

CALLED BY:

group routine

CASES CSTA CSTS

WDDSDL

WDDSDL

This SUBROUTINE is number 5 in file WDBTCH.

routine to delete a data set from the WDM SFL with no user interact

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u>	<u>status</u>	<u>explanation</u>
		<u>type</u> <u>size</u>		
1	WDMSFL	I*4	I	Fortran unit number of WDM file
2	DSN	I*4	I	dataser number to be deleted
3	RETCOD	I*4	O	return code
				0 - data set successfully deleted
				-81 - data set does not exist

COMMON USAGE:

block name status

CFBUFF WIBUFF I

CALLS:

routine

WDDSK WDFCUP WDFDUP WDRCDL WDRCGO

CALLED BY:

group routine

CASES CASES
WDIMEX PRWMIM

WDDSRN

WDDSRN

This SUBROUTINE is number 6 in file WDBTCH.

routine to renumber data sets with no user interaction

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u>	<u>status</u>	<u>explanation</u>
		<u>type</u> <u>size</u>		
1	WDMSFL	I*4	I	Fortran unit number for WDM file
2	ODSN	I*4	I	old data-set number
3	NDSN	I*4	I	new data-set number
4	RETCOD	I*4	O	return code
				0 - renumber successfully completed
				-72 - old data set does not exist
				-73 - new data set already exists

COMMON USAGE:

block name status

CFBUFF WIBUFF M
CDRLOC PTSNUM I

CALLS:

routine

WDDSCK WDFDUP WDRCGO WDRCUP

CALLED BY:

group routine

CASES CASES

WDFLCL

WDFLCL

This SUBROUTINE is number 3 in file UTWDM.D.

Remove a WDM file from the open WDM buffer and adjust buffer accordingly.

ARGUMENTS:

		declaration			
<u>order</u>	<u>name</u>	<u>type</u>	<u>size</u>	<u>status</u>	<u>explanation</u>
1	WDMSFL	I*4		I	Fortran unit number of WDM file
2	RETCOD	I*4		O	return code
					0 - everything ok
					-87 - can't remove message WDM file from buffer

COMMON USAGE:

<u>block</u>	<u>name</u>	<u>status</u>
CFBUFF	WDMCNT	M
CFBUFF	WDMFUN	O
CFBUFF	WDMOPN	O

CALLS:

routine

CALLED BY:

group routine

CASES CASES

WDLBAX

WDLBAX

This SUBROUTINE is number 6 in file WDLBLE.

add a new data-set label, but no search attributes or data.

ARGUMENTS:

		declaration			
<u>order</u>	<u>name</u>	<u>type</u>	<u>size</u>	<u>status</u>	<u>explanation</u>
1	WDMSFL	I*4		I	watershed data management file unit number
2	DSN	I*4		I	data-set number
3	DSTYPE	I*4		I	type of data set
4	NDN	I*4		I	number of down pointers
5	NUP	I*4		I	number of up pointers
6	NSA	I*4		I	number of search attributes
7	NSASP	I*4		I	amount of search attribute space

8	NDP	I*4	I	number of data pointers
9	PSA	I*4	O	pointer to search attribute space

COMMON USAGE:

block name status

CFBUFF RECNO A
 CFBUFF WIBUFF O

CALLS:

routine

WDFCUP WDFDUP WDPTCL WDRCGO WDRCGX WDRCUP

CALLED BY:

group routine

CASES CSST
 WDIMEX PRWMIM
 WDLBLE WDLBAD

WDLGET

WDLGET

This SUBROUTINE is number 3 in file WDDL.G.

retrieve DLG header or coordinate pairs from WDM file

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u>	<u>status</u>	<u>explanation</u>
		<u>type</u> <u>size</u>		
1	WDM\$FL	I*4	I	Fortran unit number for WDM file
2	DSN	I*4	I	data-set number on WDM file
3	ITYPE	I*4	I	type of DLG info (1- LINE, 2- AREA, 3- NODE)
4	ATT1	I*4	I	major attribute value
5	ATT2	I*4	I	minor attribute value
6	LEN	I*4	I	max length of information being retrieved (4 byte words)
7	ID	I*4	M	id of information retrieved (0-either, 1-header, 2-data)
8	OLEN	I*4	O	actual length of output buffer
9	DLG\$BUF	R*4 (V)	O	buffer of information being retrieved
10	RETCOD	I*4	O	return code 2 - no more data in this group 1 - more of current id remaining 0 - DLG data retrieved successfully -81 - data set does not exist -82 - data set exists, but is wrong DSTYP -50 - major and minor attributes not found on this data set

COMMON USAGE:

block name status

CFBUFF WIBUFF A

CALLS:

routine

WDLBSP WDL LCK WDNXPS WDPTSP WDRCGO WDSCHK *

CALLED BY:

group routine

CASES CSDLG
UTEXPT PRWMDE

WDLLSU

WDLLSU

This SUBROUTINE is number 5 in file WDDL.G.

summarize label information for DLG data set

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u> <u>type</u> <u>size</u>	<u>status</u>	<u>explanation</u>
1	WDMSFL	I*4	I	Fortran unit number for WDM file
2	DSN	I*4	I	data-set number on WDM file
3	ILEN	I*4	I	maximum size of information buffers
4	OLEN	I*4	O	actual amount of information returned (<= ILEN)
5	TYPE	I*4 (V)	O	buffer of information types on DSN
6	ATT1	I*4 (V)	O	buffer of major attributes on DSN
7	ATT2	I*4 (V)	O	buffer of minor attributes on DSN
8	RETCOD	I*4	O	return code

1 - more groups on DSN
 0 - label summary returned successfully
 -81 - data set does not exist
 -82 - data set exists, but is wrong DSTYP

COMMON USAGE:

block name status

CFBUFF WIBUFF I

CALLS:

routine

WDLISP WDRCGO WDSCHK

CALLED BY:

group routine

CASES CSDLG
UTEXPT PRWMDE

WDNXDV

WDNXDV

This SUBROUTINE is number 5 in file UTWDMF.

Move to the next data position and return the integer equivalent of the data value.

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u> <u>type</u> <u>size</u>	<u>status</u>	<u>explanation</u>
1	WDMSFL	I*4	I	Fortran unit number for WDM file
2	DREC	I*4	M	record number of data on WDM file
3	DPOS	I*4	M	position of data on data record (both DREC and DIND)
4	DVAL	I*4	O	data value on WDM file

COMMON USAGE:

block name status

CFBUFF WIBUFF I

CALLS:

routine

WDNXPS

CALLED BY:

group routine

CASES	CSTXT				
MSEXPT	PRMSFE	PRMSME	PRMSPE	PRMSTE	
UTEXPT	PRMSAE				
UTWDMF	WMSIDP				
WDATMS	WADGDF	WADGRA	WADGVA	WADGDS	WADGHL
WDTBLE	WDTBSP				

WDSAFI

WDSAFI

This SUBROUTINE is number 11 in file WDATRB.

Given the attribute name SAFNAM, starting at attribute index SAIND, find the first attribute name which matches SAFNAM. Return the index of the next attribute which also matches SAFNAM, if one exists.

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u>	<u>status</u>	<u>explanation</u>
		<u>type</u> <u>size</u>		
1	MESSFL	I*4	I	Fortran unit number for message file
2	SAFNAM	C*1 (6)	I	character array containing attribute name to search for
3	SAIND	I*4	M	index of next matching attribute, if one exists, otherwise, index of matching attribute
4	SANAM	C*1 (6)	O	character array of first attribute name matching SAFNAM
5	RETCOD	I*4	O	return code
				-110 - attributes not found on message file
				-111 - attribute name not found (no match)
				-112 - more attributes exist which match SAFNAM

COMMON USAGE:

none

CALLS:

routine

ASRTC	CHRCHR	LENSTR	QUPCAS	WADDSI	WADGTN
-------	--------	--------	--------	--------	--------

CALLED BY:

group routine

CASES	CSATR
WDATRB	WDBSGX

WDSAGY

WDSAGY

This SUBROUTINE is number 1 in file WDATRB.

gets general detail information about specified attribute

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u>		<u>status</u>	<u>explanation</u>
		<u>type</u>	<u>size</u>		
1	MESSFL	I*4		I	message file unit number
2	SAIND	I*4		I	attribute index number
3	SANAM	C*1	(6)	O	name of search attribute
4	DPTR	I*4		O	pointer to other details of attribute
5	SATYP	I*4		O	type of attribute
6	SALEN	I*4		O	length of attribute
7	SARQWD	I*4		O	word containing attribute requirements by dsn type
8	SAUPFG	I*4		O	attribute update flag

COMMON USAGE:

none

CALLS:

routine

WADDSI WADGTL ZIPC

CALLED BY:

group routine

CASES CSATR
WDATMS WADGTN
WDATRB WDBSAC WDBSAI WDBSAR WDBSGX
WDIMEX PRWMEX

WDTBDL

WDTBDL

This SUBROUTINE is number 11 in file WDTBLE.

delete WDM table from WDM file

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u>		<u>status</u>	<u>explanation</u>
		<u>type</u>	<u>size</u>		
1	WDM SFL	I*4		I	Fortran unit number of WDM file
2	DSN	I*4		I	Table data-set number
3	TABNAM	C*16		I	Table data-set name
4	TABIND	I*4		I	Table identifier
5	RETCOD	I*4		O	Return code

COMMON USAGE:

block name status

CFBUFF WIBUFF M

CALLS:

routine

CALLED BY:

group routine

CASES CSTA

WDTBFX

WDTBFX

This SUBROUTINE is number 2 in file WDTBLE.

determines pointer to the specified table,
also returns its message file cluster and group number

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u>	<u>status</u>	<u>explanation</u>
1	WMSFLL	I*4	I	Fortran unit number of WDM file
2	DSN	I*4	I	table data-set number
3	TABIND	I*4	I	table identifier number
4	TABNAM	C*16	I	name of table
5	TBCNT	I*4	O	total number of tables in data set
6	LREC	I*4	O	label record number
7	TGRPPT	I*4	O	table group pointer
8	MFID	C*1 (2)	O	message file name id
9	TCLU	I*4	O	table message file cluster number
10	TGRP	I*4	O	table message file group number
11	NROW	I*4	O	number of rows in table
12	RETCOD	I*4	O	return code 0 - table found, pointer and other information returned

COMMON USAGE:

block name status

CFBUFF WIBUFF A

CALLS:

routine

WDRCGO WDTBFN WTBISP

CALLED BY:

group routine

CASES CSTA

WDTBSU

WDTBSU

This SUBROUTINE is number 6 in file WDTBLE.

get WDM table label info from WDM file table data set

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u>	<u>status</u>	<u>explanation</u>
1	WMSFLL	I*4	I	Fortran unit number of WDM file
2	DSN	I*4	I	WDM table data-set number
3	TABMAX	I*4	I	Maximum number of tables to get info about

4	TABBAS	I*4	I	Table base pointer, first group to get info about
5	TABCNT	I*4	O	Total number of tables found
6	TABNAM	C*1 (16,V)	O	Name of table
7	TABID	I*4 (V)	O	Id of table
8	TABDIM	I*4 (V)	O	Dimensions of table
9	PDATVL	I*4 (V)	O	Pointer to table data values
10	RETCOD	I*4	O	Return code, (+) if more tables than TABMAX

COMMON USAGE:

block name status

CFBUFF WIBUFF I

CALLS:

routine

COPYC WDRCGO WDSCHK

CALLED BY:

group routine

CASES CSTA

WDTBTM

WDTBTM

This SUBROUTINE is number 1 in file WDTBLE.

put WDM table template on WDM file, return parameters about table

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u>	<u>status</u>	<u>explanation</u>
		<u>type</u> <u>size</u>		
1	MESSFL	I*4	I	Fortran unit number of WDM file containing table definition
2	MFID	C*1 (2)	I	Message file name id of MESSFL
3	TCLU	I*4	I	Message file cluster containing table template
4	TGRP	I*4	I	Group number containing table template
5	WDM5FL	I*4	I	Fortran unit number of WDM file to put table template in
6	DSN	I*4	I	Data-set number to put table template in
7	TABIND	I*4	I	Table identifier number of new table
8	NROW	I*4	I	Number of rows in new table
9	TFLDS	I*4	O	Number of fields in table
10	TNUM	I*4 (4)	O	Number of each variable type in table(I-1,R-2,C-3,D-4)
11	TTYP	C*1 (30)	O	Type of each field in table(I,R,C,D)
12	TLEN	I*4 (30)	O	Length of each field in table(characters)
13	TCOL	I*4 (30)	O	Starting column for each field
14	TSPA	I*4	O	Space required for each table row(words)
15	MTBNAM	C*16	O	Name of table from message file
16	TGRPPT	I*4	O	Pointer to group within DSN
17	AFLDS	I*4	O	Number of fields in table extension
18	ANUM	I*4 (4)	O	Number of each variable type in table extension(see 10)
19	ATYP	C*1 (30)	O	Type of each field in table extension
20	ALEN	I*4 (30)	O	Length of each field in table extension
21	ACOL	I*4 (30)	O	Starting column for each associated field
22	ASPA	I*4	O	Space required for table extension
23	ACLU	I*4	O	Associated table cluster number
24	AGRP	I*4	O	Associated table group number
25	RETCOD	I*4	O	Return code

COMMON USAGE:

block name status

CFBUFF RECNO I
 CFBUFF WIBUFF M

CALLS:

routine

WDPTCL WDPTSP WDRCGO WDRCGX WDRCUP WDTBFN WDTBSP WTBDCI
 WTBICL

CALLED BY:

group routine

CASES CSTA

WDTGET

WDTGET

This SUBROUTINE is number 1 in file WDTMS1.

gets timeseries information from the WDM SFL

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u>	<u>status</u>	<u>explanation</u>
1	WDM SFL	I*4	I	watershed data management file unit number
2	DSN	I*4	I	data-set number
3	DELTA	I*4	I	time step for get
4	DATES	I*4 (6)	I	starting date
5	NVAL	I*4	I	number of values
6	DTRAN	I*4	I	transformation code 0 - ave,same 1 - sum,div 2 - max 3 - min
7	QUALFG	I*4	I	allowed quality code
8	TUNITS	I*4	I	time units for get
9	RVAL	R*4 (V)	O	array to place retrieved values in
10	RETCOD	I*4	O	return code 0 - everything O.K. -8 - invalid date -14 - date specified not within valid range for data set -20 - problem with one or more of following: GPFLG, DXX, NVAL, QUALVL, LTSTEP, LTUNIT -21 - date from WDM doesn't match expected date -81 - data set does not exist -82 - data set exists, but is wrong DSTYP -84 - data set number out of range

COMMON USAGE:

block name status

CFBUFF WIBUFF A

CALLS:

routine

FLOAT MOD WDATCP WDRCGO WTDSPX WTFNDG WGTIVL
 WTPMCK WTSCSC ZIPR

CALLED BY:

group routine

CASES CSTS
 WDSPTM WSTSCP

WDTPUT

WDTPUT

This SUBROUTINE is number 17 in file WDTMS1.

Puts time series data into a WDM file. This routine traps the problem with overwriting existing data.

ARGUMENTS:

order	name	declaration	type	size	status	explanation
1	WDMSFL	I*4	I		I	watershed data management file unit number
2	DSN	I*4	I		I	data-set number
3	DELT	I*4	I		I	time step for put
4	DATES	I*4 (6)	I		I	starting date
5	NVAL	I*4	I		I	number of values
6	DTOVWR	I*4	I		I	data overwrite flag, 0 - dont overwrite 1 - overwrite O.K.
7	QUALFG	I*4	I		I	allowed quality code
8	TUNITS	I*4	I		I	time units for put
9	RVAL	R*4 (V)	I		I	array for writing out values
10	RETCOD	I*4	O		O	return code 0 - everything is O.K. -8 - invalid date -9 - data not present in current group -10 - no data in this group -11 - no non missing data, data has not started yet -14 - date specified not within valid range for data set -15 - VBTIME=1 and DELT,TUNITS do not agree with the data set -20 - problem with one or more of following: DTOVWR, NVAL, QUALFG, TUNITS, DELT -21 - date from WDM doesn't match expected date -81 - data set does not exist -82 - data set exists, but is wrong DSTYP -84 - data set number out of range -85 - trying to write to a read-only data set

COMMON USAGE:

none

CALLS:

routine

WDTPFX WTDEL

CALLED BY:

group routine

CASES CSTS
 UTIMPT PRWMTI
 WDSPTM WSTSCP

WMSBCS

WMSBCS

This SUBROUTINE is number 13 in file UTWDT1.

Split up a block control word for a message type data set into its components.

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u> <u>type size</u>	<u>status</u>	<u>explanation</u>
1	QWORD	I*4	I	message type dataset block control word
2	CLASS	I*4	O	class of information 1 - 1-dimensional parameter 4 - menu 2 - 2-dimensional parameter 5 - file 3 - text
3	ID	I*4	O	id for portion of group examples: CLASS ID Description 1 4 default value for parameter field 4 6 help for menu screen 5 3 <u>status</u> of file
4	ORDER	I*4	O	order of information
5	TLEN	I*4	O	total number of characters in the block

COMMON USAGE:

none

CALLS:

routine

MOD

CALLED BY:

group routine

CASES CSTXT
 MSEXPT PRMSFE PRMSME PRMSPE PRMSTE PRWMME
 UTWDMF WMSBCX WMSIDP
 WDMRX CHKMES
 WDTBLE WDTBSP

WMSFBC

WMSFBC

This SUBROUTINE is number 12 in file UTWDMF.

Get first block control word and its position for group GNUM in a message data set. A STOP is encountered when group GNUM does not exist.

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u> <u>type size</u>	<u>status</u>	<u>explanation</u>
1	WMSFBL	I*4	I	Fortran unit number for WDM file
2	DSN	I*4	I	data-set number
3	GNUM	I*4	I	group number
4	DREC	I*4	O	record number of block control word on WDM file
5	DPOS	I*4	O	position of block control word on DREC
6	BCWORD	I*4	O	block control word

COMMON USAGE:

block name status

CFBUFF WIBUFF A

CALLS:

routine

WDDSCK WDP TSP WDR CGO WMSQCK

CALLED BY:

group routine

CASES CSTXT
MSEXPT PRWMM
UTWDMF WMSIDP
WDTBLE WDTBSP

WMSGTE

WMSGTE

This SUBROUTINE is number 8 in file UTWDMF.

Get one record of text off WDM file.

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u>		<u>status</u>	<u>explanation</u>
		<u>type</u>	<u>size</u>		
1	WMSFSL	I*	4	I	Fortran unit number for WDM file
2	TLEN	I*	4	I	total length of text (may be more than one record)
3	LLEN	I*	4	I	maximum size of record to get
4	DREC	I*	4	M	record number of data on WDM file
5	DPOS	I*	4	M	position of data on data record
6	GLEN	I*	4	M	counter to keep track of when to read off WDM file
7	MLEN	I*	4	M	should be initialized to 0 for first call
8	OLEN	I*	4	O	number of characters retrieved so far (must be <= TLEN)
9	OBUFF	C*1	(V)	O	should be initialized to 0 for first call
10	CONT	I*	4	O	actual size of record retrieved
				O	array of size LLEN containing OLEN characters retrieved
				O	indicator flag for text
					0 - no more text available
					1 - more text available

COMMON USAGE:

block name status

CFBUFF WIBUFF I

CALLS:

routine

ICHAR MOD WDNXPS WDR CGO ZIPC

CALLED BY:

group routine

CASES CSTXT

MSEXPT PRMSTA
 UTWDMF WMSGTO
 WDATMS WADGVA WADGDS
 WDTBLE WDTBSP

WMSSKB

WMSSKB

This SUBROUTINE is number 14 in file UTWDMF.

Position DREC and DPOS at the end of the current data block. DREC and DPOS are assumed to be input as the start of the block.

ARGUMENTS:

		declaration			
<u>order</u>	<u>name</u>	<u>type</u>	<u>size</u>	<u>status</u>	<u>explanation</u>
1	WDMSFL	I*4		I	Fortran unit number for WDM file
2	TLEN	I*4		I	total number of characters to skip
3	DREC	I*4		M	record number on WDM file
4	DPOS	I*4		M	position on record DREC

COMMON USAGE:

none

CALLS:

routine

MOD WDNXPS

CALLED BY:

group routine

CASES CSTXT
 WDATMS WADGDF WADGRA WADGVA WADGDS WADGHL
 WDTBLE WDTBSP

WSTAGP

WSTAGP

This SUBROUTINE is number 3 in file WDSPTM.

add a group to a space time data set, physically allocate space for

ARGUMENTS:

		declaration			
<u>order</u>	<u>name</u>	<u>type</u>	<u>size</u>	<u>status</u>	<u>explanation</u>
1	WDMSFL	I*4		I	Fortran unit number of WDM file
2	DSN	I*4		I	Data-set number
3	NGPDAT	I*4 (6)		I	Starting date for group
4	NGPTUN	I*4		I	Time units for group
5	NGPTST	I*4		I	Time step for group
6	NGPNOV	I*4		I	Number of timesteps in group
7	RETCOD	I*4		O	Return code

0 - group added
 -42 - overlap an existing group
 -43 - can't add another space time group
 -46 - bad space time group specification parameter
 -81 - data set does not exist
 -82 - data set exists, but is wrong DSTYP
 -84 - data set number out of range

COMMON USAGE:

block name status

CFBUFF RECNO I
CFBUFF WIBUFF A

CALLS:

routine

MOD TIMADD TIMCHK WDATCL WDATSP WDPTCL WDPTSP WDRCGO
WDRCGX WDRcup WDSASV WDSCHK WSTGCL WSTGSP

CALLED BY:

group routine

CASES CSST

WSTGSU

WSTGSU

This SUBROUTINE is number 6 in file WDSPTM.

summarize a group in a space time data set

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u> <u>type</u> <u>size</u>	<u>status</u>	<u>explanation</u>
1	WMSFPL	I*4	I	Fortran unit number of WDM file
2	DSN	I*4	I	Data set number
3	GRPIND	I*4	I	Index of group to summarize
4	GSDAT	I*4 (6)	O	Start date of group
5	GEDAT	I*4 (6)	O	End date of group
6	GTUN	I*4	O	Group time units
7	GTST	I*4	O	Group time steps
8	GNOV	I*4	O	Number of values in group
9	GFRAC	R*4	O	Fraction of group containing data
10	RETCOD	I*4	O	Return code

0 - group summarized
-49 - group doesn't exist
-81 - data set does not exist
-82 - data set exists, but is wrong DSTYP
-84 - data set number out of range

COMMON USAGE:

block name status

CFBUFF WIBUFF A

CALLS:

routine

FLOAT TIMADD WDATSP WDPTSP WDRCGO WDSCHK WSTGSP

CALLED BY:

group routine

CASES CSST

WSTGTR

WSTGTR

This SUBROUTINE is number 9 in file WDSPTM.

get real space time data

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u> <u>type</u> <u>size</u>	<u>status</u>	<u>explanation</u>
1	WDMSFL	I*4	I	Fortran unit number of WDM file
2	DSN	I*4	I	Data-set number
3	STDAT	I*4 (6)	I	Date of data to get
4	NDIM	I*4	I	Number of dimensions specified
5	NUMN	I*4 (V)	I	Number of values to get in each dimension
6	BASN	I*4 (V)	I	Base value in each dimension
7	SKPN	I*4 (V)	I	Skip value in each dimension
8	NVAL	I*4	I	Total number of values to get
9	RBUFF	R*4 (V)	O	Buffer to put values in
10	RETCOD	I*4	O	Return code

0 - data retrieved
-36 - missing needed following data for a get
-37 - no data present
-38 - missing part of time required
-39 - missing data group
-40 - no data available
-41 - no data to read
-42 - overlap an existing group
-44 - trying to get/put more data than in block
-45 - types don't match
-81 - data set does not exist
-82 - data set exists, but is wrong DSTYP
-84 - data set number out of range

COMMON USAGE:

block name status

CFBUFF WRBUFF I

CALLS:

routine

WSTWNT

CALLED BY:

group routine

CASES CSST
WDSPTM WSTSCP

WSTPTR

WSTPTR

This SUBROUTINE is number 12 in file WDSPTM.

put real space time data

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u> <u>type</u> <u>size</u>	<u>status</u>	<u>explanation</u>
1	WDMSFL	I*4	I	Fortran unit number of WDM file

2	DSN	I*4	I	Data-set number
3	STDAT	I*4 (6)	I	Date of data to get
4	NDIM	I*4	I	Number of dimensions specified
5	NUMN	I*4 (V)	I	Number of values to get in each dimension
6	BASN	I*4 (V)	I	Base value in each dimension
7	SKPN	I*4 (V)	I	Skip value in each dimension
8	NVAL	I*4	I	Total number of values to get
9	RBUFF	R*4 (V)	I	Buffer to write values from
10	RETCOD	I*4	O	Return code

0 - data written
 -36 - missing needed following data for a get
 -37 - no data present
 -38 - missing part of time required
 -39 - missing data group
 -40 - no data available
 -41 - no data to read
 -42 - overlap an existing group
 -44 - trying to get/put more data than in block
 -45 - types don't match
 -81 - data set does not exist
 -82 - data set exists, but is wrong DSTYP
 -84 - data set number out of range

COMMON USAGE:

block name status

CFBUFF WRBUFF 0

CALLS:

routine

WDRUCP WSTWNT

CALLED BY:

group routine

CASES CSST
WDSPTM WSTSCP

WTBCOD

WTBCOD

This SUBROUTINE is number 7 in file WDTBLE.

convert data from full screen buffer into WDM internal format

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u>	<u>status</u>	<u>explanation</u>
		<u>type</u> <u>size</u>		
1	XFLDS	I*4	I	Number of fields in table
2	NROW	I*4	I	Number of rows in table
3	XSPA	I*4	I	Number of columns of table data(fields adjusted for size)
4	XLEN	I*4 (V)	I	Length of each field
5	XTYP	C*1 (V)	I	Type(I,R,C,D,A) of each field
6	XCOL	I*4 (V)	I	Starting column for each field
7	TBCBUF	C*1 (80,V)	I	Buffer containing output from full screen routine
8	MXTBTL	I*4	I	Size of buffer to put data in WDM internal format
9	TBRBUF	R*4 (V)	O	Buffer to put data in WDM internal format
10	RETCOD	I*4	O	Return code

COMMON USAGE:

none

CALLS:

routine

CHRDEC CHR DPR CHRINT

CALLED BY:

group routine

CASES CSTA

WTBDSP

WTBDSP

This SUBROUTINE is number 10 in file UTWDT1.

Split up dimension variable for table type data set into its components.

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u>		<u>status</u>	<u>explanation</u>
		<u>type</u>	<u>size</u>		
1	TABDIM	I*4		I	table dimension variable
2	TABIND	I*4		O	table index number
3	NROW	I*4		O	number of rows in table
4	NCOL	I*4		O	space for each column(words)
5	NEXT	I*4		O	amount of table extension space (words)

COMMON USAGE:

none

CALLS:

routine

MOD

CALLED BY:

group routine

CASES CSTA
WDMRX CHKTAB
WDTBLE WDTBFN WTBPUT WDTBDL WTBGET

WTBGET

WTBGET

This SUBROUTINE is number 10 in file WDTBLE.

get the main table data

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u>		<u>status</u>	<u>explanation</u>
		<u>type</u>	<u>size</u>		
1	WDMSFL	I*4		I	Watershed data management file unit number
2	DSN	I*4		I	table data-set number
3	TABNAM	C*16		I	table name
4	TABIND	I*4		I	table identifier number
5	DATFLG	I*4		I	data type

				1 - main table
				2 - extension
6	FROW	I*4	I	first row of data to read from
7	NROW	I*4	I	number of rows of data to read
8	FSPA	I*4	I	first data space to read from
9	NSPA	I*4	I	space for data in each row
10	TBRBUF	R*4 (V,V)	O	buffer for data to get from WDM file
11	RETCOD	I*4	O	return code

COMMON USAGE:

block name status

CFBUFF WIBUFF I

CALLS:

routine

WDPTSP WDRCGO WDTBFN WTBDSF

CALLED BY:

group routine

CASES CSTA

WTBISP

WTBISP

This SUBROUTINE is number 8 in file UTWDT1.

Split up an identifier for a table type data set into its components.

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u>		<u>status</u>	<u>explanation</u>
		<u>type</u>	<u>size</u>		
1	TABID	I*4		I	table identifier
2	MSFLID	C*1	(2)	O	message file name id
3	MCLU	I*4		O	message file cluster for table
4	MGRP	I*4		O	message file group number for table

COMMON USAGE:

none

CALLS:

routine

CHAR MOD

CALLED BY:

group routine

CASES CSTA
WDMRX CHKTAB
WDTBLE WDTBFX

WTBPUT

This SUBROUTINE is number 9 in file WDTBLE.

put WDM table data into WDM file

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u>		<u>status</u>	<u>explanation</u>
		<u>type</u>	<u>size</u>		
1	WDMSFL	I*4		I	Watershed data management file unit number
2	DSN	I*4		I	table data-set number
3	TABNAM	C*16		I	table name
4	TABIND	I*4		I	table identifier number
5	DATFLG	I*4		I	data type 1 - main table 2 - extension
6	FROW	I*4		I	first row to start writing
7	NROW	I*4		I	number of rows of data to write
8	FSPA	I*4		I	first data space to start writing
9	NSPA	I*4		I	space for data in each row
10	TBRBUF	R*4 (V,V)		I	buffer for data to write onto WDM file
11	RETCOD	I*4		O	return code

COMMON USAGE:

block name status

CFBUFF WIBUFF I

CALLS:

routine

WDPTSP WDRCGO WDRGUP WDTBFN WTBDSF

CALLED BY:

group routine

CASES CSTA

ZIPC

ZIPC

This SUBROUTINE is number 31 in file UTCHAR.

Fill the character array X of size LEN with the given value ZIP.

ARGUMENTS:

<u>order</u>	<u>name</u>	<u>declaration</u>		<u>status</u>	<u>explanation</u>
		<u>type</u>	<u>size</u>		
1	LEN	I*4		I	size of character array
2	ZIP	C*1		I	character to fill array
3	X	C*1 (V)		O	character array to be filled

COMMON USAGE:

none

CALLS:

none

CALLED BY:

group routine

CASES	CSATR					
MSIMPT	PRWMSI					
UTCHAR	DATCHR	DATLST	DECCHR	DPRCHR	INTCHR	PRTLNO
UTWDMF	WMSGTE					
WDATRB	WDSAGY					
WDIMEX	PRWMEX					

APPENDIX E. DATA STRUCTURES - COMMON BLOCKS

This appendix provides detailed information on the common blocks used internally by WDM. Included are a general description of the common block's function, a list of parameters used to dimension arrays within the common block, and the attributes of variables and equivalences defined in the common block. Attributes defined are: type (integer, real, etc.), array dimensions, and a list of subroutines where the variable is used. For each subroutine where a common variable is used, the usage is classified as follows.

set - variable value is set

ref - variable is referenced

s/r - variable is both set and referenced

arg - variable is used as an argument to a subroutine

a/r - variable is used as an argument and referenced

asr - variable is used as an argument, referenced and set

CDRLOC

This common block contains pointers to critical pieces of information on the file definition record. It is found in the include file named 'CDRLOC.INC'.

PDIRPT pointer to position where record pointer to first directory data set is stored
set: WDBFIN
ref: WDDRRC

PFNAME pointer to position where name on wdm file is stored
set: WDBFIN
ref: WDRCAD WDRCDL WDRCGN

PMXREC pointer to position where number of records in wdm file is stored
set: WDBFIN
ref: WDCREA WDFLCK WDRCAD

PTSNUM pointer to position where count of timeseries datasets in wdm file is stored
set: WDBFIN
ref: PRWMEX WADDSI WDDSRN WDFCUP

CFBUFF

This common block contains buffers for in memory storage of records from WDM files. It is found in the include file named 'CFBUFF.INC'.

```
FREPOS  next free position on the in memory record buffer
  set: WDBFIN
  s/r: WDRCGO
MAXREC  maximum number of records on each available WDM file
  set: WDBFIN WDCREA WDFLCK WDRCAD
  ref: WDRCGO
  s/r: WDFLCL
NXTPOS  forward chain of records in the in memory record buffer
  set: WDBFIN
  s/r: WDRCGO
PREPOS  backward chain of records in the in memory record buffer
  set: WDBFIN
  s/r: WDRCGO
RECNO   WDM file record numbers for each record in the in memory record buffer
  set: WDBFIN
  ref: WADADI WDRRC WDSCL WDLANG WDLBAX WDNXPS WDRCAD WDRCUP WDTBTM WMSANG WSTAGP WSTFGP
      WTNWBK
      WTSKVX
  s/r: WDRCGO
  arg: WDFDUP
WDMCNT  count of currently open WDM files
  set: WDBFIN
  ref: WDRCAD WDRCGO
  s/r: WDCREA WDFLCK WDFLCL
WDMFUN  Fortran file unit numbers for each record in the in memory record buffer
  set: WDBFIN
  s/r: WDFLCL WDRCGO
WDMOPN  Fortran file unit numbers for each open WDM file
  set: WDBFIN
  ref: WDRCAD WDRCGO
  s/r: WDCREA WDFLCK WDFLCL
WIBUFF  integer in memory record buffer
  set: WDCREA WDLBAX WDRCGO WMSADI WMSPTI WSTPTD WSTPTI WSTPTR
  ref: PRWMEX PRWMTE WADDSI WADGTL WADQCK WDCKDT WDSCK WDDSDL WDLGET WDLLCK WLLSU WDNXDV
      WDNXPS
      WDRPRS WDRCUP WDSKBK WDTBFN WDTBFX WDTBSP WDTBSU WMSANG WMSBTR WMSGTE WMSQCK WSTFDT
      WSTGSU
      WSTGTD WSTGTI WSTGTR WSTWNT WTBGET WTGPCK WTGINV
  s/r: WADADI WDRRC WDSCL WDSRN WDFCUP WDFDUP WDFLCK WDLANG WDLPUT WDRCAD WDRCDL WDRCGN
      WDTBDL
      WDTBTM WSTAGP WSTDGP WTBPUT WTDDEL WTPTVL
  arg: WDTGET WMSBCX
  a/s: WDBSAC WDBSAI WDBSAR
  a/r: WDBSGC WDBSGI WDBSGR WDSCHA WDTPEX WMSFBC WTDSCU WTFNDG WTFNDT
  asr: WDBSAD WSTFGP WTBFX WTSKVX
WRBUFF  real in memory record buffer
  set: WDBSAR WDLPUT WSTPTD WSTPTR WTBPUT WTDDEL WTPTVL
  ref: PRWMEX PRWMTE WDBSGR WDLGET WSTGTD WSTGTR WTBGET WTFNDT WTGINV
  s/r: WTSKVX
  arg: WTFNDG
```

CWTSDS

This common block is used to store internal variables used by timeseries management routines. It is found in the include file 'CWTSDS.INC'.

CURBKS starting position of current block within current record
arg: WDTPFX WTGTVL

CURCMP compression code of current block
arg: WDTPFX WTGTVL

CURCNT current position within block
arg: WDTPFX
asr: WTGTVL

CURDAT date of start of current value
arg: WDTPFX
a/r: WTGTVL

CURNOV number of values in current block
arg: WDTPFX WTGTVL

CURPOS position in current block
arg: WDTPFX WTGTVL

CURQUA quality code of current block
arg: WDTPFX
a/r: WTGTVL

CURREC current record number
arg: WDTPFX WTGTVL

CURTST current time step
arg: WDTPFX
a/r: WTGTVL

CURTUN current time units
arg: WDTPFX
a/r: WTGTVL

CURVAL current value
arg: WDTPFX
a/r: WTGTVL

PREVAL previous value
arg: WDTPFX WTGTVL

APPENDIX F.--RETURN CODES

***** describe what each one is and where it is set

CODE	SOURCE GROUP	ROUTINE	DESCRIPTION
-(ABS(FERR))	usys**.f77	WDBOPN	system specific file error number
-01	usys**.f77	WDBOPN	non specific error on WDM file open
-04	wdtms2.f77	WTDSCU	copy/update failed due to data overlap problem - part of source needed
-05	wdtms2.f77	WTDSCU	copy/update failed due to data overlap problem
-06	wdtms2.f77	WTFNDT	no data present
-08	wdtms1.f77	WTPMCK	bad dates
-09	wdtms1.f77	WTGPKC	data present in current group
-10	wdtms1.f77	WTSKVX	no data in this group
-11	wdtms1.f77	WTSKVX	no non missing data, data has not started yet
-14	wdtms1.f77	WTFNDG	date specified not within valid range for data set
-15	wdtms1.f77	WDTPFX	time units and time step must match label exactly with VBTIME=1
-20	wdtms1.f77	WTPMCK	problem with one or more of GPFLG, DXX, NVAL, QUALVL, LTSTEP, LTUNIT
-21	wdtms1.f77	WTSKVX	date from WDM doesn't match expected date
-23	wdtble.f77	WDTBSP	not a valid table
-24	wdtble.f77	WDTBSP	not a valid associated table
-25	wdtble.f77	WDTBTM	template already exists
-26	wdtble.f77	WDTBTM	can't add another table
-26	utwdmf.f77	WMSANG	no space for another question
-27	wdtble.f77	WDTBSU	no tables to return info about
-28	wdtble.f77	WDTBFX	table doesn't exist yet
-30	wdtble.f77	WTBPUT	more than whole table
-30	wdtble.f77	WTBGET	want more than whole table
-31	wdtble.f77	WTBPUT	more than whole extension
-31	wdtble.f77	WTBGET	want more than whole extension
-32	wdtble.f77	WTBPUT	data header doesn't match
-32	wdtble.f77	WTBGET	data header doesn't match
-33	wdtble.f77	WTBPUT	problems with row/space specs
-33	wdtble.f77	WTBGET	problems with row/space specs
-36	wsdptm.f77	WSTFGP	missing needed following data for a get
-37	wsdptm.f77	WSTFGP	no data present
-38	wsdptm.f77	WSTFGP	missing part of time required
-39	wsdptm.f77	WSTFGP	missing data group
-40	wsdptm.f77	WSTFGP	no data available
-41	wsdptm.f77	WSTFGP	no data to read
-42	wsdptm.f77	WSTAGP	overlap an existing group
-43	wsdptm.f77	WSTAGP	can't add another space time group
-44	wsdptm.f77	WSTWNT	trying to get/put more data than in block
-45	wsdptm.f77	WSTWNT	types don't match
-46	wsdptm.f77	WSTAGP	bad space time group specification parameter
-47	wsdptm.f77	WDSTCP	bad direction flag
-48	wsdptm.f77	WDSTCP	conflicting spec of space time dim and number of timeseries data sets
-49	wsdptm.f77	WSTGSU	group doesn't exist
-50	wddlq.f77	WDLGET	requested attributes missing from this data set
-51	wddlq.f77	WDLANG	no space for another DLG
-61	wdble.f77	WDDSCL	old data set does not exist
-62	wdble.f77	WDDSCL	new data set already exists
-71	wdbtch.f77	WDBCRL	data set already exists
-72	wdbtch.f77	WDDSRN	old data set does not exist
-73	wdbtch.f77	WDDSRN	new data set already exists
-81	utwdmd.f77	WDDSCK	data set does not exist
-82	utwdmd.f77	WDSCHA	data set exists, but is wrong DSTYP

APPENDIX F.--RETURN CODES--continued

CODE	SOURCE GROUP	ROUTINE	DESCRIPTION
-83	utwdmd.f77	WDCREA	WDM file already open, can't create it
-84	utwdmd.f77	WDDSCK	data set number out of valid range
-85	utwdmd.f77	WDSCHA	trying to write to a read-only data set
-87	utwdmd.f77	WDFLCL	can't remove message WDM file from buffer
-88	utwdmd.f77	WDFLCK	can't open another WDM file
-89	utwdmd.f77	WDFLCK	check digit on 1st record of WDM file is bad
-101	wdatrb.f77	WDBSAC	incorrect character value for attribute
-102	wdatrb.f77	WDSASP	attribute already on label
-103	wdatrb.f77	WDSASP	no room on label for attribute
-104	wdatrb.f77	WDDPAR	data present, can't update attribute
-105	wdatrb.f77	WDDPAR	attribute not allowed for this type data set
-106	wdatrb.f77	WDDPAR	can't delete attribute, its required
-107	wdatrb.f77	WDSAFN	attribute not present on this data set
-108	wdatrb.f77	WDBSAI	incorrect integer value for attribute
-109	wdatrb.f77	WDBSAR	incorrect real value for attribute
-110	wdatrb.f77	WDSAFI	attributes not found on message file
-111	wdatrb.f77	WDSAFI	attribute name not found (no match)
-112	wdatrb.f77	WDSAFI	more attributes exist which match SAFNAM
-121	wdatms.f77	WADADI	no space for another attribute
+01	wddlg.f77	WDLGET	more data available for this DLG group
+01	wddlg.f77	WDLLSU	more groups available to summarize
+01	wdtble.f77	WDTBSU	more tables exist than we have space for
+01	wdsptm.f77	WSTAGP	ok to add group
+01	wdtble.f77	WDTBFN	table template not found
+02	wddlg.f77	WDLGET	no more data available for this DLG group

APPENDIX G.

WDM FILE MAINTENANCE PACKAGE

A stand alone utility program called WDMRX (strong medicine for an ailing WDM file) is available for diagnostic use when the integrity of a WDM file is in question. WDMRX checks file internals, writes a summary of the file contents, and optionally attempts to correct major errors. The user must supply the name of the WDM file, the output summary level, and whether the attempt corrections. The following output was produced when summarizing the MESSAGE.WDM file created for use by the case studies described in this manual. Detailed output from WDMRX is always written to file DUMP.OUT with summary information written to the screen. Comments have been added to explain the particular output.

Comment	Output
summary of user options	OUTPUT LEVEL IS GROUP UPDATE OPTION IS NO
general information about WDM file	WDM FILE: message.wdm MAXREC : 60 FREREC : 42
summary of data sets by type	COUNT OF DATASETS BY TYPE 1-TIMESERIES: 0 2-TABLE : 0 3-SCHEMATIC : 0 4-PROJECT : 0 5-VECTOR : 1 6-RASTOR : 0 7-SPACE-TIME: 0 8-ATTRIBUTE : 6 9-MESSAGE : 1
dataset chains by type	CHAIN OF VECTOR DATASETS: 90 CHAIN OF ATTRIBUTE DATASETS: 10 9 8 7 5 6 CHAIN OF MESSAGE DATASETS: 12
directory records	LOOKING FOR DIRECTORY RECORDS FOUND DIR REC: 1 ON PHYS REC 3 FOR DSN: 1 - 500 END SEARCH FOR DIRECTORY RECORDS
free record chain	FREE REC CHN : 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
Comment	Output
data set 5 details	DSN: 5 TYPE: 8-ATTRIBUTE LENGTH: 4 CHAIN: 6 5 4 2 FREE POSITION OF DATA: 2 113

APPENDIX G. WDM FILE MAINTENANCE PACKAGE--continued

```

data set 6 details   DSN:    6   TYPE: 8-ATTRIBUTE   LENGTH:    6
                    CHAIN:    7   8   9   10  11  12
                    FREE POSITION OF DATA:    12   80

data set 7 details   DSN:    7   TYPE: 8-ATTRIBUTE   LENGTH:    7
                    CHAIN:   15  16  17  18  19  20  21
                    FREE POSITION OF DATA:    21   129

data set 8 details   DSN:    8   TYPE: 8-ATTRIBUTE   LENGTH:    9
                    CHAIN:   13  14  22  23  24  25  26  27  28
                    FREE POSITION OF DATA:    28   157

data set 9 details   DSN:    9   TYPE: 8-ATTRIBUTE   LENGTH:    6
                    CHAIN:   29  30  31  32  33  34
                    FREE POSITION OF DATA:    34   200

data set 10 details  DSN:   10   TYPE: 8-ATTRIBUTE   LENGTH:    3
                    CHAIN:   35  36  37
                    FREE POSITION OF DATA:    37   368

data set 12 details  DSN:   12   TYPE: 9-MESSAGE   LENGTH:    2
                    CHAIN:   38  39
                    FREE POSITION OF DATA:    39   73
                    CHECKING MESSAGE GROUPS
                    CLUSTER: CSTA
                    GROUP #:  1 IS AT   38  339
                    GROUP #:  2 IS AT   38  497
                    GROUP #:  3 IS AT   39   31

data set 90 details  DSN:   90   TYPE: 5-VECTOR   LENGTH:    2
                    CHAIN:   40  41
                    FREE POSITION OF DATA:    41   309
                    CHECKING VECTOR GROUPS
                    GROUP #:  1 IS AT   40  461 ATTR:    1    0    0    1
                    GROUP #:  2 IS AT   40  484 ATTR:    1  180  201  1
                    GROUP #:  3 IS AT   41  129 ATTR:    1  180  208  1
    
```

orphan record check

LOOKING FOR MISSING RECORDS IN CHAINS

MAP OF RECORD USAGE

codes:

-3 - fdefn

-2 - free

-1 - direct

0 - orphan

>0 - DSN

1 - 10:	-3	5	-1	5	5	5	6	6	6	6
11 - 20:	6	6	8	8	7	7	7	7	7	7
21 - 30:	7	8	8	8	8	8	8	8	9	9
31 - 40:	9	9	9	9	10	10	10	12	12	90
41 - 50:	90	-2	-2	-2	-2	-2	-2	-2	-2	-2
51 - 60:	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2

FOUND NO RECORDS MISSING FROM CHAINS

data set count check

CHECKING DATASET COUNTS

ALL DATASET COUNTS MATCH

APPENDIX H.--INDEX TO SUBROUTINES

The following is a complete list of the names of all subroutines contained in the WDM software. A short definition of the function of each subroutine also is included. The purposes of this list are two-fold: first, to prevent the application programmer from duplicating already-used routine names when naming new subroutines developed for the specific application program; and second, to alert the programmer to the capabilities of lower-level subroutines included in the WDM software. The programmer may obtain further details on these subroutines in the SYSDOC.OUT file contained on the system distribution disk. For each subroutine, the appendix provides information on the subroutine type, the file in which it resides on the distribution disk, the order number indicating where the subroutine occurs within the file, and a functional description. The subroutine types are:

E - main program
 S - subroutine
 I - integer function
 R - real function
 D - double precision function

NAME	TPY	GROUP	NUMB	DESCRIPTION
ASRTC	S	UTSORT	1	Sorts character strings.
ASRTI	S	UTSORT	2	Sorts integers.
ASRTIP	S	UTSORT	3	Sorts integers in their array.
ASRTR	S	UTSORT	4	Sorts decimal numbers.
ASRTRP	S	UTSORT	5	Sorts decimal numbers in their array.
BLDWRT	S	WDMEX	4	Write TBUFF to screen for program Messie.
CARVAR	S	UTCHAR	1	Convert a character*1 array of size LENA to a character variable of length LEN.
CASES	E	CASES	1	Demonstrate use of WDM library.
CHAIN	S	WDMRX	2	Determine chain of records.
CHDECE	S	UTCHAR	2	Convert a character array to its real equivalent.
CHINTE	S	UTCHAR	3	Convert a character array to its integer equivalent.
CHKDPR	S	UTNUMB	3	Check the double precision DVAL against the minimum and maximum values.
CHKINT	S	UTNUMB	1	Check the integer IVAL against the minimum and maximum values.
CHKMES	S	WDMRX	5	Check groups and blocks of a message data set.
CHKREA	S	UTNUMB	2	Check the real RVAL against the minimum and maximum values.
CHKSTR	I	UTCHAR	4	Search thru STR2 for a match to the character array STR1.
CHKTAB	S	WDMRX	4	Check groups and blocks of a table data set.
CHKTIM	S	WDMRX	3	Check groups and blocks of a timeseries data set.
CHKVEC	S	WDMRX	6	Check groups and blocks of a vector data set.
CHRCHR	S	UTCHAR	5	Copy LEN characters from array STR1 to array STR2.
CHRDEC	R	UTCHAR	6	Convert a character array to its real equivalent.
CHRDEL	S	UTCHAR	8	Delete the character in array position POS and shift the rest left one position.
CHRDIG	I	UTCHAR	9	Convert a single character to an integer.
CHRDPR	D	UTCHAR	7	Convert a character array to its double precision equivalent.
CHRINS	S	UTCHAR	10	Insert the character CHAR into position COL in the array STRING.
CHRINT	I	UTCHAR	11	Convert a character array to its integer equivalent.
CKDATE	S	UTDATE	1	Determine the calendar order of two dates. The dates are assumed to be valid.
CKNBLK	I	UTCHAR	12	Check character array CBUF for all blanks.
CMPTIM	S	UTDATE	2	Compare one time unit and step to a second time unit and step.
CNVDLG	E	CNVDLG	1	Convert nmd format dlg data to wdm export format.
COPYC	S	UTCHAR	13	Copy the character array ZIP of size LEN to the character array X of size LEN.
COPYD	S	UTNUMB	4	Copy the double precision array ZIP of size LEN to the double precision array X.
COPYI	S	UTNUMB	5	Copy the integer array ZIP of size LEN to the integer array X.
COPYR	S	UTNUMB	6	Copy the real array ZIP of size LEN to the real array X.
CRINTE	I	UTCHAR	14	Convert a character array to its integer equivalent.
CRINTX	I	UTCHAR	15	Convert a character array to its integer equivalent.
CSATR	S	CASES	7	Case study example for attribute datasets.

APPENDIX H. INDEX TO SUBROUTINES--continued

NAME	TYP	GROUP	NUMB	DESCRIPTION
CSDLG	S	CASES	4	Case study example for dlg datasets.
CSST	S	CASES	3	Case study example for space time datasets.
CSTA	S	CASES	6	Case study example for table datasets.
CSTS	S	CASES	2	Case study example for timeseries datasets.
CSTXT	S	CASES	5	Case study example for text and table template datasets.
CTRSTR	S	UTCHAR	16	Center the characters within the array TITLE.
CVARAR	S	UTCHAR	17	Convert a character variable of length LENV to a character array of size LENA.
DATCHK	S	UTDATE	3	Check DATE for valid entries.
DATCHR	S	UTCHAR	18	Put a date into a character array.
DATLST	S	UTCHAR	19	Put DATE into the character array DSTRNG in the format 1986 FEB. 14 10:30:00.
DATNXT	S	UTDATE	4	Adds or subtracts the time interval INTRVL from the current date and time DATE.
DAYMON	I	UTDATE	5	Return the number of days in the given month for the given year.
DECCHR	S	UTCHAR	20	Convert a real number to a character array.
DECCHX	S	UTCHAR	22	Right justifies the real number REAIN into the character array STR of size LEN.
DIGCHR	C	UTCHAR	23	Convert a single digit to a character.
DLIMIT	S	UTDATE	6	Depending on the value of FSLS, find earliest or latest date in array of dates.
DPRCHR	S	UTCHAR	21	Convert a double precision number to a character array.
INTCHR	S	UTCHAR	24	Convert the integer number INTIN to a character array.
JDMODY	S	UTDATE	7	Convert a julian day to month and day, leap year taken into account.
LENSTR	I	UTCHAR	25	Return the actual length of the character array, excluding trailing blanks.
LFTSTR	S	UTCHAR	26	Left justify characters in the array TITLE.
NUMPTS	S	UTDATE	8	Calculate the number of time steps between two dates.
PRADIT	S	UTIMPT	5	Put text from any id of attribute information on WDM file.
PRATIM	S	UTIMPT	4	Import attribute data information for an attribute group.
PRMSAE	S	UTEXPT	3	Export attribute type question.
PRMSFE	S	MSEXPT	5	Export file type group.
PRMSFI	S	MSIMPT	5	Import file type group.
PRMSIT	S	MSIMPT	6	Put text from any ID of information on WDM file.
PRMSME	S	MSEXPT	4	Export menu type group.
PRMSMI	S	MSIMPT	4	Import menu type group.
PRMSPA	S	MSIMPT	2	Import 1-dimensional and 2-dimensional type groups.
PRMSPE	S	MSEXPT	2	Export 1-dimensional and 2-dimensional type groups.
PRMSTA	S	MSEXPT	6	Export block with header (CHDR) and information on same line.
PRMSTE	S	MSEXPT	3	Export text type group.
PRMSTI	S	MSIMPT	3	Import text type group.
PRTLIN	S	UTCHAR	27	Convert DATE and an array of real numbers into a character array.
PRTLNO	S	UTCHAR	28	Convert DATE and an array of real numbers into a character array.
PRTSTR	S	UTCHAR	33	Write a character array to the given file unit.
PRWMAE	S	UTEXPT	2	Export attribute data.
PRWMAI	S	UTIMPT	3	Import attribute type question from sequential message file.
PRWMDE	S	UTEXPT	4	Export DLG type datasets.
PRWMDI	S	UTIMPT	2	Import DLG type datasets.
PRWMEX	S	WDIMEX	3	Export data sets (clusters) from WDM file to sequential file.
PRWMIM	S	WDIMEX	2	Import data sets (clusters) from sequential file to WDM file.
PRWMME	S	MSEXPT	1	Export message file clusters (data sets).
PRWMSI	S	MSIMPT	1	Import message file clusters (data sets).
PRWMTE	S	UTEXPT	1	Export timeseries data.
PRWMTI	S	UTIMPT	1	Import WDM timeseries data from sequential file to WDM file.
QUPCAS	S	UTCHAR	36	To convert a character string from lower case to upper case.
STRFND	I	UTCHAR	29	Return position in array STR of size LEN that array FSTR of size FLEN occurs.
STRLNK	I	UTCHAR	30	Return the number of characters in the array.
TIMADD	S	UTDATE	9	Add NVALS time steps to first date to compute second date.
TIMBAK	S	UTDATE	10	Subtract one time interval at the given units TCODE from DATE.
TIMCHK	I	UTDATE	11	Determine the calendar order of two dates.
TIMCNV	S	UTDATE	12	Convert date that uses midnight convention of 00:00 to the convention 24:00.
TIMCVT	S	UTDATE	13	Convert date that uses midnight convention of 24:00 to the convention 00:00.
TIMDFX	S	UTDATE	15	Calculate number of values between two dates, including units and time step.
TIMDIF	S	UTDATE	14	Calculate the number of time steps between two dates.
WADADI	S	WDATMS	3	Put attribute data set information on WDM message file.
WADDSI	S	WDATMS	6	Determine the data-set numbers containing the attribute data sets.
WADGDF	S	WDATMS	9	Get the default value for an attribute off the message file.

APPENDIX H. INDEX TO SUBROUTINES--continued

NAME	TYP	GROUP	NUMB	DESCRIPTION
WADGDS	S	WDATMS	10	Get the description for an attribute off the message file.
WADGHL	S	WDATMS	11	Get length and starting record/pos of help info for attribute off message file.
WADGRA	S	WDATMS	7	Get the min and max values for an attribute off the message file.
WADGTL	S	WDATMS	5	Given attribute index, get type, length, data set usage, etc.
WADGTN	S	WDATMS	4	Given attribute index, return attribute name.
WADGVA	S	WDATMS	8	Get the valid values for an attribute off the message file.
WADNSA	S	WDATMS	2	Given a search attribute index, determine if it exists on the message file.
WADQCK	S	WDATMS	1	Given attribute offset for data set check to see if attribute already exists.
WATTCL	I	UTWDT1	24	Calculate a word containing parameters for attribute type data set.
WATTSP	S	UTWDT1	25	Split up a word containing parameters for attribute type data set.
WATTUS	S	UTWDT1	26	Split up word containing array of indicator flags for dataset attribute usage.
WATWDC	I	UTWDT1	27	Calculate a word containing parameters for attribute type data set.
WATWDS	S	UTWDT1	28	Split up a word containing parameters for attribute type data set.
WBCWCL	I	UTWDT1	4	Calculate a block control word for time-series type data.
WBCWSP	S	UTWDT1	3	Split up block control word for time-series data set to determine info stored.
WBCWSQ	S	UTWDT1	11	Split up block control word for time-series data set.
WDATCL	I	UTWDT1	6	Calculate a date in compressed format from its components (year- hour).
WDATCP	S	WDTMS2	6	Copies an old array date into a new one.
WDATSP	S	UTWDT1	5	Split up a WDMS date word.
WDBCR1	S	WDBTCH	1	Add a label to a wdmsfl.
WDBFIN	S	UTWDM1	13	Initialize pointers and counters for WDM buffer of records.
WDBOPN	S	USYSPP	1	Open a WDM file. File is opened as new or old, depending on value of RONWFG.
WDBSAC	S	WDATRB	2	Adds (or modifies) character search attribute on given dsn.
WDBSAD	S	WDATRB	5	Deletes search attribute on given dsn.
WDBSAI	S	WDATRB	3	Adds (or modifies) integer search attribute on given dsn.
WDBSAR	S	WDATRB	4	Adds (or modifies) real search attribute on given dsn.
WDBSGC	S	WDBTCH	2	Gets values of character search attribute for a dsn.
WDBSGI	S	WDBTCH	3	Gets the values of integer search attribute for a dsn.
WDBSGR	S	WDBTCH	4	Get the values of real search attribute for a data set.
WDBSGX	S	WDATRB	12	Routine gets information about search attribute from message file.
WDCKDT	I	UTWDM1	11	Check data set for existence and type.
WDCREA	S	UTWDM1	14	Adds directory record and 19 empty records to a new WDM file.
WDDPAR	S	WDATRB	10	Determines if either data present and attrib can't update or attrib not allowed.
WDDPRC	I	UTWDM1	8	Determine WDM file directory record number for data-set number DSN.
WDDSCK	S	UTWDM1	12	Check data set existence and return record number of first record in data set.
WDDSCL	S	WDLBLE	1	Copies an old data-set label into a new data-set label.
WDDSDL	S	WDBTCH	5	Routine to delete a data set from the WDMSFL with no user interact.
WDDSRN	S	WDBTCH	6	Routine to renumber data sets with no user interaction.
WDDTFG	I	WDATRB	7	Determines if data is present in a WDMS data set.
WDFCUP	S	WDLBLE	3	Updates file definitions record data set counters and pointers.
WDFDUP	S	WDLBLE	2	Updates a WDMS file directory record.
WDFLCK	S	UTWDM1	2	Check directory of WDM for major errors.
WDFLCL	S	UTWDM1	3	Remove a WDM file from the open WDM buffer and adjust buffer accordingly.
WDIMEX	E	WDIMEX	1	Import/export data sets of all types to/from a WDM file.
WDLANG	S	WDDL1	1	Add new DLG group to WDM file.
WDLBAD	S	WDLBLE	5	Add new data set label, but no search attrib or data, use default label sizing.
WDLBAX	S	WDLBLE	6	Add new data-set label, but no search attrib or data.
WDLBCV	I	UTWDT1	22	Calculate a block control word for vector (DLG) type data set.
WDLBSP	S	UTWDT1	23	Split up a block control word for vector (DLG) type data set.
WDLGET	S	WDDL1	3	Retrieve DLG header or coordinate pairs from WDM file.
WDLICV	I	UTWDT1	20	Calculate a word containing parameters for a vector (DLG) type data set.
WDLISP	S	UTWDT1	21	Split up a word containing parameters for a vector (DLG) type data set.
WDL1CK	S	WDDL1	4	Search for group with desired data type and attributes.
WDL1SU	S	WDDL1	5	Summarize label information for DLG data set.
WDL1PUT	S	WDDL1	2	Store DLG header or coordinate pairs on WDM file.
WDMRX	E	WDMRX	1	Strong medicine for an ailing wdm file.
WDNXDV	S	UTWDM1	5	Move to the next data position and return the integer equivalent of data value.
WDNXPS	S	UTWDM1	4	Get the next data position on a WDM file.
WDPRPS	S	UTWDM1	6	Get the previous data position on a WDM file.
WDPTCL	I	UTWDT1	1	Calculate a pointer value from record number and the offset within the record.
WDPTSP	S	UTWDT1	2	Split up a pointer into record number and offset within record.

APPENDIX H. INDEX TO SUBROUTINES--continued

NAME	TYP	GROUP	NUMB	DESCRIPTION
WDRCAD	S	UTWDM	7	Add NUMADD records to the WDM file. Update directory record on the WDM file.
WDRCDL	I	WDLBLE	4	Deletes a record in the WDMSFL and updates pointers as required.
WDRCGN	I	UTWDM	5	Get the next free record from the WDM file and add it to WDM buffer of records.
WDRCGO	I	UTWDM	1	Determine index of user requested record in WDM in memory buffer of records.
WDRCGX	I	UTWDM	6	Get the next free record from the WDM file.
WDRCUP	S	UTWDM	4	Write record index number RIND from the buffer of records to the WDM file.
WDSAFI	S	WDATRB	11	Find the first attribute name which matches SAFNAM.
WDSAFI	S	WDATRB	8	Determines where values for particular search attribute start in data-set label.
WDSAGY	S	WDATRB	1	Gets general detail information about specified attribute.
WDSASP	S	WDATRB	6	Adds space for search attribute on a dsn label if not present on it.
WDSASV	I	WDATRB	9	Determines where values for particular search attribute start in data-set label.
WDSCHA	S	UTWDM	10	Check WDM data set existence, type and ability to update.
WDSCHK	S	UTWDM	9	Check WDM data set existence and type.
WDSKBK	S	WDTMS1	10	Skips to next WDMSFL block.
WDTBDL	S	WDTBLE	11	Delete WDM table from WDM file.
WDTBFN	S	WDTBLE	3	Determines pointer to specified table.
WDTBFX	S	WDTBLE	2	Determines pointer to specified table and returns its cluster and group number.
WDTBSP	S	WDTBLE	4	Get WDM table specifications from message file.
WDTBSU	S	WDTBLE	6	Get WDM table label info from WDM file table data set.
WDTBTM	S	WDTBLE	1	Put WDM table template on WDM file, return parameters about table.
WDTGET	S	WDTMS1	1	Gets timeseries information from the WDMSFL.
WDTPFX	S	WDTMS1	13	Puts timeseries information into the WDMSFL.
WDTPUT	S	WDTMS1	17	Puts time series data into a WDM file.
WMSADI	S	UTWDMF	3	Add portion of group-number GNUM information to DSN.
WMSANG	S	UTWDMF	2	Check that data-set DSN exists and there is space in data set for new group.
WMSBCS	S	UTWDT1	13	Split up a block control word for a message type data set into its components.
WMSBCV	I	UTWDT1	12	Calculate a block control word for message type data set.
WMSBCX	S	UTWDMF	11	Given pointer to a block control word, return its record number, position, etc.
WMSBTR	S	UTWDMF	9	Back up NREC text records in a text group and reset the pointers for record.
WMSFBC	S	UTWDMF	12	Get first block control word and position for group GNUM in message data set.
WMSGTE	S	UTWDMF	8	Get one record of text off WDM file.
WMSGTO	S	UTWDMF	10	Get and write text to a sequential file until end of text is reached.
WMSIDP	S	UTWDMF	13	Return the record number and position on the WDM file for the given ID.
WMSMNS	S	UTWDT1	15	Split up word containing integer parameters for a message type data-set menu.
WMSMNV	I	UTWDT1	14	Calculate word containing parameters for a message type data-set menu.
WMSP2S	S	UTWDT1	19	Split up word containing integer parameters for message type data-set PRM2 scr.
WMSP2V	I	UTWDT1	18	Calculate word containing parameters for message type data-set PRM2 class scr.
WMSPIS	S	UTWDT1	17	Split up word into two integer values.
WMSPIV	I	UTWDT1	16	Calculate word from two integer values (1st value- 16 bits, 2nd- 15 bits).
WMSPT	S	UTWDMF	7	Add one record of text to WDM file.
WMSQCK	S	UTWDMF	1	Check to see if a group already exists in a data set on WDM file.
WMSSKB	S	UTWDMF	14	Position DREC and DPOS at the end of the current data block.
WSTAGP	S	WDSPTM	3	Add a group to a space time data set, physically allocate space for it.
WSTDGP	S	WDSPTM	4	Delete group from a space time data set, free up space it uses.
WSTDIM	S	WDSPTM	5	Determine dimensions a space time data set.
WSTFDT	S	WDSPTM	2	Find start position of data from group pointer.
WSTFGP	S	WDSPTM	1	Find space time data within a data set, return pointers to it.
WSTGCL	I	UTWDT1	29	Calculate a block control word for space-time type data set.
WSTGSP	S	UTWDT1	30	Split up a block control word for space-time type data set.
WSTGSU	S	WDSPTM	6	Summarize a group in a space time data set.
WSTGTD	S	WDSPTM	10	Get double precision space time data.
WSTGTI	S	WDSPTM	8	Get integer space time data.
WSTGTR	S	WDSPTM	9	Get real space time data.
WSTPTD	S	WDSPTM	13	Put double precision space time data.
WSTPTI	S	WDSPTM	11	Put integer space time data.
WSTPTR	S	WDSPTM	12	Put real space time data.
WSTSCP	S	WDSPTM	14	Copy from WDM space time data set to timeseries data sets or vv.
WSTWNT	S	WDSPTM	7	Determine next wdm space time data offset (from beginning of data).
WTBCOD	S	WDTBLE	7	Convert data from full screen buffer into WDM internal format.
WTBDCD	S	WDTBLE	8	Convert data from WDM internal format into full screen buffer.
WTBDCL	I	UTWDT1	9	Calculate the table dimension for a table type data set.

APPENDIX H. INDEX TO SUBROUTINES--continued

NAME	TYP	GROUP	NUMB	DESCRIPTION
WTBDSP	S	UTWDT1	10	Split up dimension variable for table type data set into its components.
WTBGET	S	WDTBLE	10	Get the main table data.
WTBICL	I	UTWDT1	7	Calculate the identifier for a table type data set.
WTBISP	S	UTWDT1	8	Split up an identifier for a table type data set into its components.
WTBPUT	S	WDTBLE	9	Put WDM table data into WDM file.
WTBSPA	S	WDTBLE	5	Calculates space required for a row in table data set and count types of data.
WTBYFX	S	WDTMS1	6	Add base year attribute to a timeseries data set.
WTDATE	S	WDTMS2	7	Find common period with data from a list of time-series data sets on a WDM file.
WTDDEL	S	WDTMS2	3	Delete all data following a specified date in the given data set.
WTDSCU	S	WDTMS2	2	Copy from a data set to another any timeseries data between start and end dates.
WTDSPM	S	WDTMS1	5	Obtains values for a variety of TIMSER parms from labels or defaults.
WTDSPX	S	WDTMS1	7	Obtains values for a variety of timeseries parms from labels or defaults.
WTEGRP	S	WDTMS2	4	Determines end of group which contains a given date.
WTFNDG	S	WDTMS1	4	Check the data set, computes ending date, start and end group pointers, etc.
WTFNDT	S	WDTMS2	1	Determine starting and ending dates of data in data set.
WTGPCK	S	WDTMS1	14	Checks information related to group, skip to start value, fill in current info.
WTGTNV	S	WDTMS1	12	Routine to get the next value from a WDS timeseries DSN.
WTGTVL	S	WDTMS1	11	Fills in RVAL array with data values from WDMS DSN.
WTNWBK	S	WDTMS1	16	Starts a new WDMS timeseries block, on a new record if req.
WTPMCK	S	WDTMS1	3	Checks the parameters supplied to either WDTPUT or WDTGET.
WTPTVL	S	WDTMS1	15	Writes all or part of a WDMS group into a WDMS timeseries data set.
WTSCSC	S	WDTMS1	2	Converts the given time units and time step to seconds.
WTSGRP	S	WDTMS2	5	Determines start of group which contains given date.
WTSKVL	S	WDTMS1	8	Skips values within a WDMSFL timeseries group.
WTSKVX	S	WDTMS1	9	Skips values within a WDMSFL timeseries group.
ZIPC	S	UTCHAR	31	Fill the character array X of size LEN with the given value ZIP.
ZIPD	S	UTNUMB	7	Fill the double precision array X of size LEN with the given value ZIP.
ZIPI	S	UTNUMB	8	Fill the integer array X of size LEN with the given value ZIP.
ZIPR	S	UTNUMB	9	Fill the real array X of size LEN with the given value ZIP.
ZLJUST	S	UTCHAR	32	Remove leading blanks from a character variable.
ZLNTXT	I	UTCHAR	34	Determine length of text in a character variable.
ZTRIM	S	UTCHAR	35	Remove embeded blanks in a character variable.

APPENDIX I. DLG CONVERSION UTILITY

***** need to put stuff describing how to convert optional format dlg data to export format somewhere, an appendix?

The input data for this example are digital line graph data in the National Mapping Division DLG format; the data represent a railroad network in the Chickamauga TN area; the file format for the external data file is illustrated in this excerpt.

Excerpt

```

USGS-NMD DLG DATA - CHARACTER FORMAT - 09-29-82 VERSION
CHICKAMAUGA, GA AL TN 01          1981,          100000. F01
100K 4059                          RO1.RR      EMC86          4 40 0
      3      1      16      2 0.25400000000D+01      4      0      4      1
-0.850520300000000D+08  0.340520300000000D+08  0.0000000000000D+00  0.0000000000000D+00
0.000000000000000D+00  0.000000000000000D+00  0.000000000000000D+00  0.0000000000000D+00
0.000000000000000D+00  0.000000000000000D+00  0.000000000000000D+00  0.0000000000000D+00
0.000000000000000D+00  0.000000000000000D+00  0.000000000000000D+00  0.0000000000000D+00
0.000000000000000D+00  0.000000000000000D+00  0.000000000000000D+00  0.0000000000000D+00
0.10000000000D+01  0.00000000000D+00  0.00000000000D+00  0.00000000000D+00
SW      34.750000  -86.000000          591534.17  3845580.49
NW      35.000000  -86.000000          591256.91  3873300.74
NE      35.000000  -85.750000          614069.78  3873558.63
SE      34.750000  -85.750000          614415.64  3845836.61
RAILROADS          0      29      29 010      9      9 010      36      36      1
N      1      591534.17  3845580.49          2          0      0
      28      -30
N      2      591256.91  3873300.74          2          0      0
      1      -28
*** (records deleted)
A      5      602818.50  3859568.47          3      0      0      0      0
      4      2      -3
A      6      602818.50  3859568.47          3      0      0      0      0
      14      -9      -12
A      7      602818.50  3859568.47          3      0      0      0      0
      12      -10      -11
A      8      602818.50  3859568.47          3      0      0      0      0
      21      22      20
A      9      602818.50  3859568.47          3      0      0      0      0
      36      -25      35
L      1      2      5      1      2          2      0      0
591256.91  3873300.74  599976.35  3873385.63
L      2      7      6      3      5          2      2      0
613107.83  3865320.27  613840.85  3866098.14
      180      201      181      2
L      3      8      6      5      4          3      1      0
613479.74  3865680.05  613813.56  3866039.41  613840.85  3866098.14
      180      208
L      4      8      7      4      5          4      1      0
613479.74  3865680.05  613206.20  3865382.33  613128.12  3865323.04
      613107.83  3865320.27
      180      208
L      5      9      7      3      4          2      2      0
613047.64  3865251.01  613107.83  3865320.27
      180      201      181      2
L      6      10      9      4      4          6      1      0
612953.52  3864581.89  612921.96  3864678.07  612932.87  3864838.22
612982.13  3864975.94  613025.23  3865210.12  613047.64  3865251.01
      180      208
**** (records deleted)

```

The following pages contain code segments and descriptions for a program called CSDLG. The program reads the digital line graph data shown above and transforms the data to a format that will allow subsequent storage in the WDM file.

As described above, the computer program CSDLG produces two new files: (1) a WDM sequential import/export file containing the reformatted DLG data and (2) a file that reports the number of "lines" and "areas" successfully saved in the import/export file. The contents of the two output files are shown below; the import/export file illustration is an excerpt from a larger file.

**Program
CNVDLG**

```

C
C
C
C      PROGRAM  CNVDLG
C
C      + + + PURPOSE + + +
C      convert nmd format dlq data to wdm export format
C
C      + + + LOCAL VARIABLES + + +
C      INTEGER      FL, I, J, K, NLIN, NATTR, FOU, DSN,
2      LINID(2000), AL, AID, IH, IA, IB, IC,
3      OLIN, TLIN, LCNT, IAT1, IAT2, ITMP, ILEN,
4      LINSTR(11000), IFR, ILS, CPTS, IDIR, IX,
5      ACNT, AREATR(500), ARECNT(500), AREVAL(25, 500),
6      AATMIN, AATMAX, LAT1MN, LAT1MX, LAT2MN, LAT2MX,
7      LINAT1(11000), LINAT2(11000), AATCNT, LATCNT
C      REAL          TX(4800), TY(4800), XMIN, XMAX, YMIN, YMAX,
1      AX(1000), AY(1000)
C      CHARACTER*1  ID
C      CHARACTER*8  FNAME
C      CHARACTER*80 BUFF
C
C      COMMON /SCR/LINX, LINY
C      REAL          LINX(37000), LINY(37000)
C
C      + + + INTRINSICS + + +
C      INTRINSIC    ABS
C
C      + + + INPUT FORMATS + + +
1000  FORMAT (A80)
1010  FORMAT (12I6)
1020  FORMAT (A1, 41X, 2I6)
1030  FORMAT (6F12.2)
1060  FORMAT (A8)
1070  FORMAT (A1, I5, 2F12.2, 6X, I6, 6X, I6, I6)
C
C      + + + OUTPUT FORMATS + + +
2000  FORMAT (' XMIN, XMAX:', 2F12.0, '/', ' YMIN, YMAX:', 2F12.0)
2010  FORMAT ('GROUP TYPE LINE MAJ ATTR', I5, ' MIN ATTR', I5)
2020  FORMAT (' 1', I5, I6, 2F12.2)
2030  FORMAT (' 2', I5)
2040  FORMAT (6F12.2)
2070  FORMAT ('DATE', '/', 'WDMSFL', '/', 'SYSTEM', '/', 'COMMENT', '/', 'END COMMENT',
1      '/', 'DSN', ' ', I6, ' TYPE VECT NDN 1 NUP 1',
2      ' NSA 10 NSP 20 NDP 400',
3      '/', ' LABEL', '/', ' TSPREC 0',
4      '/', ' END LABEL',
5      '/', ' DATA')
2080  FORMAT (' END DATA')
C

```

```

C      + + + END SPECIFICATIONS + + +
C
WRITE(*,*) 'enter base file name '
READ(*,1060) FNAME
FOU= 40
OPEN (UNIT=FOU,FILE=FNAME//'.MSG')
FL = 41
OPEN (UNIT=FL,FILE=FNAME//'.DLG',STATUS='OLD')

C
AATMIN= 100000
AATMAX= -100000
AATCNT= 0
LAT1MN= 100000
LAT1MX= -100000
LAT2MN= 100000
LAT2MX= -100000
LATCNT= 0
TLIN = 0
ACNT = 0
50 CONTINUE
   READ (FL,1000,END=60,ERR=50) BUFF
   IF (BUFF(1:1).EQ.'L') THEN
C     a line header, get details
   READ (BUFF,1020) ID,NLIN,NATTR
C     a line
   LCNT= LCNT+ 1
   OLIN= TLIN+ 1
   LINSTR(LCNT)= OLIN
   TLIN= TLIN+ NLIN
   READ (FL,1030) (LINX(I),LINY(I),I=OLIN,TLIN)
   IF (NATTR.GT.0) THEN
C     what are these attributes
   READ (FL,1010) LINAT1(LCNT),LINAT2(LCNT)
   IF (LINAT1(LCNT).GT.LAT1MX) LAT1MX= LINAT1(LCNT)
   IF (LINAT1(LCNT).LT.LAT1MN) LAT1MN= LINAT1(LCNT)
   IF (LINAT2(LCNT).GT.LAT2MX) LAT2MX= LINAT2(LCNT)
   IF (LINAT2(LCNT).LT.LAT2MN) LAT2MN= LINAT2(LCNT)
   LATCNT= LATCNT+ 1
   ELSE
C     no attribute available
   LINAT1(LCNT)= -1
   LINAT2(LCNT)= -1
   END IF
   ELSE IF (BUFF(1:1).EQ.'A') THEN
C     an area header, get details
   ACNT= ACNT+ 1
   READ (BUFF,1070) ID,AID,AX(ACNT),AY(ACNT),ARECNT(ACNT),NATTR
   IF (ARECNT(ACNT).GT.25) THEN
C     too many lines make up this area
   WRITE (*,*) '**** ARECNT: ',ARECNT(ACNT)
   READ (FL,1010) (LINID(I),I=1,ARECNT(ACNT))
   ACNT= ACNT- 1
   READ (FL,1010) ITMP,IX
   WRITE (*,*) '      SKIP: ',ITMP,IX
   WRITE(FOU,*) '      SKIP: ',ITMP,IX
   ELSE
C     an valid area header
   READ (FL,1010) (AREVAL(I,ACNT),I=1,ARECNT(ACNT))
   IF (NATTR.GT.0) THEN
C     attribute available
   READ (FL,1010) AREATR(ACNT),IX
   IF (IX.NE.0) THEN
C     whats this second attribute

```

```

        WRITE(FOU,*) 'second attribute:',IX
        WRITE(*,*) 'second attribute:',IX
        END IF
        IF (AREATR(ACNT).LT.AATMIN) AATMIN= AREATR(ACNT)
        IF (AREATR(ACNT).GT.AATMAX) AATMAX= AREATR(ACNT)
        AATCNT= AATCNT+ 1
    ELSE
C        no attribute
        AREATR(ACNT)= -1
    END IF
    END IF
    END IF
    GO TO 50
C
60 CONTINUE
C    all done
    LINSTR(LCNT+1)= TLIN+ 1
    CLOSE(UNIT=FL)
C
    WRITE (FOU,*) 'lines saved:',LCNT,' with attributes:',LATCNT
    WRITE (*,*) 'lines saved:',LCNT,' with attributes:',LATCNT
    WRITE (FOU,*) 'areas saved:',ACNT,' with attributes:',AATCNT
    WRITE (*,*) 'areas saved:',ACNT,' with attributes:',AATCNT
    XMIN = 1.0E30
    YMIN = 1.0E30
    XMAX = -1.0E30
    YMAX = -1.0E30
C
    DO 70 I= 1,TLIN
        IF (LINX(I).GT.XMAX) XMAX= LINX(I)
        IF (LINX(I).LT.XMIN) XMIN= LINX(I)
        IF (LINY(I).GT.YMAX) YMAX= LINY(I)
        IF (LINY(I).LT.YMIN) YMIN= LINY(I)
70 CONTINUE
C
    WRITE (FOU,2000) XMIN,XMAX,YMIN,YMAX
    WRITE (*,2000) XMIN,XMAX,YMIN,YMAX
C
    OPEN (UNIT=FL,FILE=FNAME//'.EXP',STATUS='NEW',ERR=900)
    write header
C    WRITE (*,*) 'enter dsn for dlg in export format: '
    READ(*,*) DSN
    WRITE (FL,2070) DSN
C
    IF (AATCNT.GT.0) THEN
C    process areas with attributes
        IAT1= 0
        IAT2= 0
        WRITE (*,*) 'looking for areas:',AATMIN,AATMAX
        DO 190 IH= AATMIN,AATMAX
            IA= 0
            DO 180 IC= 1,ACNT
                IF (AREATR(IC).EQ.IH) THEN
                    IA= IA+ 1
                    IF (IA.EQ.1) THEN
                        IAT1= IH
                        WRITE(*,*) 'attribute:',IAT1,IAT2
C                        we need to write type 1 info
                        WRITE(FL,2010) IAT1,IAT2
                        AL= 3
                        WRITE(FL,2020) AL,IAT1,AX(IC),AY(IC)
                    END IF
                END IF
                WRITE(*,*) ' for area:',IC,' with:',ARECNT(IC),' lines'
C

```

```

CPTS= 0
DO 130 I= 1,ARECNT(IC)
C   put lines into buffer
   J= AREVAL(I,IC)
   IF (J.NE.0) THEN
C     not an island, save points
     IF (J.LT.0) THEN
C       go last to first
         J = ABS(J)
         IDIR= -1
         IFR = LINSTR(J+1)- 1
         ILS = LINSTR(J)
     ELSE IF (J.GT.0) THEN
C       first first
         IDIR= 1
         IFR = LINSTR(J)
         ILS = LINSTR(J+1)- 1
     END IF
     IF (I.GT.1) THEN
C       dont need first point
         IFR = IFR+ IDIR
     END IF
     DO 110 K= IFR, ILS, IDIR
       CPTS= CPTS+ 1
       TX(CPTS)= LINX(K)
       TY(CPTS)= LINY(K)
110    CONTINUE
     END IF
     WRITE (*,*) 'line:',I,LINID(I),J,IFR,ILS,IDIR,CPTS
     IF (J.EQ.0 .OR. I.EQ.ARCNT(IC)) THEN
C       IF (CPTS.GT.5) THEN
         enough points to define area
         IFR= 1
115        CONTINUE
           ILEN= CPTS
           IF (ILEN.GT.1200) THEN
C             only write part to fit in a dlq buffer
               ILEN= 1200
               WRITE(FOU,*) 'part buffer:',IAT1,CPTS
             END IF
             ILS= IFR+ ILEN- 1
             IF (IFR.GT.1) THEN
C               WRITE(FOU,*) 'more buffer:',IFR,ILS,ILEN
             END IF
             write out line coordinates
             WRITE(FL,2030) ILEN*2
             WRITE(FL,2040) (TX(K),TY(K),K=IFR,ILS)
             WRITE(*,*) 'output line:',IAT1,IAT2,I,ILEN
C             see if more to write
               CPTS= CPTS- ILEN+ 1
               IFR = ILS
               IF (CPTS.GT.1) GO TO 115
             ELSE
C               skip line, not enough points
               WRITE (FOU,*) 'skip',CPTS
               WRITE (*,*) 'skip',CPTS
             END IF
             start at new position
             CPTS= 0
C           END IF
130          CONTINUE
        END IF
180        CONTINUE
190        CONTINUE

```

```

END IF
C
IF (LATCNT.GT.0) THEN
C
  process lines with attributes
  WRITE(*,*) 'looking for lines:',LAT1MN,LAT1MX,LAT2MN,LAT2MX,LCNT
  DO 250 IA=LAT1MN,LAT1MX
    DO 240 IB=LAT2MN,LAT2MX
      IX= 0
      DO 230 IC= 1,LCNT
        IF (LINAT1(IC).EQ.IA .AND. LINAT2(IC).EQ.IB) THEN
C
          got one that matches
          IX= IX+ 1
          IF (IX.EQ.1) THEN
C
            first match
            WRITE(*,*) 'attribute:', IA, IB
C
            we need to write type 1 info
            WRITE(FL,2010) IA, IB
C
            a handle at the mid point of the first line
            AL= 3
            K = (LINSTR(IC)+ LINSTR(IC+1))/2
            WRITE(FL,2020) AL, IA, LINX(K), LINY(K)
          END IF
C
          write out line coordinates
          WRITE(FL,2030) (LINSTR(IC+1)- LINSTR(IC))* 2
          WRITE(FL,2040) (LINX(K), LINY(K),
1
            K=LINSTR(IC), LINSTR(IC+1)-1)
          END IF
230
          CONTINUE
240
          CONTINUE
250
          CONTINUE
        END IF
C
      WRITE (FL,2080)
      CLOSE (UNIT=FL)
C
      GOTO 920
C
      errors on file opens
900
      CONTINUE
      WRITE (*,*) '*** ERROR ON OPEN OF OUTPUT DLG FILE ***'
920
      CONTINUE
C
C
      STOP
      END

```

**Export format
DLG data**

```

DATE
WDMSFL
SYSTEM
COMMENT
END COMMENT
DSN          99,   TYPE VECT  NDN   1   NUP   1   NSA  10   NSP  20
NDP 400
  LABEL
    TSPREC          0
  END LABEL
  DATA
GROUP TYPE LINE   MAJ ATTR   0  MIN ATTR   0
1     3     0  602818.50 3859568.50
2     16
  591534.19 3845580.50 596116.06 3845629.25 614415.62 3845836.50
  614159.69 3866434.50 614069.75 3873558.75 599976.37 3873385.75
  591256.94 3873300.75 591534.19 3845580.50
GROUP TYPE LINE   MAJ ATTR 180  MIN ATTR 201
1     3    180  613840.87 3866098.25
2     4
  613107.81 3865320.25 613840.87 3866098.25
2     4
  613047.62 3865251.00 613107.81 3865320.25
2     8
  606622.19 3859245.25 607422.25 3859932.50 611278.00 3863359.00
  613047.62 3865251.00

```

**Summary Output
From CNVDLG**

```

lines saved:          36 with attributes:          29
areas saved:          9 with attributes:           1
XMIN, XMAX:         591257.         614416.
YMIN, YMAX:         3845580.         3873559.

```