RESOURCE AND PATIENT MANAGEMENT SYSTEM

(RPMS)

IHS/VA UTILITIES

TECHNICAL MANUAL

XB/ZIB

Version 3.0

February, 1997

Office of Information Resource Management
Indian Health Service
Albuquerque, New Mexico

PREFACE

This document is designed primarily for RPMS application programmers. Area and site IRM personnel can find this document helpful in understanding how the XB/ZIB utility routines operate.

TABLE OF CONTENTS

1. INTRODUCTION 1-1

2. FACILITY PARAMETERS 2-1

3. AREA OFFICE PARAMETERS 3-1

4. SECURITY KEYS 4-1

5. OPTIONS 5-1

6. FIELDS IN FILES 6-1

7. ARCHIVING AND PURGING 7-1

8. CALLABLE ROUTINES 8-1

9. EXTERNAL RELATIONS 9-1

10. INTERNAL RELATIONS 10-1

11. HOW TO GENERATE ON-LINE DOCUMENTATION 11-1

12. GLOSSARY 12-1

13. SYSTEM REQUIREMENTS 13-1

14. KILL OF UNSUBSCRIPTED GLOBALS 14-1

1. INTRODUCTION

The IHS/VA UTILITIES are in the XB namespace, for routines that are not MUMPS implementation specific. Routines that are implementation specific will be in the ZIB namespace.

Programmer tools are available from programmer mode through the menu-driver routine XB.

There are no files associated with this package.

To aid in your reading the routines, if required, the following style guidelines have been followed in most of the routines:

1) all NEW and KILL commands are not abbreviated;
2) only one command scope per line;
3) unconditional GOs/QUITs are followed by a comment line;
4) lines with labels have no executable code.

2. FACILITY PARAMETERS

There are no facility parameters for this application.

3. AREA OFFICE PARAMETERS

There are no Area Office parameters for this package.

4. SECURITY KEYS

5. OPTIONS

There are no options distributed with the package.

There is one option associated with the Remote Patch Installer (ZIBRPI), which is used to schedule the task. That option is installed by ZIBRPI when the local facility installs it.

If you have Remote Error Reporting (ZIBRER) installed, there will be options in that namespace.

Any other XB or ZIB listed option will have been created on your local machine.

XBHFMAN ** no parents **
_____
TYPE: run routine
TEXT: Print a Help Frame Manual DESCRIPTION:

XBPKGSRV ** no parents ** _____
TYPE: server
TEXT: XB PACKAGE SERVER
DESCRIPTION : THIS IS THE PACKAGE SERVER OPTION

ZIB REMOTE PATCH INSTALLATION ** no parents **
_____
TYPE: run routine
TEXT: Remote Patch Installation
ENTRY ACTION: I 2 S:0 %="/usr/spool/uucppublic^1"
DESCRIPTION:

6. FIELDS IN FILES

No files are distributed with this package. Any fields listed, below, will have been created locally. The list will be an alphabetical list of fields in the package's files.

These are the files in the package:

These are the alphabetized fields in the files:

7. ARCHIVING AND PURGING

At the present time there are no archiving and/or purging capabilities with this package.

8. CALLABLE ROUTINES

These are the routine descriptions, which are usually contained in the commented lines prior to the first label or executable line.

Each line label is also listed. The internally documented entry points (" ;EP") are listed.

Routines and sub-routines in namespace:

XB - UTILITY MENU

SEE ROUTINE XB1 FOR FURTHER DOCUMENTATION AND THE MENU OPTIONS.

This routine lists available utilities in the form of a menu with a brief description of what the utility does. New utilities may be added to this routine by adding the appropriate ";;" entries to the bottom of this routine.

START ;
MENU ;
HELP ;
LIST ; List menu options.
LETTERS ;
OPTION ;
CALL ;
RECURSE ;
TRAP ; ERROR TRAP
PAUSE ;EP
CHECK ; CHECK XB OPTION ROUTINES (EXECUTED FROM ^XB MENU OPTIO
RCHK ;EP - Check Existence of Routine in X
EOJ ;
OSNO ;EP

XB1 - XB MENUS AND DOCUMENTATION

Each label represents a menu. The label must begin with "M" and be followed by 1 or more 1 digit numbers 1-9.

Each digit represents the link # of the parent option. E.g., 'M1' is the label for submenu options for the main menu option with a link # of 1. 'M11' would be the label for subsubmenu options for the option in 'M1' with a link # of 1. Within a menu options are listed and selected positionally. The only purpose of the link # is to link options to their parent menu.

Example:

```
M1  ;;FILES/DICTIONARIES
;;Submenu example one;;1
;;Submenu example two;;2
M11 ;;SUBMENU ONE
;;Submenuoption;;^ROUTINEX
M12 ;;SUBMENU TWO
;;Submenuoption;;^ROUTINEY
```

This label naming technique allows the menu tree to go to seven levels. No more than nine options on one menu may also be menus.

For menu options the 2nd ";;" piece is the title, the 3rd ";;" piece must be a number if the option is a submenu, a valid routine or label^routine or executable code if the 3rd piece begins with a !. The code following the ! will be placed in a variable and the variable will be executed. A P in the 4th ";;" piece indicates pause after execution.

```
M ;;MAIN XB UTILITY MENU
M1 ;;FILES/DICTIONARIES
M2 ;;GLOBALS
M3 ;;ROUTINES
M31 ;;LIST ROUTINES
M4 ;;MISCELLANEOUS
M5 ;;DEVELOPERS
```

XBARRAY - BUILD AN ARRAY

This utility provides a word processing format of free text and local variable references to build an array. A file is necessary that has a .01 field for the form name and a WP field to hold the WP form. Please refer to routine XBFORM0 for documentation.

```
GEN(XBFORM,XBWPDIC,XBWPFLD,XBREF,XBFMT,XBLAST) ;EP  **  generate
EDIT(XBFORM,XBWPDIC,XBWPFLD) ;EP  Edit a Form
EDIT2 ;
EDITWP ;** edit WP array WPGET ;** get WP array
BUILD ;** scan WP array to build XBL
LINE ;** process one line of the WP array
ZBUILD ;** build Z array from XBL
REFBUILD ; %RCR BACK TO CALL FILL ;** fill one line
TEXT ;**
VAR ;** add .5 to column count to indicate a variablevs text
MAP ;** map shorthand for variables
OUT ;** output tranform of data field
TABS ;
```

EXIT ;
MDY(X) ;external date to mm/dd/yy x :: var or ~"NOW"~ or ~" WP(X) ;build wp entry
X #:: WP(FLD,n)=TEXTn

XBARRAY0 - Documentation for XBARRAY

This utility provides a word processing format of free text and local variable references to build an array. A file is necessary that has a .01 field for the form name and a WP field to hold the WP form.

Two Entry points

EDIT^XBARRAY(.NAME,DIC,FIELD) Edits and Displays the form. Place the call to EDIT in the code where the data or variables have been gathered. Typically this is one line previous to the call to $$GEN^XBARRAY. Once the form is designed the EDIT call is commented out.

$$GEN^XBARRAY(.NAME,DIC,FIELD,ROOT,FORMAT,LINE)

Generates the form into the ARRAY indicated by the ROOT. The call to $$GEN must have all variables used gathered. The return value of $$GEN is equal to the last line set in the array.

VARIABLES

NAME - The name space variable that holds the name of the form to be used. A pass by reference is needed for efficiency so that the pre-compilation of the form is held for repetitive use. The compilation is stored in the sub array as NAME(@NAME,line,.....). IE one local variable can be used for all form references.

Ex: S BARFORM="A/R BILL" will store and use the form compilation in BARFORM("A/R BILL",line,....,). When finished K BARFORM(BARFORM) will retrieve the local variable space from the last form used.

DIC - The root or file number of the file holding the forms.

FIELD ; The field number of the WP field holding the form.

ROOT - The root of the target array to be built. Either a global or a variable root as in the format used for a %XY^%RCR call. (%RCR is actually used)

FORMAT
null or zero The array is built ROOT(line)="...
1 The array is built ROOT(line,0)="....

LINE - The offset in line numbers in building the array. The array will start construction at LINE +1. The value of the last line created is returned $$GEN.

WP FORMAT INSTRUCTIONS

Free Text: Free text is key striken where desired. Do not use ~ as it is used to mark variable

Variables: The reference to a variable is marked with a beginning ~ and a trailing ~. The trailing ~ is always required even if the variable is last item on the line.

Mnemonics: A short hand for variables is available.

Comments: Programmers comments can be put into the for which are ignored by the generator.

Output Transform: Mumps output transforms can be
indicated for execution upon selected variables.

WP SPECIAL FUNCTIONS Located at the top of the form.

Comment line Begin the line with a ';'

Variable Mnemonic Reference: Name spaced variables can be long. A mnemonic reference is available to make life simple. Multiple mnemonic lines can be used if desired.

SETUP

#mnemonic1|variable1*mnemonic2|variable2*...
#mnemonicZ|variableZ*.....|

Example: #D|DUZ*V|BARVPT
#I|BARIPT |

(BARIPT array is storing IHS Patient Information)
(BARVPT array is storing VA Patient Information)

'#' Marker placed in the first column

mnemonic1 User's choice
ex: D to denote DUZ
'|' Separator

variable1 User's choice of the local variable

ex: DUZ
'*' Repetitive marker if more than one mnemonic is indicated

USE The mnemonic reference can be used any where in the WP form.

Format ~mnemonic|variable subscript~|

'~' Beginning marker for the variable

mnemonic1 User's mnemonic

'|' Separator|

subscript The subscript of the variable to be used

'~' Ending marker for the variable

ex: ~D|~ for DUZ|
x ~D|0~ for DUZ(0)|
~I|.01~ for BARIPT(.01)|

MUMPS OUTPUT TRANSFORM

A simple MUMPS output transform is also provided to aid in form design. A variable or mnemonic indicated will have its output transformed prior to being put into the form.

SETUP
*var1!mumps code1*var2!mumps code2
*mnemonic3!mumps code3*mnemonic4!mumps code4

Ex: *DUZ(2)!$J(X,10,2) will output $J(DUZ(2),10,2)
*D | 2!$J(X,10,2) mnemonic notation of same|

'*' Output Transform marker in column one. At T

Variable/ Variable or mnemonic as it would appear in Mnemonic form between '~'s.

'!' Separator

mumps code Mumps code expression as a function of x.
Do not state 'S X=f(x)'
Enter the function only, f(x).

'*' Separator if more than one is put on one line

SPECIAL OUTPUT TRANSFORMS provided by XBARRAY

xxx!$$MDY(X) a literal ~"NOW"~ or variable ~IT|9~|
ex: *"NOW"!$$MDY(X) or *IT|9!$$MDY(X)| returns mmdd/ yy

xxx!$$WP("X") for a word processing field
NOTE: "X" IS ABSOLUTELY NECESSARY
The variable array must have the form
VAR(subcript,n) where n = 1:1

DOCE ;
TEST ;

If you have A/R installed, uncomment the following line

XBBJ - GO TO %BJ^%ZTMS

START ;

XBBPI - BUILD PACKAGE PRE-INIT ROUTINE

This routine builds a pre-init routine for a specified package. The pre-init routine will delete FileMan dictionaries being created by the package. Data globals and templates will be saved.

START ;
PACKAGE ;
BUILD ;
CHECKRTN ;
CR2 ;
EOJ ;
EOJ2 ;
EOJ3 ;
DTA ;

XBCDIC - CLEAN UP ^DIC AND ^DD

THIS FUNCTIONALITY HAS BEEN INCLUDED IN THE FILEMAN DD UTILITIES, BEGINNING WITH V 19.0. WE RECOMMEND IT'S USE AS IT IS MORE LIKELY TO BE CURRENT. 3-20-96

This routine cleans up ^DIC and ^DD by a range of dictionary numbers. All files in ^DIC within the range of dictionary numbers are checked for the following:

They must have a NAME in ^DIC. The NAME in ^DIC must match the NAME in ^DD. The NAME must exist in ^DIC("B" with the correct number, and that number cannot

occur more than once in ^DIC("B". They must have a data global specified in ^DIC. The data global must be in the correct form. The data global must exist. The data global must have a 0th node. The NAME and NUMBER in the data global must match ^DIC. The data globals 0th node must be consistent with the data (Exact count not checked).

They must have valid entries in ^DD as follows: The ^DD entry must have a .01 field. All "SB" pointers must point to existing sub-files. All sub-files must point back to correct parent. All “TRB” entries must exist. All “PT” entries must exist. All “ACOMP” entries must exist.

When discrepancies are found the entries are corrected automatically where ever possible. If this is not possible, operator interaction occurs to make the corrections. If the file cannot be corrected, it will be deleted.

After all dictionaries within the range of dictionary numbers are checked, all other entries within the range will be deleted.

The last step is to set the 0th node of the FILE OF FILES to the correct high DFN and the correct count of entries.

```
BEGIN ;
LO ;
HI ;
BCHK ;
EOJ ;
EOJ3 ;
DTA ;
DOCE ;
TEST ;
```

If you have A/R installed,uncomment the following line

```
XBCDIC2 ;  - CHECK DICTIONARY NAMES AND DATA GLOBALS ;
```

Part of XBCDIC

```
START ;
XBCDNC ;
READNAME ;
GCHK ; CHECK DATA GLOBAL
GCHK2 ; CHECK 3RD AND 4TH PIECE
GCHK3 ; CHECK FILE NUMBER IN DATA GLOBAL
G2R1 ;
READGBL ;
DICB ; CHECK DIC("B"
PICKNAME ;
```

```
P1 ;
NAMESET ;
GNMCHK ; CHECK DATA GLOBAL NAME AGAINST ^DIC
GNMFIX ;
GNMR1 ;
```

XBCDIC3 - CHECK ^DD

Part of XBCDIC

```
START ;
XBCDDDC ; CHECK ^DD ENTRY CHKDD0
CHECK 0TH NODE
CHKPT ; CHECK "PT" NODE PT
CHKTRB ; CHECK "TRB" NODE
TRB ; THIS CAN CHECK MORE THAN IT DOES ***
CHKACOMP ; CHECK "ACOMP" ENTRIES
CHKFIELD ;
ACOMP ;
SBTRACE ; CHECK ALL SUB-FILES
SBTRACE2 ;
SBCHECK ;
SBTRACE3 ;
SBTRACE4 ;
SBTFIX ; FIX "NM"
```

XBCDICD - DELETE BAD FILES ;

Part of XBCDIC

```
START ;
ERRORS ; RESOLVE ERRORS SET BY ^XBCDIC2 OR ^XBCDIC3
ACTR ;
```

XBCFIX - COUNT ENTRIES IN FILEMAN FILES AND FIX

This routine counts primary entries in FileMan files and fixes the 3rd and 4th piece of the 0th node.

```
START ;
XBCFIXFL ;
EOJ ;
```

XBCFXREF - CHECK/FIX XREFS

This routine checks all REGULAR xrefs at the file level for selected files to insure all pointed to entries exist.

START ;
INIT ;
FILES ; CHECK FILES
FILE ; CHECK ONE FILE
XREFILE ; CHECK EACH FILE/FIELD CREATING XREFS BACKUP ; BACKUP
TREE (CALLED RECURSIVELY) XREFIELD ; CHECK EACH FIELD CREATING
XREFS XREF ; CHECK XREFS ON FIELD
SAVE ; SAVE XREF TO CHECK
CHECK ; CHECK DATA GLOBAL FOR XREFS
CHKXREF ; CHECK ONE XREF
EOJ ;

XBCLM - COLUMN LISTER

This routine displays a column number header followed by the passed string.

EP(STR) ;PEP - ColumnLister
LINE ; WRITE HEADER AND ONE LINE
QUIT(L) ;

XBCLS - CLEAR SCREEN

XBCNODE - COUNT ENTRIES IN GLOBAL NODE

This program counts unique values in a selected global node.

START ;
LOOP ;

XBCOUNT - COUNT ENTRIES IN FILEMAN FILE

This routine counts primary entries in a FileMan file and corrects the 0th node.

START ;
LOOP ;
ENT ;
EOJ ;

XBCSPC - CHECK POTENTIAL SPECIFIER FIELDS

This routine checks selected field to see what percent of the time it exists in the entries in a file, and if it should be unique, it makes sure it is unique.

START ;
FILE ;
FIELD ;
LIST ;
CHKXREF ; SEE IF UNIQUE SPECIFIER HAS REGULAR XREF
CHKDATA ; CHECK DATA IN SELECTED FIELD EN(FILE,FIELD,XREF,UNIQUE)
EXTERNAL ENTRY POINT TO ALLOW SPE EOJ ;

XBDAD0 - SET ALTERNATE DA/D0

This routine sets the DA array from D0,D1 etc. or D0,D1, etc., from the DA array. If the variable XBDAD0=2 it sets the DA array, otherwise it sets D0,D1 etc.

The variable XBDAD0 will be killed upon exiting this routine.

The entry point KILL kills D0, D1, etc.

START ;
DAD0 ; ----- Set D0 (etc.) from DA array.
D0DA ; ----- Set DA array from D0 (etc.).
KILL ;PEP - KILL D0, D1, ETC.

XBDATE - ADAPTATION OF %RS TO SELECT ROUTINES EDITED

This routine limits routines selected by RSEL to routines edited after some date.

START ;
DIR ;

XBDBQDOC - DOUBLE QUEUING SHELL HANDLER DOCUMENTATION

NOTES FOR PROGRAMMERS

%ZIS with "PQM" is called by XBDBQUE if '$D(XBIOP).

The user will be asked to queue if queuing has not been selected. IO variables as necessary are automatically stored. XBxx variables are killed after loading into an XB array. XBDBQUE can be nested.

The compute and print phases can call XBDBQUE individually, (XBIOP is required). The appropriate %ZTSK node is killed.

Example:
S XBRC="C^AGTEST",XBRP="P^AGTEST",XBRX="END^AGTEST",XBNS="A
D ^XBDBQUE and handles foreground and tasking
Q

VARIABLES NEEDED FROM CALLING PROGRAM

MANDATORY
EITHER XBRC-Compute Routine or XBRP-Print Routine.

OPTIONAL
XBRC-Compute Routine.
XBRP-Print Routine.
XBRX-Exit Routine that cleans variables (HIGHLY SUGGESTED)
XBNS-name space of variables to auto load in ZTSAVE("NS*")=" "="DG;AUPN;PS;..."
(will add '*' if missing).
XBNS("xxx")="" - ZTSAVE variable arrays where xxx is as described for
ZTSAVE("xxxx")="".
XBFQ=1 ForceQueing.
XBDTH=FM date time of computing/printing XBIOP=pre-selected printer device with
constructed with ION ; IOST ; IOSL ; IOM (mandatory if the calling routine is a
queued routine XBPAR= %ZIS("IOPAR") values for host file with XBIOP if
needed.

TEST ;
TEST1 ; test of stacking a second call to XBDBQUE in the print PA ;
PB ;
TEST2 ; TEST FOR COMPUTING ONLY
RC S %H=$H D YX^%DTC F XBI=1:1:20 S ^PWDBT(XBI)=XBI_Y

XBDBQUE - DOUBLE QUEUING SHELL HANDLER

|refer to XBDBQDOC for instructions, examples, and tests |

START ;
ZIS ;
ZISQ ;
QUE1 ;
DEQUE1 ;> 1st deque
COMPUTE ;>do computing |routine|
QUE2 ;
DEQUE2 ;>EP 2ndDeque |printing|
PRINT ;>print
END ;>End |cleanup|
END0 ;EP - from compute cycle when XB("RP") EXISTS

END1 ;EP clean outxb as passed in
ENDC ;EP - end computing cycle
SUB ;>Subroutines
NORC ;used if no XBRC identified

XBDH - HEADER EDITOR MAIN ROUTINE

VAR ;
TITLE ;
XBDHD ;
EXIT ;

XBDHD - GET BASIC INFO ABOUT FILE AND FIELDS

NEW ;
XBDHPDFN ;
HEADER ;
THLW ;
START ;
STACK ;
FIELD ;
HDR ;
HDW ;
WIDTH ;
GLOB ;
RESET ;
N1 ;
EXIT ;
DIRCK ;

XBDHD1 - COMPILES HEADER LINE

NEW ;
INIT ;
WLINE ;
FIN ;
EXIT ;
LINE ;
VAR ;
L3 ;
NOTES ;

XBDHD2 - SPECIAL CHOICES

NEW ;
START ;
CNEXT ;
C1 ;
C2 ;
C3 ;
C31 ;
C4 ;
C5 ;
C6 ;
C7 ;
C8 ;
MOVE ;
TEXT ;
NOTES ;

XBDHDF - GETS FIELD INFO FOR HEADER LINE EDITOR

NEW ;
VAR ;
MORE ;
GETFIELD ;
OK1 ;
CKMUMPS ;
CKM1 ;
OK ;
COMPUTED ;
C1 ;
MUMPS ;
MULTIPLE ;
PATH ;
DECI ;
LAST ;
LINE ;
OUT ;
NOTES ;

XBDHDF1 - CHECKS JUMP SYNTAX

STRIP ;
EXIT ;
CKF ;
CKPT ;
J1 ;

JUMPQ ;

XBDHDIP - OVERLAY OF DIP2 FOR AUTO FILEMAN

AUTO ;
NOTES ;

XBDHDSP - PUTS SPACES BETWEEN HEADERS

NEW ;
INIT ;
CQ ;
MANUAL ;
EXIT ;
MAN ;
SP ;
AUTO ;
REP ;
EREP ;
NOTES ;

XBDHDSV - COMPILES HEADER INFO FOR AUTO ENTRY INTO

NEW ;
INIT ;
INCN ;
CLOSE ;
EXIT ;
SET ;
MUMPS ;
PRELIM ;
DOWN ;
UP ;
PATH ;
NORMAL ;
STD ;
DIP ;
NOTES ;

XBDICV - SET DICTIONARY VERSION NUMBERS

This routine sets FileMan dictionary version numbers.

START ;
GETDICS ; GET SET OF DICTIONARIES
SHOW ; SHOW CURRENT VERSION NUMBERS

HIGH ; SAVE HIGH VERSION NUMBER
ASK ;
VER ;
CHANGE ; CHANGE VERSION NUMBERS
PROCESS ;
P2 ;
P2ERR ;
EOJ ;

XBDIE - NESTING OF DIE

PROGRAMMERS NOTE: PLEASE USE THE MORE GENERIC ^XBNEW. ;
XBRET has the form "TAG^ROUTINE:VAR,NSVAR*" This allows for the nesting of die calls by:

1. Building and executing an exclusive new from preselected kernel variables and any local variables &/or name spaces identified by the calling parameter.
2. After executing the new (....) XBDIE performs a DO call the program entry point identified by the calling parameter. The entry point passed should build the variables and execute the DIE call to be nested.
3. As XBDIE quits to return to the calling program it pops the variable stack.

The passing parameter is built by t'ag^ routine:var;vns*"

The die call to be nested is structured with a tag entry and a Quit.

The call is made with DO ^XBDIE("TAG^ROUTINE:AGSITE,ABM*") where the variable AGSITE and the namespace ABM is included in the exclusive new for illustration.

Proper logic flow after the XBDIE call usually needs some attention.

A TEST entry point is provided in this routine for illustration.

S ;
RETURN ;
END ;
XBKVAR ;;DUZ,DTIME,DT,DISYS,IO,IOF,IOBS,IOM,ION,IOSL,IOST,IOT, TEST ;
T2 ;

XBDIFF - RETURN DIFFERENCE BETWEEN TWO DATE/TIMES

Passed two date/times this routine returns the difference in days, hours, minutes, seconds separated by colons ":".

The date/times must be passed in the variables X and X1.

The result will be returned in X. X1 will be killed. If either X or X1 are invalid X will be returned as -1 and ; X1 will be killed. The date/times may be passed in $HOROLOG format or in internal FileMan format. See also, $$FMDIFF^XLFDT, and $$HDIFF^XLFDT.

START ;
EDIT ; EDIT INPUT
EDITX ; EDIT X
EDITX1 ; EDIT X1

XBDINUM - CONVERTS NON-DINUM FILE TO DINUM FILE

START ;
X1 ;
EOJ ;

XBDIQ0 - Documentation for XBDIQ1

Documentation for XBDIQ1

This routine provides a friendly front end to EN^DIQ1 and an assortment of other features.

1. Data arrays are returned into 'DIQ in a variety of formats controlled by the parameter set into DIQ(0). The default is 'DIQ(FLDNUM)= external value of field FLDNUM is the DD number of the field as used in DR.

2. Data retrieval is non-intrusive! Does not disturb the partition.

3. Input Variables used are the same as for EN^DIQ1 with more friendly results.

4. DR (filenumber and DA filenumber arrays are automatically built when needed.

ENTRY POINTS

ENP^XBDIQ1(DIC,DA,DR,DIQ,DIQ(0))
Returns 'DIQ(FLDNUM)= data for One Entry.

ENPM^XBDIQ1(DIC,DA,DR,DIQ,DIQ(0))
Returns 'DIQ(DA,FLDNUM)= data for Multiple Entries.
DIC("S") can be set and used for screening entries.

$$VAL^XBDIQ1(DIC,DA,DR)
Returns External value of one field.

$$VALI^XBDIQ1(DIC,DA,DR)

Returns Internal value of one field.
$$DIC^XBDIQ1(DIC) Returns constructed DIC from file/ subfile number.

PARSE^XBDIQ1(DA)
Returns a DA array from a literal string made from Variables or Numbers mixed in descending order.

Example: "1,DFN,56" => DA=56,DA(1)=34,DA(2)=1 where DFN=34
also: S VAR(I)="1,DFN,56" D PARSE^XBDIQ1(VAR(I)) => as above.

EN Returns one Entry (DR) fields. Needs DIC,DA,DR,DIQ,DIQ(0) as set up for calls to EN^DIQ1.

ENM Returns Multiple Entry's (DR) fields
1) upper DA arrayie: DA(1),DA(2), ...
2) DA="" in the passing array
3) optional DIC("S")
Needs DIC,DA,DR,DIQ,DIQ(0) as set up for calls to EN^DIQ1.
DIQ(0)=1 by default.

DIQ(0) Format Options.

DIQ(0) If DIQ(0) is not present the default is set to NULL.

0 OR NULL DIQ(FLD)=
1 DIQ(DA,FLD)=
2 DIQ(DA(x),..,DA,FLD)=
nI DIQ(... ,FLD,"I")=internal value(s)returne
nN NULL fields are not returned

DA can be the array .DA or a literal string in descending order. "1,23,45"
"1,PATDFN,BLDFN" variables will be unfolded.
BARVDA("EOBSUB")
("EOBSUB")="BAFCLDA,BARITDA,BAREDA"

XBDIQ1 - SPECIAL EN^DIQ1 DATA PULLER

Documentation for the APIs in this routine can be found in routine XBDIQ0.

DOC ;
EN ;PEP - Returns single entries
ENP(DIC,DA,DR,DIQ,XBFMT)
PEP - param pass into EN
ENPM(DIC,DA,DR,DIQ,XBFMT) ;
PEP - param pass into EN ENM ;

PEP - get multiple entries
ENDIQ1 ;
EP - call EN^DIQ1ENDIQ1X ;
EP - to call DIQ1 with new
ENDIQ1XN ;EP
EXIT ;EP
PULLDIQ1 ;EP - PULL FROM ^UTILITY("DIQ1",$J) %XY
EP - set %X & %Y to format

0 I +XBDIQ(0)=0 D Q
1 I +XBDIQ(0)=1 D Q 2 I +XBDIQ(0)=2 D Q %XYE Q
DICFNGL(X) ;EP - set XBFN & XBGL0 return 1 error
DICFNGLX ;
VAL(DIC,DA,DR) ;PEP - extrinsic pull a value for a field VALI(DIC,DA,DR)

PEP - extrinsic pull a value for a field
PARSE(XBDA) ;

PEP - parse DA literal into array

DIC(XBFN)

PEP - Extrinsic entry to return DIC from global LEVELS
EP - setup XB_FN_DA_DR_FLD arrays for upper levels it PARENT gather parent information

EPAR ;
SETDIQ1

EP - set DR (fn and DA (fn arrays for DIQ1

XBDIR - DIR INTERFACE

The purpose of routine XBDIR is to provide interface methodology for a call to ^DIR, to ensure correct handling of variables, and to provide for the expressiveness of an extrinsic function.

There is no requirement to use the entry point, below.

The format of the call is to SET a local variable to the output of the call to DIR^XBDIR(), which will be Y at the bottom of this routine, or, less likely, WRITE the value. An example of the call is:

S %=$$DIR^XBDIR(<actual_parameter_list>)
where the <actual_parameter_list> is:

(DIR(0),DIR("A"),DIR("B"),DIR("T"),DIR("?"),DIR("??"),<skip>
where <skip> is the number of lines to skip before the call to ^DIR.
Examples:

S %=$$DIR^XBDIR("N^1:2","Select report method",2,"","Produce report by FY or Dates", "^D HELP^<your_routine>",300,2)

S <namespace>FY=$$DIR^XBDIR("NO","Object Class Code Summary for FISCAL YEAR ",FY,$G(DTIME,500),"Enter a FOUR DIGIT FISCAL YEAR","^D SB1^<your_routine>")

DIR(O,A,B,T,Q,H,R)
PEP - Extrinsic interface to ^DIR.

XBDR - BUILDS DIR STRING

This routine builds a string which sets variable DIR, and it's descendants, for use in a routine. The string is stored in the variable "%", and in the "Temp" storage area for the screen editor for the current device.

START ;
RUN ;
EXIT ;
LOC ;
HELP ;
QQ ;
SET ;
S1 ;
NAR ;
DFLT ;
TEST ;
TQ ;
SAVE ;

XBDR1 - XBDR SUBROUTINE

Part of XBDR

TYPE ;
E ;
Y ;
MINMAX ;
F ;
L ;
N ;

S ;
S1 ;
S2 ;
D ;
DTS ;
ADTS ;
Z ;

XBDSET - BUILDS LIST OF FILEMAN FILES

This routine selects FileMan dictionaries individually, by a range, or for a specific package. This routine can be called from another routine by setting the variables XBDSLO, XBDSHI and then D EN1^XBDSET.

If the variable XBDSND exist upon entry no default menu option will be displayed.

START ;
GETFILES ;
OPTION ; GET FILES FOR SELECTED OPTION
ONEFILE ; GET ONE FILE AND EXIT
SELECT ; GET SELECTED FILES
RANGE ; GET RANGE OF FILES
RANGE2 ; LABEL FOR EXTERNAL ENTRY POINT EN1 PACKAGE

GET FILES FROM SPECIFIC PACKAGE LIST ; LIST FILES ALREADY SELECTED PXBSE, GIVE USER A CHANCE TO SEE LAST PAGE AND QUIT

EN1 ;EP - Non-interactive selection of range of files.
EOJ ;

XBENHANC - DISPLAY/PRINT ENHANCEMENTS FIELD IN PACK

Print enhancements to a package, from the entry in the PACKAGE file. Entry point EN^XBENHANC(ns) is used, with \ the caller providing the namespace of the package.

EN(XB) ;PEP - XB = Namespace of package to print enhancements.
DEV ;
K ;
START ;EP - TaskMan.
DIWP(X) ;
TOF ;

XBFCMP - COMPARES FILEMAN FILES IN TWO UCIs

Ignores the following:
^DD(file,0,"PT",
^DD(file,field,1,0)
^DD(file,field,21
^DD(file,field,"DT"

If a field does not exist in one file, a message is displayed and all sub-nodes of that field are ignored.

If the compare is limited to fields containing a particular GROUP, the second pass, which checks for entries in the secondary UCI not in the primary UCI, is not executed. On the first pass the GROUP multiple in the secondary UCI is ignored.

START ;
XBFCMPFL ;
COMPARE ;
SBTRACE ; CHECK ALL SUB-FILES
SBTRACE2 ;
SBTRACE3 ;
GET2ND ; GET SECONDARY UCI
EOJ ;
XBGCMP ; COMPARES GLOBAL TREES
SEARCH ;
EXTR ;
SUB ;
CHKGROUP ;

XBFDINFO - RETURN FIELD INFORMATION

ATTENTION PROGRAMMERS: Use line label FLD() for entry. Do not use the first line for entry.

Given a file/subfile number, a field number, and an array root, this routine will return information about the specified field. The information will be returned as subscripted variables from the root passed by the caller.

The field information returned will be a subset of the following:
ROOT("NAME") = name of field
ROOT("NODE") = node in data global
ROOT("PIECE") = piece in node
ROOT("TYPE") = FileMan field type or "M" for multiple, or "C" for computed
ROOT("SFILE") = subfile number if the field is a multiple
ROOT("PFILE") = file number of pointed to file

ROOT("PGBL") = gbl of pointed to file
ROOT("DINUM") =existance indicates DINUM pointer
ROOT("VPFILE",file) = variable pointer prefix. 'file' is pointed to file
ROOT("VPGBL",file) = variable pointergbl of pointed to file. 'file' is pointed to file.

Formal list:

1) FILE = file/subfile number (call by value)
2) FIELD = field number (call by value)
3) ROOT = array root (call by reference)

START ;
FLD(FILE,FIELD,ROOT)

PEP - Return information about a field.

XBFIXL1 - STANDARDIZE LINE 1 OF SELECTED ROUTINES

This routine asks user to select a set of routines, asks the user for the programmer information and standardizes the format of the first line of each routine.

The form of the first line will be as follows:

label: agency/site/developer - comment: edit date Ex:XBFIXL1, - FIX LINE 1 [11/08/90 10:41 AM]

START ;
CHECK ;
REVER ;
MODIFY ; Modify Line 1.
EXTDATE ; Extract date and remove from Line.
VERIFY ; Ask user to verify mod.
FIX ; Get comment from user.
EOJ ;

XBFIXPT- FIX ALL "PT" NODES

This routine fixes all "PT" nodes for files 1 through the highest file number in the current UCI.

START ;
FPOS ; CHECK FOR FALSE POSITIVES
CHKIT ;
FNEG ; CHECK FOR FALSE NEGATIVES
PTRCHK ;
PTRCHK2 ; VARIABLE POINTER CHECK

XBFLD - DICTIONARY LISTING

This routine lists dictionaries which may be selected individually or by a range of dictionary numbers.

This routine requires the 89 MUMPS Standard, FileMan Version 17.7 or greater, Kernel Version 6 or greater, and the following routines must exist in the UCI in which this routine is running:

XBKVAR, XBSFGBL

START ;
LOOP ; LIST FILES UNTIL USER SAYS STOP
LIST ; LIST RANGE OF FILES
FILE ; LIST ONE FILE
FIELDS ; LIST ALL FIELDS IN ONE FILE/SUBFILE (CALLED RECURSIVE FIELD
LIST ONE FIELD COMPUTED ; COMPUTED FIELD
MULTIPLE ; LIST MULTIPLE, THEN FIELDS IN SUBFILE WRITE ; WRITE ONE LINE WRITELF ; WRITE ONE LINE FEED
HEADING ; DICTIONARY HEADERS HEADING2 ; HARD COPY HEADERS PAGE EP - PAGE HEADERS
PAUSE ; GIVE USER A CHANCE TO SEE LAST PAGE AND QUIT INIT ; INITIALIZATION
FORMAT ;EP - select format
TXT ;
EN ; EXTERNAL ENTRY POINT
EOJ ; END OF JOB

XBFLD0 - PRINT FIELD TRIGGERS

S ;
MU ;MUMPS
MN ;MNEMONIC
RG ;REGULAR
BU ;BULLETIN
KW ;KWIC
TR ;TRIGGER

XBFLD2 - INITIALIZATION FOR ^XBFLD

Part of XBFLD

^UTILITY("XBDSET",$J, is used to store the list of files to be listed so that other software can pass files to be listed to the external entry point EN^XBFLD, and the other software could select files by using ^XBDSET.

INIT ; INITIALIZATION
DEVICE ; GET DEVICE (QUEUEING ALLOWED)
XBLM ;
XBLME .Q

XBFMK - KILL FILEMAN VARIABLES

This routine kills variables left around by FileMan.

XBFORM - BUILD ARRAY FROM WP FORMAT

Please refer to routine XBFORM0 for documentation.

EDIT(XBFORM,XBWPDIC,XBWPFLD)
EP Edit a Form
EDIT2 ;
MARK ;
GEN(XBFORM,XBWPDIC,XBWPFLD,XBREF,XBFMT,XBLAST) ;EP ** generate
EDITWP ;** edit WP array
WPGET ;** get WP array
BUILD ;** scan WP array to build XBLLINE
;** process one line of the WP array
ZBUILD ;** build Z array from XBL
REFBUILD ; %RCR BACK TO CALL FILL ;** fill one line
TEXT ;** VAR ;** add .5 to column count to indicate a variable text
MAP ;** map shorthand for variables
OUT ;** output transform of data field
TABS ;
EXIT ;
MDY(X) ;external date to mm/dd/ yy x ::var or ~"NOW"~ or ~" WP(X) ;buildwp entry X #:: WP(FLD,n)=TEXTn
FL(X) ; FL fill lines until line X
FMSUB(X) ;process popular ;D8 ;L20 ;R20

XBFORM0 - Documentation for XBFORM

This utility provides a word processing format of free text and local variable references to build an array.

A file is necessary that has a .01 field for the form name and a WP field to hold the WP form.

Two Entry points

EDIT^XBFORM(NAME,DIC,FIELD) Edits and Displays the form.

Place the call to EDIT in the code where the data or variables have been gathered. Typically this is one line previous to the call to $$GEN^XBFORM. Once the form is designed the EDIT call is commented out.

$$GEN^XBFORM(NAME,DIC,FIELD,%Y,FORMAT,OFFSET) Generates the form into the root array indicated by %Y. The call to $$GEN must have all variables used gathered. The return value of $$GEN is equal to the last line set in the array.

INPUT VARIABLES
NAME The name space variable that holds the name of the form to be used.

DIC The root or file number of the file holding the forms.

FIELD The field number of the WP field holding the form

%Y The root of the target array to be built. Either a global or a variable root as in the format used for a %XY^%RCR call. (%RCR is actually used)

FORMAT null or zero The array is built %Y(line)="...

1 The array is built %Y(line,0)="....

OFFSET The offset in line numbers in building the array. The array will start construction at OFFSET+1. The value of the last line created is returned $$GEN.

WP FORMAT INSTRUCTIONS

Free Text: Free text is key striked in where desired. Do not use | as it is used to mark variables|

Variables: The reference to a variable is marked with a beginning | and a trailing |. The trailing | is always required even if the variable is | last item on the line.

To use a " in your display use |" in the | form.

Mnemonics Comments - A short hand for variables is available. Programmers' comments can be put into the form which are ignored by the generator.

Output Transform - Mumps output transforms can be indicated for execution upon selected variables.

WP SPECIAL FUNCTIONS Located at the top of the form.

Comment line Begin the line with a ';'

Variable Mnemonic - Reference Name spaced variables can be long. A mnemonic reference is available to make life simple. Multiple mnemonic lines can be used if desired.

SETUP

#mneum1=variable1|mneum2=variable2*...|
#mneumZ=variableZ|.....|

Example:

#D=DUZ | V =BARVPT|
#I=BARIPT

(BARIPT array is storing IHS Patient Information)
(BARVPT array is storing VA Patient Information)

'#' Marker placedin the first column

mnemonic1	User's choice Ex: D to denote DUZ '=' EQUALS
variable1	User's choice of the local variable Ex: DUZ
'|'	Repeat separator if more than one | mnemonic is indicated on a line

USE	The mnemonic reference can be used any where in the WP form.
Format	~mnemonic|variablesubscript~|
'|'	Beginning marker for thevariable|

mnemonic1	User's mnemonic
'@'	Mnemonic substitution marker
subscript	The subscript ofthe variable to be used.
'|'	Ending marker for the variable

ex: |D@| for DUZ
|D@0| for DUZ(0)
|I@.01| for BARIPT(.01)

MUMPS OUTPUT TRANSFORM - A simple mumps output transform is also provided to aid in form design. A variable or mnemonic indicated will have its output transformed prior to being put into the form.

SETUP

*var1:mumps code1|var2:mumps code2| *mnemonic3:mumps code3|mnemonic4:mumps code4|

Ex: *DUZ(2):$J(X,10,2) will output $J(DUZ(2),10,2)
*D@2! :$J(X,10,2) mnemonic notation of same

'*' Output Transform marker in column one. Near TO

Variable/ Mnemonic	Variable or mnemonic as it would appear in the form between '~'s.
':'	Separator
mumps code	Mumps code expression as a function of x. Do not state 'S X=f(x)' Enter the function only, f(x).
'|'	Repeat Separator if more than one is put on one line.

SPECIAL OUTPUT TRANSFORMS provided by XBFORM

$$MDY(X)
*xxx:$$MDY(X) a literal NOW or variable |IT@9|
ex: *"NOW":$$MDY(X) or *IT@9:$$MDY(X) returns mm/dd /yy

$$WP("X")
*xxx:$$WP("X") for a word processing field array x|xx |

NOTE: "X" IS ABSOLUTELY NECESSARY

The variable array must have the form xxx (n) where n = 1:1 xxx may be B@101 as if returned by XBDIQ1 in the node 101 of B@

EX:
*B@:$$WP("X") |B@| for B=BARWP with BARWP(n) define
*B@101:$$WP("X") |B@101| for B=BARWP with BARWP(101,n) de $$FL(X)
*19:$$FL(X) |19| in form: fill lines through 19

D10
*xxx:;D10 Performs $J(xxx,10,2)
R20
*xxx:;R20 Performs $J(xxx,20)
L15
*xxx:;L15 Performs $E(xxx,1,15)

TEST ;;
TESTE ;;END
PRT ;

XBFORM1 - sub x in output transforms

XBV1=NEWCODE,XBLINX=original out transform SUB(XBV1,XBLINX)
EP extrinsic to return new output transform XSUB
EP - do it
SCAN
EP - scan for X
CHKMK ;
SCANE ;
BLDLIN1 ;
BLDLIN1E ;

XBFRESET - RESET FILE GLOBALS

This routine removes all data from FileMan files by saving the 0th node, killing the global, then resetting the 0th node.

START ;
EN1
PEP - Interactive entry, files already selected.
EN2
PEP - Non-interactive entry, files already selected.

XBFUNC - FUNCTION LIBRARY

FNDPATRN(STR,PAT)
PEP - Find pattern in string. Return begin GETPATRN(STR,PAT)
PEP - Retrieve pattern from string. INTSET(FILE,FIELD,EXTVAL)
PEP - Get Intnl Field Value Given E EXTSET(FILE,FIELD,INTVAL)
PEP - Get Extnl Field Value Given I DECFRAC(X)
PEP - Convert Decimal to Fraction (X containsDeci C(X,Y)
PEP - Center X in field length Y/IOM/80. GDT(JDT)
PEP - Return Gregorian Date, given Julian Date. JDT(XBDT)
PEP - Return Julian Date, given FM date. USR()
PEP - Return name of current user for ^VA(200. LOC()
PEP - Return location name from file 4 based on DUZ(2). CV(X)
PEP - Given a Namespace, return current version. XBFUNC1

XBFUNC1 - FUNCTION LIBRARY CONTINUED

PROVCLS(PROV,FORM)
PEP - Retrieve Provider Class from New Per PROVCLSC(PROV)
PEP - Retrieve Provider Class Code given New P PROVAFFL(PROV,FORM)
PEP - Retrieve provider affiliation in PROVCODE(PROV)
PEP - Retrieve provider code PROVINI(PROV)
PEP - Retrieve provider initials
ENDIQ1 ;

XBFUNC2 - FUNCTION LIBRARY : PCC RELATED FUNCTIONS

PCCPPINT(XBVISIT)
PEP - Return primary provider in VA(200 PCCPPN(XBVISIT)
PEP - Return a visit's primary provider (NAME PCCPPI(XBVISIT)
PEP - Return a visit's primary provider (INIT
PCCPPCLS(XBVISIT,FORM)
PEP - Return a visit's primary provide PCCPPCLC(XBVISIT)
PEP - Return a visit's primary provider class PCCPPAFF(XBVISIT,FORM)
PEP - Return a visit's primary provide

XBGC - COPY GLOBAL (ANY LEVEL)

START ;
GSGL ;
GDGL ;
WALK ; TRAVERSE TREE AT CURRENT SUBSCRIPT LEVEL GOTNODE ;
PROCESS ONE NODE

XBGCMP - COMPARES TWO DIFFERENT GLOBALS

This utility is to be used to compare two globals. The init globals entered must be identically subscripted. The utility indicate which nodes of the first global have values differ than similarly subscipted nodes of the second global. It will also indicate if a node in one global exists and if a similar subscripted node in the other does not exist. You may util [UCI,VOLUME] syntax to compare across UCIs and volume group.

###

A ;
X1 ;
INIT
Setup
ASK
Get globals to be compared

```
        1 ;
        2 ;
        X2 ;
        CHECK ; Check each global
        X4 . Q
        X6 ;
        TRAP ; Error trap for missing quotes
        CHECK2 ; Check both  globals
        X5 ;
        SETUP ; Get print parameters, task?
        NOQUE ;
        QUE ;
        PROCESS ; Compare
        X3 . Q
        CHANGE ; Temp change double quotes to single
        PRINT ; Prints or displays results
        PAUSE ; Quit display?
        SCHED ; Schedules another task to print
        EOJ ;
        HELP ;EP - Dooda about the utility

XBGLDFN  - GET LAST DFN

        START ;
        LOOP ;

XBGSAVE  - GENERIC GLOBAL SAVE FOR TRANSMISSION GLOB

XBGL = name of global (mandatory, all others optional)

        XBCON= if defined, stops if first-level subscript is non-ca
        XBDT = date of save, in FM format, default NOW
        XBE  = ending first-level numeric subscript
        XBFLT= 1, saves as flat file
        XBFN = output file name, default "<ns ><asufac>.<JulianDate>
        XBF  = beginning first-level numeric subscript, seed for $O
        XBIO = output device number
        XBMED= media to which to save global (user asked, if not ex
        XBNAR= description displayed to user, if user asks for help
        XBPAR= CR parameter (DSM only)
        XBQ  = Y/N, to place file in uucp q, default "Y"
        XBQTO= 'sendto' destination, default AOsysid in RPMS SITE
        XBTLE= comment for dump header (facility name is  concatenated)
        XBUF = directory, default " /usr /spool/uucppublic"
        SETUP ;
```

```
CHECK ;
CKGLOB ;
SETUPMSM ;
SETUPDSM ;
EOJ ;
```

XBGTI - RESTORE GLOBALS SAVED IN DSM %GTO FORMAT

This routine restores globals saved from DSM using ^ZIBGTOT into MSM environment on Altos. Globals are specially bracketed because of problems with dev/pipe.

The globals are read from a unix pipe. The script to read tape and pass the data to the pipe follows:

```
bs=$1
if [ "$1" = "" ]then echo "Enter block size: 8192// \c" read bs
if [ "$bs" = "" ]
then
bs=8192
fi
fi
echo Copying current tape on /dev/rmt0 to dev/pipe
unset noclobber
dd if=/dev/rmt0ibs=$bs skip=1 |tr -d '\015' > /dev/pipe| ; set noclobber
if [ $? != 0 ]
then echo Tape copy failed
exit 1
fi
echo Tape copy succeeded
```

```
START ;
LOAD
LOAD GLOBALS
FILE
PROCESS FILE
READ ; READ %GR AND %GV AND REPAIR IF NECESSARY
RGR ; READ GLOBAL REFERENCE
EOT ; END OF TAPE FLUSHP ; FLUSH THE PIPE
NEXTVOL ; START NEXT VOLUME EOJ ; EOJ HOUSEKEEPING CLOSE ; CLOSE
INPUT FILE KILL ; KILL VARIABLES
ERR ;
INIT ; INITIALIZATION DEVICE ; DEVICE HANDLING MSG ; MESSAGE TO
OPERATOR
```

XBGTOT - UTILITY, GLOBALS, FAST SAVE TO TAPE

This functionality was moved from XBGTOT to ZIBGTOT because of the use of non-standard $Z special variables. The below GO is provided for backwards compatibility.

XBGXFR - TRANSFERS GLOBAL TREES

```
START ;
SEARCH ;
EXTR ;
SUB ;
EN(FROM,TO,TALK) ;PEP - Transfer global trees.
```

XBGXREFS - GET XREFS FOR ONE FIELD IN ONE FILE

ATTENTION PROGRAMMERS: Do not use line one for entry. Use label XREF for entry.

Given a file/subfile number, a field number, and a variable from which to assign subscripted values, this routine will return the xrefs for the specified field.

The returned xrefs will besubscripted from the ROOT as follows:

```
ROOT(FIELD,n)     = file^subfile^xref (e.g. 9000010^AC)
ROOT(FIELD,n,"K") = executable kill logic
ROOT(FIELD,n,"S") = executable set logic
```

Formal list:

```
1)      FILE   = file or subfile number (call by value)
2)      FIELD  = field number (call by value)
3)      ROOT   = array root (call by reference)
XREF(FILE,FIELD,ROOT)
PEP - Return x-ref info for a field.
START ;
```

XBHELP - DISPLAY HELP TEXT FROM ROUTINE

Display text from the named routine, beginning at the named label. The fourth semi-colon piece is displayed. If the third semi-colon piece is "@", the indirection of the fourth semi-colon piece is written. The display ends if null or "###" is returned. E.g: D HELP^XBHELP("LABEL","ROUTINE",0) will print the text after LABEL:

ROUTINE ;
LABEL ;
Please enter what I think you should enter.
;;@;*7
;;@;!
;;###

HELP(L,R,T) ;PEP - Display text at label L, routine R, tab T s

XBHFMAN - HELP FRAME MANUAL (1/2)

Print a help frame manual for an IHS application, using OPTION descriptions and HELP FRAME texts in the namespace of the application selected from the PACKAGE file.

Information for the title and preface pages, and for ; indexed words, is expected to be in a routine named <namespace>HFMN. The title page lines are expected to begin at line TITLE+1, and the preface page at PREFACE+1. Any words to be indexed are expected to begin at line INDEX+1. See routine XBHFMAN2 for an example.

If entered from the top, user is asked for application. Entry point EN() must have the namespace of the application as the parameter. That allows programmers to create their own option and call it, without forcing user to select the application.

EN(XBSEL)
PEP ----- From application options, with
EN1 ;
DEV ;
K ;
START ;EP ----- FromTaskMan.
MENU(I)
Assign chapter number toOPTIONs. Recurse if O DATA(I,N,P) ;
RTRN
EP
If interactive, ask user to press RETURN. SETTMP(I,N)
Set option IEN and chapter designationint

XBHFMAN1 - HELP FRAME MANUAL (2/2)

Print Title and Preface page.

MAIN
$ORDER thru the list of OPTIONS, and print them.
INDEX ; ----- Print the index.
CONTENTS ; ----- Print the table of Contents.
END ;EP - Paginate, close, kill, quit.
HATOUT ;
PR(X) ;EP - Process one line of text.
INDX(X) ; ----- Parse/capitalize one line of text. Check for HDR(XB)
Print a chapter heading.
TOF ;EP ----- Move to bottom of page, print footer, paginate, MAKEHDRS
Make headers for odd and even pages.
CONT(X) ; ----- Add chapter number, title, and page number to DESC(A)
Print descriptions of theOPTIONs as the first HF(A)
Print the HELP FRAME text.

XBHFMAN2 - HELP FRAME MANUAL INFO EXAMPLE

TITLE ;EP
PREFACE ;EP
INDX ;EP

XBKD - KILLS DICs and GLOBALS

This routine deletes FileMan dictionaries, and optionally their globals, TEMPLATES and AUTHORITIES, by a range of dictionary numbers, or if called from another routine, by a predefined set of dictionaries. The assumptions made by this routine are that ^UTILITY, ^DIC, and ^DD are not UCI TRANSLATED. Any other globals may be translated, but the KILLs will take place in the current UCI only.

This routine can be called from another routine by setting the variables XBKDLO, XBKDHI, XBKDDEL, XBKDTMP and then D EN1^XBKD, or by creating the array ^UTILITY("XBDSET",$J) and then D EN2^XBKD.

The array ^UTILITY("XBDSET",$J) is subscripted by the file numbers and has a value of 'v1^v2' where v1 applies to the data global, and v2 applies to the TEMPLATES attached to the file. The allowable values of v1 and v2 are 'S' for save, 'D' for delete, 'A' for ask.

This routine will execute ^XBRESID to delete any residual entries in ^DD if dictionaries are deleted by a range of numbers.

BEGIN ;
LO ;
HI ;
DEL ;
TMP ;
EN1
PEP - Variables XBKDLO, XBKDHI, XBKDDEL, XBKDTMP must be CHECKDD ;
CHECK ^DD FOR DANGLING ENTRIES
EN2
PEP - Array ^UTILITY("XBDSET",$J) must exist when entering CHKVAL
CHECK G^T VALUES
CONFIRM
SHOW AND ASK
LIST ; LIST FILE INFO
MODIFY ;
ASK ;
ASK2 ;
EOJ ;

XBKD1 - XBKD SUBROUTINES

Part of XBKD
BX ;
NCK ;
NCKER ;
NCKOK ;
FGLB ;
FGOK ;
END ;
TEMPLP ;
TEMP ;
TEMP1 ;
TEMP2 ;
TEMPE ;
SB1 ;

XBKD2 - CHECK DICTIONARY NAMES AND DATA GLOBALS

Part of XBKD

START ;
XBKDNC ;
GCHK ; CHECK DATA GLOBAL
DICB ; CHECK DIC("B"
NAMESET ;
GNMCHK ; CHECK DATA GLOBAL NAME AGAINST ^DIC

GNMCHK2 ; DATA GLOBAL MISMATCH

XBKD3 - KILLSDICs and GLOBALS (PART 3)

Part of XBKD

Upon entry into this routine ^DIC(file #,0) must contain the file name, and, if the data global is to be deleted piece 3 of ^UTILITY("XBDSET",$J,file #) must be a valid global reference.

START ;
KILL ;
SBTRACE ; Delete all Sub-Files.
SBTRACE2 ;
SAVE ; Save "PT", "TRB", and "ACOMP" node from ^DD.
SAVE2 ;

XBKERCLN - CLEAN OUT KERNEL NAMESPACE ITEMS PRIOR

This routine is a modified XBPKDEL for use specifically to clean out KERNEL package items prior to new KERNEL install. This routine does not delete any security keys.

PKDEL ;
ASK ;ASK USER IF WANTS TO CONTINUE
DELETE ;
LIST ; ENTRY POINT FOR LISTING NAMESPACED ITEMS
LIST1 ;
LIST2 ;
EOJ ;

XBKSET - SET MINIMUM KERNEL VARIABLES

This routine is for programmers in direct mode only. It should not be called within a routine.

START ;
KERNEL ;

XBKTMP - CLEAN ^TMP NODES FOR CURRENT JOB

Called from the top, this routine KILLs entries in the ^TMP global that have $J as the first or second subscript.

XBKVAR - SET MINIMUM KERNEL VARIABLES

START ;

XBL - List Template Exporter

XBLCALL - LIST CALLABLE SUBROUTINES

This routine lists callable subroutines that are known to this routine. To add subroutines to this routine just add them to the end of this routine in same manner.

START ;
QUIT() ;
TARDATE(X) ; ReturnStardate of FM date/time.
L ;

XBLM - LIST MANAGER API'S

Documentation APIs for XBLM Generic Display. This utility uses the Veterans Administration List Manager ; (VALM).

APIs

FILE^XBLM("Directory", "File Name") Displays file indicated.

SFILE^XBLM Selection of host file for display.

VIEWR^XBLM("TAG^ ROUTINE","Header") Displays printout of the routine. (non - FM, using IO)

VIEWD^XBLM("Tag^ Routine","Header") Displays printout of the routine. (FM - using EN1^DIP)

DIQ^XBLM("DIC","DA") Displays EN1^DIQ for the DIC,DA.
ARRAY^XBLM("array(","Header") Displays the array(..,n,0) (%RCR notation)

>>GUI<<

GUIR^XBLM ("TAG^ROUTINE","root(") Returns the hard coded output in the array specified. "(" not required.

GUID^XBLM ("TAG^ROUTINE","root(") Returns the output of the FM routine specified in the array specified. Most often the call is "EN1^DIP".

S XBGUI=1,XBY="root(" D entry_point^XBLM The entry points sense these two variables and will put the output into the array specified.

EN ;EP -- main entry point for XB DISPLAY
HDR ;EP -- header code

INIT ;EP -- init variables and list array
MARKERS ;
HELP ;EP -- help code
EXIT ;EP -- exit code
K ;
EXPND ;EP -- expand code
FILE(XBDIR,XBFN) ;PEP - pull up a file into the TMP global for EFILE ;
SFILE
PEP - Select a host file for display.
OPEN ;
FNAME ;
FNAME1 ;
ES ;
ESFILE ;
VIEWR(XBROU,XBHDR)
PEP ** USING XBROU print to a host filefo GUIR(XBROU,XBY)
PEP - give routine and target array GUID(XBROU,XBY)
PEP give routine and target array for FM print VIEWD(XBROU,XBHDR)
PEP ** USING XBROU print to a host file for DIQ(DIC,DA)
PEP - Display DIC and DA after call to EN^DIQ ARRAY(XBAR,XBHDR)
PEP Display an array that has (...,n,0)st ARRAYE ;
STRIP(Z)
REMOVE CONTROLL CHARACTERS
OPENROOT(XBY)
EP - return OPen RooT form of XBY .. for %RCR u CONT
BROWSE
EP - support for device P-BROWSE-HFS

XBLMGUI - LIST MANAGER API'S

EN(XBROU,XBREF)
EP ** USING XBROU print to a hostfil OPENXB(ZISH1,ZISH2,ZISH3)
STRIP(Z) ;REMOVE CONTROLL CHARACTERS
DF(X) ; ----- Directory format.
PWD(ZISH1) ; ----- Print working directory.
JW ; -- MSM extrinsic.
STATUS() ; ----- EndOfFile flag.
PWDDOS S ZISH1(1)=$ZOS(14),ZISH1(1)=ZISH1(1)_":",ZISH1(1)=ZISH

XBLML - ENTER OR RESET XB DISPLAY IN LIST TEMPLATE

XBLUTL - LIST ^UTILITY FOR $J

This routine lists all entries in the ^UTILITY global for the current $J where $J is the first or second subscript.

This is most useful from programmer mode. If used through the XB menu, ^UTILITY($J) is killed in ^XBKSET before this routine is run.

START ;
LIST ;

XBLZRO - LISTS 0TH NODES

This routine lists the 0th nodes of FileMan files.

START ;
EN
PEP - List 0th node of pre-selected list of FileMan files. LIST ;
PAGE ; PAGE BREAK
HEADER ; PRINT HEADER
DEVICE ; GET DEVICE (QUEUEING ALLOWED)

XBMAIL - MAIL MESSAGE TO SECURITY KEY HOLDERS

This utility generates a mail message to everyone on the local machine that holds a security key according to the namespace, range, or single key provided in the parameter. The text of the mail messages must be provided by you, and passed to the utility as a line reference. The utility uses the first line after the line reference as the mail message subject, and subsequent lines as the body of the message, until a null string is encountered. This places an implicit limit on your mail messages to the maximum size of a routine.

Suggested text would be to inform the users that a patch has been installed, and describe any changes in displays or functionality, or problems addressed, and provide a contact number for questions, e.g.,

Please direct your questions or comments about RPMS software to:

OIRM / DSD (Division of Systems Development)
5300 Homestead Road NE
Albuquerque NM 87110
505-248-4189

Call examples are:

D MAIL^XBMAIL("ACHS*","MSG^ACHSP56")
D MAIL^XBMAIL("AG*,XUMGR-XUPROGMODE,APCDZMENU","LABEL^AGP5"
The second example would deliver a mail message containing the text beginning at LABEL+2^AGP5, and continuing to the end of routine AGP5, to each local user that holds a security key in the AG namespace, in the range from XUMGR to XUPROGMODE (inclusive), and to holders of the APCDZMENU security key.

If you are indicating a namespace, your namespace must end with a star ("*") character.

If you are indicating a range of security keys, the beginning and ending keys must be separated with a dash ("-"). If the utility encounters a dash in a comma-piece of the first parameter, it will consider it to be range-indicated, and not part of the name of the key. Use caution not to begin or end with a key that has a dash in its name.

If a comma-piece does not contain a star or dash, a single key is assumed.

The subject of the message is assumed to be the first line after LABEL^AGP5: LABEL EP - Mail msg text.

PATIENT REG, PATCH 5 CHANGES.

The utility will return Y=0 if successful, and Y=-1 if not successful. The message "Message delivered." will be displayed if the routine is called interactively.

MAIL(XBNS,XBREF)
PEP - XBNS is namespace, XBREF is line refer GETRECIP
SINGLE(K) ; Get holders of a single key K.
RANGE(R) ; Get holders of a range of keys.
NS(N) ; Get holders of keys in namespace N.
WRITDESC ;

XBNEW - NESTING OF DIE

PROGRAMMERS: Do not use the first line for entry. Use label EN^XBNEW() for entry.

EN^XBNEW("TAG^ ROUTINE", "variable list")

Variable list has the form "AGDFN;AGINS;AGP*".
Wild card * allowed.

XBRET has the form "TAG^ROUTINE:VAR;NSVAR*"

This allows for the nesting of die calls by

1. Building and executing an exclusive new from pre-selected kernel variables and any local variables &/or name spaces identified by the calling parameter.

2. After executing the new (....) XBNEW performs a DO call to the program entry point identified by the calling parameter. The entry point passed should build the variables and execute the DIE call to be nested.

3. As XBNEW quits to return to the calling program it pops the variable stack.

```
EN(XBRT,XBNS) ;PEP XBRT=TAG^ROUTINEXBNS=variable list ";'wi S ;
RETURN ;
NEW ;
END ;
XBKVAR ;;DUZ,DTIME,DT,DISYS,IO,IOF,IOBS,IOM,ION,IOSL,IOST,IOT,
```

XBNODEL - PREVENT USER FROM DELETING ENTRIES

This routine sets FileMan dictionaries so users cannot delete entries. Protection is provided bySET'ing the "DEL" node of the .01 fields in the selected dd's to "I 1".

```
START ;
ASK ;
PROCESS ;
P2 ;
P2ERR ;
EOJ ;
```

XBOFF - SET REVERSE VIDEO OFF

Original routine from IHS/OHPRD/EDE. 08-25-95. See also routine XBVIDEO.

START ;

XBON - SET REVERSE VIDEO ON

Original routine from IHS/OHPRD/EDE. 08-25-95. See also routine XBVIDEO.

START ;

XBPATC - CHECK PATIENT GLOBALS

$O thru the PATIENT and 3RD party globals looking for missing entries.

ST;
LOOP ;
LOOP1 ;
LOOP2 ;
LOOP3 ;
LOOP4 ;
LOOP5 ;
EXIT ;
PRT ;PRINT FOR ENTRIES IN AUPNPAT NOT IN DPT
PRT1 ;PRINT FOR ENTRIES IN DPT NOT IN AUPNPAT
PRT2 ;PRINT FOR ENTRIES IN 3RD PARTY FILES BUT NOT IN AUPNPAT PRT3
PRINT FOR ENTRIES IN MEDICAID GLOBAL BUT NOT IN AUPNPAT

XBPATSE - SEARCH ROUTINES FOR PATCHES

Search Routines for Patch Versions.

MAIN ;
INIT ;
RSEL ;
DEVICE ;
QUE ;
SRCH ;
SRCH1 ;
SRCH2 ;
SCHDR ;
PRT ;
HDR ;
HDR1 ;
HDR2 ;

EXIT ;

XBPFTV - RETURN POINTER FIELD TERMINAL VALUE

NOTE TO PROGRAMMERS; Use entry point PFTV. Do not use the first line of this routine, as pending initiatives in MDC might make a formal list on the first line of a routine invalid. GTH 07-10-95

Given a file number, file entry number, and variable name into which the results will be placed, return the terminal value after following the pointer chain.

U must exist and have a value of "^"

Formal list:

1) F = file number (call by value)
2) E = file entry number (call by value)
3) V = variable for results (call by reference)

Scratch vars:
D = Flag, 1 = Done, 0 = continue
G = Global for file F.
*** NO ERROR CHECKING DONE ***
PFTV(F,E,V) ;PEP - Return Pointer Field Terminal Value.
START ;
TRACE ; FOLLOW POINTER CHAIN

XBPKDEL - REMOVE OPTIONS, INPUT, SORT, PRINT TEMPLATE _____

XBPKNSP must be set to the namespace, e.g. "AICD" if this routine is called from a pre-init.

If you want security keys deleted, set XBPKEY=1 if this routine is called from a pre-init.

Call LIST^XBPKDEL to list all namespaced options, templates, etc.

Call RUN^XBPKDEL to delete all namespaced options, templates, etc.

The RUN and LIST entry points are for programmer use and are not to be called from a pre-init. Pre-init calls XBPKDEL directly with variables set as indicated above.

START ;
A ;
ASK ;ASK USER IF WANTS TO CONTINUE
DELETE ;
LIST ; ENTRY POINT FOR LISTING NAMESPACED ITEMS
LIST2 ;

PAUSE ; Screen control for LIST
RUN ; ENTRY POINT FOR ACQUIRING CONTROL ARGUMENTS AND DOING DE GETNSP ; CODE TO ACQUIRE NAMESPACE
GETKEY ; CODE TO ACQUIRE SECURITY KEY FLAG
EOJ ;

XBPOST - XB/ZIB INSTALLATION POSTINIT

ORD101 ;
MGR ;
DESC ;

XBPRE - PREINIT, CHK RQMNTS, ETC.

C(X,Y) ; Center X in field length Y/IOM/80.
SORRY ;

XBRESID - CLEAN UP RESIDUAL ENTRIES IN ^DD

This routine deletes residual entries in ^DD by a range of dictionary numbers. A residual entry is one that has no parent. The process is reiterative, so an entry who has a parent in ^DD, but the parent is deleted because it has no parent, will also be deleted. The parent of an entry in ^DD is defined as another entry in ^DD for sub-files, and an entry in ^DIC for primary files.

The range of dictionary numbers is inclusive but residual entries for the high file number will not be deleted at the sub-file level. This is because sub-files are numbered with the primary file number with decimal numbers appended. The terminating check is ^DD entry greater than high file number specified, so by definition all sub-files for the high number are greater than the high number.

This routine can be called by another routine by setting XBRLO and XBRHI and then D EN1^XBRESID.

START ;
LO ;
HI ;
EN1 ;
PEP - Clean residual entries in ^DD(. HiLo file numbers RESID ;
LOOP ;
CHK ;
EOJ ;

XBRLL - LIST ROUTINE LINES WITH LENGTHS

This routine lists a single routine line by line noting the length of the line plus the cumulative character count.

START ;

XBRPRTBD - ROUTINE PRINT

This functionality has been moved to ZIBRPTRD because of the use of non-standard $Z special variables. The below GO is provided for backwards compatibility.

XBRPTL - PRINT ROUTINE TO FIRST LABEL

This routine prints selected routines down to the first label.

START ;
PRINT ;
TOP ;
PAGE ;
EN ;
PEP - Print routines down to first label.

XBRSIZ - List routine names and sizes w/overall totals

List routine names, sizes, and total bytes.

START ;
SELD ;
WAIT ;
DET ;
A1 ;
PRT ;
EXIT ;

XBRSRCH - SEARCH DD FOR CALLED ROUTINES

This routine searches a dictionary for called routines, excluding %DT*, DIC, DIK, and DIQ.

START ;
CHECK ; CHECK FILES UNTIL ALL DONE
LIST ; LIST ROUTINE NAMES
EOJ ;

XBRSRCH1 - COMMON CHECK LOGIC

Part of XBRSRCH

CHECK ; EXCLUDE ^%DT,^DIC,^DIK,^DIQ, AND GLOBALS

XBRSRCH2 - SEARCH INPUT TRANSFORM FOR ROUTINES

Part of XBRSRCH

START ;
EN ;EP - Search for routines in Input Transforms.
SBTRACE ; CHECK ALL SUB-FILES
SBTRACE2 ;
FILE ; CHECK ONE FILE OR SUB-FILE
FIELD ; Check Field's Input Transform.
WRITE ;
LIST ; List Routine Names.
EOJ ;

XBRSRCH3 - SEARCH OUTPUT TRANSFORM FOR ROUTINES

Part of XBRSRCH

START ;
EN ;EP - ENTRY POINT FOR CALLING ROUTINES
SBTRACE ; CHECK ALL SUB-FILES
SBTRACE2 ;
FILE ; CHECK ONE FILE OR SUB-FILE
FIELD ; CHECK FIELD'S INPUT TRANSFORM
WRITE ;
LIST ; LIST ROUTINE NAMES
EOJ ;

XBRSRCH4 - SEARCH XREFS FOR ROUTINES

Part of XBRSRCH

START ;
EN ;EP - ENTRY POINT FOR CALLING ROUTINES
SBTRACE ; CHECK ALL SUB-FILES
SBTRACE2 ;
FILE ; CHECK ONE FILE OR SUB-FILE
FIELD ; CHECK FIELD'S XREFS CHKXREF ; CHECK ONE XREF
CHKSK ; CHECK XREF SET/KILL WRITE ;LIST ; LIST ROUTINE NAMES
EOJ ;

XBRSRCH5 - SEARCH MISCELLANEOUS FOR ROUTINES

Part of XBRSRCH
START ;
EN ;EP - ENTRY POINT FOR CALLING ROUTINES
FILE ; CHECK ONE FILE
FIELD ; CHECK ONE FIELD
FIELD2 ;
FILE2 ;
WRITE ;
LIST ; LIST ROUTINE NAMES
EOJ ;

XBRXREF - RE-XREF SELECTED XREFS

This routine re-xrefs selected cross references for a file. The xrefs are killed at the highest level and then reset. This is very different from what FileMan does when you RE-INDEX a field. FileMan does a logical kill and then sets the new xrefs. The reason for this is multiple fields may set the same xref so you would want to kill only the ones set by the field being RE-INDEXed. You must re-xref all fields that set any one of the xrefs being killed and reset, unless the xref is set the same from multiple fields. Very hard to explain. If you do not understand the problem, you probably shouldn't be running this routine.

This routine executes an entry point in ^DIK to build the xref logic for all xrefs on the file. It then deletes the logic for all xrefs not selected, and executes another entry point in ^DIK to actually xref the file.

TRIGGERS are very complex animals, which do not have a xref to kill plus may be conditional and may have no affect on the SET side at all. Because of the uncertainties here this routine will not do TRIGGERS.

Xrefs at the multiple level would require a $O through the data global to kill, therefore, sub-file xrefs are not selectable unless they are cross-referenced to the whole file.

START ;
EN ; EXTERNAL ENTRY POINT
SETUP ; SETUP XREFS FOR ^DIK
SETUPB ; KILL OFF DIK FILE/SUB-FILES NOT NEEDED
SETUPC ; KILL OFF DIK FIELDS NOT NEEDED
SETUPD ; KILL OFF DIK XREFS NOT NEEDED
KILL ; KILL XREFS
KILL2 ;
KILL3 ;
XREF ; SET XREFS FOR ALL ENTRIES
EOJ ; EOJ HOUSEKEEPING

XBRXREF2 - INITIALIZATION ROUTINES FOR DRIVER

Part of XBRXREF

START ;
INIT ;EP - INITIALIZATION
GETFILE ;EP - GET FILE TO BE RE-XREFED
BLDXRT ;EP - BUILD XREF TABLE GETFIELD ; GET FIELD TO XREF
GFRCR ; GET FIELD RECURSION RECURSE ;XREFS ; DISPLAY XREFS FOR FIELD
XREFS2 ; DISPLAY XREFSGETXREF ; GET XREFS FROM FIELD
INFOSAVE ; GET XREF/NODE/PIECE INFO AND SAVE
CONFIRM ;EP - GET USER CONFIRMATION
CONFIRM2 ;
CONFIRM3 ;

XBSAUD - SET AUDIT AT FILE LEVEL

This routine sets 'audit' on at the file level for selected files

START ;
EOJ ;

XBSAUTH - SET AUTHORITIES

This routine sets FileMan dictionary authorities: "AUDIT" "DD" "DEL" "LAYGO" "RD" "WR"

START ;
ASK ;
ASK2 ;
GETAUTH ; GET DICTIONARY AUTHORITIES
PRTAUTH ; PRINT DICTIONARY AUTHORITIES
PROCESS ;
P2 ;
P2SETS ;
P2ERR ;
EOJ ;

XBSFGBL - RETURN SUBFILE GLOBAL REFERENCE

NOTE TO PROGRAMMERS; Use entry point EN. Do not use the first line of this routine, as pending initiatives in MDC might make a formal list on the first line of a routine invalid. GTH 07-10-95

Given a file or subfile number and global reference form, this routine will return the global reference in the form specified.

F (form) is optional but if passed should equal 1 or 2. If F is not passed the default form will be 1.

F = 1 will be in the form ^GLOBAL(DA(2),11,DA(1),11,DA,
F = 2 will be in the form ^GLOBAL(D0,11,D1,11,D2,

Formal list:
1) S = subfile number (call by value)
2) G = global reference (call by reference)
3) F = global reference form (call by value) ;

*** NO ERROR CHECKING DONE ***

START ;
BACKUP ; BACKUP TREE
NOPARENT ; for no parent
DIC(S) ;PEP - Extrinsic entry to return root global from FILE EN(S,G,F)
PEP - RETURN SUBFILE GLOBAL REFERENCE

XBSITE - SET "DUZ(2)"

L1 ;
B1 ;
ASK ;
SET ;
PEP - Request Set of DUZ(2) from applications.
ERRMSG ;
ERRMSG1 ;
ERRMSG2 ;

XBSUMBLD - ROUTINE INTEGRITY CHECK GENERATOR

This routine requests the user to select a set of routines generates an integrity checking routine for the selected routine. The user is asked to enter the name of the generated routine. The VA's equivalent routine is XTSUMBLD, which will also create integrity checking routine(s).

XBTM - TECH MANUAL : MAIN

This routine, and subsequent routines in the XBTM* namespace, produce a technical manual from information contained in the package. The manual is approximately 80 pages. All, or individual chapters can be printed.

```
SEL ;
DEV ;
K ;
START ;EP - TaskMan.
BODY ;
INDEX ;
CONTENTS ;
END ;
AT ;
TEXT(XBLAB) ;
PR(X) ;EP
INDX(X) ;
HDR(XB) ;
TOF ;EP
MAKEHDRS ; CONT(X) ;
CHAPS ;EP - From DIR
        1 ;;Facility Parameters
        2 ;;Area Office Parameters
        3 ;;Security Keys
        4 ;;Options
        5 ;;Fields in Files
        6 ;;Archiving and Purging
        7 ;;Callable Routines
        8 ;;External Relations
        9 ;;Internal Relations
        10 ;;How to GenerateOn-Line Documentation
        11 ;;Glossary
        12 ;;System Requirements
        13 ;;Installation notes
        14 ;;Enhancements
        15 ;;KILL ofUnsubscripted Globals

XBTM1 - TECH MANUAL : Facility Parameters

        DIWW NEW A,B,C D ^DIWW Q
        PR(X) NEW A,B,C D PR^XBTM(X) Q

XBTM10 - TECH MANUAL : ONLINE DOC

        PR(X) NEW A D PR^XBTM(X) Q

XBTM11 - TECH MANUAL : GLOSSARY

        PR(X) NEW A D PR^XBTM(X) Q
```

XBTM12 - TECH MANUAL : SYSTEM REQUIREMENTS

```
PR(X) NEW %,A,B,C,I D PR^XBTM(X) Q
```

XBTM13 - TECH MANUAL : INSTALL NOTES

```
PR(X) NEW %,A,B D PR^XBTM(X) Q
```

XBTM14 - TECH MANUAL : ENHANCEMENTS

```
PR(X) NEW %,A,B D PR^XBTM(X) Q
```

XBTM15 - TECH MANUAL : KILL UNSUBSCRIPTED GLOBALS

```
PR(X) NEW A D PR^XBTM(X) Q
```

XBTM2 - TECH MANUAL : AREA OFFICE PARAMETERS;

```
PR(X) NEW A D PR^XBTM(X) Q
```

XBTM3 - TECH MANUAL : SECURITY KEYS

```
SK(A) ; Printinfo on security keys for namespace A.
PR(X) NEW A,B,C D PR^XBTM(X) Q
```

XBTM4 - TECH MANUAL : OPTIONS

```
OP(A) ; Printinfo on options in namespace A.
PR(X) NEW A,B,C,T,XB D PR^XBTM(X) Q
```

XBTM5 - TECH MANUAL : FIELDS IN THE FILES

```
PR(X) NEW %,A,B,C,I,J D PR^XBTM(X) Q
ALPHA ;
FLD ;
```

XBTM6 - TECH MANUAL : ARCHIVING & PURGING

```
PR(X) NEW A D PR^XBTM(X) Q
```

XBTM7 - TECH MANUAL : ROUTINES

```
PR(X) NEW %,A,B D PR^XBTM(X) Q
```

XBTM8 - TECH MANUAL : EXTERNAL RELATIONS

```
PR(X) NEW %,A,B D PR^XBTM(X) Q
```

XBTM9 - TECH MANUAL : INTERNAL RELATIONS

```
PR(X) NEW A D PR^XBTM(X) Q
```

XBTMI - TECH MANUAL : INDEXED WORDS

```
ALPHA ;
FLD ;
1 ;;
```

XBTMPR - TECH MANUAL : PREFACE

```
PREFACE ;;
```

XBTMTI - TECH MANUAL : TITLE PAGE

```
TITLE ;;
```

XBUPCASE - UPCASE VALUE IN X

```
Uppercase value in X

START ;
```

XBVCH - routine to intelligently change variable name

```
START ;
V0 ;
V1 ;
V2 ;
SELROU ;
PROCESS ;
SHOVAR ;
PAGE ;
EXIT
EP - Paginate, print, kill, quit.
PRINT ;print variables and routines changed
PRINT1 ; Continue print
XBPG ;
EP PAGE CONTROLLER
XBHDR ;EP write page header
XBHD ;EP Write column header / message
EXBPG ;
```

XBVCH1 - CONTINUE VARIABLE CHANGER

PROCESS ;
DISPROU ;display routine list
LIN ;PROCESS LINE FROM TOP
SCAN0 ;
SCAN1 ;
DISP0 ;
DISP1 ;
SCAN ;
CHKMK ;
EDIT ;
DISPLAY ; display line
TAB ;
TAB1 ;
UPT ; SET TAB
BLDLIN1 ;
ACCEPT ;
%EDIT ; USE %E EDITOR
SAVE ; SAVE NEW ROUTINE TO DISK

XBVCHV - pull in variables and routines from a %IN

OPEN ;
FNAME ;
FNAME1 ;
ES ;
READ ;

XBVIDEO - SET VIDEO ATTRIBUTES

Set various video attributes. $X is saved and the cursor is returned to its original position through X IOXY (except certain attributes).

In addition to the attributes supported by ENDR^%ZISS, some color attributes are supported, and other mnemonics to provide for backward compatibility.

EN(XB) ;PEP - Set video attribute in XB. E.g. D EN^XBVIDEO("I 10 ;;5;;1;;1;;TEN PITCH;;IOPTCH10;;1

12 ;;5;;2;;2;;TWELVE PITCH;;IOPTCH12;;1
16 ;;12.1;;1;;250;;SIXTEEN PITCH;;IOPTCH16;;1
BLF ;;5;;9;;9;;BLINK OFF;;IOBOFF;;1
BLN ;;5;;8;;8;;BLINK ON;;IOBON;;1
CLR ;;6;;1;;1;;RESET;;IORESET;;1
CUP ;;8;;1;;1;;CURSOR UP;;IOCUU;;0

DTB ;;17;;2;;2;;DOUBLE HIGH BOTTOM HALF;;IODHLB;;0 DTP
;;17;;1;;1;;DOUBLE HIGH TOP HALF;;IODHLT;;0 HIF ;;7;;2;;2;;HI INTENSITY
OFF;;IOINORM;;1
HIN ;;7;;1;;1;;HI INTENSITY ON;;IOINHI;;1
HOM ;;5;;3;;3;;HOME CURSOR;;IOHOME;;0
IOF ;;1;;2;;2;;FORM FEED/CLEAR SCREEN;;;;0 RVF ;;5;;5;;5;;REVERSE VIDEO
OFF;;IORVOFF;;1 RVN ;;5;;4;;4;;REVERSE VIDEO ON;;IORVON;;1 ULF
;;6;;5;;5;;UNDERLINE OFF;;IOUOFF;;1
ULN ;;6;;4;;4;;UNDERLINE ON;;IOUON;;1
CYB ;;C;;3;;3;;CYAN BACKGROUND;;;;1 GRF ;;C;;1;;1;;GREEN FOREGROUND;;;;1
REB ;;C;;5;;5;;RED BACKGROUND;;;;1 WHF ;;C;;4;;4;;WHITE FOREGROUND;;;;1
YEF ;;C;;2;;2;;YELLOW FOREGROUND;;;;1

XBVK - LOCAL VARIABLE KILLER FRONT END

This is the front end for killing local variables in the namespaced parameter. Implementation specific routines are called from this routine. Those routines are in the ZIBVK* namespace.

This routine is intended to be called by applications that are thru executing, in order to KILL any remaining namespaced local variables. E.g., D EN^XBVK("AG") will KILL any local variables that exist in the AG namespace. Notice that if called in background, and the OS is not supported, the routine will quit, unpleasantly. If your implementation is other than what is supported, below, and your vendor has implemented all Type A extensions to the 1990 ANSI M standard, you can safely remove the two lines that check for OS, and use the existing call to the MSM-specific routine.

EN(XBVK)
PEP - Killvars in namespace of parameter variable.
MSM ; Micronetics Standard MUMPS.

XBVL - LOCAL VARIABLE LISTER FRONT END

This is the front end for listing local variables. Implementation specific routines are called from this routine. Those routines are in the ZIBVL* namespace.

MSM ; Micronetics Standard MUMPS.
MESSAGE ;
EP - Tell user of limitations.

XBVLINE - SET LINE TWO OF SELECTED ROUTINES

This routine asks user to select a set of routines, asks the user for the version number, package, and the date, and sets the second line of each routine.

The form of the version line will be as follows:

n;packagename;patchlevel;date E.G.
1.1;PCC DATA ENTRY;**1,2**; Sep 9, 1989

ZIBCKPKG - CHECK UCI FOR PACKAGE CONTENT

ZIS ; SELECT DEVICE
NOQUE ;
QUE ;
EXIT ;
EN1 ; ENTRY FOR SILENT OPERATION
INIT ; INITIALIZATION
EN ; COMMON INTERNAL ENTRY
KILLS ;
SCAN ;
CHKPKG ; CHECK FOR PACKAGE
NOTPKG ;
GETPKG ;
SKIP ;
GETNXT ;
DSPLY ;
SHOWNPR ;
SHOWLC ;
ASKIT ; ASK A YES/NO QUESTION
ASKIT2 ;
DSPHLP ;
QUEST ;
1 ;;Ignore non-package routines? ;;YES
2 ;;Display routine names containing lower case letters? ;;YES
3 ;;Display names of non-package routines?;; YES
4 ;;

ZIBCLU - GENERAL PURPOSE CLEAN UP UTILITY GLOBALS

This routine will initiate a job running ^%ZIBCLU0 in each U and then wait 5 seconds to elapse before getting the next UCI, skip the UCI this task is in and then run ^%ZIBCLU0 here. - %ZIBCLU0 will remove all dangling ^UTILITY,^XUTL,^ZUT EN, this routine is usually started viaTaskMan by scheduling the -ZIBCLU- option which runs this routine.

DSM ONLY - $ZU(ZIBI) returns <NOUCI> error at end of UCI LIS
MSM ONLY - $ZU(ZIBI) returns -null- value at end of UCI LIS ;

EN ;
ZT ;END OF UCI LIST

ZIBCLU0 - GENERAL PURPOSE CLEAN UP UTILITY GLOBALS

EN ;
MSM ; MSM specific look up of activeJOBs.
MER ;EP - MSM error trap.
DSM ; DSM specific look up of activeJOBs.
XUT ; Clenaup ^XUTL in MGR separate from otheUCIs.
GO ; $O down ^ZUT or ^UTILITY looking for jobnbr OR (namespac GOQ ;
N1 ; Check first subscript value and remove if its a dangling N1Q ;
N2 ; Process second node if first is non-numeric or ^UTILITY(" RM ; Remove dangling ^UTILITY node.
RMQ ;
OUT ;

ZIBERR - REPORT ERROR

The ERR PEP will return the name of the error currently in the implementation-specific intrinsic variable for errors. E.g., SET ERR=$$ERR^ZIBERR will return the value of $ZE on MSM systems.

PROGRAMMERS NOTE: Entry point Z^ZIBNSSV() is recommend instead of this routine.

ERR() ;PEP - Implementation-specific error.
MSM ; Micronetics Standard MUMPS.

ZIBFIND - FIND MSM BLOCKS WITH CONTAIN SPECIFIC GLOBALS

MSM-specific utility for finding blocks which contain a specific GBL.

ZIBCC=common count,ZIBUC=unique count
ZIBCHAR=string of characters

```
S X="ERR^ZIBFIND",@^%ZOSF("TRAP") K X

GETINFO ;
ASK1 ;
ASK2 ;
ASK3 ;
LOOP ;
END ;
ERR ;
```

ZIBFMD - DISPLAY FILEMAN INSTALLATION DATA

```
SETUP ;
ZIS ;
NOQUE ;
QUE ;
EN ;EP - From TaskMan.  Common processing for tasked or direct SCAN ;
NMUCI ;
DSPLY ;
HDR ;
SHOW ;
DDPROT ;
SUMM ;
SHOWF ;
SHOWF2 ;
NODIC ;
SHOWF3(X) ;
EXIT ;
ERR ;
```

ZIBFR - LIST UCI'S FOR A GIVEN ROUTINE

Given a routine name, this routine searches all UCIs and reports the first line of the selected routine to the user.

```
EN ;
EX ;
ENQ ;
ZT ; ERROR TRAP
ZTQ ;
```

ZIBGCHAR - NONINTERACTIVE MODIFICATIONS OF GLOBAL C

Not all capabilities of the implementation-specific global characteristics routines are reflected in this routine.

The argument for each entry point is the unsubscripted name of the global whose

characteristics you want to change, with the circumflex present. If the call is successful, 0 is returned.

If the call is not successful, a positive integer is returned, and the cause can be retrieved at the ERR()

Examples:

```
S %=$$NOJOURN^ZIBGCHAR("^AUTTSITE")
I % W !,$$ERR^ZIBGCHAR(%)
```

```
KILLOK(ZIBGLOB) ;PEP - Allow kill of global. KILLNO(ZIBGLOB)
PEP - Prevent kill'ing of global. JOURN(ZIBGLOB)
PEP - Set Journaling to ALWAYS. NOJOURN(ZIBGLOB)
PEP - Set Journaling for global to NEVER. UCIJOURN(ZIBGLOB)
PEP - Journal when UCI is Journaled. PROCESS(ZIBFLAG,ZIBVAL)
MSM ; Micronetics Standard MUMPS.
ERR(Z)
PEP - Return cause of error.
1 ;;NO GLOBAL SPECIFIED IN PARAMETER
2 ;;GLOBAL DOES NOT EXIST
3 ;;OPERATING SYSTEM NOT SUPPORTED
4 ;;WRONG VERSION OF MSM'S ^%GCH
5 ;;BAD GLOBAL NAME
TEST ;
T1(AZHB) ;
T2(AZHB) ;
DATA ;
```

ZIBGCHR - SEARCH FOR CONTROL CHAR. IN GLOBALS

```
%GLCHR ;SEARCH FOR CONTROL CHAR. IN GLOBALS [ 04/15/85  9:13 A %ST ;
%STL ;
%SCR ;
IHS1 ;
%PAG ;
%ASKC ;
%OPT ;
%OPT1 ;
%DO ;
%START ;
%GET ;
%WT ;
%NEXT ;
IHS2 ;
IHS3 ;
```

%OUT ;
%OUT1 ;
%DSP1 ;
%WRT ;
%DSP2 ;
%CTL ;
%FIXO ;
%LST ;
%LIN ;
%SC ;
%HELP ;
%Q1 ;
%Q2 ;
%Q3 ;
%EX ;
%ERR ;
%END ;

ZIBGD - DIRECTORY OF SELECTED GLOBALS

This routine displays a selected range or sub-set of the global directory. Save, or copy, to the MGR UCI as %ZIBGD, or %AZGD.

ZIBGSVED - SAVE GLOBAL TO TAPE, DSM SPECIFIC

ASK ;
NOSELT ;
HELP ;
CART ;
TAPE ;
PROCESS ;
RETRY ;
S9 ;
WRITPROT ;
EXIT ;
GOODREW ;
CLOSE ;
END ;
SAVEDSM ;

ZIBGSVEM - SAVE GLOBAL TO MSM UNIX

ASK ;
NOSELT ;
HELP ;
DISK ; ----- Transfer TX Global to floppy disk.
UNIX ; ----- Transfer TX Global tounix file.
TAPE ;
CART ;
TAPETST ; ----- Transfer global to cartridge or 9-track.
S ;
MAGOPEN ;
SW ;
ERRMESS ;
EXIT ;
CLOSE ;
END ;
TAPETEST ;
WRITE ;
TESTERR ;
UUCPQ ;auto queue to UUCP subroutine, must have system ID in R SAVEMSM
EP - $QUERY through global, write to output. FOLLOW(Y,XBE) If Y follows XBE return 1. Else return 0.

ZIBGSVEP - SAVE GLOBAL TO DOS MEDIA

ASK ;
NOSELT ;
HELP ;
DISK ;TRANSFER TX GLOBAL TO FLOPPY DISK
DOS ;TRANSFER TX GLOBAL TO DOS FILE.
ERRMESS ;
END ;

ZIBGTOT - UTILITY, GLOBALS, FAST SAVE TO TAPE

EDE;PLAGERIZED FASTGTO & %GTO;2-6-88

TAPE ONLY;MODE CDT ONLY;NO PARTIAL GLOBALS;ONE FILE PER TAPE ; This routine creates a global save in the same format as ^%GTO. The difference is it uses $ZO which is much faster. Because tests showed no significant difference in speed between CDT and CAVL4 this routine accepts CDT only. It would be difficult, although not impossible, to allow partial global. Therefore, this routine will save complete globals only. This routine also always rewinds the tape before saving the global. That means only one file per tape but one file may use multiple volumes.

The primary purpose of this routine is to move globals from DSM to MSM. Therefore, the limitation of tape only.

START ;
EN ; EXTERNAL ENTRY POINT
GLOBALS ; PROCESS GLOBALS IN ^UTILITY($J,
GBL ; PROCESS ONE GLOBAL
NODES ; PROCESS ALL GLOBAL NODES AFTER THE FIRST ONE
EOG ; END OF ONE GLOBAL
EOT ; END OF TAPE
DSMEOT ; STANDARD DSM EOT HANDLING
MSMEOT ; FIX EOT FOR MICRONETICS
EOTERR ; ERROR TRAP WHILE EOT PROCESSING
GETGBLS ; GET GLOBALS TO SAVE
SETZE ; SET $ZE TO OTHER THAN <UNDEF>
SETZET ; ERROR TRAP FOR SETTING $ZE
GETDEV ; GET OUTPUT DEVICE
TAPECHK ; CHECK TAPE READY *** DSM *** MSMCHK ; CHECK IF OUTPUT FOR MICRONETICS
HEADING ; WRITE HEADING RECORDS ON OUTPUT DEVICE EOF ; END OF FILE
CLOSE ; CLOSE OUTPUT DEVICE
KILL ; KILL VARIABLES
SUICIDE ; IMPOSSIBLE ERROR

ZIBJRNI - AUTOINITIALIZE JOURNAL AREAS

This utility is used to initialize all except the active journal areas of a system. The STU entry point is used by the automatic partition reference for a particular configuration. This was developed for the PC Network Configuration and has only been tested using MSM-PC/386.

START ;
END ;
MAIN ;
EOJ ;
STU ;EP - Entry Point for when started up upon enteringonfig

ZIBNSSV - NONSTANDARD SPECIAL VARIABLES

Return Non-Standard ($Z) Special Variables.
E.g.: W $$Z^ZIBNSSV("ERROR") will write the contents of the error message most recently produced by the OS. These are the variables supported:

ERROR : Text of error message most recently produced.
LEVEL : Number of the current nesting level.

NAME : Name of routine currently loaded in memory.
ORDER : Data value of the next global node that follows the current global reference.
TRAP : Line label and routine name of the program that is to receive control when an error occurs.
VERSION : Name and release of M implementation.

Z(NSSV) ;PEP - Return Non-Standard ($Z) Special Variables.
MSM ; Micronetics specific Non-Standard Special Variables. MSMZE Q $ZE
MSMZL Q $ZL
MSMZN Q $ZN
MSMZO Q $ZO
MSMZR() ;PEP -MSM's last global reference.
MSMZT Q $ZT
MSMZV Q $ZV

ZIBPKGF - INSTALLATION STATUS REPORT

Q2 ;EP - From DIR
OPT ;EP - Set option in OPTION file.
START ;EP - FromTaskMan.
MAIN ;
ENDMAIN ;
Q ;
JDT NEW X1,X2 S X2=$E(DT,1,3)_"0101",X1=DT D ^%DTC S X=X+1,X=" SYTM ;

cmbsyb

ZIS NEW A,D,F,I,J,L,M,P,R,S,V D ^%ZIS Q
ZISC NEW A,D,F,I,J,L,M,P,R,S,V D ^%ZISC Q
10 ;;abr-ab
11 ;;bji-ao
20 ;;albisc
30 ;;akarea
40 ;;bilcsy
50 ;;okc-ao
51 ;;nsa-oa
60 ;;phx-ao
61 ;;cao-as
70 ;;pordps
80 ;;nav-aa
00 ;;tucdev

ZIBPKGP - PROCESS IMPLEMENTATION STATUS FILES

OPT ; Set option in OPTION file.
IN ;EP - From TaskMan.
MAIN ;
PKG ;
RPI ;
ENDMAIN ;
Q ;
FAC ;
FILE NEW A,D,DD,DO,F,L,P,S,T,V,Z D FILE^DICN K DIC Q
DIE NEW A,D,F,L,P,S,T,V,Z D ^DIE K DA,DR,DIE Q
XMB NEW A,D,F,L,P,S,T,V,Z D ^XMB Q
ZIS NEW A,D,F,L,P,S,T,V,Z D ^%ZIS Q
ZISC NEW A,D,F,L,P,S,T,V,Z D ^%ZISC Q

ZIBRD - DISPLAY MSM DIRECTORY OF SELECTED RTNS

Generate routine directory of selected routines. Save, or %RCOPY, this routine to the MGR uci, named as %ZIBRD. It may also be name %AZRD.

ZIBRER - REMOTE ERROR REPORTING

dpssyg AnyTimeplex 9600 .00-15 n:--n:--n:uucpdps word:uu
dpssyg Any ACU2400 FTS-505-262-6250 n:--n:--nuucpdps word ;
INTRO ;
OPT ;EP - Set option in OPTION file.
START ;EP - From TaskMan.
MAIN ;
ENDMAIN ;
Q ;DIE ; LOCK node and call DIE. Always called interactively.
JDT ; SET today's Julian date into ZIBJUDT.
OS ; The "OUT" DOS directory is retrieved from the RPMS SITE f PUB
/usr/spool/uucppublic/
EXPORT ;;C:\EXPORT\
SYTM ;;dpssyg
10 ;;abr-ab
30 ;;akarea
20 ;;albtrn
40 ;;bilcsy
11 ;;bji-ao
61 ;;cao-as
80 ;;nav-aa
51 ;;nsa-oa
50 ;;okc-ao
60 ;;phx-ao
70 ;;pordps

00 ;;tucdev
HELP(L) ;EP
DIR(O,A,B,Q,H,R)
EP Q1 ;
PP() ; Return pseudo prefix.
FREQ() ; Return frequency that option runs.

ZIBRER1 - REMOTE ERROR REPORTING, ETC.

INQ ;EP - From Option.
INQ1 ;
DMP ;EP - From Option.
DMPEND ;EP - Possible from ^C.
HDR ;EP - Print menu header.
PHDR ;EP - Print parent menu header.
SHDR ;EP - Screen header.
C(X) ;
RT ;EP

ZIBRERP - PROCESS ERRORS FROM FILES FOR RER

OPT ;
ZIB REMOTE ERR FILE PROCESSING;
START ;EP - FromTaskMan.
MAIN ;
BULLETIN ; ^TMP($J,"COUNT",namespace,$H-date,facility)=count
PURGE ;
Q ;
FILE NEW A,D,E,F,H,I,J,L,N,P,R,V KILL DD,DO S DIC(0)="L" D FIL DIE NEW A,D,E,F,H,I,J,L,N,P,R,V D ^DIE K DA,DR,DIE Q
DIK NEW A,D,E,F,H,I,J,L,N,P,R,V D ^DIK Q
DIR NEW A,D,E,F,H,I,J,L,N,P,R,V D ^DIR Q
XMB NEW A,D,E,F,H,I,J,L,N,P,R,V D ^XMB Q ZIS NEW A,D,E,F,H,I,J,L,N,P,R,V D ^%ZIS Q ZISC NEW A,D,E,F,H,I,J,L,N,P,R,V D ^%ZISC Q
ERR ;EP - Possible entry if error ($ZT), probably due toincom Q2 ;EP - From DIR
IMPORT() ;
LOC(X) ; Given a filename, return the Location IEN.

ZIBRNSPC - NAMESPACE PREVIOUSLY WRITTEN ROUTINES

INIT ;
START ;
EXIT ;
SETUP ; INITIALIZE UTILITY
PLOOP ;
SHEXT ;

EXC ;
^Y^DIE^DIC^DT^U^DUZ^DTIME^ZTSK^ZTDESC^ZTSAVE^ZTLOAD^ZTR PEXC
IO^D^XB^Z
RLOAD ; LOAD ROUTINE INTO GLOBAL
RLOADX ;;S ^TMP("ZIBRNSPC",$J,"T",0)=ZIBRRTN ZL @ZIBRRTN F ZIB RSAVE
SAVE GLOBAL TEXT AS ROUTINE
RSAVEX S ZIBRRTN=^TMP("ZIBRNSPC",$J,"T",0) ZR X "F ZIBRI=1:1 RNDX ;
PRINT INDEX OF ROUTINE CONVERSION
RNDXP ;
RFIX ; FIX ROUTINE LINES STORED IN GLOBAL
LSCAN ; SCAN LINE AND REPLACE VARIABLES
CMD ;
COPY1 ;
ADDOBJ ;
ADVPOS ;
EXPR ;
COPYOBJ ; COPY AN OBJECT, CHECKING FOR VARIABLES QSTR
COPY QUOTED STRING (INCLUDED DOUBLED QUOTES) ARGS
COPY ARGUMENTS -- 'DO' AND 'GO' SPECIAL CASES DGARG
PROCESS DO/GO ARGUMENTS CRPYTCOM
COPIES OBJECTS THRU ZERO-LEVEL COMMA CPYTKN
COPIES A TOKEN, MODIFYING PARENTHESIS LEVEL TSTOBJ
CONDITIONALLY REPLACES A VARIABLE NAME CHKPART
VERIFY MATCH WITH EXCLUSION PARTIAL NAME LIST VERPART
MANAGE PARTIAL MATCH VERCAND
MANAGE AUTO CANDIDATE SELECTION GETALT
GET ALTERNATE FOR PROPOSED CANDIDATE REPLACEMENT NAME

ZIBRPI - REMOTE PATCH INSTALLATION

For a description of this utility, see the text in routine ; ZIBRPI2.

D = Directory containing patch files
D("OUT") = Directory with results files
E = "Action" routine, named (A/B)9<namespace><patch_number>
F = Name of a file containing a patch
J = Today's Julian date
L = Facility's Pseudo Prefix
N = Namespace derived from the name of the file
O = Operating System, and OS-specific commands
P = PACKAGE file IEN
V = Version derived from the name of the file
W = Work file
OPT ;EP - Set option in OPTION file. Called by a programmer.
START ;EP - FromTaskMan.

```
MAIN ;
ENDMAIN ;
Q ;
DIE NEW D,E,F,I,J,L,N,O,P,V,W D ^DIE Q
DIR NEW D,E,F,I,J,L,N,O,P,V,W D ^DIR Q
DTC NEW D,E,F,I,J,L,N,O,P,V,W D ^%DTC Q
FILE NEW D,E,F,I,J,L,N,O,P,V,W K DD,DO D FILE^DICN Q
HC(%) NEW D,E,F,I,J,L,N,O,P,V,W S %=$$JOBWAIT^%HOSTCMD(%) Q
RTN(%) NEW D,E,F,I,J,L,N,O,P,V,W D @(%) Q
XMD NEW D,E,F,I,J,L,N,O,P,V,W D ^XMD Q
ZIS NEW D,E,F,I,J,L,N,O,P,V,W D ^%ZIS Q ZISC NEW D,E,F,I,J,L,N,O,P,V,W D
^%ZISC Q HFS ;
JDT ;
OS ; The "IN" directory is retrieved from the OPTION entry.
SYTM ;;dpssyg
PATTERN ;;2.4L.2"_"4N1"."1"p"1.2N
WORK ;;ZIBRPI.WRK
PUB ;;/usr/spool/uucppublic/
NS ;;ZIB_P
```

ZIBRPI1 - REMOTE PATCH INSTALLATION (1)

```
OPT ;EP - Set option in OPTION file.
DIE NEW D,E,F,I,J,L,N,O,P,V,W D ^DIE Q DIR NEW D,E,F,I,J,L,N,O,P,V,W D ^DIR
Q
FILE NEW D,E,F,I,J,L,N,O,P,V,W K DD,DO D FILE^DICN Q
C(%) NEW D,E,F,I,J,L,N,O,P,V,W S %=$$JOBWAIT^%HOSTCMD(%) Q
XMD NEW D,E,F,I,J,L,N,O,P,V,W D ^XMD Q
ZIS NEW D,E,F,I,J,L,N,O,P,V,W D ^%ZIS Q
ZISC NEW D,E,F,I,J,L,N,O,P,V,W D ^%ZISC Q HFS ;
OS ;
SYTM ;;dpssyg
10 ;;abr-ab
11 ;;bji-ao
20 ;;albisc
30 ;;akarea
40 ;;bilcsy
50 ;;okc-ao
51 ;;nsa-oa
61 ;;cao-as
60 ;;phx-ao
70 ;;pordps
80 ;;nav-aa
00 ;;tucdev
```

ZIBRPI2 - REMOTE PATCH INSTALLATION (2)

GEN ; General description
SYSID ; Select system id's to receive result files.
DIRECT ;
ACTION ;
HELP(L) ;EP - From DIR

ZIBRPRTD - ROUTINE PRINT

This routine lists routines edited after given date.

BEGIN ;
RSEL ;
SDEV ;
F1 ;
F2 ;
CMT ;
CMT1 ;
START ;
EXIT ;
ERR ;EP - If error, from error trap.
FORMAT ;
F3 ;
F4 ;

ZIBRSEL - NONINTERACTIVE ROUTINE SELECT

Return the number of selected routines set into the indicated variable.

E.g.:
I '$$RSEL^ZIBRSEL("B-BZZZZZZZ","ARRAY(") W "NONE SELECTED"
If routines exists in the list or range, their name will be returned as the last subscript of indicated variable in the 2nd parameter. The default is ^TMP("ZIBRSEL",$J. If routine B exists, then node ^TMP("ZIBRSEL",$J,"B") will be null.

It is the programmer's responsibility to ensure the name of the array is correctly formed.

Variables used:
X = String indicating list or range of routines.
Y = String indicating variable into which to set the selected routines.
Default = ^TMP("ZIBRSEL",$J, ; F = First routine, if range. L = Last routine, if range.
N = Number of routines returned.
Q = Quote character.

RSEL(X,Y) ;PEP - Select a list or range of routines, return in MSM.

ZIBRUN - CHECK FOR ACTIVE ROUTINE IN A SPECIFIC UCI

EN ;
MSM ; MSM specific look up active JOBs.
MER ; MSM error trap.
DSM ; DSM specific look up active JOBs.
CK ; Check %ZIB("JOB TABLE) for match of ROUTINE^UCI.
OUT ;

ZIBSSD - SHUTDOWN FOR MSM-PC

This utility is used as the nightly shutdown routine and is initiated by the scheduled option AZSJ SHUTDOWN. It was developed for the PC Network configuration.

START ;

ZIBTCP - TCP PRINT TEST

This routine must be DONE from the CLOSE EXECUTE when printing to a TCP printer. See below for further documentation.

H = Host IP address
P = Port number
I = Counter

EN ;
EXIT ;
EN1 ;
OPEN ;OPEN HOST FILE

ZIBVCHV - READ VARS AND RTNS FROM A %INDEX

OPEN ;
FNAME ;
FNAME1 ;
ES ;
READ ;

ZIBVGE - CHANGE VOLUME GROUP NAME FOR NETWORK CONFI

This routine changes the Volume Group Name in ^SYS (and ^%ZOSF. It is used for the rapid change of the volume group names in the PC Network Configuration. It has only been tested using MSM-PC/386.

START ;
END ;
ASK ;
ASKX ;
VGLABEL ;
VG ;
VOLUME ;
SGCNFG ;
ZOSF ;
ZOSFX ;
EOJ ;

ZIBVKIL - BUILD A KILL VARIABLE ROUTINE

Build a name space variable killer routine in ^s. KVAR.

Select a %INDEX host file summary from which to build the routine. Select a name space for the variables and the routine to be built. Enter any package wide variables.

Add D ^.ns.VKL0 to all menu exit actions where package variables are to remain.
Add D KILL^XUSCLEAN to the exit of all other menus.

NS ;
PKGVAR ;
KROU ;
S ;
EXIT ;
1 ;;S XBNUM=0 X XBLD(2),XBLD(3),XBLD(4),XBLD(6) ZS @XBROU W !,
2 ;;ZR S XBROU=XBNS_"VKL"_XBNUM,X=XBROU_" ; - kill variables"
3 ;;S XBVAR=XBNS,XBHD=" K ",X=" K " F S XBVAR=$O(^XBVROU($J,"
4 ;;S XBLX=$L(X) I XBLX>3 S X=$E(X,1,XBLX-1) ZI X
5 ;;S XBNUM=XBNUM+1 S X=" D ^"_XBNS_"VKL"_XBNUM ZI X ZS @XBROU
6 ;;I $D(XBKROU) S X=XBKROU ZI X W !,"ADDING ",X,!

ZIBVKMSM - KILL VARIABLES

This routine kills variables in the namespace of the variable passed in the parameter. This routine is accessedthru the front end routine XBVK.

EN(ZIBVKNS) ;EP - KILL Local variables in the passed namespace

ZIBVLMSM - LIST MSM VARIABLES

This routine lists variables that begin with the string entered by the user. Selection of variables is case sensitive.

This routine is specific to Micronetics. It will work with any M implementation that has all Type A extensions to the 1990 M ANSI standard implemented. The front end routine, XBVL, stops if any other than an MSM implementation is encountered.

Routine provided by DonEnos, OHPRD, 5 Feb 96.

START ;
LOOP ; WRITE NAME SPACED VARIABLES UNTIL USER IS THROUGH
NMSPACE ; LIST VARIABLES IN NAME SPACE
ALL ; LIST ALL VARIABLES
QUERY ; $Q THROUGH ARRAYS
WRITE ; WRITE ONE VARIABLE NAME AND VALUE
READ ; READ USER INPUT
HELP ; DISPLAY HELP MESSAGE
PAUSE ; PAUSE FOR USER

ZIBZUCI - SWAP UCI BETWEEN VOLUME SETS FOR MSM-UNIX

SAVE THIS ROUTINE AS %ZUCI IN THE MGR UCI

This utility permits switching between UCIs and Volume Groups when run in programmer mode. D ^%ZUCI

If switching to a UCI in a Volume Group other than the System Volume Group (0), you must enter either the Volume Group Number or Volume Group Name along with the UCI Number or Name. A 'help' display identifies all UCIs and Volume Groups that are currently mounted. Use a '?' for 'help'.

A routine may be tied to the UCI,VOL switch. This routine will be called immediately after the UCI,VOL switch occurs.

EN ; ENTRY - Ask for [UCI,VOL]
ASK ; Get new UCI name or number.
ED . ; Edit input from user.

HLP .. ; Display UCI list.
EXAMP .. ;
VER .. ; Verify if UCI,VOL exists.
SW . ; SWITCH TO NEW UCI
EX ; EXIT
ZT ; Error trap for DSM.

Documented entry points:

OSNO^XB : ;EP
PAUSE^XB : ;EP
RCHK^XB : ;EP - Check Existence of Routine in X
EDIT(XBFORM,XBWPDIC,XBWPFLD)^XBARRAY : ;EP Edit a Form
GEN(XBFORM,XBWPDIC,XBWPFLD,XBREF,XBFMT,XBLAST)^XBARRAY : ;EP
** EN(FILE,FIELD,XREF,UNIQUE)^XBCSPC : ; EXTERNAL ENTRY POINT TO
AL END0^XBDBQUE : ;EP - from compute cycle when XB("RP") EXISTS
END1^XBDBQUE : ;EP clean outxb as passed in
ENDC^XBDBQUE : ;EP - end computing cycle
%XY^XBDIQ1 : ;EP - set %X & %Y to format DICFNGL(X)^XBDIQ1 : ;EP - set
XBFN & XBGL0 return 1 error ENDIQ1^XBDIQ1 : ;EP - call EN^DIQ1
ENDIQ1X^XBDIQ1 : ;EP - to call DIQ1 with new ENDIQ1XN^XBDIQ1 : ;EP
EXIT^XBDIQ1 : ;EP
LEVELS^XBDIQ1:
EP - setup XB_FN_DA_DR_FLD arrays for upperlev PULLDIQ1^XBDIQ1 :
EP - PULL FROM ^UTILITY("DIQ1",$J) SETDIQ1^XBDIQ1 :
EP - set DR(fn and DA(fn arrays for DIQ1 EN1^XBDSET :
EP - Non-interactive selection of range of files. RANGE2^XBDSET :
LABEL FOR EXTERNAL ENTRY POINT EN1 START^XBENHANC :
EP - TaskMan.
EN^XBFLD :
EXTERNAL ENTRY POINT FORMAT^XBFLD :
EP - select format PAGE^XBFLD :
EP - PAGE HEADERS
EDIT(XBFORM,XBWPDIC,XBWPFLD)^XBFORM :
EP Edit a Form
GEN(XBFORM,XBWPDIC,XBWPFLD,XBREF,XBFMT,XBLAST)^XBFORM :
EP ** SCAN^XBFORM1 :
EP - scan for X
SUB(XBV1,XBLINX)^XBFORM1 :
EP extrinsic to return new output t XSUB^XBFORM1 :
EP - do it
HELP^XBGCMP : ;EP -Dooda about the utility
GL^XBHEDD10 : ;EP - List Globals in ASCII order DTYPE1^XBHEDD5 :
EP - Called by DATATYPE^XBHEDD4 DTYPE2^XBHEDD5 :
EP - Called by DATATYPE^XBHEDD4 PAGE^XBHEDD5 :

EP WORD^XBHEDD5 : ;EP -String=text - Prints a string in lines of 5 INIT^XBHEDD7
EP MULT^XBHEDD7 : ;EP PRINT^XBHEDD7 :
EP SCROLL^XBHEDD7 :
EP - Adjust scroll rate TXT^XBHEDD7 :
EP GLOBAL^XBHEDD9 :
EP - Find File when user enters global RTRN^XBHFMAN :
EP ----- If interactive, ask user to press RETURN START^XBHFMAN :
EP ----- From TaskMan. END^XBHFMAN1 :
EP - Paginate, close, kill, quit. PR(X)^XBHFMAN1 :
EP - Process one line of text. TOF^XBHFMAN1 :
EP - Move to bottom of page, print footer, p INDX^XBHFMAN2 :
EP PREFACE^XBHFMAN2 :
EP TITLE^XBHFMAN2 :
EP LIST^XBKERCLN :
ENTRY POINT FOR LISTING NAMESPACED ITEMS
BROWSE^XBLM :
EP - support for device P-BROWSE-HFS
EN^XBLM :
EP -- main entry point for XB DISPLAY EXIT^XBLM :
EP -- exit code
EXPND^XBLM :
EP -- expand code HDR^XBLM :
EP -- header code HELP^XBLM :
EP -- help code
INIT^XBLM :
EP -- init variables and list array OPENROOT(XBY)^XBLM :
EP - return OPen RooT form of XBY .. for % EN(XBROU,XBREF)^XBLMGUI :
EP ** USING XBROU print to a LIST^XBPKDEL :
ENTRY POINT FOR LISTING NAMESPACED ITEMS RUN^XBPKDEL :
ENTRY POINT FOR ACQUIRING CONTROL ARGUMENTS & EN^XBRSRCH2 :
EP - Search for routines in Input Transforms. EN^XBRSRCH3 :
EP - ENTRY POINT FOR CALLING ROUTINES
EN^XBRSRCH4 :
EP - ENTRY POINT FOR CALLING ROUTINES EN^XBRSRCH5 :
EP - ENTRY POINT FOR CALLING ROUTINES EN^XBRXREF :
EXTERNAL ENTRY POINT
BLDXRT^XBRXREF2 :
EP - BUILD XREF TABLE CONFIRM^XBRXREF2 :
EP - GET USER CONFIRMATION GETFILE^XBRXREF2 :
EP - GET FILE TO BE RE-XREFED INIT^XBRXREF2 :
EP - INITIALIZATION CHAPS^XBTM :
EP - From DIR PR(X)^XBTM :
EP START^XBTM :
EP - TaskMan. TOF^XBTM :
EP EXIT^XBVCH :

EP - Paginat, print, kill, quit. XBHD^XBVCH :
EP Write column header / message XBHDR^XBVCH :
EP write page header
XBPG^XBVCH :
EP PAGE CONTROLLER
MESSAGE^XBVL :
EP - Tell user of limitations. MER^ZIBCLU0 :
EP - MSM error trap.
EN^ZIBFMD :
EP - From TaskMan. Common processing for tasked or SAVEMSM^ZIBGSVEM :
EP - $QUERY thru global, write to output. EN^ZIBGTOT :
EXTERNAL ENTRY POINT STU^ZIBJRNI :
EP - Entry Point for when started up upon enterin OPT^ZIBPKGF :
EP - Set option in OPTION file. Q2^ZIBPKGF :
EP - From DIR START^ZIBPKGF :
EP - From TaskMan. IN^ZIBPKGP :
EP - From TaskMan. DIR(O,A,B,Q,H,R)^ZIBRER :
EP HELP(L)^ZIBRER :
EP OPT^ZIBRER :
EP - Set option in OPTION file. START^ZIBRER : ;EP - From TaskMan.
DMP^ZIBRER1 :
EP - From Option. DMPEND^ZIBRER1 :
EP - Possible from ^C. HDR^ZIBRER1 : EP - Print menu header. INQ^ZIBRER1 :
EP - From Option. PHDR^ZIBRER1 :
EP - Print parent menu header. RT^ZIBRER1 :
EP SHDR^ZIBRER1 :
EP - Screen header. ERR^ZIBRERP :
EP - Possible entry if error ($ZT), probably due Q2^ZIBRERP :
EP - From DIR START^ZIBRERP :
EP - From TaskMan. OPT^ZIBRPI :
EP - Set option in OPTION file. Called by a programmer. START^ZIBRPI :
EP - From TaskMan. OPT^ZIBRPI1 :
EP - Set option in OPTION file. HELP(L)^ZIBRPI2 :
EP - From DIR ERR^ZIBRPRTD :
EP - If error, from error trap. EN(ZIBVKNS)^ZIBVKMSM :
EP - KILL Local variables in the passed .

Compiled/Generated routines:

9. EXTERNAL RELATIONS

There are several published entry points that may be called from other packages. Some XB/ZIB routines were programmed to be available to call from the top of the routine, and are so noted in the routine.

Published entry points and supported routines:

EP^XBCLM(STR) :
PEP - ColumnLister
^XBCLS : Clears the screen
KILL^XBDAD0 :
PEP - KILL D0, D1, ETC.
^XBDAD0 : Returns DA array for D0,D1, etc. or vice versa
^XBDATE : Limits selected routines to those edited after given d ^XBDBQUE : Double Q'uing shell handler
^XBDIE : Exclusive NEW of Kernel vars for nesting DIE calls ^XBDIFF : Returns difference between two date/times DIC^XBDIQ1(XBFN) :
PEP - Extrensic entry to return DIC fromgl EN^XBDIQ1 :
PEP - Returns single entries ENM^XBDIQ1 :
PEP - get multiple entries ENP^XBDIQ1(DIC,DA,DR,DIQ,XBFMT) :
PEP - param pass into EN ENPM^XBDIQ1(DIC,DA,DR,DIQ,XBFMT) :
PEP - param pass into EN PARSE^XBDIQ1(XBDA) :
PEP - parse DA literal into the array VAL^XBDIQ1(DIC,DA,DR) :
PEP - extrinsic pull a value for a file VALI^XBDIQ1(DIC,DA,DR) :
PEP - extrinsic pull a value for a file DIR^XBDIR(O,A,B,T,Q,H,R) :
PEP - Extrinsic interface to ^DIR. EN^XBENHANC(XB) :
PEP - XB = Namespace of package to print enhanced FLD^XBFDINFO
(FILE,FIELD,ROOT) :
PEP - Return information about ^XBFMK : Kills variables left around by FileMan
EN1^XBFRESET :
PEP - Interactive entry, files already selected. EN2^XBFRESET :
PEP - Non-interactive entry, files already selected C^XBFUNC(X,Y) :
PEP - Center X in field length Y/IOM/80. CV^XBFUNC(X) :
PEP - Given a Namespace, return current version. DECFRAC^XBFUNC(X) :
PEP - Convert Decimal to Fraction (X contains
EXTSET^XBFUNC(FILE,FIELD,INTVAL) :
PEP - Get Extnl Field Value FNDPATRN^XBFUNC(STR,PAT) :
PEP - Find pattern in string. Return GDT^XBFUNC(JDT) :
PEP - Return Gregorian Date, given Julian Date GETPATRN^XBFUNC(STR,PAT) :
PEP - Retrieve pattern from string. INTSET^XBFUNC(FILE,FIELD,EXTVAL) :
PEP - Get Intnl Field Value JDT^XBFUNC(XBDT) :
PEP - Return Julian Date, given FM date. LOC^XBFUNC() :
PEP - Return location name from file 4 based on USR^XBFUNC() :

PEP - Retn name of current user for ^VA(200. ROVAFFL^XBFUNC1(PROV,FORM) :
PEP - Retrieve provider affiliating PROVCLS^XBFUNC1(PROV,FORM) :
PEP - Retrieve Provider Class from PROVCLSC^XBFUNC1(PROV) :
PEP - Retrieve Provider Class Code give PROVCODE^XBFUNC1(PROV) :
PEP - Retrieve provider code PROVINI^XBFUNC1(PROV) :
PEP - Retrieve provider initials PCCPPAFF^XBFUNC2(XBVISIT,FORM) :
PEP - Return a visit's primary PCCPPCLC^XBFUNC2(XBVISIT) :
PEP - Return a visit's primaryprov PCCPPCLS^XBFUNC2(XBVISIT,FORM) :
PEP - Return a visit's primary PCCPPI^XBFUNC2(XBVISIT) :
PEP - Return a visit's primaryprovid PCCPPINT^XBFUNC2(XBVISIT) :
PEP - Return primary providerien i PCCPPN^XBFUNC2(XBVISIT) :
PEP - Return a visit's primaryprovid ^XBGSAVE : Generic global save for transmission globals EN^XBGXFR(FROM,TO,TALK) :
PEP - Transfer global trees. ^XBGXFR : Copies global to another global ^XBGXREFS : Returns xrefs for file/subfile,field XREF^XBGXREFS(FILE,FIELD,ROOT) :
PEP - Return x-ref info for a HELP^XBHELP(L,R,T) :
PEP - Display text at label L, routine R, EN^XBHFMAN(XBSEL) :
PEP ----- From application options, withina EN1^XBKD :
PEP - Variables XBKDLO, XBKDHI, XBKDDEL, XBKDTMP must EN2^XBKD :
PEP - Array ^UTILITY("XBDSET",$J) must exist when en ^XBKERCLN : Clean out kernel namespace items prior to install ^XBKTMP : KILL nodes in ^TMP(that have $J as 1st or 2nd subscript ^XBKVAR : Set minimum Kernelvars ARRAY^XBLM(XBAR,XBHDR) :
PEP Display an array that has (...,n, DIQ^XBLM(DIC,DA) :
PEP - Display DIC and DA after call to EN^DI FILE^XBLM(XBDIR,XBFN) :
PEP - pull up a file into the TMP global GUID^XBLM(XBROU,XBY) :
PEP give routine and target array for FM GUIR^XBLM(XBROU,XBY) :
PEP - give routine and target array SFILE^XBLM :
PEP - Select a host file for display. VIEWD^XBLM(XBROU,XBHDR) :
PEP ** USING XBROU print to a host file VIEWR^XBLM(XBROU,XBHDR) :
PEP ** USING XBROU print to a host file EN^XBLZRO :
PEP - List 0th node of pre-selected list of FileMan MAIL^XBMAIL(XBNS,XBREF) :
PEP - XBNS is namespace, XBREF is line EN^XBNEW(XBRT,XBNS) :
PEP XBRT=TAG^ROUTINE XBNS=variable list ^XBOFF : Set reverse video off ^XBON : Set reverse video on
PFTV^XBPFTV(F,E,V) :
PEP - Return Pointer Field Terminal Value. ^XBPKDEL : Delete parts of package, namespace in XBPKNSP EN1^XBRESID :
PEP - Clean residual entries in ^DD(. Hi/Lo file EN^XBRPTL :
PEP - Print routines down to first label. DIC^XBSFGBL(S) :
PEP - Extrinsic entry to return root global from EN^XBSFGBL(S,G,F) :
PEP - RETURN SUBFILE GLOBAL REFERENCE SET^XBSITE :
PEP - Request Set of DUZ(2) from applications. ^XBSITE : Ask user to select site to set DUZ(2) ^XBUPCASE :Upcases value in X EN^XBVIDEO(XB) :
PEP - Set video attribute in XB. E.g. D EN^XB EN^XBVK(XBVK) :

PEP - Killvars in namespace of parameter variable ^XBVL : List variables in the selected namespace ERR^ZIBERR() :
PEP - Implementation-specific error. ERR^ZIBGCHAR(Z) :
PEP - Return cause of error. JOURN^ZIBGCHAR(ZIBGLOB) :
PEP - Set Journaling to ALWAYS. KILLNO^ZIBGCHAR(ZIBGLOB) :
PEP - Prevent kill'ing of global. KILLOK^ZIBGCHAR(ZIBGLOB) :
PEP - Allow kill of global. NOJOURN^ZIBGCHAR(ZIBGLOB) :
PEP - Set Journaling for global to UCIJOURN^ZIBGCHAR(ZIBGLOB) :
PEP - Journal when UCI isJournale MSMZR^ZIBNSSV() :
PEP - MSM's last global reference. Z^ZIBNSSV(NSSV) :
PEP - Return Non-Standard ($Z) Special Variable RSEL^ZIBRSEL(X,Y) :
PEP - Select a list or range of routines, R ^ZIBRUN : Sets $T based on whether routine in X is running.

10. INTERNAL RELATIONS

XB/ZIB contains routines and entry points that are for use both interactively by programmers, and as calls from applications. The XB menu can be accessed from programmer mode through routine XB, i.e., DO ^XB.

11. HOW TO GENERATE ON-LINE DOCUMENTATION

The package documentation presented in this manual is extensive and should provide most site managers, ISC staff members, and developers with sufficient information.

This manual can be generated from programmer mode by DO'ing ^XBTM. This is a CPU-intensive routine. Please queue to TaskMan to run after hours, and expect approximately 150 pages of output.

12. GLOSSARY

13. SYSTEM REQUIREMENTS

Kernel v. 8.0
FileMan v. 20

14. KILL OF UNSUBSCRIPTED GLOBALS

Unsubscripted Globals KILLed in the package.

Routine	Line	Global