DIMENSIDNING ADJUSTED FDR CDC AND CU DNLY.....          DBE14910
```

```
      DIMENSIDN       MSGSFL(11)  ,USEFIL(11)  ,INFDFL(11)  ,          DBE14920
     1                DBSFIL(11)  ,NEWSFL(11)  ,HELPFL(11)  ,          DBE14930
     3                DUTFIL(11)  ,INPFIL(11)  ,                       DBE14940
     2                LDADFL(11)  ,NDT1FL(11)  ,NDT2FL(11)             DBE14950
      COMMON /DBCDM/                                                   DBE14960
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBE14970
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBE14980
     2               INTTYP,RELTYP,ARRTYP,                              DBE14990
     3               DBCLSD,DBACTV,FILEDB                               DBE15000
      INTEGER                                                          DBE15010
     1               TITLE ,BUCKET,IDENT ,NODTYP,DATUM ,LINK  ,LINKF ,  DBE15020
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBE15030
     2               INTTYP,RELTYP,ARRTYP                               DBE15040
      LDGICAL                                                          DBE15050
     3               DBCLSD,DBACTV,FILEDB                               DBE15060
      DIMENSION       TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),    DBE15070
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)        DBE15080
      INTEGER    FILNAM,RCDDE,DXMENU                                   DBE15090
      INTEGER    NAME(1),TEMP(20)                                      DBE15100
      LDGICAL    NAMEDF,DBSAVE,DXPRMT                                  DBE15110
      DIMENSIDN FILNAM(987)                                            DBE15120
      DIMENSIDN MESG(16),MENNAM(2),ITEMS(4)                            DBE15130
      DATA       MENNAM /4HDXYE,4HS-NO/                                DBE15140
      DATA       MSGWDS/16/,NITEMS /2/                                 DBE15150
      DATA       ITEMS /4HYES ,4H    ,4HND  ,4H    /                  DBE15160
      DATA       NAMSUB/4HCLUT/                                        DBE15170
      DATA       NAME  /4HOPUT/                                        DBE15180
C...                                                                   DBE15190
      IF (DBCLSD) GO TO 11111                                          DBE15200
      CALL DXFNMV(DBSFIL,TEMP)                                         DBE15210
         IF (.NOT.(NAMEDF)) GO TO 1000                                 DBE15220
            CALL DXFNMV(FILNAM,DBSFIL)                                 DBE15230
            GO TO 4000                                                 DBE15240
1000  CONTINUE                                                         DBE15250
C...           +++ ANNOUNCE THE NAME OF THE OPEN DATABASE              DBE15260
         CALL DXMSGZ(NAMSUB,1)                                         DBE15270
         CALL DXFNPR(DBSFIL)                                           DBE15280
C...       .   +++ SHOULD DATABASE BE SAVED\                           DBE15290
         CALL DXMSGF(NAMSUB,2,MESG,MSGWDS)                             DBE15300
         ITEM=DXMENU(MENNAM,NITEMS,ITEMS,MESG)                         DBE15310
         IF (DXREQS) GO TO 99999                                       DBE15320
      IF (ITEM.EQ.1) GO TO 2000                                        DBE15330
      DBACTV=.FALSE.                                                   DBE15340
      GO TO 6000                                                       DBE15350
 2000 CONTINUE                                                         DBE15360
C...           +++ SHOULD IT BE SAVED WITH A NEW FILE NAME\            DBE15370
               CALL DXMSGF(NAMSUB,3,MESG,MSGWDS)                       DBE15380
```

```
                ITEM=DXMENU(MENNAM,NITEMS,ITEMS,MESG)                    DBE15390
                IF (DXREQS) GO TO 99999                                  DBE15400
                   GO TO (3000,4000),ITEM                                DBE15410
3000               CONTINUE                                              DBE15420
C...            +++ GET THE NEW FILE NAME                                DBE15430
                      CALL DXGFNM(DBSFIL)                                DBE15440
                      IF (DXREQS) GO TO 99999                            DBE15450
                         GO TO 4000                                      DBE15460
4000               CONTINUE                                              DBE15470
C...            +++ SAVE THE FILE                                        DBE15480
                      DBACTV=DBSAVE(DBSFIL)                              DBE15490
                      FILEDB=DBACTV                                      DBE15500
                      IF (DBACTV) GO TO 6000                             DBE15510
                         IF (.NOT.(NAMEDF)) GO TO 5000                   DBE15520
C...            +++ THE DATABASE COULD NOT BE SAVED AS NAMED IN          DBE15530
C...            +++ FILNAM                                               DBE15540
                            DBACTV=.TRUE.                                DBE15550
                            CALL DXFNMV(TEMP,DBSFIL)                     DBE15560
                            RCODE=2                                      DBE15570
                            GO TO 99999                                  DBE15580
C...                                                                     DBE15590
5000                      CONTINUE                                       DBE15600
C...            +++ THE DATABASE COULD NOT BE SAVED WITH THE             DBE15610
C...            +++ INTERACTIVELY OBTAINED NAME. TRY AGAIN BUT           DBE15620
C...            +++ PROMPT FIRST                                         DBE15630
                            CALL DXMSGZ(NAMSUB,5)                        DBE15640
                            CALL DXFNPR(DBSFIL)                          DBE15650
                            DBACTV=.TRUE.                                DBE15660
                            CALL DXFNMV(TEMP,DBSFIL)                     DBE15670
                            CALL DXMSGF(NAME,6,MSG,MSGWDS)               DBE15680
                            ITEM=DXMENU(MENNAM,NITEMS,ITEMS,MESG)        DBE15690
                            IF (DXREQS) GO TO 99999                      DBE15700
                            GO TO (1000,5100),ITEM                       DBE15710
 5100                       CONTINUE                                     DBE15720
                           RCODE=2                                       DBE15730
                            GO TO 99999                                  DBE15740
6000                  CDNTINUE                                           DBE15750
                         RCODE=0                                         DBE15760
                         GO TO 7000                                      DBE15770
7000                        CDNTINUE                                     DBE15780
                              DBCLSD=.TRUE.                              DBE15790
                              GO TO 99999                                DBE15800
11111 CDNTINUE                                                           DBE15810
C...            +++ THERE IS NO ACTIVE DATABASE                          DBE15820
         IF (.NOT.(NAMEDF)) CALL DXMSGZ(NAMSUB,4)                        DBE15830
         RCODE=1                                                         DBE15840
         GO TO 99999                                                     DBE15850
```

```
99999 CONTINUE                                                          DBE15860
      RETURN                                                            DBE15870
      END                                                               DBE15880
```

```
C                                                                       DBD00010
C ***********************************************************************DB000020
      LDGICAL FUNCTIDN OBDPEN(NAMEDF,NEWBAS,FILNAM,RCDOE)               DBD00030
C +---------------------------------------------------------------------+DB000040
C                                                                       DB000050
C        DBOPEN -                                                       DBD00060
C                                                                       DBD00070
C             RDUTINE TO CALL DBOPUT TD DPEN A DATABASE                 DBD00080
C                                                                       DB000090
C             RCODE AS RETURNED FRDM DBDPUT                             DB000100
C                                                                       DB000110
C +---------------------------------------------------------------------+DB000120
C                                                                       DB000130
      INTEGER FILNAM,RCODE                                              DBD00140
      LDGICAL NAMEOF,NEWBAS                                             0BD00150
      OIMENSIDN FILNAM(987)                                             DBD00160
      CALL DBOPUT(NAMEOF,NEWBAS,FILNAM,RCDOE)                           DBD00170
      DBOPEN=(RCDDE.EQ.O)                                               DBD00180
99999 CDNTINUE                                                          0BD00190
           RETURN                                                       DB000200
      END                                                               DBD00210
```

```
C                                                                       DB000220
C ***********************************************************************DB000230
      LOGICAL FUNCTION DBCLOS(NAMEDF,FILNAM,RCODE)                      DB000240
C +-------------------------------------------------------------------+ DB000250
C                                                                       DB000260
C        DBCLOS -                                                       DB000270
C                                                                       DB000280
C             ROUTINE TO CALL DBCLUT TO CLOSE AN OPEN DATABASE          DB000290
C                                                                       DB000300
C             RCODE AS RETURNED FROM DBCLUT                             DB000310
C                                                                       DB000320
C +-------------------------------------------------------------------+ DB000330
C                                                                       DB000340
      INTEGER FILNAM,RCODE                                              DB000350
      LOGICAL NAMEOF                                                    DB000360
      DIMENSION FILNAM(987)                                             DB000370
      CALL DBCLUT(NAMEDF,FILNAM,RCODE)                                  DB000380
      DBCLOS=(RCODE.EQ.0)                                               DB000390
      RETURN                                                            DB000400
      END                                                               DB000410
```

```
C                                                                       DBD00420
C ***********************************************************************DBD00430
      LOGICAL FUNCTION OBVINS(ONAME,TYPE,ARSIZE,RCDDE)                  DBD00440
C +---------------------------------------------------------------------+DBD00450
C                                                                       DBD00460
C        OBVINS -                                                       DBD00470
C                                                                       DB000480
C             OBVINS INSERTS THE VARIABLE CONTAINED IN ONAME INTO THE   DB000490
C             DPEN DATABASE.                                            DB000500
C                                                                       DB000510
C             RCOOE IS A RETURN CDDE;                                   DB000520
C                                                                       DBD00530
C                  RCDOE = O SUCCESSFUL INSERT                          DB000540
C                  RCODE = 1 NO DPEN OATABASE                           DB000550
C                  RCOOE = 2 NDOE ALREAOY EXISTS                        DBD00560
C                  RCOOE = 3 NO MDRE SPACE TO MAKE NODES                DB000570
C                  RCDOE = 4 WRDNG TYPE                                 DB000580
C                                                                       DBD00590
C +---------------------------------------------------------------------+DB000600
C                                                                       DBD00610
      COMMON /DBCOM/                                                    DBD00620
     1               TITLE ,BUCKET,IOENT ,NODTYP,DATUM ,LINK  ,LINKF ,  DB000630
     1               CMTPTR,NAVAIL,MAXNDO,MAXARR,NCHTIT,NCHCMT,OELCNT,  DB000640
     2               INTTYP,RELTYP,ARRTYP,                              DB000650
     3               DBCLSD,OBACTV,FILEOB                               DB000660
      INTEGER                                                           DB000670
     1               TITLE ,BUCKET,IOENT ,NDOTYP,OATUM ,LINK  ,LINKF ,  DB000680
     1               CMTPTR,NAVAIL,MAXNOO,MAXARR,NCHTIT,NCHCMT,OELCNT,  DB000690
     2               INTTYP,RELTYP,ARRTYP                               DB000700
      LOGICAL                                                           DB000710
     3               OBCLSO,OBACTV,FILEDB                               DB000720
      OIMENSIDN      TITLE(16),BUCKET(32),IOENT(200,2),NOOTYP(200),     DB000730
     1               OATUM(200),LINK(200),LINKF(200),CMTPTR(200)        DB000740
      INTEGER DNAME,TYPE,RCDDE,DBGETN,BKTPTR,OBNSKR                     DB000750
      INTEGER RSCODE,STYPE,ARSIZE                                       DB000760
      OIMENSION DNAME(2)                                                DB000770
      IF (DBCLSD) GD TO 11111                                           DB000780
         IF (TYPE.GT.ARRTYP) GD TO 44444                                DB000790
            NOOE=DBNSKR(ONAME,STYPE,BKTPTR,RSCODE)                      DB000800
            IF (NODE.NE.O) GD TO 22222                                  DBD00810
               NDDE=OBGETN(RCOOE)                                       DB000820
               IF (RCODE.NE.O) GO TD 33333                              DB000830
                                                                        DB000840
C...                                                                    DBD00850
C...              +++ INSERT THE NODE IN THE OATABASE                   DBD00860
C...              +++ LINK CHAINS UPWARO IN THE OATABASE TO PREVIOUS    DBD00870
C...              +++ NOOES THAT HASHED TO THE SAME BUCKET.  LINKF LINKSDBD00880
C...              +++ FORWARO TO PROVIDE A 2-WAY LINK BETWEEN THE NOOES.
```

```
C...              +++ THE NDDE NUMBER IS INSERTEO IN THE BUCKET AS A     OBOOO890
C...              +++ POINTER FRDM THE BUCKET TO THE NODE. IOENT(NOOE,1)DBOOO900
C...              +++ ANO IDENT(NDDE,2) CONTAIN THE OATUM NAME.  NOOTYP OBOOO910
C...              +++ IS SET TO THE NEGATIVE OF THE TYPE TO INDICATE ND OBOOO920
C...              +++ OATA IS STDRED YET.  IF NDOTYP IS 3 THE EXPECTED  OBDOO930
C...              +++ ARRAY SIZE IS ENTEREO IN OATUM.                    OBOOO940
C...                                                                     OBOOO950
                  LINK(NDDE)=BUCKET(BKTPTR)                              OBDOO960
                  IF(LINK(NDOE).NE.O) LINKF(LINK(NODE))=NDOE             OBOOO970
                  BUCKET(BKTPTR)=NDOE                                    OBDOO980
                  LINKF(NODE)=-BKTPTR                                    OBDOO990
                  IDENT(NDDE,1)=DNAME(1)                                 OBDO1000
                  IOENT(NDDE,2)=DNAME(2)                                 OBDO1010
                  NOOTYP(NODE)=-TYPE                                     DBOO1020
                  OATUM(NDOE)=O                                          OBOO1030
                  IF (TYPE.EQ.ARRTYP) DATUM(NODE)=ARSIZE                 OBOO1040
                  CMTPTR(NOOE)=O                                         OBOO1050
                  GO TD 99999                                            OBOO1060
11111 CONTINUE                                                           OBDO1070
C...     NO DPEN OATA BASE                                               OBDO1080
         RCODE=1                                                         OBDO1090
         GD TD 99999                                                     OBDO1100
22222 CONTINUE                                                           OBOO1110
C...     NDDE ALREAOY EXISTS                                             DBOO1120
         RCDDE=2                                                         DBOO1130
         GD TD 99999                                                     DBOO1140
33333 CONTINUE                                                           DBOO1150
C...     NO MDRE SPACE TO MAKE NDOES                                     DBOO1160
         RCDDE=3                                                         DBOO1170
         GD TO 99999                                                     OBOO1180
44444 CONTINUE                                                           OBOO1190
C...     TYPE HAS IMPROPER VALUE                                         DBOO1200
         RCDOE=4                                                         OBOO1210
         GO TO 99999                                                     OBOO1220
99999 CDNTINUE                                                           OBOO1230
         DBVINS=(RCOOE.EQ.O)                                             OBOO1240
         RETURN                                                          OBOO1250
      END                                                                OBOO1260
```

```
C                                                                       OB001270
C ***********************************************************************OB001280
      LOGICAL FUNCTION OBVOEL(DNAME,RCODE)                              OB001290
C +---------------------------------------------------------------------+OB001300
C                                                                       OB001310
C        OBVOEL -                                                       OB001320
C                                                                       OB001330
C             DELETES THE VARIABLE CONTAINED IN DNAME FROM THE OPEN     OB001340
C             DATABASE.  IT ADJUSTS THE LINK AND LINKF AS APROPRIATE    OB001350
C                                                                       OB001360
C             RCODE IS THE RETURN CODE;                                 OB001370
C                                                                       OB001380
C                  RCODE = 1 NO OPEN DATA BASE                          OB001390
C                                                                       OB001400
C +---------------------------------------------------------------------+OB001410
C                                                                       OB001420
      COMMON /OBCOM/                                                    OB001430
     1               TITLE ,BUCKET,IDENT ,NDTYP,DATUM ,LINK  ,LINKF ,   OB001440
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,  OB001450
     2               INTTYP,RELTYP,ARRTYP,                              OB001460
     3               OBCLSD,OBACTV,FILEOB                               OB001470
      INTEGER                                                           OB001480
     1               TITLE ,BUCKET,IDENT ,NDTYP,DATUM ,LINK  ,LINKF ,   OB001490
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,  OB001500
     2               INTTYP,RELTYP,ARRTYP                               OB001510
      LOGICAL                                                           OB001520
     3               OBCLSD,OBACTV,FILEOB                               OB001530
      DIMENSION      TITLE(16),BUCKET(32),IDENT(200,2),NDTYP(200),      OB001540
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)        OB001550
      INTEGER   DNAME,DBNSKR,PRNODE,RCODE,STYPE                         OB001560
      DIMENSION DNAME(2)                                                OB001570
C...          +++ SEARCH FOR THE NODE TO BE DELETED                     OB001580
      NODE=DBNSKR(DNAME,STYPE,PRNODE,RCODE)                             OB001590
      IF (OBCLSD) GO TO 11111                                           OB001600
         OBVOEL=(NODE.NE.0)                                             OB001610
         IF (.NOT.OBVOEL) GO TO 99999                                   OB001620
C...                                                                    OB001630
C...          +++ MAKE IT UNAVAILABLE BY SETTING NDTYP = -9             OB001640
C...          +++ IF THERE IS A LINK UPDATE THE LINKF OF THE NODE TO    OB001650
C...          +++ WHICH LINKED.  COPY THE LINK INTO EITHER THE LINK OF  OB001660
C...          +++ THE PREVIOUS NODE OR INTO THE BUCKET, DEPENDING ON    OB001670
C...          +++ POSITION IN THE CHAIN.  IF THE NODE IS AN ARRAY SET   OB001680
C...          +++ ITS DATUM (ARRAY POINTER) TO THE NEGATIVE OF ITS      OB001690
C...          +++ CURRENT VALUE.  SIMILARLY FOR COMMENTS.               OB001700
C...                                                                    OB001710
               IF (LINK(NODE).NE.0) LINKF(LINK(NODE))=LINKF(NODE)       OB001720
               IF (RCODE.EQ.0) LINK(PRNODE)=LINK(NODE)                  OB001730
```

```
                IF (RCODE.EQ.1) BUCKET(PRNODE)=LINK(NODE)                  DBD01740
                DATUM(NODE)=0                                              DBD01750
                IF (NODTYP(NODE) .EQ. 3) DATUM(NODE)=-DATUM(NODE)          DBD01760
                IF (CMTPTR(NODE) .NE. 0) CMTPTR(NODE)=-CMTPTR(NODE)        DBD01770
                LINKF(NODE)=0                                              DBD01780
                NODTYP(NODE)=-9                                            DBD01790
                DELCNT=DELCNT+1                                            DBD01800
                GO TO 99999                                                DBD01810
11111 CONTINUE                                                             DBD01820
C...           +++ NO OPEN DATA BASE                                       DBD01830
          RCODE=1                                                          DBD01840
          DBVDEL=.FALSE.                                                   DBD01850
          GO TO 99999                                                      DBD01860
99999 CONTINUE                                                             DBD01870
          RETURN                                                           DBD01880
      END                                                                  DBD01890
```

```
C                                                                       DBD01900
C ***********************************************************************DBD01910
      LOGICAL FUNCTIDN IPUT(NAME,IVALUE,RCDDE)                          DB001920
C +---------------------------------------------------------------------+DB001930
C                                                                       DBD01940
C        IPUT -                                                         DBD01950
C                                                                       DBD01960
C             PUT AN INTEGER VALUE (IVALUE) IN THE DATABASE AT THE      DBD01970
C             NODE ASSIGNED TD NAME.                                    DBD01980
C                                                                       DBD01990
C             RCDDE IS AS RETURNED FROM DBNDIN                          DBD02000
C                                                                       DBD02010
C +---------------------------------------------------------------------+DB002020
C                                                                       DBD02030
      CDMMDN /DBCDM/                                                    DB002040
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DB002050
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DB002060
     2               INTTYP,RELTYP,ARRTYP,                              DB002070
     3               DBCLSD,DBACTV,FILEDB                               DBD02080
      INTEGER                                                           DB002090
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBD02100
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBD02110
     2               INTTYP,RELTYP,ARRTYP                               DBD02120
      LOGICAL                                                           DBD02130
     3               DBCLSD,DBACTV,FILEDB                               DB002140
      DIMENSION      TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),     DB002150
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)        DBD02160
      INTEGER NAME  ,IVALUE,RCDDE                                       DB002170
      LOGICAL DBNDIN                                                    DB002180
      DIMENSIDN NAME(2)                                                 DB002190
      IPUT=DBNDIN(NAME,INTTYP,IVALUE,RCDDE)                             DB002200
99999 CDNTINUE                                                          DBD02210
         RETURN                                                         DB002220
      END                                                               DBD02230
```

```
C                                                                       DBD02240
C ***********************************************************************DBD02250
      LOGICAL FUNCTION IGET(NAME,IVALUE,RCODE)                          DBD02260
C +---------------------------------------------------------------------+DBD02270
C                                                                       DBD02280
C        IGET -                                                         DBD02290
C                                                                       DBD02300
C             GET AN INTEGER VALUE FROM THE DATABASE AT THE NODE        DBD02310
C             ASSIGNED TO NAME AND RETURN IT IN IVALUE.                 DBD02320
C                                                                       DBD02330
C             RCODE IS AS RETURNED FROM DBNDRD                          DBD02340
C                                                                       DBD02350
C +---------------------------------------------------------------------+DB002360
C                                                                       DB002370
      COMMON /DBCOM/                                                    DB002380
     1               TITLE ,BUCKET,IDENT ,NODTYP,DATUM ,LINK  ,LINKF ,  DBD02390
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBD02400
     2               INTTYP,RELTYP,ARRTYP,                              DBD02410
     3               DBCLSD,DBACTV,FILEDB                               DB002420
      INTEGER                                                           DB002430
     1               TITLE ,BUCKET,IDENT ,NODTYP,DATUM ,LINK  ,LINKF ,  DBD02440
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBD02450
     2               INTTYP,RELTYP,ARRTYP                               DB002460
      LOGICAL                                                           DB002470
     3               DBCLSD,DBACTV,FILEDB                               DB002480
      DIMENSION      TITLE(16),BUCKET(32),IDENT(200,2),NODTYP(200),     DB002490
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)        DBD02500
      INTEGER NAME  ,IVALUE,RCODE                                       DBD02510
      LOGICAL DBNDRD                                                    DBD02520
      DIMENSION NAME(2)                                                 DBD02530
      IGET=DBNDRD(NAME,INTTYP,IVALUE,RCODE)                             DBD02540
99999 CONTINUE                                                          DBD02550
         RETURN                                                         DBD02560
      END                                                               DBD02570
```

```
C                                                                       DBD02580
C ***********************************************************************DBD02590
      LDGICAL FUNCTIDN RPUT(NAME,RVALUE,RCDDE)                          DBD02600
C +-------------------------------------------------------------------+ DBD02610
C                                                                       DBD02620
C        RPUT -                                                         DBD02630
C                                                                       DBD02640
C             PUT A REAL VALUE (RVALUE) IN THE DATABASE AT THE NDDE     DBD02650
C             ASSIGNED TD NAME.                                         DBD02660
C                                                                       DBD02670
C             RCDDE IS AS RETURNED FRDM DBNDIN                          DBD02680
C                                                                       DBD02690
C +-------------------------------------------------------------------+ DBD02700
C                                                                       DBD02710
      CDMMDN /DBCDM/                                                    DBD02720
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBD02730
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBD02740
     2               INTTYP,RELTYP,ARRTYP,                              DBD02750
     3               DBCLSD,DBACTV,FILEDB                               DBD02760
      INTEGER                                                           DBD02770
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBD02780
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBD02790
     2               INTTYP,RELTYP,ARRTYP                               DBD02800
      LDGICAL                                                           DBD02810
     3               DBCLSD,DBACTV,FILEDB                               DBD02820
      DIMENSIDN      TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),     DBD02830
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)        DBD02840
      INTEGER NAME,RCDDE,ITEMP                                          DBD02850
      LDGICAL DBNDIN                                                    DBD02860
      DIMENSIDN NAME(2)                                                 DBD02870
      REAL RVALUE,RTEMP                                                 DBD02880
      EQUIVALENCE (ITEMP,RTEMP)                                         DBD02890
      RTEMP = RVALUE                                                    DBD02900
      RPUT=DBNDIN(NAME,RELTYP,ITEMP,RCDDE)                              DBD02910
99999 CDNTINUE                                                          DBD02920
         RETURN                                                         DBD02930
      END                                                               DBD02940
```

```
C                                                                       DBD02950
C ***********************************************************************DBD02960
      LDGICAL FUNCTIDN RGET(NAME,RVALUE,RCDDE)                          DBD02970
C +---------------------------------------------------------------------+DBD02980
C                                                                       DBD02990
C        RGET-                                                          DBD03000
C                                                                       DBD03010
C             GET A REAL VALUE FRDM THE DATABASE NODE ASSIGNED TD NAME  DB003020
C             AND RETURN IT IN RVALUE.                                  DBD03030
C                                                                       DBD03040
C             RCDDE IS AS RETURNED FRDM DBNDRD                          DBD03050
C                                                                       DBD03060
C +---------------------------------------------------------------------+DBD03070
C                                                                       DBD03080
      CDMMDN /DBCDM/                                                    DBD03090
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DB003100
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBD03110
     2               INTTYP,RELTYP,ARRTYP,                              DB003120
     3               DBCLSD,DBACTV,FILEDB                               DB003130
      INTEGER                                                           DBD03140
     1               TITLE ,BUCKET,IDENT ,NODTYP,DATUM ,LINK  ,LINKF ,  DB003150
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DB003160
     2               INTTYP,RELTYP,ARRTYP                               DBD03170
      LDGICAL                                                           DBD03180
     3               DBCLSD,DBACTV,FILEDB                               DBD03190
      DIMENSIDN      TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),     DBD03200
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)        DBD03210
      INTEGER NAME,RCDDE,ITEMP                                          DBD03220
      DIMENSIDN NAME(2)                                                 DBD03230
      REAL RVALUE,RTEMP                                                 DBD03240
      LDGICAL DBNDRD                                                    DBD03250
      EQUIVALENCE (ITEMP,RTEMP)                                         DBD03260
      RGET=DBNDRD(NAME,RELTYP,ITEMP,RCDDE)                              DBD03270
      RVALUE=RTEMP                                                      DB003280
99999 CDNTINUE                                                          DB003290
         RETURN                                                         DB003300
      END                                                               DBD03310
```

```
C                                                                       DBD03320
C ***********************************************************************DBD03330
      LDGICAL FUNCTIDN AGET(NAME,RARRAY,NGET,NSTDRD,RCDDE)              DBD03340
C +---------------------------------------------------------------------+DBD03350
C                                                                       DBD03360
C        AGET -                                                         DBD03370
C                                                                       DBD03380
C             GET THE VALUES OF AN ARRAY FRDM THE ECS STDRAGE AREA      DBD03390
C             ASSIGNED TD NAME AND RETURN IN THE ARRAY RARRAY.          DBD03400
C                                                                       DBD03410
C             RCDDES 1-4 ARE AS RETURNED FRDM DBNDRD                    DBD03420
C             RCDDE = 5 NGET > NSTDRD                                   DBD03430
C             RCDDE = 6 NGET < NSTDRD                                   DBD03440
C                                                                       DBD03450
C +---------------------------------------------------------------------+DBD03460
C                                                                       DBD03470
      CDMMDN /DBCDM/                                                    DBD03480
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBD03490
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBD03500
     2               INTTYP,RELTYP,ARRTYP,                              DBD03510
     3               DBCLSD,DBACTV,FILEDB                               DBD03520
      INTEGER                                                           DBD03530
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBD03540
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBD03550
     2               INTTYP,RELTYP,ARRTYP                               DBD03560
      LDGICAL                                                           DBD03570
     3               DBCLSD,DBACTV,FILEDB                               DBD03580
      DIMENSIDN      TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),     DBD03590
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)        DBD03600
      INTEGER NAME,NGET,NSTDRD,RCDDE,PDINTR,START                       DBD03610
      DIMENSIDN NAME(2),RARRAY(987)                                     DBD03620
      LDGICAL DBNDRD,DBNARD                                             DBD03630
      DATA START /1/                                                    DBD03640
      AGET=DBNDRD(NAME,ARRTYP,PDINTR,RCDDE)                             DBD03650
      IF (RCDDE.EQ.3) AGET=.FALSE.                                      DBD03660
      IF (RCDDE.EQ.4) AGET=.FALSE.                                      DBD03670
      IF (RCDDE.EQ.4) NSTDRD=PDINTR                                     DBD03680
      IF (.NDT.AGET) GD TD 99999                                        DBD03690
         AGET=DBNARD(PDINTR,RARRAY,START,NGET,NSTDRD)                   DBD03700
         IF (NGET.GT.NSTDRD) RCDDE=5                                    DBD03710
            IF (NGET.LT.NSTDRD) RCDDE=6                                 DBD03720
99999 CDNTINUE                                                          DBD03730
         RETURN                                                         DBD03740
      END                                                               DBD03750
```

```
C                                                                       DBD03760
C ***********************************************************************DBD03770
      LDGICAL FUNCTIDN APUT(NAME,RARRAY,NPUT,NSTORD,RCDDE)              DBD03780
C +---------------------------------------------------------------------+DBD03790
C                                                                       DBD03800
C        APUT -                                                         DBD03810
C                                                                       DBD03820
C             PUT THE VALUES DF THE ARRAY (RARRAY) INTD THE ECS         DBD03830
C             STDRAGE AREA ASSIGNED TD NAME.                            DBD03840
C                                                                       DBD03850
C             RCDDE = O SUCCESS                                         DBD03860
C             RCDDE 1-3 AS IN DBNDIN                                    DBD03870
C             RCDDE = 4 NPUT > NSTDRD                                   DBD03880
C             RCDDE = 5 NPUT < NSTDRD                                   DBD03890
C                                                                       DBD03900
C +---------------------------------------------------------------------+DBD03910
C                                                                       DBD03920
      CDMMDN /DBCDM/                                                    DBD03930
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBD03940
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBD03950
     2               INTTYP,RELTYP,ARRTYP,                              DBD03960
     3               DBCLSD,DBACTV,FILEDB                               DBD03970
      INTEGER                                                           DBD03980
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBD03990
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBD04000
     2               INTTYP,RELTYP,ARRTYP                               DBD04010
      LDGICAL                                                           DBD04020
     3               DBCLSD,DBACTV,FILEDB                               DBD04030
      DIMENSIDN      TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),     DBD04040
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)        DBD04050
      INTEGER NAME,NPUT,NSTDRD,RCDDE,PDINTR,START                       DBD04060
      LDGICAL   DBNDRD,DBNAIN,DBNDIN,NEW                                DBD04070
      DIMENSIDN NAME(2),RARRAY(987)                                     DBD04080
      DATA START /1/, NDDAT /4/                                         DBD04090
      APUT=DBNDRD(NAME,ARRTYP,PDINTR,RCDDE)                             DBD04100
      IF(RCDDE.EQ.O) GD TD 1000                                         DBD04110
      IF(RCDDE.LT.4) GD TD 99999                                        DBD04120
1000  CDNTINUE                                                          DBD04130
         NEW=(RCDDE.EQ.NDDAT)                                           DBD04140
         APUT=DBNAIN(NEW,NAME,PDINTR,RARRAY,START,NPUT,NSTDRD)          DBD04150
         IF (NEW.AND.APUT) APUT=DBNDIN(NAME,ARRTYP,PDINTR,RCDDE)        DBD04160
         IF (APUT) GD TD 99999                                          DBD04170
            IF (NPUT.GT.NSTDRD) RCDDE=4                                 DBD04180
               IF (NPUT.LT.NSTDRD) RCDDE=5                              DBD04190
99999 CDNTINUE                                                          DBD04200
         RETURN                                                         DBD04210
      END                                                               DBD04220
```

```
C                                                                       DBD04230
C ***********************************************************************DBD04240
      LOGICAL FUNCTION TPUT(STRING)                                     DBD04250
C +---------------------------------------------------------------------+DBD04260
C                                                                       DBD04270
C        TPUT -                                                         DBD04280
C                                                                       DBD04290
C             PUT THE TITLE CONTAINED IN STRING INTO THE DATABASE.      DBD04300
C                                                                       DBD04310
C +---------------------------------------------------------------------+DBD04320
C                                                                       DBD04330
C...                  *THE CDMMDN CDNTRDL PARAMETERS,CHARACTERS, AND    DBD04340
C...                  *GRAPHIC DISPLAY CDNTRDL PARAMETERS               DBD04350
      COMMON /DEXCDM/                                                   DBD04360
     1                OXMDDE,LOAOED,MOPEND,DXREQS,DLREQS,KEYBRD,       DBD04370
     2                TERSE ,ECHDMD,DXSC  ,CPWOXS,MDSC  ,CURMDD,       DBD04380
     3                NULMSG,DXNCPW,NAMLEN,                            DBD04390
     4                CLRMOD,MUNPST,ERASE,UPDATE,UPAINT,GCNTRL         DBD04400
      INTEGER                                                           DBD04410
     1                OXSC  ,CPWOXS,MOSC  ,CURMOD,NULMSG,DXNCPW,       DBD04420
     2                NAMLEN,CLRMDD,MUNPST,ERASE,UPOATE,UPAINT         DBD04430
      LDGICAL                                                           DBD04440
     1                DXMDDE,LOADED,MOPEND,DXREQS,DLREQS,KEYBRD,       DBD04450
     2                TERSE ,ECHOMD,GCNTRL                             DBD04460
      OIMENSIDN       DXSC(1),CURMDO(2)  ,NULMSG(1)                     DBD04470
      CDMMON /DBCDM/                                                    DBD04480
     1               TITLE ,BUCKET,IOENT ,NODTYP,DATUM ,LINK  ,LINKF ,  DBD04490
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBD04500
     2               INTTYP,RELTYP,ARRTYP,                              DBD04510
     3               DBCLSD,DBACTV,FILEDB                               DBD04520
      INTEGER                                                           DBD04530
     1               TITLE ,BUCKET,IDENT ,NDOTYP,DATUM ,LINK  ,LINKF ,  DBD04540
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBD04550
     2               INTTYP,RELTYP,ARRTYP                               DBD04560
      LOGICAL                                                           DBD04570
     3               DBCLSD,OBACTV,FILEDB                               DBD04580
      DIMENSIDN       TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),    DBD04590
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)        DBD04600
      INTEGER   STRING                                                  DBD04610
      OIMENSIDN STRING(987)                                             DBD04620
      IF (DBCLSO) GD TO 11111                                           DBD04630
         LENTIT=NCHTIT/DXNCPW                                           DBD04640
         DO 1000 I=1,LENTIT                                             DBD04650
            TITLE(I)=STRING(I)                                          DBD04660
1000     CONTINUE                                                       DBD04670
         TPUT=.TRUE.                                                    DBD04680
         GD TO 99999                                                    DBD04690
```

```
11111 CONTINUE                                                  DB004700
         TPUT=.FALSE.                                           DB004710
         GO TO 99999                                            DB004720
99999 CONTINUE                                                  DB004730
         RETURN                                                 DB004740
      END                                                       DB004750
```

```
C                                                                       DB004760
C ***********************************************************************DBD04770
      LDGICAL FUNCTIDN TGET(STRING)                                     DB004780
C +---------------------------------------------------------------------+DB004790
C                                                                       DB004800
C        TGET -                                                         DB004810
C                                                                       DBD04820
C             GET THE TITLE STORED IN THE DATABASE AND RETURN IT        DBD04830
C             IN STRING.                                                DBD04840
C                                                                       DB004850
C +---------------------------------------------------------------------+DB004860
C                                                                       DBD04870
C...                  *THE CDMMDN CONTRDL PARAMETERS,CHARACTERS, AND    DBD04880
C...                  *GRAPHIC DISPLAY CONTROL PARAMETERS               DB004890
      COMMON /DEXCDM/                                                   DBD04900
     1                DXMODE,LOADED,MDPEND,DXREQS,DLREQS,KEYBRD,        DBD04910
     2                TERSE ,ECHDMD,DXSC  ,CPWOXS,MOSC  ,CURMDD,        DB004920
     3                NULMSG,DXNCPW,NAMLEN,                             DB004930
     4                CLRMDD,MUNPST,ERASE,UPOATE,UPAINT,GCNTRL          DB004940
      INTEGER                                                           DB004950
     1                DXSC  ,CPWOXS,MOSC  ,CURMOD,NULMSG,DXNCPW,        DBD04960
     2                NAMLEN,CLRMOD,MUNPST,ERASE,UPOATE,UPAINT          DBD04970
      LDGICAL                                                           DBD04980
     1                OXMOOE,LDADED,MDPEND,OXREQS,OLREQS,KEYBRD,        DBD04990
     2                TERSE ,ECHDMO,GCNTRL                              DBD05000
      DIMENSIDN       DXSC(1),CURMOD(2)  ,NULMSG(1)                     DBD05010
      CDMMON /DBCDM/                                                    DB005020
     1               TITLE ,BUCKET,IDENT ,NDOTYP,DATUM ,LINK  ,LINKF ,  DBD05030
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DB005040
     2               INTTYP,RELTYP,ARRTYP,                              DB005050
     3               DBCLSD,OBACTV,FILEDB                               DB005060
      INTEGER                                                           DB005070
     1               TITLE ,BUCKET,IOENT ,NODTYP,DATUM ,LINK  ,LINKF ,  DB005080
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DB005090
     2               INTTYP,RELTYP,ARRTYP                               DB005100
      LOGICAL                                                           DB005110
     3               DBCLSO,OBACTV,FILEOB                               DB005120
      DIMENSIDN      TITLE(16),BUCKET(32),IDENT(200,2),NOOTYP(200),     DBD05130
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)        DBD05140
      INTEGER   STRING                                                  DBD05150
      OIMENSION  STRING(987)                                            DBD05160
      IF (DBCLSO) GD TD 11111                                           DBD05170
         LENTIT=NCHTIT/DXNCPW                                           DBD05180
         TGET=(TITLE(1).NE.O)                                           DBD05190
         IF (.NOT.(TGET)) GO TO 99999                                   DBD05200
            DD 1000 I=1,LENTIT                                          DBD05210
               STRING(I)=TITLE(I)                                       DBD05220
```

```
1000        CONTINUE                                                    DB005230
          GO TO 99999                                                   DB005240
11111 CONTINUE                                                          DB005250
C...            +++ NO OPEN DATABASE                                    DB005260
            TGET = .FALSE.                                              DB005270
            GO TO 99999                                                 DB005280
99999 CONTINUE                                                          DB005290
           RETURN                                                       DB005300
      END                                                               DB005310
```

```
C                                                                       DBD05320
C ***********************************************************************DBD05330
      LDGICAL FUNCTION CMTGET(NAME,CMTSTR,RCDDE)                        DBD05340
C +---------------------------------------------------------------------+DBD05350
C                                                                       DBD05360
C        CMTGET -                                                       DBD05370
C                                                                       DBD05380
C             GET THE COMMENT STDRED FOR VARIABLE NAME FRDM THE         DBD05390
C             ECS STDRAGE AREA AND RETURN IT IN CMTSTR.                 DBD05400
C                                                                       DBD05410
C             RCDDE IS A RETURN CODE;                                   DBD05420
C                                                                       DBD05430
C                  RCODE = O SUCCESS                                    DBD05440
C                  RCDDE = 1 ND DPEN DATABASE                           DB005450
C                  RCODE = 2 NODE NOT IN DATABASE                       DB005460
C                  RCDDE = 3 NO CDMMENT STDRED                          DBD05470
C                                                                       DBD05480
C +---------------------------------------------------------------------+DB005490
C                                                                       DBD05500
      CDMMDN /DBCOM/                                                    DBD05510
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBD05520
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DB005530
     2               INTTYP,RELTYP,ARRTYP,                              DB005540
     3               DBCLSD,DBACTV,FILEDB                               DB005550
      INTEGER                                                           DBD05560
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DB005570
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DB005580
     2               INTTYP,RELTYP,ARRTYP                               DB005590
      LDGICAL                                                           DB005600
     3               DBCLSD,DBACTV,FILEDB                               DBD05610
      DIMENSION      TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),     DBD05620
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)        DB005630
       INTEGER   CMTSTR,RCODE,DBNSKR,PDINTR                             DB005640
       LDGICAL   DBNCRD                                                 DB005650
       DIMENSIDN CMTSTR(987),NAME(2)                                    DB005660
       IF (DBCLSD) GD TO 11111                                          DBD05670
          NODE=DBNSKR(NAME,IDUM1,IDUM2,RCDDE)                           DB005680
          CMTGET=(NODE.NE.O)                                            DBD05690
          IF (.NOT.CMTGET) GO TD 22222                                  DBD05700
             POINTR=CMTPTR(NODE)                                        DBD05710
             CMTGET=DBNCRD(POINTR,CMTSTR)                               DBD05720
             IF (.NOT.(CMTGET)) GO TD 33333                             DBD05730
                RCDDE=O                                                 DBD05740
                GD TD 99999                                             DBD05750
11111 CONTINUE                                                          DBD05760
C...           +++ ND DPEN DATABASE                                     DBD05770
         RCDDE=1                                                        DB005780
```

```
         GO TO 99999                                                    0B005790
22222 CONTINUE                                                          0B005800
C...           +++ NOT IN OATA BASE                                     0B005810
         RCOOE=2                                                        0B005820
         GO TO 99999                                                    0B005830
33333 CONTINUE                                                          0B005840
C...           +++ NO COMMENT IS STOREO                                 0B005850
         RCOOE=3                                                        0B005860
         GO TO 99999                                                    0B005870
99999 CONTINUE                                                          0B005880
         RETURN                                                         0B005890
       END                                                              0B005900
```

```
C                                                                       DBD05910
C ***********************************************************************DB005920
      LDGICAL FUNCTION CMTPUT(NAME,CMTSTR,RCDDE)                        DB005930
C +---------------------------------------------------------------------+DB005940
C                                                                       DBD05950
C        CMTPUT -                                                       DBD05960
C                                                                       DBD05970
C             PUT THE COMMENT CONTAINED IN CMTSTR INTD THE ECS          DBD05980
C             STRDAGE AREA ASSIGNED TO NAME.                            DBD05990
C                                                                       DBD06000
C             RCDDE IS A RETURN CODE;                                   DBD06010
C                                                                       DBD06020
C                  RCODE = O SUCCESS                                    DBD06030
C                  RCDDE = 1 NO OPEN DATABASE                           DBD06040
C                  RCDDE = 2 NODE NOT IN DATABASE                       DB006050
C                  RCODE = 3 NDT ABLE TD GET SPACE                      DBD06060
C                                                                       DB006070
C +---------------------------------------------------------------------+DB006080
C                                                                       DB006090
      COMMON /DBCDM/                                                    DB006100
     1               TITLE ,BUCKET,IDENT ,NODTYP,DATUM ,LINK  ,LINKF ,  DB006110
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DB006120
     2               INTTYP,RELTYP,ARRTYP,                              DB006130
     3               DBCLSD,DBACTV,FILEDB                               DB006140
      INTEGER                                                           DB006150
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DB006160
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DB006170
     2               INTTYP,RELTYP,ARRTYP                               DB006180
      LDGICAL                                                           DBD06190
     3               DBCLSD,DBACTV,FILEDB                               DB006200
      DIMENSIDN      TITLE(16),BUCKET(32),IDENT(200,2),NODTYP(200),     DBD06210
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)        DBD06220
      INTEGER CMTSTR,RCDDE,DBNSKR,POINTR                                DB006230
      LDGICAL   DBNCIN                                                  DB006240
      DIMENSIDN CMTSTR(987),NAME(2)                                     DB006250
      IF (DBCLSD) GO TO 11111                                           DB006260
         NDDE=DBNSKR(NAME,IDUM1,IDUM2,RCDDE)                            DB006270
         CMTPUT=(NODE.NE.O)                                             DB006280
         IF (.NDT.CMTPUT) GO TO 22222                                   DBD06290
            POINTR=CMTPTR(NODE)                                         DBD06300
            CMTPUT=DBNCIN(POINTR,CMTSTR,NDDE)                           DBD06310
            IF (.NOT.(CMTPUT)) GO TO 33333                              DBD06320
               CMTPTR(NDDE)=PDINTR                                      DBD06330
               RCDDE=O                                                  DBD06340
               GO TO 99999                                              DBD06350
11111 CONTINUE                                                          DBD06360
C...          +++ NO DPEN DATABASE                                      DBD06370
```

```
         RCODE=1                                                        DB006380
         GO TO 99999                                                    DB006390
22222 CONTINUE                                                          DB006400
C...           +++ NOT IN DATA BASE                                     DB006410
         RCODE=2                                                        DB006420
         GO TO 99999                                                    DB006430
33333 CONTINUE                                                          DB006440
C...           +++ NOT ABLE TO GET SPACE FOR COMMENT                    DB006450
         RCODE=3                                                        DB006460
         GO TO 99999                                                    DB006470
99999 CONTINUE                                                          DB006480
         RETURN                                                         DB006490
      END                                                               DB006500
```

```
C                                                                       DBL00010
C***********************************************************************DBL00020
      LDGICAL FUNCTIDN DBLDAD(RCDDE)                                    DBL00030
C +-------------------------------------------------------------------+ DBL00040
C                                                                       DBL00050
C        DBLOAD -                                                       DBL00060
C                                                                       DBL00070
C             LDAD AN EXISTING DATABASE FROM DISK.                      DBL00080
C                                                                       DBL00090
C             RCDDE IS A RETURN CDDE;                                   DBL00100
C                                                                       DBL00110
C                  RCODE = O DATABASE SUCCESSFULLY LOADED               DBL00120
C                  RCDDE = 1 DATABASE FILE NDT FDUND                    DBL00130
C                  RCDDE = 2 FILE IS NDT A DEX DATABASE                 DBL00140
C                  RCDDE = 3 ECS STORAGE AREA IS FULL                   DBL00150
C                  RCDDE = 4 PREMATURE EDF ENCDUNTERED                  DBL00160
C                                                                       DBL00170
C +-------------------------------------------------------------------+ DBL00180
C                                                                       DBL00190
C+++  THIS ROUTINE IS MACHINE DEPENDENT                                 DBL00200
C+++  THIS RDUTINE IS SITE DEPENDENT                                    DBL00210
C+++  USES ECS                                                          DBL00220
C     +-----------------------------------------------------------+     DBL00230
C     .  ESTABLISH DATA BASE FDR 'DIRECT' ACCESS IN ECS           .     DBL00240
C     +-----------------------------------------------------------+     DBL00250
C...                 *THE COMMON CONTRDL PARAMETERS,CHARACTERS, AND     DBL00260
C...                 *GRAPHIC DISPLAY CONTRDL PARAMETERS                DBL00270
      CDMMDN /DEXCOM/                                                   DBL00280
     1               DXMDDE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,        DBL00290
     2               TERSE ,ECHDMD,DXSC  ,CPWDXS,MDSC  ,CURMDD,        DBL00300
     3               NULMSG,DXNCPW,NAMLEN,                             DBL00310
     4               CLRMOD,MUNPST,ERASE,UPDATE,UPAINT,GCNTRL          DBL00320
      INTEGER                                                           DBL00330
     1               DXSC  ,CPWDXS,MDSC  ,CURMOD,NULMSG,DXNCPW,        DBL00340
     2               NAMLEN,CLRMOD,MUNPST,ERASE,UPDATE,UPAINT          DBL00350
      LDGICAL                                                           DBL00360
     1               DXMODE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,        DBL00370
     2               TERSE ,ECHOMD,GCNTRL                              DBL00380
      DIMENSIDN      DXSC(1),CURMOD(2)  ,NULMSG(1)                     DBL00390
C...                 *I/O DEVICES AND FILES                             DBL00400
      CDMMON /DEXFIL/                                                   DBL00410
     1               MSGSDV,USEDEV,INFDDV,DBSDEV,NEWSDV,HELPDV,        DBL00420
     2               DUTDEV,INPDEV,NFWDS ,                             DBL00430
     3               MSGSFL,USEFIL,INFDFL,DBSFIL,NEWSFL,HELPFL,        DBL00440
     4               OUTFIL,INPFIL,LDADFL,NOT1FL,NDT2FL                DBL00450
      INTEGER                                                           DBL00460
     1               MSGSDV,USEDEV,INFDDV,DBSDEV,NEWSDV,HELPDV,        DBL00470
```

```
     2                OUTDEV,INPDEV,                                          DBL00480
     3                MSGSFL,USEFIL,INFOFL,DBSFIL,NEWSFL,HELPFL,              DBL00490
     4                OUTFIL,INPFIL,LOADFL,NOT1FL,NDT2FL,NFWDS                DBL00500
C...  FILE NAME DIMENSIDNING ADJUSTED FOR COC AND CU DNLY.....                DBL00510
      DIMENSION       MSGSFL(11)  ,USEFIL(11)  ,INFDFL(11)  ,                 DBL00520
     1                DBSFIL(11)  ,NEWSFL(11)  ,HELPFL(11)  ,                 DBL00530
     3                OUTFIL(11)  ,INPFIL(11)  ,                              DBL00540
     2                LOADFL(11)  ,NOT1FL(11)  ,NOT2FL(11)                    DBL00550
      COMMON /DBCOM/                                                          DBL00560
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,        DBL00570
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,        DBL00580
     2               INTTYP,RELTYP,ARRTYP,                                    DBL00590
     3               OBCLSO,DBACTV,FILEDB                                     DBL00600
      INTEGER                                                                 DBL00610
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,        DBL00620
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,        DBL00630
     2               INTTYP,RELTYP,ARRTYP                                     DBL00640
      LOGICAL                                                                 DBL00650
     3               DBCLSD,DBACTV,FILEDB                                     DBL00660
      OIMENSIDN       TITLE(16),BUCKET(32),IDENT(200,2),NDOTYP(200),          DBL00670
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)              DBL00680
      COMMDN /OBOAID/                                                         DBL00690
     1               MAXECS,NEXECS,PRECS,ECSPTR,ECSLEN                        DBL00700
      INTEGER                                                                 DBL00710
     1               PRECS,ECSPTR,ECSLEN                                      DBL00720
      DIMENSION                                                               DBL00730
     1               ECSPTR(2),ECSLEN(2)                                      DBL00740
      INTEGER   DBFCHK, CHKSTR,RCODE,WDSTIT,WDSCMT                            DBL00750
      LDGICAL   OXCKDV,LDG,DBAPTR                                             DBL00760
      DIMENSION ARRAY(200), DBFCHK(2), CHKSTR(2)                              DBL00770
      DATA      CHKSTR/4HDATA, 4HBASE/, NAMSUB/4HDBLD/                        DBL00780
      OBLDAO = .TRUE.                                                         DBL00790
      IF (.NOT.DXCKOV(OBSDEV,DBSFIL)) GO TO 11111                             DBL00800
         REWINO DBSOEV                                                        DBL00810
         READ (DBSDEV,00001,END=44444) OBFCHK                                 DBL00820
         IF (DBFCHK(1).NE.CHKSTR(1)) GO TO 22222                              DBL00830
         IF (DBFCHK(2).NE.CHKSTR(2)) GO TO 22222                              DBL00840
            CALL DBINIT                                                       DBL00850
            WDSTIT=NCHTIT/DXNCPW                                              DBL00860
            WDSCMT=NCHCMT/DXNCPW                                              DBL00870
C...          +++ LDAD THE DATABASE FROM THE DBSDEV                           DBL00880
            READ (OBSDEV,00002,END=44444) (TITLE(I),I=1,WDSTIT)               DBL00890
            READ (DBSDEV,00003,ENO=44444) NAVAIL,OELCNT,MAXECS,NEXECS,        DBL00900
     +            PRECS,BUCKET                                               DBL00910
            DO 1000 I=1,MAXNDO                                                DBL00920
               READ (DBSDEV,00004,END=44444) IOENT(I,1),IDENT(I,2),           DBL00930
     +              NDDTYP(I),DATUM(I),LINK(I),LINKF(I),CMTPTR(I)             DBL00940
```

```
1000      CONTINUE                                                          DBL00950
             DO 4000 I=1,MAXNOO                                             DBL00960
C...            +++ LOAD THE ARRAYS AND COMMENTS                            DBL00970
C...            +++ LOAD ARRAYS                                             DBL00980
                IF (NODTYP(I) .NE. 3) GO TO 2000                            DBL00990
                   READ (OBSDEV,00005,END=44444) L,(ARRAY(J),J=1,L)         DBL01000
                   IF (.NOT.(DBAPTR(IPOINT,IDUM,1))) GO TO 33333            DBL01010
                      DATUM(I)=IPOINT                                       DBL01020
                      CALL WRITEC(L,IPOINT,1,ECSPTR(1),ECSLEN(1))           DBL01030
                      IF(.NOT.DBAPTR(IPOINT,IDUM,L)) GO TO 33333            DBL01040
                         CALL WRITEC(ARRAY,IPOINT,L,ECSPTR(1),ECSLEN(1))    DBL01050
2000            CONTINUE                                                    DBL01060
C...            +++ LOAD THE COMMENTS                                       DBL01070
                   IF(CMTPTR(I).EQ.0) GO TO 3000                            DBL01080
                      READ (OBSDEV,00006,END=44444) (ARRAY(J),              DBL01090
     +                                              J=1,WDSCMT)             DBL01100
                      IF(.NOT.DBAPTR(IPOINT,IDUM,WDSCMT)) GO TO 33333       DBL01110
                        CALL WRITEC(ARRAY,IPOINT,16,ECSPTR(1),ECSLEN(1))    DBL01120
                        CMTPTR(I)=IPOINT                                    DBL01130
3000                CONTINUE                                                DBL01140
4000          CONTINUE                                                      DBL01150
              RCODE=0                                                       DBL01160
              GO TO 99999                                                   DBL01170
11111 CONTINUE                                                              DBL01180
C...             +++ DATABASE FILE NOT FOUND                                DBL01190
           CALL DXMSGZ(NAMSUB,2)                                            DBL01200
           RCODE=1                                                          DBL01210
           GO TO 99999                                                      DBL01220
22222 CONTINUE                                                              DBL01230
C...             +++ FILE IS NOT A DEX DATABASE                             DBL01240
           CALL DXMSGZ(NAMSUB,3)                                            DBL01250
           RCODE=2                                                          DBL01260
           GO TO 99999                                                      DBL01270
33333 CONTINUE                                                              DBL01280
C...             +++ DIRECT ACCESS STORAGE FULL                             DBL01290
           RCODE=3                                                          DBL01300
           GO TO 99999                                                      DBL01310
44444 CONTINUE                                                              DBL01320
C...             +++ PREMATURE EOF ENCOUNTERED                              DBL01330
           CALL DXMSGZ(NAMSUB,1)                                            DBL01340
           RCODE=4                                                          DBL01350
           GO TO 99999                                                      DBL01360
99999 CONTINUE                                                              DBL01370
           DBLOAD=(RCODE.EQ.0)                                              DBL01380
           RETURN                                                           DBL01390
00001 FORMAT (1X,2A4)                                                       DBL01400
00002 FORMAT (1X,16A4)                                                      DBL01410
```

```
00003 FORMAT (1X,5I10/(16I5))                                    DBLO1420
C+++  THE NEXT TWO FORMATS ARE SITE DEPENDENT                    DBLO1430
00004 FORMAT (2X,2A4,1X,I4,1X,Z8,1X,4I5)                         DBLO1440
00005 FORMAT (2X,Z8,1X,Z8,1X,Z8)                                 DBLO1450
00006 FORMAT (2X,16A4)                                           DBLO1460
      END                                                        DBLO1470
```

```
C                                                                       DBLO1480
C ***********************************************************************DBLO1490
      LDGICAL FUNCTIDN DBSAVE(FILNAM)                                   DBLO1500
C +---------------------------------------------------------------------+DBLO1510
C                                                                       DBLO1520
C        DBSAVE -                                                       DBLO1530
C                                                                       DBLO1540
C             SAVE A DATABASE FDR FUTURE USE.                           DBLO1550
C                                                                       DBLO1560
C +---------------------------------------------------------------------+DBLO1570
C                                                                       DBLO1580
C+++  THIS RDUTINE IS MACHINE DEPENDENT                                 DBLO1590
C+++  THIS RDUTINE IS SITE DEPENDENT                                    DBLO1600
C+++  FILE MANIPULATIDN                                                 DBLO1610
C     +---------------------------------------------------------------+DBLO1620
C...                  *THE CDMMDN CDNTRDL PARAMETERS,CHARACTERS, AND    DBLO1630
C...                  *GRAPHIC DISPLAY CDNTRDL PARAMETERS               DBLO1640
      CDMMDN /DEXCDM/                                                   DBLO1650
     1                DXMDDE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,        DBLO1660
     2                TERSE ,ECHDMD,DXSC  ,CPWDXS,MDSC  ,CURMDD,        DBLO1670
     3                NULMSG,DXNCPW,NAMLEN,                             DBLO1680
     4                CLRMDD,MUNPST,ERASE,UPDATE,UPAINT,GCNTRL          DBLO1690
      INTEGER                                                           DBLO1700
     1                DXSC  ,CPWDXS,MDSC  ,CURMDD,NULMSG,DXNCPW,        DBLO1710
     2                NAMLEN,CLRMDD,MUNPST,ERASE,UPDATE,UPAINT          DBLO1720
      LDGICAL                                                           DBLO1730
     1                DXMDDE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,        DBLO1740
     2                TERSE ,ECHDMD,GCNTRL                              DBLO1750
      DIMENSIDN       DXSC(1),CURMDD(2)  ,NULMSG(1)                     DBLO1760
C...                  *I/D DEVICES AND FILES                            DBLO1770
      CDMMDN /DEXFIL/                                                   DBLO1780
     1                MSGSDV,USEDEV,INFDDV,DBSDEV,NEWSDV,HELPDV,        DBLO1790
     2                DUTDEV,INPDEV,NFWDS ,                             DBLO1800
     3                MSGSFL,USEFIL,INFDFL,DBSFIL,NEWSFL,HELPFL,        DBLO1810
     4                DUTFIL,INPFIL,LDADFL,NDT1FL,NDT2FL                DBLO1820
      INTEGER                                                           DBLO1830
     1                MSGSDV,USEDEV,INFDDV,DBSDEV,NEWSDV,HELPDV,        DBLO1840
     2                DUTDEV,INPDEV,                                    DBLO1850
     3                MSGSFL,USEFIL,INFDFL,DBSFIL,NEWSFL,HELPFL,        DBLO1860
     4                DUTFIL,INPFIL,LDADFL,NDT1FL,NDT2FL,NFWDS          DBLO1870
C...  FILE NAME DIMENSIDNING ADJUSTED FDR CDC AND CU DNLY.....           DBLO1880
      DIMENSIDN       MSGSFL(11)  ,USEFIL(11)  ,INFDFL(11)  ,          DBLO1890
     1                DBSFIL(11)  ,NEWSFL(11)  ,HELPFL(11)  ,          DBLO1900
     3                DUTFIL(11)  ,INPFIL(11)  ,                        DBLO1910
     2                LDADFL(11)  ,NDT1FL(11)  ,NDT2FL(11)              DBLO1920
      CDMMDN /DBCDM/                                                    DBLO1930
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBLO1940
```

```
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,   DBLO1950
     2               INTTYP,RELTYP,ARRTYP,                               DBLO1960
     3               DBCLSD,DBACTV,FILEDB                                DBLO1970
      INTEGER                                                            DBLO1980
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,   DBLO1990
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,   DBLO2000
     2               INTTYP,RELTYP,ARRTYP                                DBLO2010
      LDGICAL                                                           DBLO2020
     3               DBCLSD,DBACTV,FILEDB                                DBLO2030
      DIMENSIDN      TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),      DBLO2040
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)         DBLO2050
      CDMMDN /DBDAID/                                                   DBLO2060
     1               MAXECS,NEXECS,PRECS,ECSPTR,ECSLEN                   DBLO2070
      INTEGER                                                           DBLO2080
     1               PRECS,ECSPTR,ECSLEN                                 DBLO2090
      DIMENSIDN                                                         DBLO2100
     1               ECSPTR(2),ECSLEN(2)                                 DBLO2110
      INTEGER   RCDDE,WDSTIT,WDSCMT,FILNAM,DBFCHK                       DBLO2120
      LDGICAL   DBNARD,CMTGET,LDG                                       DBLO2130
      LDGICAL   DXCKDV                                                  DBLO2140
      DIMENSIDN FILNAM(987),NAME(2),ITEMP(200),TEMP(200),DBFCHK(2)      DBLO2150
      EQUIVALENCE (ITEMP(1),TEMP(1))                                    DBLO2160
      DATA      NAMSUB /4HSAVE/                                         DBLO2170
      DATA      DBFCHK /4HDATA,4HBASE/                                  DBLO2180
C                                                                       DBLO2190
      DBSAVE=.TRUE.                                                     DBLO2200
      WDSTIT=NCHTIT/DXNCPW                                              DBLO2210
      WDSCMT=NCHCMT/DXNCPW                                              DBLO2220
      IF(.NDT.DXCKDV(DBSDEV,FILNAM)) GD TD 11111                        DBLO2230
      REWIND DBSDEV                                                     DBLO2240
C...          +++ SAVE THE DATABASE                                     DBLO2250
      WRITE (DBSDEV,00001) DBFCHK                                       DBLO2260
      WRITE (DBSDEV,00002) (TITLE(I),I=1,WDSTIT)                        DBLO2270
      WRITE (DBSDEV,00003) NAVAIL,DELCNT,MAXECS,NEXECS,                 DBLO2280
     1      PRECS,BUCKET                                                DBLO2290
      DD 1000 I=1,MAXNDD                                                DBLO2300
         WRITE (DBSDEV,00004) IDENT(I,1),IDENT(I,2),NDDTYP(I),          DBLO2310
     +             DATUM(I),LINK(I),LINKF(I),CMTPTR(I)                  DBLO2320
1000  CDNTINUE                                                          DBLO2330
C...          +++ SAVE THE ARRAYS                                       DBLO2340
      DD 4000 I=1,MAXNDD                                                DBLO2350
         IF (NDDTYP(I).NE.3) GD TD 2000                                 DBLO2360
            IPDINT=DATUM(I)                                             DBLO2370
            LDG=DBNARD(IPDINT,TEMP,1,MAXARR,L)                          DBLO2380
            WRITE (DBSDEV,00005) L,(ITEMP(J),J=1,L)                     DBLO2390
2000     CDNTINUE                                                       DBLO2400
C...          +++ DUMP CDMMENTS TD A SAVE DEVICE                        DBLO2410
```

```
            IF (CMTPTR(I).EQ.O) GD TD 3000                                  DBLO2420
               NAME(1)=IDENT(I,1)                                           DBLO2430
               NAME(2)=IDENT(I,2)                                           DBLO2440
               LOG=CMTGET(NAME,ITEMP,RCDDE)                                 DBLO2450
               WRITE (DBSDEV,00006) (ITEMP(J),J=1,WDSCMT)                   DBLO2460
3000        CDNTINUE                                                        DBLO2470
4000  CONTINUE                                                              DBLO2480
      REWIND DBSDEV                                                         DBLO2490
C...             +++ ANNOUNCE THAT DATABASE HAS BEEN SAVED                  DBLO2500
         CALL DXMSGZ(NAMSUB,1)                                              DBLO2510
         GO TD 99999                                                        DBLO2520
11111 CONTINUE                                                              DBLO2530
C...           +++ FAILED TO SAVE DATABASE                                  DBLO2540
         DBSAVE=.FALSE.                                                     DBLO2550
         CALL DXMSGI(NAMSUB,2,IRC)                                          DBLO2560
         GD TD 99999                                                        DBLO2570
99999 CONTINUE                                                              DBLO2580
         RETURN                                                             DBLO2590
00001 FORMAT (1H ,2A4)                                                      DBLO2600
00002 FORMAT (1H ,16A4)                                                     DBLO2610
00003 FDRMAT (1H ,5I10/(16I5))                                              DBLO2620
C...            +++ THE NEXT TWD FORMATS ARE MACHINE  DEPENDENT             DBLO2630
00004 FORMAT (2H *,2A4,1H*,I4,1H*,Z8,1H*,4I5)                               DBLO2640
00005 FORMAT (2H *,Z8,1H*,Z8,1H*,Z8,1H*)                                    DBLO2650
00006 FDRMAT (2H *,16A4,1H*)                                                DBLO2660
      END                                                                   DBLO2670
```

```
C                                                                       DBLO2680
C ********************************************************************* DBLO2690
      INTEGER FUNCTION DBHASH(DNAME)                                    DBLO2700
C +-------------------------------------------------------------------+ DBLO2710
C                                                                       DBLO2720
C        DBHASH -                                                       DBLO2730
C                                                                       DBLO2740
C             HASH DNAME TO A NUMBER BETWEEN 1 AND 32 TO BE USED AS     DBLO2750
C             AN INDEX TO ITS LOCATION IN THE DATABASE.                 DBLO2760
C                                                                       DBLO2770
C +-------------------------------------------------------------------+ DBLO2780
C                                                                       DBLO2790
C+++  THIS ROUTINE IS MACHINE DEPENDENT                                 DBLO2800
C...  THIS ROUTINE USES THE INTERNAL CODE VALUES FOR EACH               DBLO2810
C...  CHARACTER TO SUM THEM AND TRUNCATE THE SUM TO OBTAIN              DBLO2820
C...  VALUES IN THE RANGE OF 1 TO 32.                                   DBLO2830
C...  THE SUMMATION CEASES WHEN THE FIRST BLANK IN THE NAME             DBLO2840
C...  IS ENCOUNTERED.                                                   DBLO2850
C     +-------------------------------------------------------------+   DBLO2860
C                                                                       DBLO2870
      INTEGER DNAME,NAMWRD,IFWORD,ZBLANK                                DBLO2880
      LOGICAL*1 NAMBYT,IFBYTE                                           DBLO2890
      DIMENSION DNAME(2),NAMWRD(2)                                      DBLO2900
      DIMENSION NAMBYT(8),IFBYTE(4)                                     DBLO2910
      EQUIVALENCE (NAMWRD(1),NAMBYT(1)),(IFWORD,IFBYTE(1))              DBLO2920
      DATA ZBLANK /Z00000040/                                           DBLO2930
      NAMWRD(1)=DNAME(1)                                                DBLO2940
      NAMWRD(2)=DNAME(2)                                                DBLO2950
      IFWORD=ZBLANK                                                     DBLO2960
      DBHASH=0                                                          DBLO2970
      DO 1000 I=1,8                                                     DBLO2980
         IFBYTE(4)=NAMBYT(I)                                            DBLO2990
         IF (IFWORD.EQ.ZBLANK) GO TO 2000                               DBLO3000
         DBHASH=DBHASH+IFWORD                                           DBLO3010
 1000 CONTINUE                                                          DBLO3020
 2000 CONTINUE                                                          DBLO3030
         DBHASH=IABS(MOD(DBHASH,32))+1                                  DBLO3040
99999 CONTINUE                                                          DBLO3050
         RETURN                                                         DBLO3060
      END                                                               DBLO3070
```

```
C                                                                       DBLO3080
C ***********************************************************************DBLO3090
      LDGICAL FUNCTIDN DBAPTR(PDINTR,PREPTR,NEED)                       DBLO3100
C +-------------------------------------------------------------------+ DBLO3110
C                                                                       DBLO3120
C        DBAPTR -                                                       DBLO3130
C                                                                       DBLO3140
C             DATABASE ARRAY PDINTER - DBTAINS THE POINTER TO THE       DBLO3150
C             NEXT AVAILABLE LDCATIDN IN THE ECS AREA IN PDINTR.        DBLO3160
C                                                                       DBLO3170
C             PREPTR IS A POINTER TD THE LOCATION OF THE LAST ARRAY     DBLO3180
C             DR CDMMENT ENTERED IN THE STORAGE AREA.                   DBLO3190
C                                                                       DBLO3200
C             NEED IS THE NUMBER DF LDCATIONS WHICH NEED TD BE ENTERED. DBLO3210
C                                                                       DBLO3220
C +-------------------------------------------------------------------+ DBLO3230
C                                                                       DBLO3240
C+++  THIS ROUTINE IS MACHINE DEPENDENT                                 DBLO3250
C+++  THIS RDUTINE IS SITE DEPENDENT                                    DBLO3260
C+++  USES ECS                                                          DBLO3270
C +-------------------------------------------------------------------+ DBLO3280
C                                                                       DBLO3290
      COMMON /DBCDM/                                                    DBLO3300
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBLO3310
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBLO3320
     2               INTTYP,RELTYP,ARRTYP,                              DBLO3330
     3               DBCLSD,DBACTV,FILEDB                               DBLO3340
      INTEGER                                                           DBLO3350
     1               TITLE ,BUCKET,IDENT ,NODTYP,DATUM ,LINK  ,LINKF ,  DBLO3360
     1               CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBLO3370
     2               INTTYP,RELTYP,ARRTYP                               DBLO3380
      LOGICAL                                                           DBLO3390
     3               DBCLSD,DBACTV,FILEDB                               DBLO3400
      DIMENSION      TITLE(16),BUCKET(32),IDENT(200,2),NODTYP(200),     DBLO3410
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)        DBLO3420
      CDMMON /DBDAID/                                                   DBLO3430
     1               MAXECS,NEXECS,PRECS,ECSPTR,ECSLEN                  DBLO3440
      INTEGER                                                           DBLO3450
     1               PRECS,ECSPTR,ECSLEN                                DBLO3460
      DIMENSIDN                                                         DBLO3470
     1               ECSPTR(2),ECSLEN(2)                                DBLO3480
      INTEGER POINTR,PREPTR                                             DBLO3490
      DATA      NAMSUB/4HAPTR/                                          DBLO3500
C...          +++ CHECK THAT HAVEN'T EXCEEDED MAX SIZE                  DBLO3510
      DBAPTR=((NEXECS+NEED-1).LE.MAXECS)                                DBLO3520
      IF (.NDT.DBAPTR) GD TD 88888                                      DBLO3530
C...          +++ SET PDINTER TD NEXT AVAILABLE ECS, AND PREPTR TD THE  DBLO3540
```

```
C...          +++ PREVIOUS ECS LOCATION.                                  DBLO3550
         POINTR=NEXECS                                                    DBLO3560
         PREPTR=PRECS                                                     DBLO3570
C...          +++ UPDATE THE NEXECS AND PRECS FOR NEXT TIME.              DBLO3580
         PRECS=NEXECS                                                     DBLO3590
         NEXECS=NEXECS+NEED                                               DBLO3600
         GO TO 99999                                                      DBLO3610
88888 CONTINUE                                                            DBLO3620
      CALL DXMSGZ(NAMSUB,1)                                               DBLO3630
      DBAPTR=.FALSE.                                                      DBLO3640
      GO TO 99999                                                         DBLO3650
99999 CONTINUE                                                            DBLO3660
         RETURN                                                           DBLO3670
      END                                                                 DBLO3680
```

```
C                                                                       DBLO3690
C ***********************************************************************DBLO3700
      LDGICAL FUNCTIDN DBCPTR(PDINTR,PREPTR,NEED)                       DBLO3710
C +-------------------------------------------------------------------+ DBLO3720
C                                                                       DBLO3730
C        DBCPTR -                                                       DBLO3740
C                                                                       DBLO3750
C             DATABASE CDMMENT PDINTER - OBTAINS THE PDINTER TD THE     DBLO3760
C             NEXT AVAILABLE LDCATIDN IN THE ECS AREA IN PDINTR.        DBLO3770
C                                                                       DBLO3780
C             PREPTR IS A PDINTER TD THE LOCATIDN OF THE LAST ARRAY     DBLO3790
C             DR CDMMENT ENTERED IN THE STORAGE AREA.                   DBLO3800
C                                                                       DBLO3810
C             NEED IS THE NUMBER OF LDCATIDNS WHICH NEED TD BE ENTERED. DBLO3820
C                                                                       DBLO3830
C +-------------------------------------------------------------------+ DBLO3840
C                                                                       DBLO3850
C+++  THIS ROUTINE IS MACHINE DEPENDENT                                 DBLO3860
C+++  THIS RDUTINE IS SITE DEPENDENT                                    DBLO3870
C+++  USES ECS                                                          DBLO3880
      INTEGER  PDINTR,PREPTR                                            DBLO3890
      LDGICAL  DBAPTR                                                   DBLO3900
      DBCPTR=DBAPTR(PDINTR,PREPTR,NEED)                                 DBLO3910
99999 CONTINUE                                                          DBLO3920
         RETURN                                                         DBLO3930
      END                                                               DBLO3940
```

```
C                                                                       DBL03950
C ***********************************************************************DBL03960
      LDGICAL FUNCTIDN DBNAIN(NEW,NAME,POINTR,RARRAY,START,NPUT,        DBL03970
     1                        NSTORD)                                   DBL03980
C +-------------------------------------------------------------------+ DBL03990
C                                                                       DBL04000
C        DBNAIN -                                                       DBL04010
C                                                                       DBL04020
C             DATABASE NODE ARRAY IN - STDRES THE VALUES DF THE ARRAY   DBL04030
C             RARRAY INTD THE ECS STDRAGE AREA FOR VARIABLE NAME.       DBL04040
C                                                                       DBL04050
C +-------------------------------------------------------------------+ DBL04060
C                                                                       DBL04070
C+++  THIS RDUTINE IS MACHINE DEPENDENT                                 DBL04080
C+++  THIS RDUTINE IS SITE DEPENDENT                                    DBL04090
C+++  USES ECS                                                          DBL04100
C +-------------------------------------------------------------------+ DBL04110
C                                                                       DBL04120
      CDMMDN /DBDAID/                                                   DBL04130
     1               MAXECS,NEXECS,PRECS,ECSPTR,ECSLEN                  DBL04140
      INTEGER                                                           DBL04150
     1               PRECS,ECSPTR,ECSLEN                                DBL04160
      DIMENSIDN                                                         DBL04170
     1               ECSPTR(2),ECSLEN(2)                                DBL04180
      INTEGER PDINTR,START,NPUT,NSTDRD,PREPTR,DBNSKR                    DBL04190
      LDGICAL NEW,DBAPTR                                                DBL04200
      DIMENSION RARRAY(NPUT),NAME(2)                                    DBL04210
      IF (NEW) GD TD 1000                                               DBL04220
C...        +++ ARRAY IS ALREADY STDRED USE SAME AREA.                  DBL04230
         CALL READEC(NSTORD,POINTR,1,ECSPTR(1),ECSLEN(1))               DBL04240
         DBNAIN=(NPUT.EQ.NSTDRD)                                        DBL04250
         IF (.NOT.(DBNAIN)) GO TD 99999                                 DBL04260
            CALL WRITEC(RARRAY,POINTR+3,MINO(NPUT,NSTDRD),ECSPTR(1),    DBL04270
     1                  ECSLEN(1))                                      DBL04280
         GD TO 99999                                                    DBL04290
1000  CDNTINUE                                                          DBL04300
      NSTDRD=NPUT                                                       DBL04310
C...        +++ GET SPACE FOR ARRAY ITS LENGTH AND POINTER INFDRMATIDN  DBL04320
            DBNAIN=DBAPTR(POINTR,PREPTR,NSTDRD+3)                       DBL04330
            IF (.NOT.(DBNAIN)) GO TD 99999                              DBL04340
C...        +++ PUT IN THE NUMBER STORED                                DBL04350
               CALL WRITEC(NSTORD,PDINTR,1,ECSPTR(1),ECSLEN(1))         DBL04360
C...        +++ PUT IN A PDINTER TO THE PREVIDUS ARRAY OR COMMENT STREDDBL04370
               CALL WRITEC(PREPTR,POINTR+1,1,ECSPTR(1),ECSLEN(1))       DBL04380
C...        +++ PUT IN A PDINTER BACK TO THE NDDE                       DBL04390
               NODE=DBNSKR(NAME,IDUM1,IDUM2,IDUM3)                      DBL04400
               CALL WRITEC(NDDE,PDINTR+2,1,ECSPTR(1),ECSLEN(1))         DBL04410
```

```
C...          +++ PUT IN THE ARRAY VALUES                                        DBLO4420
                 CALL WRITEC(RARRAY,POINTR+3,MINO(NPUT,NSTORD),                 DBLO4430
     1                       ECSPTR(1),ECSLEN(1))                               DBLO4440
                 IF (NPUT.EQ.NSTORD) GO TO 99999                                DBLO4450
                    IBEG=NPUT+1                                                 DBLO4460
                    DO 2000 I=IBEG,NSTORD                                       DBLO4470
                       CALL WRITEC(0.0,POINTR+3+I,1,ECSPTR(1),ECSLEN(1))        DBLO4480
2000                 CONTINUE                                                   DBLO4490
                    GO TO 99999                                                 DBLO4500
99999 CONTINUE                                                                  DBLO4510
         RETURN                                                                 DBLO4520
       END                                                                      DBLO4530
```

```
C                                                                               DBL04540
C ******************************************************************************DBL04550
      LDGICAL FUNCTIDN DBNARD(PDINTR,RARRAY,START,NGET,NSTDRD)                  DBL04560
C +--------------------------------------------------------------------+        DBL04570
C                                                                               DBL04580
C        DBNARD -                                                               DBL04590
C                                                                               DBL04600
C             DATABASE NDDE ARRAY READ - READS THE CDNTENTS DF THE              DBL04610
C             ARRAY PDINTED TD BY PDINTR FRDM THE ECS STDRAGE AREA              DBL04620
C             INTD RARRAY.                                                      DBL04630
C                                                                               DBL04640
C +--------------------------------------------------------------------+        DBL04650
C                                                                               DBL04660
C+++  THIS RDUTINE IS MACHINE DEPENDENT                                         DBL04670
C+++  THIS RDUTINE IS SITE DEPENDENT                                            DBL04680
C+++  USES ECS                                                                  DBL04690
      CDMMDN /DBDAID/                                                           DBL04700
     1               MAXECS,NEXECS,PRECS,ECSPTR,ECSLEN                          DBL04710
      INTEGER                                                                   DBL04720
     1               PRECS,ECSPTR,ECSLEN                                        DBL04730
      DIMENSIDN                                                                 DBL04740
     1               ECSPTR(2),ECSLEN(2)                                        DBL04750
      INTEGER PDINTR,START,NGET,NSTDRD                                          DBL04760
      DIMENSIDN RARRAY(987)                                                     DBL04770
C...          +++ READ THE NUMBER STDRED                                        DBL04780
      CALL READEC(NSTDRD,PDINTR,1,ECSPTR(1),ECSLEN(1))                          DBL04790
C...          +++ READ THE ARRAY ELEMENTS                                       DBL04800
      CALL READEC(RARRAY,PDINTR+3,MINO(NSTDRD,NGET),ECSPTR(1),ECSLEN(1))DBL04810
      IF (NSTDRD.GE.NGET) GD TD 2000                                            DBL04820
         IB=NSTDRD+1                                                            DBL04830
         DD 1000 I=IB,NGET                                                      DBL04840
            RARRAY(I)=O.                                                        DBL04850
1000     CDNTINUE                                                               DBL04860
2000  CDNTINUE                                                                  DBL04870
      DBNARD=.TRUE.                                                             DBL04880
      GD TD 99999                                                               DBL04890
99999 CDNTINUE                                                                  DBL04900
         RETURN                                                                 DBL04910
      END                                                                       DBL04920
```

```
C                                                                       DBL04930
C ***********************************************************************DBL04940
      LDGICAL FUNCTIDN DBNCRD(PDINTR,CMTSTR)                            DBL04950
C +---------------------------------------------------------------------+DBL04960
C                                                                       DBL04970
C        DBNCRD -                                                       DBL04980
C                                                                       DBL04990
C             DATABASE NDDE CDMMENT READ - READS THE CDMMENT PDINTED    DBL05000
C             TD BY PDINTR FRDM THE ECS STDRAGE AREA INTD CMTSTR.       DBL05010
C                                                                       DBL05020
C +---------------------------------------------------------------------+DBL05030
C                                                                       DBL05040
C+++  THIS RDUTINE IS MACHINE DEPENDENT                                 DBL05050
C+++  THIS RDUTINE IS SITE DEPENDENT                                    DBL05060
C+++  USES ECS                                                          DBL05070
C +---------------------------------------------------------------------+DBL05080
C                                                                       DBL05090
C...                  *THE CDMMDN CDNTRDL PARAMETERS,CHARACTERS, AND    DBL05100
C...                  *GRAPHIC DISPLAY CDNTRDL PARAMETERS               DBL05110
      CDMMDN /DEXCDM/                                                   DBL05120
     1                DXMDDE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,        DBL05130
     2                TERSE ,ECHDMD,DXSC  ,CPWDXS,MDSC  ,CURMDD,        DBL05140
     3                NULMSG,DXNCPW,NAMLEN,                             DBL05150
     4                CLRMDD,MUNPST,ERASE,UPDATE,UPAINT,GCNTRL          DBL05160
      INTEGER                                                           DBL05170
     1                DXSC  ,CPWDXS,MDSC  ,CURMDD,NULMSG,DXNCPW,        DBL05180
     2                NAMLEN,CLRMDD,MUNPST,ERASE,UPDATE,UPAINT          DBL05190
      LDGICAL                                                           DBL05200
     1                DXMDDE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,        DBL05210
     2                TERSE ,ECHDMD,GCNTRL                              DBL05220
      DIMENSIDN       DXSC(1),CURMDD(2)  ,NULMSG(1)                     DBL05230
      CDMMDN /DBCDM/                                                    DBL05240
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBL05250
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBL05260
     2               INTTYP,RELTYP,ARRTYP,                              DBL05270
     3               DBCLSD,DBACTV,FILEDB                               DBL05280
      INTEGER                                                           DBL05290
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBL05300
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBL05310
     2               INTTYP,RELTYP,ARRTYP                               DBL05320
      LDGICAL                                                           DBL05330
     3               DBCLSD,DBACTV,FILEDB                               DBL05340
      DIMENSIDN      TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),     DBL05350
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)        DBL05360
      CDMMDN /DBDAID/                                                   DBL05370
     1               MAXECS,NEXECS,PRECS,ECSPTR,ECSLEN                  DBL05380
      INTEGER                                                           DBL05390
```

```
     1               PRECS,ECSPTR,ECSLEN                                    DBLO5400
      DIMENSION                                                             DBLO5410
     1               ECSPTR(2),ECSLEN(2)                                    DBLO5420
      INTEGER   POINTR,CMTSTR,CMTWDS                                        DBLO5430
      DIMENSION CMTSTR(987)                                                 DBLO5440
      CMTWDS=NCHCMT/DXNCPW                                                  DBLO5450
      DBNCRD=(POINTR.GT.0)                                                  DBLO5460
C...          +++ READ THE COMMENT                                          DBLO5470
      IF (DBNCRD) CALL READEC(CMTSTR,POINTR+2,CMTWDS,ECSPTR(1),             DBLO5480
     1                        ECSLEN(1))                                    DBLO5490
99999 CONTINUE                                                              DBLO5500
         RETURN                                                             DBLO5510
      END                                                                   DBLO5520
```

```
C                                                                       DBLO5530
C ***********************************************************************DBLO5540
      LDGICAL FUNCTIDN DBNCIN(PDINTR,CMTSTR,NDDE)                       DBLO5550
C +---------------------------------------------------------------------+DBLO5560
C                                                                       DBLO5570
C        DBNCIN -                                                       DBLO5580
C                                                                       DBLO5590
C             DATABASE NDDE CDMMENT IN - ENTERS THE CDMMENT CDNTAINED   DBLO5600
C             IN CMTSTR INTD THE ECS STDRAGE AREA AND RETURNS THE       DBLO5610
C             PDINTER TD ITS LDCATIDN IN PDINTR.  THE CDMMENT IS        DBLO5620
C             LINKED BACK TD ITS NDDE.                                  DBLO5630
C                                                                       DBLO5640
C +---------------------------------------------------------------------+DBLO5650
C                                                                       DBLO5660
C+++  THIS RDUTINE IS MACHINE DEPENDENT                                 DBLO5670
C+++  THIS RDUTINE IS SITE DEPENDENT                                    DBLO5680
C+++  USES ECS                                                          DBLO5690
C +---------------------------------------------------------------------+DBLO5700
C                                                                       DBLO5710
C...                  *THE CDMMDN CDNTRDL PARAMETERS,CHARACTERS, AND    DBLO5720
C...                  *GRAPHIC DISPLAY CDNTRDL PARAMETERS               DBLO5730
      CDMMDN /DEXCDM/                                                   DBLO5740
     1                DXMDDE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,        DBLO5750
     2                TERSE ,ECHDMD,DXSC  ,CPWDXS,MDSC  ,CURMDD,        DBLO5760
     3                NULMSG,DXNCPW,NAMLEN,                             DBLO5770
     4                CLRMDD,MUNPST,ERASE,UPDATE,UPAINT,GCNTRL          DBLO5780
      INTEGER                                                           DBLO5790
     1                DXSC  ,CPWDXS,MDSC  ,CURMDD,NULMSG,DXNCPW,        DBLO5800
     2                NAMLEN,CLRMDD,MUNPST,ERASE,UPDATE,UPAINT          DBLO5810
      LDGICAL                                                           DBLO5820
     1                DXMDDE,LDADED,MDPEND,DXREQS,DLREQS,KEYBRD,        DBLO5830
     2                TERSE ,ECHDMD,GCNTRL                              DBLO5840
      DIMENSIDN       DXSC(1),CURMDD(2)  ,NULMSG(1)                     DBLO5850
      CDMMDN /DBCDM/                                                    DBLO5860
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBLO5870
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBLO5880
     2               INTTYP,RELTYP,ARRTYP,                              DBLO5890
     3               DBCLSD,DBACTV,FILEDB                               DBLO5900
      INTEGER                                                           DBLO5910
     1               TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,  DBLO5920
     1               CMTPTR,NAVAIL,MAXNDD,MAXARR,NCHTIT,NCHCMT,DELCNT,  DBLO5930
     2               INTTYP,RELTYP,ARRTYP                               DBLO5940
      LOGICAL                                                           DBLO5950
     3               DBCLSD,DBACTV,FILEDB                               DBLO5960
      DIMENSIDN       TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),    DBLO5970
     1               DATUM(200),LINK(200),LINKF(200),CMTPTR(200)        DBLO5980
      CDMMDN /DBDAID/                                                   DBLO5990
```

```
     1               MAXECS,NEXECS,PRECS,ECSPTR,ECSLEN                  OBL06000
                                                                        OBL06010
      INTEGER
     1               PRECS,ECSPTR,ECSLEN                                DBL06020
                                                                        OBL06030
      OIMENSION
     1               ECSPTR(2),ECSLEN(2)                                OBL06040
      INTEGER   POINTR,CMTSTR,CMTWDS,PREPTR                             OBL06050
      LOGICAL   DBCPTR                                                  OBL06060
      DIMENSION CMTSTR(987)                                             OBL06070
      CMTWOS=NCHCMT/OXNCPW                                              DBL06080
      DBNCIN=(POINTR.GT.O)                                              OBL06090
C...          +++ IF ALREADY HAVE A POINTER DON'T GET ANOTHER           DBL06100
C...          +++ OTHERWISE GET THE POINTER TO THE NEXT AVAILABLE ECS   OBL06110
      IF (.NOT.(DBNCIN)) DBNCIN=OBCPTR(POINTR,PREPTR,CMTWDS+2)          OBL06120
      IF (.NOT.(OBNCIN)) GO TO 99999                                    OBL06130
C...          +++ PUT IN A POINTER BACK TO PREVIOUS COMMENT OR ARRAY    OBL06140
C...          +++ LOCATION IN THE ECS AREA.  PUT IN A POINTER BACK TO   OBL06150
C...          +++ THE NOOE, AND THEN PUT IN THE COMMENT.                OBL06160
         CALL WRITEC(PREPTR,POINTR,1,ECSPTR(1),ECSLEN(1))               OBL06170
         CALL WRITEC(NOOE,POINTR+1,1,ECSPTR(1),ECSLEN(1))               OBL06180
         CALL WRITEC(CMTSTR,POINTR+2,CMTWDS,ECSPTR(1),ECSLEN(1))        OBL06190
99999 CONTINUE                                                          OBL06200
         RETURN                                                         OBL06210
      ENO                                                               OBL06220
```

```
C*************************************************************************DBX00010
      LOGICAL FUNCTION DBXECS(IDUM)                                       DBX00020
C +-------------------------------------------------------------------+   DBX00030
C                                                                         DBX00040
C        DBXECS -                                                         DBX00050
C                                                                         DBX00060
C             DATABASE EXPAND ECS - EXPANDS THE DATABASE SIMULATED ECS    DBX00070
C             STORAGE AREA BY INCREMENTS OF 2000.                         DBX00080
C                                                                         DBX00090
C +-------------------------------------------------------------------+   DBX00100
C                                                                         DBX00110
      COMMON /DBDAIO/                                                     DBX00120
     1               MAXECS,NEXECS,PRECS,ECSPTR,ECSLEN                    DBX00130
      INTEGER                                                             DBX00140
     1               PRECS,ECSPTR,ECSLEN                                  DBX00150
      DIMENSION                                                           DBX00160
     1               ECSPTR(2),ECSLEN(2)                                  DBX00170
      LOGICAL        GETMN,FREEMN,LOG                                     DBX00180
      INTEGER        RFCODE                                               DBX00190
      COMMON /CODE/ RFCODE                                                DBX00200
C                                                                         DBX00210
C             +++ SAVE THE OLD MAXECS AND INCREMENT MAXECS BY 2000        DBX00220
C                                                                         DBX00230
      ECSLEN(1)=MAXECS                                                    DBX00240
      MAXECS=MAXECS+2000                                                  DBX00250
      ECSLEN(2)=MAXECS                                                    DBX00260
C                                                                         DBX00270
C             +++ GET STORAGE VIA GETMAIN AND COPY THE DATABASE ARRAYS    DBX00280
C                                                                         DBX00290
      DBXECS=GETMN(MAXECS,ECSPTR(2))                                      DBX00300
      IF (.NOT. DBXECS) GO TO 11111                                       DBX00310
         KLEN = ECSLEN (1)                                                DBX00320
      DO 1000 I=1,KLEN                                                    DBX00330
         CALL READEC(RARRAY,I,1,ECSPTR(1),ECSLEN(1))                      DBX00340
         CALL WRITEC(RARRAY,I,1,ECSPTR(2),ECSLEN(2))                      DBX00350
 1000 CONTINUE                                                            DBX00360
C                                                                         DBX00370
C             +++ FREE THE ORIGINAL STORAGE AREA VIA FREEMAIN             DBX00380
C                                                                         DBX00390
      LOG=FREEMN(ECSLEN(1),ECSPTR(1),RFCODE)                              DBX00400
      ECSPTR(1)=ECSPTR(2)                                                 DBX00410
      ECSLEN(1)=ECSLEN(2)                                                 DBX00420
      GO TO 99999                                                         DBX00430
11111 MAXECS = MAXECS - 2000                                              DBX00440
99999 RETURN                                                              DBX00450
      END                                                                 DBX00460
```

```
C*****************************************************************OXA00010
      SUBROUTINE DXABND                                            OXA00020
C +-------------------------------------------------------------+  OXA00030
C                                                                  OXA00040
C        OXABND -                                                  OXA00050
C                                                                  DXA00060
C             DEX ABNORMAL ENO - WHEN AN ABNORMAL ENO OCCURS THE   OXA00070
C             ABENO CONOITION IS TRAPPEO BY THE ASSEMBLY LANGUAGE  OXA00080
C             ROUTINE ABTRAP, ANO CONTROL IS PASSEO TO THIS ROUTINE OXA00090
C             UPON ENTRY THE VALUE OF RFCODE INOICATES THE ROUTINE DXA00100
C             WHICH CAUSEO THE ABNORMAL END.                       OXA00110
C                                                                  DXA00120
C             THE CONOITION IS ANNOUNCEO TO THE USER.              OXA00130
C                                                                  OXA00140
C             THE USER IS GIVEN THE OPTION OF SAVING ANY OPEN      OXA00150
C             OATABASE.                                            DXA00160
C                                                                  OXA00170
C             THE USER IS THEN GIVEN THE OPTION OF CONTINUING OEX  DXA00180
C             OPERATION, OR STOPING.  THE LATTER IS RECOMMENOEO.   DXA00190
C                                                                  OXA00200
C +-------------------------------------------------------------+  OXA00210
C                                                                  OXA00220
C+++  THIS ROUTINE IS MACHINE OEPENDENT                            OXA00230
C+++  THIS ROUTINE IS SITE DEPENOENT                               DXA00240
C +-------------------------------------------------------------+  OXA00250
C                                                                  OXA00260
C...                  *I/O DEVICES ANO FILES                       DXA00270
      COMMON /OEXFIL/                                              OXA00280
     1                MSGSOV,USEOEV,INFOOV,OBSOEV,NEWSDV,HELPDV,   OXA00290
     2                OUTOEV,INPOEV,NFWOS ,                        DXA00300
     3                MSGSFL,USEFIL,INFOFL,OBSFIL,NEWSFL,HELPFL,   OXA00310
     4                OUTFIL,INPFIL,LOAOFL,NOT1FL,NOT2FL           OXA00320
      INTEGER                                                      OXA00330
     1                MSGSOV,USEOEV,INFOOV,DBSOEV,NEWSOV,HELPDV,   OXA00340
     2                OUTOEV,INPOEV,                               OXA00350
     3                MSGSFL,USEFIL,INFOFL,OBSFIL,NEWSFL,HELPFL,   OXA00360
     4                OUTFIL,INPFIL,LOAOFL,NOT1FL,NOT2FL,NFWDS     OXA00370
C...  FILE NAME OIMENSIONING ADJUSTED FOR COC ANO CU ONLY.....     OXA00380
      OIMENSION       MSGSFL(11)  ,USEFIL(11)  ,INFOFL(11)  ,      DXA00390
     1                OBSFIL(11)  ,NEWSFL(11)  ,HELPFL(11)  ,      DXA00400
     3                OUTFIL(11)  ,INPFIL(11)  ,                   OXA00410
     2                LOAOFL(11)  ,NOT1FL(11)  ,NOT2FL(11)         OXA00420
C...  OATABASE COMMON                                              OXA00430
      COMMON /DBCOM/                                               OXA00440
     1               TITLE ,BUCKET,IOENT ,NOOTYP,OATUM ,LINK  ,LINKF ,  OXA00450
     1               CMTPTR,NAVAIL,MAXNOO,MAXARR,NCHTIT,NCHCMT,OELCNT,  OXA00460
     2               INTTYP,RELTYP,ARRTYP,                         OXA00470
```

```
     3                DBCLSD,DBACTV,FILEDB                                    DXA00480
      INTEGER                                                                 DXA00490
     1                TITLE ,BUCKET,IDENT ,NDDTYP,DATUM ,LINK  ,LINKF ,       DXA00500
     1                CMTPTR,NAVAIL,MAXNOD,MAXARR,NCHTIT,NCHCMT,DELCNT,       DXA00510
     2                INTTYP,RELTYP,ARRTYP                                    DXA00520
      LOGICAL                                                                 DXA00530
     3                DBCLSD,DBACTV,FILEDB                                    DXA00540
      DIMENSIDN       TITLE(16),BUCKET(32),IDENT(200,2),NDDTYP(200),          DXA00550
     1                DATUM(200),LINK(200),LINKF(200),CMTPTR(200)             DXA00560
      COMMON /CDDE/ RFCDDE                                                    DXA00570
      LOGICAL SAVEFL,DBSAVE                                                   DXA00580
      INTEGER FILENM                                                          DXA00590
      DIMENSIDN       FILENM(5)                                               DXA00600
      DATA       FILENM/4H    ,4H    ,4H DAT,4HABAS,4HE A /                  DXA00610
      DATA       NAMSUB/4HABND/                                               DXA00620
C                                                                             DXA00630
C...  ANNOUNCE THAT AN ABNORMAL TERMINATIDN HAS DCCURED.                      DXA00640
C...  IF RFCDDE = 1 THEN ANNOUNCE THAT THE ERRDR WAS IN ROUTINE LOAD         DXA00650
C...  WHILE LDADING A MODULE.                                                 DXA00660
C...  IF RFCDDE = 2 THEN ANNOUNCE THAT THE ERRDR WAS IN RDUTINE START         DXA00670
C...  WHILE STARTING A MODULE.                                               DXA00680
C...  IF RFCDDE = 3 THEN ANNOUNCE THAT THE ERRDR WAS IN RDUTINE FREEMN.      DXA00690
C...  IF RFCODE = 4 THEN ANNOUNCE THAT THE ERRDR WAS IN THE MDDULE.          DXA00700
C                                                                             DXA00710
      WRITE (DUTDEV,00001)                                                    DXA00720
      IF (RFCDDE .EQ. 1) WRITE (OUTDEV,00002)                                 DXA00730
      IF (RFCDDE .EQ. 2) WRITE (DUTDEV,00003)                                 DXA00740
      IF (RFCODE .EQ. 3) WRITE (DUTDEV,00004)                                 DXA00750
      IF (RFCODE .EQ. 4) WRITE (OUTDEV,00005)                                 DXA00760
      IF (.NDT.DBCLSD) GD TD 77777                                            DXA00770
C                                                                             DXA00780
C        IF THERE WAS AN DPEN DATABASE SAVE IT                                DXA00790
C                                                                             DXA00800
      WRITE (DUTDEV,00006)                                                    DXA00810
      READ (INPDEV,00007) IANS                                                DXA00820
      IF (IANS .EQ. 0) GD TD 88888                                            DXA00830
      WRITE (OUTDEV,00008)                                                    DXA00840
      READ (INPDEV,00009) (FILENM(I),I=1,2)                                   DXA00850
      SAVEFL=DBSAVE(FILENM)                                                   DXA00860
      IF (SAVEFL) WRITE (DUTDEV,00010)                                        DXA00870
      IF (.NDT. SAVEFL) WRITE (DUTDEV,00011)                                  DXA00880
      GO TD 88888                                                             DXA00890
C                                                                             DXA00900
C        ANNOUNCE THAT THERE IS ND DPEN DATABASE                              DXA00910
C                                                                             DXA00920
77777 CONTINUE                                                                DXA00930
      WRITE (DUTDEV,00012)                                                    DXA00940
```

```
C                                                                       DXA00950
C        CHECK IF THE USER WANTS TO RETURN TO DEX OR STOP               DXA00960
C                                                                       DXA00970
88888 CONTINUE                                                          DXA00980
      WRITE (OUTDEV,00013)                                              DXA00990
      READ (INPDEV,00007) IANS                                          DXA01000
      IF (IANS .EQ. 0) GO TO 99999                                      DXA01010
         CALL DXMAJC                                                    DXA01020
         STOP                                                           DXA01030
99998 RETURN                                                            DXA01040
C                                                                       DXA01050
C        THATS IT THE USER CHOSE TO STOP                                DXA01060
C                                                                       DXA01070
99999 CONTINUE                                                          DXA01080
         STOP                                                           DXA01090
C                                                                       DXA01100
C        FORMAT STATEMENTS                                              DXA01110
C                                                                       DXA01120
00001 FORMAT (1X,'AN ABNORMAL TERMINATION OF EXECUTION HAS OCCURED.',/, DXA01130
     1           'THE SYSTEM ABEND HAS BEEN INTERCEPTED BY DEX.',//)    DXA01140
00002 FORMAT (1X,'THE ABEND WAS CAUSED IN ROUTINE LOAD WHILE LOADING',/ DXA01150
     1           'A MODULE PROGRAM.'/)                                  DXA01160
00003 FORMAT (1X,'THE ABEND WAS CAUSED IN ROUTINE START WHILE STARTING',DXA01170
     1         /,'A MODULE PROGRAM.'/)                                  DXA01180
00004 FORMAT (1X,'THE ABEND WAS CAUSED IN ROUTINE FREEMN',/)            DXA01190
00005 FORMAT (1X,'THE ABEND WAS CAUSED BY THE EXECUTING MODULE',/)      DXA01200
00006 FORMAT (1X,'DEX IS NOW UNPREDICTABLE.  DO YOU WANT TO SAVE YOUR',/DXA01210
     1           'OPEN DATABASE. (ENTER 1-YES OR 0-NO)',/)              DXA01220
00007 FORMAT (I1)                                                       DXA01230
00008 FORMAT (1X,'ENTER AN 8 CHARACTER FILENAME FOR YOUR DATABASE.',/,  DXA01240
     1           'A FILETYPE OF DATABASE IS ASSUMED.',//)               DXA01250
00009 FORMAT (2A4)                                                      DXA01260
00010 FORMAT (1X,'YOUR DATABASE HAS BEEN SAVED.',/)                     DXA01270
00011 FORMAT (1X,'UNABLE TO SAVE YOUR DATABASE.',/)                     DXA01280
00012 FORMAT (1X,'THERE IS NO OPEN DATABASE.',/)                        DXA01290
00013 FORMAT (1X,'SINCE THE SYSTEM AND POSSIBLY DEX HAVE BEEN HAMMERED',DXA01300
     1         /,'FURTHER EXECUTION WILL HAVE UNPREDICTABLE RESULTS.',/,DXA01310
     2           'WE RECOMMEND THAT YOU STOP AND RESTART DEX.  DO YOU',/DXA01320
     3           'WISH TO CONTINUE WITH DEX OR STOP?',/,                DXA01330
     4           '(ENTER 1 TO CONTINUE, OR 0 TO STOP)',/)               DXA01340
      END                                                               DXA01350
```

Thesis
S73875
c.1

Stone

199470

A further development of the Massachusetts Institute of Technology computer aided design executive system.