

NRC Publications Archive Archives des publications du CNRC

A program for magnetic tape input and control on the IBM 360 Bradt, D.

For the publisher's version, please access the DOI link below. / Pour consulter la version de l'éditeur, utilisez le lien DOI ci-dessous.

Publisher's version / Version de l'éditeur:

<https://doi.org/10.4224/21275884>

Report (National Research Council of Canada. Radio and Electrical Engineering Division : ERB), 1969-12

NRC Publications Archive Record / Notice des Archives des publications du CNRC :

<https://nrc-publications.canada.ca/eng/view/object/?id=fb2f65c2-a001-42c4-a336-b0020f31f521>

<https://publications-cnrc.canada.ca/fra/voir/objet/?id=fb2f65c2-a001-42c4-a336-b0020f31f521>

Access and use of this website and the material on it are subject to the Terms and Conditions set forth at

<https://nrc-publications.canada.ca/eng/copyright>

READ THESE TERMS AND CONDITIONS CAREFULLY BEFORE USING THIS WEBSITE.

L'accès à ce site Web et l'utilisation de son contenu sont assujettis aux conditions présentées dans le site

<https://publications-cnrc.canada.ca/fra/droits>

LISEZ CES CONDITIONS ATTENTIVEMENT AVANT D'UTILISER CE SITE WEB.

Questions? Contact the NRC Publications Archive team at

PublicationsArchive-ArchivesPublications@nrc-cnrc.gc.ca. If you wish to email the authors directly, please see the first page of the publication for their contact information.

Vous avez des questions? Nous pouvons vous aider. Pour communiquer directement avec un auteur, consultez la première page de la revue dans laquelle son article a été publié afin de trouver ses coordonnées. Si vous n'arrivez pas à les repérer, communiquez avec nous à PublicationsArchive-ArchivesPublications@nrc-cnrc.gc.ca.

Sw
Q.C.
N 21
417
ERB-811

ERB-811

UNCLASSIFIED

~~417~~ NATIONAL RESEARCH COUNCIL OF CANADA

RADIO AND ELECTRICAL ENGINEERING DIVISION



ANALYZED

A PROGRAM FOR MAGNETIC TAPE INPUT AND
CONTROL ON THE IBM 360

-D. BRADT-

OTTAWA
DECEMBER 1969

ANALYZED

ABSTRACT

This report describes a computer subprogram for magnetic tape input and control on the IBM 360. The subprogram is written in OS/360 assembler language and is compatible with both Fortran and PL/1. This subprogram gives the Fortran or PL/1 programmer more direct control over read errors, ends-of-file, and extremely short and long blocks.

A PROGRAM FOR MAGNETIC TAPE INPUT AND CONTROL
ON THE IBM 360

TABLE OF CONTENTS

1	INTRODUCTION
2	METHOD
2	DESCRIPTION OF ENTRY POINTS
4	HOW TO USE THE PROGRAM
4	GETBLK/GETBLOK
5	GETPOS/GETPOSN
5	SETLEN
6	MAXERR/MAXERRS
6	SETERR/SETERRS
7	GETNAM/GETNAME
7	SETMOD/SETMODE
8	SETDDN/SETDDNM
8	CLOSEF/CLOSEFL
9	PUTBLK/PUTBLOK
9	PUTEOF/PUTEOF
10	PREPARATION OF THE DATA DEFINITION (LD) CARD
12	ERROR AND ABNORMAL CONDITIONS
12	The Fortran coded subroutine MESOUT
14	ABENDS issued by TPCNTRL
15	EXAMPLES
16	EXAMPLE 1.
16	EXAMPLE 2.
16	EXAMPLE 3.
17	EXAMPLE 4.
17	EXAMPLE 5.
17	EXAMPLE 6.
17	EXAMPLE 7.
18	EXAMPLE 8.
18	EXAMPLE 9.
18	EXAMPLE 10.
18	EXAMPLE 11.
19	EXAMPLE 12.
19	EXAMPLE 13.
19	EXAMPLE 14.
21	EXAMPLE 15.
23	STATUS INFORMATION
23	UNIT STATUS BYTE
25	SENSE INFORMATION
25	SENSE DATA

A PROGRAM FOR MAGNETIC TAPE INPUT AND CONTROL
ON THE IBM 360

25 SENSE OPERATION
26 SENSE BYTE 0
26 SENSE BYTE 1
28 SENSE BYTE 2
28 SENSE BYTE 3
29 SENSE BYTE 4
29 SENSE BYTE 5

29 APPENDIX A

29 A source copy of the program TPCNTRL

29 APPENDIX B

29 A source copy of the Fortran subroutine MESOUT.

A PROGRAM FOR MAGNETIC TAPE INPUT AND CONTROL
ON THE IBM 360

1

INTRODUCTION

The advantages of writing programs in high-level languages such as FORTRAN and PL/1 are indisputable¹. The time gained in ease of programming and debugging usually more than compensates for the less efficient execution time of the resultant program. A major disadvantage to this trend in programming is that there are always things which cannot be done easily in FORTRAN or PL/1, or sometimes cannot be done at all using these high-level languages. It has been found that in most cases it is still advantageous to write the body of the program in the language best suited for the job, and to code the troublesome areas in assembler language in the form of a function subprogram, subroutine subprogram, or procedure.

One of these troublesome areas was found to be magnetic tape input and control. Difficulties are frequently encountered in reading tapes on one computer that have been produced on another computer, or on a data acquisition system that is not computer controlled. Such tapes frequently contain a large number of files and must be processed in a non-sequential manner. In such cases, it is more efficient to utilize the "advance file" and "backspace file" control commands which are not available using the high-level languages.

¹Panel Discussion on "System Debugging and Startup Problems", Spring Joint Computer Conference, Atlantic City, N.J., May 1, 1968.

METHOD

All entry points and their associated routines are collectively called TPCNTRL for easy reference within this report. The routines are coded in OS/360 assembler language in the form of a single control section (TPCNTRL). All I/O is accomplished by the Execute Channel Program (EXCP) access method, with flags set in the Data Control Block (DCB) to inhibit system error correction on input and control. System error correction is used in the output portion of the routine. By using EXCP, tape marks do not require the closing and reopening of the file, and only one DD statement is required, regardless of the number of files on the tape. The input and output operations consist of two Channel Command Words (CCW'S) chained together. Initially, the first CCW is a set mode no-operation, and the second CCW is a read or write operation. The system chains its own set mode command to the beginning of every magnetic tape Channel Program, and does a sense operation at the termination of the Channel Program. This is necessary in a multi-programming environment. The subroutine/procedure SETMOD/SETMODE changes the no-operation CCW in the Channel Program to a set mode operation specifying the correct parameters. The address of the user's array or character string (A) is moved directly into the input or output CCW. This makes the I/O unbuffered as the data is read directly into A. The multi-programming facility of OS/360 is relied upon to make this type of I/O efficient. All input or output is of type Undefined (RECFM=U) and no blocking or deblocking is performed. The DCB subparameters of BLKSIZE, LRECL, and RECFM are ignored if they are coded in the DD statement.

DESCRIPTION OF ENTRY POINTS

This subprogram appears to the user as a collection of Fortran subroutines, and PL/1 procedures. All Fortran entry points have six character names, and all PL/1 entry points have seven character names. The entry points are:

GETBLK/GETBLOK a subroutine/procedure for magnetic tape input and positioning.

GETPOS/GETPOSN a subroutine/procedure used to obtain the position on the tape in the form of a file and block count.

SETLEN a subroutine used to set an upper limit to the number of error recovery attempts

GETBLK is to use when a parity error is detected.

MAXERR/MAXERRS a subroutine/procedure used to set an upper to the number of parity errors to be allowed before execution of the program is terminated.

SETERR/SETERRS a subroutine/procedure used to set the number of error correction attempts to be used upon detection of an input parity error.

GETNAM/GETNAME a subroutine/procedure used to obtain the volume serial number of the tape being used.

SETMOD/SETMODE a subroutine/procedure used to specify the density and mode to be used for the following input or output if a seven-track tape is being used.

SETDDN/SETDDNM a subroutine/procedure used to specify the ddname of the DD card to be used by GETBLK/GETBLOK to open the file.

CLOSEF/CLOSEFL a subroutine/procedure used to close the file.

PUTBLK/PUTBLOK a subroutine/procedure used to output a block on magnetic tape.

PUTEOF/PUTEOFL a subroutine/procedure used to write a tape mark or end-of-file mark on magnetic tape.

MESOUT a Fortran coded subroutine which is given control upon detection of abnormal conditions. The user may easily modify this subroutine to suit his own particular needs. The PL/1 entry points do not have an analagous procedure to display error conditions.

HOW TO USE THE PROGRAM

GETBLK/GETBLOK

This is the entry point used for both input and control. The four possible calling sequences are:

i) from Fortran

```
CALL GETBLK(A,L)
CALL GETBLK(A,L,I)
CALL GETBLK(A,L,I,J)
CALL GETBLK(A,L,I,J,K)
```

ii) or from PL/1

```
CALL GETBLOK(A,L);
CALL GETBLOK(A,L,I);
CALL GETBLOK(A,L,I,J);
CALL GETBLOK(A,L,I,J,K);
```

GETBLK/GETBLOK assumes that the missing arguments are zero. The arguments coded have the following meanings:

A An array into which the block will be read. In Fortran this may be any type of array of any dimension. In PL/1, A must be a variable length character string. GETBLOK will insert the length of the block input into the Dope Vector for the character string A, in addition to returning the length in L as specified below.

L a) Set by the user:

L=-1 indicates to GETBLK or GETBLOK that the tape is to be positioned only. (Input will be suppressed).

L= 0 indicates to GETBLK or GETBLOK that after the tape is positioned (if necessary), a block will be input.

b) Returned by GETBLK or GETBLOK:

L>0 the length in bytes of the block just input.

L=0 the block just input was an end-of-file mark.

I I>0 the block to which the tape should be spaced

before input.

I=0 no block positioning is attempted.

J J>0 the file to which the tape should be spaced before input.

J=0 no file positioning is attempted. The block specified by I is read. If I is also zero, the next sequential block is input.

K A return code set by GETBLK or GETBLOK indicating the status of the completed I/O.

K=-1 a read error was detected while reading the current block.

K=0 normal return, no errors.

K=1 an end-of-file was detected while reading the last block, or while spacing forward to the block requested.

** Integer constants should never be used for I or K in the CALL statement, as GETBLK/GETBLOK modifies these arguments.

GETPOS/GETPOSN

All entry points which cause the tape to be moved, update a block and file count. The current position on the tape is always known, and is returned to the user by the GETPOS/GETPOSN subroutine/procedure.

The calling sequence is:

CALL GETPOS(I,J)
or CALL GETPOSN(I,J);

where I and J are returned to the user and represent the block and file at which the tape is currently positioned.

SETLEN

This routine should be used only by a Fortran program. It sets a maximum length to the input and prevents the array A from overflowing. The calling sequence is:

CALL SETLEN(N)

where N is the size of the array in bytes. For example the value of N for the following arrays is:

INTEGER A(20,50) N is 4000 bytes.

REAL*8 A(20) N is 160 bytes.

The Dope Vector describing the PL/1 character string contains the string's maximum length. This length is used by GETBLOK each time it is called.

MAXERR/MAXERRS

This subroutine/procedure sets an upper limit to the number of read errors to be allowed before execution of the program is terminated. The calling sequence is:

CALL MAXERR(N)
or CALL MAXERRS(N);

where N is the maximum number of errors permitted. After N read errors have been detected, control is passed to the Fortran subroutine MESOUT. A PL/1 procedure will ABEND with a user code of 2 if this maximum is reached.

SEIERR/SETERRS

This subroutine/procedure determines the number of times a block will be read in the event of a parity error. GETBLK/GETBLOK does not use the built-in system routines for error recovery. A reread attempt consists of a backspace block, followed by a read operation. The calling sequence for SETERR/SETERRS is:

CALL SETERR(N)
or CALL SETERRS(N);

where N-1 is the number of reread attempts.

CALL SETERR(1)
or CALL SETERRS(1);

will inhibit the error recovery attempt. No error recovery will be attempted for blocks with a length of less than 12 bytes, regardless of the value of N used in SETERR/SETERRS.

GETNAM/GETNAME

This subroutine/procedure will return the volume serial number of the tape being used. The volume serial number is obtained from the Unit Control Block (UCB) which is in protected core. This routine must not be used on computers with fetch protect as it will cause a protection interrupt. The calling sequence for GETNAM/GETNAME is:

```
CALL GETNAM(B)
or CALL GETNAME(C);
```

where B is a variable or array with a length of at least 8 bytes, and C is a character string also with a length of at least 8 bytes.

SETMOD/SETMODE

This subroutine/procedure will set the density and mode of a seven-track magnetic tape unit. The density of the tape is measured in bits per inch (bpi). The seven-track magnetic tape units have a byte converter and a BCD translator. These hardware options permit the packing of data, or translation from BCD to EBCDIC upon input. This subroutine/procedure must not be used with nine-track tapes. The nine-track tape units will accept odd parity, high density (800 bpi) tapes only. The calling sequence for SETMOD/SETMODE is:

```
INTEGER DEN, MODE
...
CALL SETMOD(DEN,MODE)
```

```
or DECLARE (DEN,MODE) BINARY FIXED (31,0);
...
CALL SETMODE(DEN,MODE);
```

where DEN=0 specifies low density (200 bpi),
 =1 specifies medium density (556 bpi)
 =2 specifies high density (800 bpi)

and MODE=0 specifies that the density and
 mode on the DD card is to be used.
 =1 should not be used.
 =2 specifies odd parity, converter on, translator
 off.
 =3 should not be used.
 =4 even parity, converter off, translator off.

=5 specifies even parity, converter off, translator on.
=6 specifies odd parity, converter off, translator off.
=7 specifies odd parity, converter off, translator on.

The byte converter on input will pack four six-bit characters into three eight-bit bytes in core.

The BCD translator on input will convert each six-bit BCD character into its eight-bit equivalent and store it in one eight-bit byte in core.

With both of the above options off, on input each six-bit character on tape will be input into one eight-bit byte in core with two high-order zeros (the first two bits).

SETDDN/SETDDNM

This subroutine/procedure gives the ddname to be used when the file is opened. This routine may be used only when the file is not open. An infraction of the above rule will cause an ABEND with a user code of 3. The subroutines/procedures GETBLK/GFTBLOK, GETNAM/GETNAME, PUTBLK/PUTBLOK and PUTEOF/PUTEOF1 cause the file to be opened. The calling sequence is:

```
REAL*8 DD/8HINPUTAPE/
```

```
    ...  
    CALL SETDDN(DD)
```

```
or DECLARE DD CHAR(8) INITIAL ('INPUTAPE');
```

```
    ...  
    CALL SETDDNM(DD);
```

The file may be closed by using CLOSEF/CLOSEFL, the tape used changed by SETDDN/SETDDNM, and the new file opened automatically by the next input.

CLOSEF/CLOSEFL

This subroutine/procedure is used to close the file. The system will close the file automatically at the termination of the job if this subroutine/procedure is not used. The calling sequence is:

```
CALL CLOSEF
```

A PROGRAM FOR MAGNETIC TAPE INPUT AND CONTROL
ON THE IBM 360

9

or CALL CLOSEFL;

PUTBLK/PUTBLOK
PUTEOF/PUTEOF

The subroutines/procedures PUTBIK/PUTBLOK and
PUTEOF/PUTEOF are included for completeness only. It is
highly recommended that they not be used, and that for any
tape output, one of the standard access methods be employed
directly from FORTRAN or PL/1.

PREPARATION OF THE DATA DEFINITION (DD) CARD

```
                {TAPE7}                {NL}
//INPUTAPE DD UNIT= {TAPE9}, DISP=OLD, LABEL= (n, {SL}),
                {BLP}

                {C}
                {0}                {E}
//                VOLUME=SER=XXXXXX, DCB= (DEN= {1}, TRTCH= {ET})
                {2}                {T}
                {omitted}
```

Where:

{ } indicates that a choice must be made.

(The "{" and "}" are not punched).

INPUTAPE is the ddname for the tape being used by GETBLK/GETBLOK. INPUTAPE is the default value if the ddname is not set by the subroutine/procedure SETDDN/SETDDNM. If this tape is used during the GO step of a catalogued procedure, the ddname becomes GO.INPUTAPE.

TAPE7
or
TAPE9

must be chosen to specify a seven or nine track tape.

DISP=OLD

must be used to specify input. Coding this parameter will cause the tape to be rewound and kept when the file is closed, or when the job is terminated.

n is the file to which the tape should initially be positioned by the system. If n is omitted, n=1 is assumed, and the parameter is coded: LABEL=(,...).

NL specifies No Label, or an unlabelled tape.

SL specifies a Standard Label.

BLP (Bypass Label Processing) prevents the system from checking the label. The default if this operand is omitted is LABEL=(,SL).

XXXXXX is the volume serial number of the tape. This is the value returned by the subroutine/procedure GETNAM/GETNAME.

The only DCB subparameters which may be coded are DEN and TRTCH, and these may be used for a seven-track magnetic tape only. The hardware of the nine track magnetic tape units will accept odd parity, 800 bpi tapes only.

DEN=0 for low density (200 bpi)
=1 for medium density (556 bpi)
=2 for high density (800 bpi)

TRTCH=C for odd parity, byte converter on, and BCD translator off. This corresponds to MODE=2 in the subroutine/procedure SETMOD/SETMODE.

=E for even parity, byte converter off, and BCD translator off. This corresponds to MODE=4 in the subroutine/procedure SETMOD/SETMODE.

=T for odd parity, byte converter off, and BCD translator on. This corresponds to MODE=7 in the subroutine/procedure SETMOD/SETMODE.

=ET for even parity, byte converter off, and BCD translator on. This corresponds to MODE=5 in the subroutine/procedure SETMOD/SETMODE.

TRTCH omitted for odd parity, byte converter off, and BCD translator off. This corresponds to MODE=6 in the subroutine/procedure SETMOD/SETMODE.

For further information about preparing the DD statement, the user should consult the IBM publication: IBM System/360 Operating System Job Control Language.

ERROR AND ABNORMAL CONDITIONS

The Fortran entry points of TPCNTRL pass control to a Fortran coded subroutine MESOUT upon detection of error or abnormal conditions. This allows Fortran users to easily code their own messages and actions. Both Fortran and PL/1 programs can check the return code (K) from GETBLK/GETBLOK. Terminal errors will cause an ABEND for PL/1 users.

The Fortran coded subroutine MESOUT

Control is passed to the Fortran coded subroutine MESOUT upon detection of any of the following conditions, as specified by the variable CODE. MESOUT should not change any of the parameters passed to it. Its only purpose is to display them. A coded example of MESOUT is shown in Appendix B. The calling sequence that TPCNTRL uses to invoke MESOUT is:

CALL MESOUT(CODE,IOB,DCB,ECB,P,DEB,UCB)

where CODE indicates the nature of the error or condition.

- CODE=1 The block just read was an end-of-file, or GETBLK was attempting to space forward to the block requested (I), and detected an end-of-file.
- =2 A parity error was detected (unit status byte=00001110). If the length of the block was greater than 12 bytes, the error persisted after the specified number of error correction attempts.
- =3 An error was detected on the last read. The status byte is different than that specified above and should be checked.
- =4 This is a warning condition only. The block just read contained a parity error, but GETBLK was able to correct the error by successive rereads. The number of read attempts may be calculated by: $B(15)-B(10)+1$ where B is described below.
- =5 The maximum number of read errors permitted (as set by MAXERR) has been reached. MESOUT should terminate the job. The execution of a RETURN statement from MESOUT will cause an ABEND with a user code of two.
- =6 TPCNTRL was unable to open the file. The usual cause of this error is a missing or misspelled

ddname. MESOUT should terminate the job. The execution of a RETURN statement from MESOUT will cause an ABEND with a user code of one.

- =7 The subroutine PUTBLK was called with a zero or negative length(L).
- =8 PUTBLK was unable to output the block correctly after 100 attempts (System error recovery is used for output). This error is usually caused by defective tape.
- =9 The user attempted to change the ddname via SETDDN and the file was already open. MESOUT should terminate the job. The execution of a RETURN statement from MESOUT will cause an ABEND with a user code of three.
- =10 GETBLK was called with the block requested (I) and/or the file requested (J) less than zero. MESOUT should terminate the job. The execution of a RETURN statement from MESOUT will cause an ABEND with a user code of four.
- IOB The Input/Output Block, an array 40 bytes long. If the array is declared as INTEGER IOB(10) then IOB(3) and IOB(4) contain the Channel Status Word (CSW).
- DCB The Data Control Block, an array 56 bytes long. If the array is declared as INTEGER DCB(14) then DCB(11) and DCB(12) contain the ddname before the file is open.
- ECB The Event Control Block, a single word long.
- B An array used within TPCNTRL. It contains positional information and other parameters which may be displayed.
- B(1) is the block requested (I).
- B(2) is the file requested (J).
- B(3) is the return code (K).
- B(4) is the block at which the tape is currently positioned.
- B(5) is the file in which the above block occurs.

- B(6) is internal to GETBLK.
- B(7) is the number of blocks in this file and is set only when an end-of-file is detected.
- B(8) is the file corresponding to the above block count.
- B(9) is the length in bytes of the last input or output.
- B(10) is the number of reads to use for error correction as set by SETERR.
- B(11) is the maximum number of errors permitted as set by MAXERR.
- B(12) is the number of errors to date.
- B(13) is internal to GETBLK.
- B(14) is the code (the first argument).
- B(15) is the number of read tries used. This variable is decremented from the value in B(10) to zero.

DEB The Data Extent Block, an array 36 bytes long.

UCB The Unit Control Block, an array 44 bytes long. If the array is declared as INTEGER*2 UCB(22) then UCB(7) and UCB(8) contain the physical address of the device. UCB(12), UCB(13) and UCB(14) contain the six sense bytes from the magnetic tape control unit. UCB(15), UCB(16) and UCB(17) contain the volume serial number of the tape.

ABENDS issued by TPCNTRL

User ABENDS are issued by TPCNTRL upon detection of certain errors by the PL/1 compatible routines. These ABENDS are also issued by the Fortran compatible routines if MESOUT attempts to return control after a terminal error. These ABENDS are:

- 1 TPCNTRL was unable to open the file. Check for missing or misspelled DD card for the tape.
- 2 The maximum number of read errors permitted, as set by

MAXERR/MAXERRS has been reached.

- 3 The user attempted to change the ddname when the file was still open.
- 4 GETBLK/GETBLOK was invoked with invalid (negative) block (I) and/or file (J) arguments.

EXAMPLES

The use of these routines is illustrated in the following examples. Examples 1 to 10 illustrate the input and positioning entry point of the subroutine. PL/1 programs should use the seven character entry points and are similar to the Fortran examples. Example 11 illustrates the method by which positional information is obtained from the subroutine (GETBLK maintains a block and file count). Initially, the programmer should set the number of error retries (Example 12) and set a size limit on the input array size if using Fortran (Example 13). Programmers coding in Fortran should use the subroutine GETBLK, and those coding in PL/1 should use the procedure GETBLOK.

1. Input the next block.
2. Input block I within this file.
3. Input block I in file J.
4. Position tape to block I within this file but do not input the block.
5. Position tape to block I file J but do not input the block.
6. Rewind the tape. (no input)
7. Backspace to beginning of previous file. (no input)
8. Advance to beginning of next file and input the first block.
9. Advance N files. (no input)
10. Backspace N records, or to beginning of tape, or to a file mark (whichever comes first) then input a block.
11. Get position on the tape. (The file number and block

number of the next sequential input).

12. Set the number of error retries to N.
13. Set the buffer length to N.
14. A Fortran program to dump the first n files of a tape onto the line printer.
15. A PL/1 program to dump a magnetic tape onto the line printer. The dump starts at block I1, file J1 and continues to block I2, file J2.

EXAMPLE 1.

Input the next block.

```
L=0
CALL GETBLK(A, L)
```

or if the return code is desired

```
L=0
CALL GETBLK(A, L, 0, 0, K)
```

or

```
L=0
I=0
J=0
CALL GETBLK(A, L, I, J, K)
```

EXAMPLE 2.

Input block I within this file.

```
L=0
CALL GETBLK(A, L, I)
```

or if the return code is desired

```
L=0
CALL GETBLK(A, L, I, 0, K)
```

EXAMPLE 3.

Input block I in file J.

```
L=0
CALL GETBLK(A, L, I, J)
```

or if the return code is desired

```
L=0
CALL GETBLK(A,L,I,J,K)
```

EXAMPLE 4.

Position tape to block I within this file but do not input the block.

```
L=-1
CALL GETBLK(A,L,I,0,K)
```

or if you wish to find out which file you are in

```
L=-1
CALL GETPOS(IBLOCK,IFILE)
CALL GETBLK(A,L,I,IFILE,K)
```

See Example 11 for use of GETPOS routine.

EXAMPLE 5.

Position the tape to block I file J but do not input the block.

```
L=-1
CALL GETBLK(A,L,I,J,K)
```

EXAMPLE 6.

Rewind the tape (no input).

```
L=-1
CALL GETBLK(A,L,1,1,K)
```

Requesting block 1, file 1 will cause successive backspace blocks to be executed if the tape is currently positioned in file 1. If the tape is not in file 1, an actual rewind command is issued.

EXAMPLE 7.

Backspace to the beginning of the previous file (no input).

```
L=-1
CALL GETPOS(IBLOCK,IFILE)
CALL GETBLK(A,L,0,IFILE-1,K)
```

or

```
L=-1
CALL GETPOS(IBLOCK,IFILE)
CALL GETBLK(A,L,1,IFILE-1,K)
```

EXAMPLE 8.

Advance to the beginning of the next file and input the first block.

```
L=0
CALL GETPOS(IBLOCK,IFILE)
CALL GETBLK(A,L,0,IFILE+1,K)
```

or

```
L=0
CALL GETPOS(IBLOCK,IFILE)
CALL GETBLK(A,L,1,IFILE+1,K)
```

EXAMPLE 9.

Advance N files (no input).

```
L=-1
CALL GETPOS(IBLOCK,IFILE)
CALL GETBLK(A,L,0,IFILE+N,K)
```

EXAMPLE 10.

Backspace N blocks, or to the beginning of the tape, or to a file-mark (whichever comes first), then input a block.

```
L=0
CALL GETPOS(IBLOCK,IFILE)
CALL GETBLK(A,L,IBLOCK-N,0,K)
```

or

```
L=0
CALL GETPOS(IBLOCK,IFILE)
CALL GETBLK(A,L,IBLOCK-N,IFILE,K)
```

The expression $IBLOCK-N$ must be greater than zero. $IBLOCK-N < 0$ will cause an ABEND, and $IBLOCK-N = 0$ will cause no positioning to take place.

EXAMPLE 11.

Get the current position of the tape. (The file number and block number of the next sequential input).

```
CALL GETPOS(IBLOCK,IFILE)
```

IBLOCK and IFILE are returned by GETPOS. They are the block and file at which the tape is positioned.

EXAMPLE 12.

Set the number of error retries to N.

```
CALL SETERR(N)
```

EXAMPLE 13.

Set the buffer length to N bytes.

```
CALL SETLEN(N)
```

where N is the size of the array in bytes. This routine does not have to be used from PL/1 since GETBLOK will set the buffer length to the maximum length of the variable size character string as specified in the declare statement.

EXAMPLE 14.

A Fortran program to dump the first n files of a tape onto the line printer in character format. Assume the tape is seven-track and the following information is to be read from a data card.

- i) N the number of files to be dumped,
- ii) DEN the density of the tape,
- iii) MODE the mode to be used to set-up the tape unit,
- iv) NERRS the number of error correction attempts to use,
- v) MXERRS the maximum number of errors permitted,
- vi) DDNAME the ddname of the DD card to be used for the tape.

```
//DJBPORT JOB DJB,D.BRADT,MSGLEVEL=1
// EXEC PORTCLG
//FORT.SYSIN DD *
      INTEGER A(250), N, DEN, MODE, NERRS, MXERRS
      INTEGER READER/1/, PRINTR/3/
      REAL*8 DDNAME, VOLSER
C
C----SET LENGTH TO BUFFER A (4X250=1000).
C
      CALL SETLEN(1000)
C
C----READ THE DATA CARD.
C
      100 READ(READER,101) N, DEN, MODE, NERRS, MXERRS, DDNAME
```

A PROGRAM FOR MAGNETIC TAPE INPUT AND CONTROL
ON THE IBM 360

20

```

101 FORMAT(5I5, 2X, A8)
C
C-----CALL THE SUBROUTINES FOR ARGUMENTS READ
C-----FROM THE DATA CARD.
C
      CALL SETMOD(DEN,MODE)
      CALL SETERR(NERRS)
      CALL MAXERR(MXERRS)
      CALL SETDDN(DDNAME)
C
C----GET THE VOLUME SERIAL NUMBER OF THE TAPE AND
C----WRITE IT OUT.
C
      CALL GETNAM(VOLSER)
      WRITE(PRINTR,102) VOLSER
102 FORMAT('1THE VOLUME SERIAL NUMBER IS', A8, /)
C
C----SET UP THE TWO LOOPS: J FOR FILES
C                          I FOR BLOCKS
C
C          999999 IS AN ARBITRARILY LARGE NUMBER
C
      DO 120 J=1,N
C
      DO 115 I=1,999999
      L=0
      CALL GETBLK(A,L,I,J,K)
C
C-----IF(END-OF-FILE) GET OUT OF INNER LOOP.
      IF(K.EQ. 1) GO TO 120
      L=(L+3)/4
C          THE LENGTH IS NOW IN WORDS.
      WRITE(PRINTR,103) (A(M),M=1,L)
103   FORMAT(1H , 30A4)
115   CONTINUE
120 CONTINUE
C
      WRITE(PRINTR,130)
130  FORMAT('ONORMAL TERMINATION.')
```

CALL EXIT
END

```

/*
//LKED.SYSIN DD *
      INSERT BINARY DECK FOR TPCNTRL HEPE.
      INSERT BINARY DECK FOR MESOUT HERE
/*
//GO.FT03F001 DD SYSOUT=A
//GO.DJBTAPE1 DD UNIT=TAPE7,DISP=OLD,LABEL=(,BIP),
//          DCB=(DEN=1,TRTCH=ET),VOLUME=SER=ARL014
//GO.FT01F001 DD *
```

```
1      1      5      10     15  DJBTAP1
/*
```

EXAMPLE 15.

A PL/I program to dump the contents of a magnetic tape onto the line printer. The dump starts at block I1, file J1 and continues to block I2, file J2. I1, J1, I2, J2 and the ddname are read from a data card.

```
//DJBPL1 JOB DJB,D.PRADT,MSGLEVEL=1
// EXEC PL1CLG
//PL1.SYSIN DD *
TEST: PROCEDURE OPTIONS(MAIN);
    DECLARE A CHAR(4000) VARYING,
            DDNAME CHAR(8),
            (L,I,J,K) BINARY FIXED(31,0),
            (STARTFILE,STARTBLOCK,STOPFILE,STOPBLOCK)
                    BINARY FIXED(31,0);
    DECLARE (START,STOP) BINARY FIXED(31,0);
    DECLARE (DEN, MODE, NERRS) BINARY FIXED(31,0);
    GET LIST(STARTFILE,STARTBLOCK,STOPFILE,
            STOPBLOCK,DDNAME);
    CALL SETDDNM(DDNAME);
    DEN=1; MODE=5;
    CALL SETMODE(DEN,MODE);
    NERRS=5;
    CALL SETERRS(NERRS);
OUTERLOOP: DO J=STARTFILE TO STOPFILE;
    IF J=STARTFILE THEN START=STARTBLOCK;
    ELSE START=1;
    IF J=STOPFILE THEN STOP=STOPBLOCK;
    ELSE STOP=999999;
INNERLOOP: DO I=START TO STOP;
    L=0; K=0;
    CALL GETBLOK(A,L,I,J,K);
    IF K<0 THEN GOTO ENDOUTERLOOP;
    ELSE PUT SKIP EDIT(A) (A);
    END INNERLOOP;
ENDOUTERLOOP: END OUTERLOOP;
END;
*PROCESS;
MESOUT: PROCEDURE;
/* DUMMY MESOUT ROUTINE TO SATISFY LINK EDITOR */
END MESOUT;
/*
//LKED.SYSIN DD *
    INSERT BINARY DECK FOR TPCNTRL HERE.
/*
//GO.PL1TAPE DD UNIT=TAPE7,DISP=OLD,VOLUME=SER=ARL014,
```

A PROGRAM FOR MAGNETIC TAPE INPUT AND CONTROL
ON THE IBM 360

22

```
//      DCB=(DEN=1,TRTCH=ET),LABEL=(,NL)
//GO.SYSIN DD *
1,1,1,100,'PL1TAPE '
/*
```

STATUS INFORMATION

The description of the status and sense information is taken directly from the IBM publication: IBM System/360 Component Description 2400-Series Magnetic Tape Units.

UNIT STATUS BYTE

Status information consists of the channel status byte (eight bits of information that indicate the status of the channel) and the unit status byte (eight bits of information that indicate the status of the selected control unit and tape unit). Status information is present in the tape control or channel until the channel status word (CSW) is stored or another operation is accepted. Status information is renewed at the beginning of each tape operation (except Test I/O) to indicate initial status; it is updated during the tape operation to provide ending status at termination of the I/O operation. This information is set in the status portion, bits 32-47, of the channel status word by an I/O interruption. Under certain conditions, the status information may be set in the CSW by a Start I/O, Test I/O, or Halt I/O instruction. Bits 32-39 of the CSW (unit status byte) identify to the program the conditions in the tape control that caused the storing of the CSW. Bits 40-47 of the CSW indicate conditions associated with the channel and are described in the publication IBM System/360 Principles of Operation.

Bit 0 Attention
Not used.

Bit 1 Status Modifier
Present with busy to indicate TCU busy, or has interrupt pending.

Bit 2 Control Unit End
Signaled by the TCU:

1. At completion of an operation during which a TCU busy was indicated.
2. At the completion of the control unit portion of an operation during which a unit check or unit exception is detected.
3. At completion of a command on the alternate interface of a simultaneous read/write TCU that caused the TCU busy to be given.

Bit 3 Busy
The busy status indication can change at any time. When presented without bit 1 (status modifier),

indicates:

1. That the addressed tape unit is busy (i.e., rewinding),
 2. The addressed tape unit is switched.
- When presented with bit 1, indicates:

1. That the TCU has an interrupt pending, or
2. For the simultaneous TCU, the addressed tape unit is in operation on the alternate interface, or
3. The interface section required by the command is already in operation with a similar type command.

Bit 4 Channel end

Indicates that a read, read backward, write, control, or sense command has been completed, or in the case of certain control commands involving tape motion, that the operation has been initiated at the TCU and the channel has been released.

Bit 5 Device end

Indicates:

1. The tape unit has completed a command, or
2. The tape unit has changed from not ready to ready, if an attempt had been made to select it while it was not ready, e.g., by issuing a Test I/O instruction; or the tape control initiated a rewind-unload that caused the tape unit to go not ready, or
3. A tape unit has reached load point as the result of a program initiated rewind, or
4. A rewind unload is completed at the control unit level, i.e., when the tape unit becomes not ready or,
5. A switched tape unit has become not switched, if an attempt had been made to select it while it was switched.

Bit 6 Unit check

Set whenever:

1. Any bit in sense byte 0, or bit 7 of byte 1, is set to 1.
2. The tape unit performs read backward, backspace block, or backspace file into or at load point.
3. A rewind-unload is completed at the TCU level.

Bit 7 Unit exception

Set when:

1. A write, write tape mark, or erase gap operation is performed in the end of tape area, i.e., when tape indicate is on, or
2. A tape mark is sensed during a read, read

backward, forward space block, or backspace block.

PROGRAMMING NOTES:

Tape indicate (TI), set on by sensing the end of tape (EOT) reflective marker in the forward direction, stays on (unless reset) until set off by a backward command. Unit exception, however, remains set in unit status only until the channel accepts status, except for a sense, NOP, or Test I/O operation.

If a seven-track tape mark is read in a density other than the density in which it was written, proper recognition is not guaranteed.

Data received while reading backward into load point is not to be considered valid even if there is no separate data check.

Channel end, device end, and unit check in the unit status byte, and intervention required in the sense data are indicated if a tape unit becomes not ready while performing a command.

SENSE INFORMATION

SENSE DATA

Sense data provides detailed information about the selected I/O device and the last I/O operation on the TCU. The information consists of both unusual conditions detected in the last operation and the status of the device. The status information provided by the sense command is more detailed than that supplied by the unit status byte and may describe reasons for the unit check indication.

SENSE OPERATION

The sense operation causes the transfer of the sense data from the selected tape unit and from the tape control unit to main storage. The data is placed in storage in an ascending order of addresses, starting with the address specified in the CCW. The number of bytes may be specified by the count field in the CCW, otherwise all six bytes will be transferred.

PROGRAMMING NOTE: Any sense information bits (except bits 1-6 of byte 1) pertaining to the last I/O operation are reset by the next command, other than a sense command, Test I/O instruction, or a NOP command addressed to the control unit. If the control unit detects an equipment error or invalid parity of the sense command code, the equipment check or bus-out check bits are turned on, and unit check is sent with the channel end. In the case of invalid parity of

a command, sense data that pertains to the preceding operation is not reset.

SENSE_BYTE_1

Bit 0 Command reject

Set when a write, write tape mark, or erase command is addressed to a file protected tape unit, or a data-converter-on control command that is addressed to a seven-track tape unit is recognized on a TCU with the seven-track compatibility feature but without the data converter feature. In this case, mode set is executed for parity, density, and translator.

Bit 1 Intervention required

Set whenever tape unit status A is inactive, i.e., tape unit is not ready or nonexistent. See "Sense Byte 1."

Bit 2 Bus-out check

Set whenever even parity appears on the information bus lines from the channel to the control unit.

Bit 3 Equipment check

Set when reject tape unit (bit 1, byte 4) or sequence error (bit 5, 6, or 7 of byte 4) is set.

Bit 4 Data check

Set when a data check occurs. See "Sense Byte 3."

Bit 5 Overrun

Set if service is requested, but data cannot be transferred during a read, write, or read backward operation. Data transfer stops as soon as condition is detected. Note: Data check during overrun suppresses the overrun indication.

Bit 6 Word count zero

Set during a write operation if transfer of data is prevented before the first byte of data. When word count zero is set, no tape motion occurs.

Bit 7 Data converter check

Set when reading a tape not written with converter on and the number of bytes is not a multiple of four.

SENSE_BYTE_1

Bit 0 Noise

During a read forward space block, indicates that data was recognized after the normal LRC byte time but not

long enough after to be considered a new block. Data before the LRC byte is checked and transferred; data after the LRC byte turns on the noise bit and maintains tape motion, but is not transferred.

When connected to Model 2 control, during a read backward or backspace block, if data is recognized after the disconnect sequence is started. With Model 1 control, data recognized after start of disconnect, is transferred as part of block. Noise bit is not set; data check is probable.

During a write, erase gap, or write tape mark, indicates that data (or noise caused by tape defects) was detected at the read head before the block or tape mark was written, or during erase gap while the tape was being erased. Data check and unit check are indicated.

Bit 1 TU status A
Selected and ready

Bit 2 TU status B
Not ready, or rewinding, or under the control of another TCU via the 2816 Switching Unit. Assuming no outstanding device end status, the bits determine response to initial selection as follows:

Looking at both of above bits (AB)

AB=00 nonexistent, response to initial selection is Unit check.

AB=01 Not ready, response to initial selection is Unit check, arm for device end.

AB=10 Ready and not rewinding and not switched, response to initial selection is Clear status.

AB=11 Ready and rewinding or switched or power is down on a tape unit attached through a switching unit, response to initial selection is Busy, arm for device end.

NOTE: Unit check is not signalled for a sense operation. Following unit check or busy indication, device end will be signalled when the tape unit becomes ready and not rewinding.

Bit 3 Seven-track
The selected tape unit has the seven-track feature installed.

Bit 4 Load point
The selected unit is at load point.

Bit 5 Selected and write status
The selected tape unit is in write status.

Bit 6 File protect

The selected tape unit is in file protect status.

Bit 7 Not capable

Not used, always set to zero.

SENSE BYTE 2

This sense byte contains the track-in-error (TIE) indicator bits that are set at the end of a read or read backward command if a data check has been encountered. At the end of properly executed read or read backward with no data check and at the end of all other commands, sense byte 2 contains at least bits 6 and 7 set to ones. No error correction is attempted when operating with seven-track tape units; bits 6 and 7 are set to ones in sense byte 2.

SENSE BYTE 3

Bit 0 R/W VRC

A vertical redundancy check occurred during a read or read backward operation. Indicator is not set after an overrun or after receipt of a stop signal. (Stop signal resulted from Halt I/O instruction in CPU).

Bit 1 LRCR

A longitudinal redundancy check occurred during write, write tape mark, read, or read backward operation.

Bit 2 Skew

Excessive skew detected by a read back check during a write, write tape mark, or erase operation.

Bit 3 CRC

A cyclic redundancy check occurred during a read or read backward operation (nine-track only).

Bit 4 Skew reg VRC

A character with incorrect parity detected in skew register during write, write tape mark, or erase operation.

Bit 5 Phase encoding

Not applicable; always set to zero.

Bit 6 Backward

The selected tape unit is in backward status.

Bit 7 C compare

C compare is an equipment check. It indicates that

parity of data into the data register did not equal
that out of the data register.

NOTE: Bits 0-4 and 7 of byte 3 indicate data checks. Any
of these will set data check (bit 4, byte 0).

SENSE BYTE 4

Bit 0 Not used

Bit 1 Reject TU

Selected tape unit failed to respond to set read or set
write status when instructed, or became not ready
during execution of a tape motion operation. Equipment
check (bit 3, byte 0) also set.

Bits 2-7 Maintenance aids.

SENSE BYTE 5

Bits 0-7 Always zero.

APPENDIX A

A source copy of the program TPCNTRI

APPENDIX B

A source copy of the Fortran subroutine MESOUT.

PRINT OFF	A	0
GETB TITLE 'MAGNETIC TAPE INPUT AND COX	A	10
NTROL ROUTINE'	A	20
PRINT ON	A	30
***** VERSION C-1 OCTOBER 1969. *****	A	40
*	A	50
* D.J.BRADT, RADIO ASTRONOMY SECTION, N.R.C.	A	60
*	A	70
* THE PURPOSE OF THIS ROUTINE IS TO MAKE THE CONTENTS OF A	A	80
* MULTIFILE REEL OF MAGNETIC TAPE EASILY AVAILABLE TO A FORTRAN	A	90
* OR PL/1 PROGRAMMER IN A RANDOM OR SEQUENTIAL MANNER, AND TO ALLOW	A	100
* THE PROGRAMMER TO HANDLE CONDITIONS SUCH AS READ ERRORS, AND EOF'S	A	110
* HIMSELF.	A	120
*	A	130
* FORTRAN USES THE SIX CHARACTER ENTRY POINTS.	A	140
* PL/1 USES THE SEVEN CHAPACTER ENTRY POINTS.	A	150
*	A	160
*-----	A	170
*	A	180
* I N P U T	A	190
*	A	200
* FORTRAN CALLING SEQUENCES:	A	210
*	A	220
* 1) CALL GETBLK(A,L)	A	230
* 2) CALL GETBLK(A,L,I)	A	240
* 3) CALL GETBLK(A,L,I,J)	A	250
* 4) CALL GETBLK(A,L,I,J,K)	A	260
*	A	270
*	A	280
* PL/1 CALLING SEQUENCES:	A	290
*	A	300
* 1) CALL GETBLOK(A,L);	A	310
* 2) CALL GETBLOK(A,L,I);	A	320
* 3) CALL GETBLOK(A,L,I,J);	A	330
* 4) CALL GETBLOK(A,L,I,J,K);	A	340
*	A	350
*	A	360

* SET L=-1 TO SUPPRESS INPUT AFTER POSITIONING	A 740
*	A 750
*****	A 760
* DO NOT USE AN INTEGER CONSTANT IN CALLING SEQUENCE FOR L.	A 770
*****	A 780
*	A 790
*	A 800
-----	A 810
*	A 820
* OPENING THE FILE.	A 830
*	A 840
* THE FILE IS OPENED AUTOMATICALLY DURING THE FIRST CALL TO GETBLK,	A 850
* GETBLOK, PUTBLK, PUTEOF, OR GETNAM. THE INITIAL FILE (N) AT WHICH	A 860
* THE TAPE IS OPENED IS PICKED UP FROM THE LABEL=(N,...) PARAMETER	A 870
* ON THE DD CARD. GETBLK THEN CALCULATES THE DEB AND UCB ADDRESSES	A 880
* FOR THE MESOUT PARAMETER LIST IN CASE THERE IS AN ERROR. IF PL/1	A 890
* IS USED, GETBLOK WILL CAUSE AN ABEND WITH A USER CODE OF ONE IF IT	A 900
* IS UNABLE TO OPEN THE FILE.	A 910
*	A 920
-----	A 930
*	A 940
*	A 950
-----	A 960
*	A 970
* REWIND..	A 980
* CALLING SEQUENCE..	A 990
* L=-1	A 1000
* CALL GETBLK(A,L,1,1,K)	A 1010
*	A 1020
* FILES AND RECORDS CAN BE BACKSPACED AND SKIPPED BY PERFORMING	A 1030
* INTEGER ARITHMETIC ON THE RECORD AND FILE ARGUMENTS I AND J.	A 1040
*	A 1050
*	A 1060
*	A 1070
-----	A 1080
*	A 1090
* GETPOS....A ROUTINE TO OBTAIN CURRENT POSITION OF TAPE.	A 1100

*	GETPOSN		A 1110
*	CALLING SEQUENCE..		A 1120
*		CALL GETPOS(I,J)	A 1130
*			A 1140
*	THE TAPE IS POSITIONED AT THE ITH RECORD OF THE JTH FILE		A 1150
*			A 1160
*	I AND J ARE RETURNED TO THE USER.		A 1170
*			A 1180
*			A 1190
*	-----		A 1200
*	SETLEN..A ROUTINE TO SET A MAXIMUM SIZE ON THE BUFFER AREA.		A 1210
*			A 1220
*	FORTRAN CALLING SEQUENCE.... CALL SETLEN(N)		A 1230
*			A 1240
*	WHERE..		A 1250
*		N=THE BUFFER SIZE IN BYTES.	A 1260
*		IF THE BLOCK BEING INPUT IS LONGER THAN THE	A 1270
*		BUFFER AREA, THE DATA TRANSMISSION WILL BE	A 1280
*		SUPPRESSED WHEN THE BUFFER AREA IS FULL.	A 1290
*		IF PL/I IS USED GETBLOK PERFORMS THE SAME FUNCTION	A 1300
*		AS THIS ROUTINE USING THE MAXIMUM LENGTH OF THE	A 1310
*		CHARACTER STRING A OBTAINED FROM THE DOPE VECTOR	A 1320
*		OF A.	A 1330
*			A 1340
*			A 1350
*	-----		A 1360
*			A 1370
*	SETERR	A ROUTINE TO SET THE NUMBER OF ATTEMPTS TO READ A	A 1380
*	SETERRS	RECORD WHEN AN ERROR IS DETECTED.	A 1390
*		PRESET TO 25.	A 1400
*			A 1410
*	CALLING SEQUENCE..	N=10	A 1420
*		CALL SETERR(N)	A 1430
*	INVOKING THIS ROUTINE ONCE WILL SET THE NUMBER OF READ		A 1440
*	ATTEMPTS FOR THE DURATION OF THE PROGRAM.		A 1450
*			A 1460
*			A 1470

*		A 1480
*		A 1490
*	-----	A 1500
*		A 1510
*	MAXERR A ROUTINE TO SET AN UPPER LIMIT ON THE NUMBER	A 1520
*	MAXERRS OF ERRORS TO BE PERMITTED.	A 1530
*		A 1540
*	PRESET TO 25.	A 1550
*		A 1560
*		A 1570
*	-----	A 1580
*		A 1590
*		A 1600
*	SETMOD A ROUTINE TO SET THE MODE OF A SEVEN TRACK MAGNETIC	A 1610
*	SETMODE TAPE UNIT.	A 1620
*		A 1630
*	CALLING SEQUENCE: CALL SETMOD(DEN, MODE)	A 1640
*		A 1650
*		A 1660
*	DEN=0 FOR 200 BPI DENSITY.	A 1670
*	1 FOR 556 BPI DENSITY.	A 1680
*	2 FOR 800 BPI DENSITY.	A 1690
*		A 1700
*	MODE=0 USE DEN AND TRTCH SUBPARAMETERS FROM DD CARD.	A 1710
*	1 MUST NOT BE USED.	A 1720
*	2 FOR ODD PARITY, CONVERTER ON, TRANSLATOR OFF.	A 1730
*	3 MUST NOT BE USED.	A 1740
*	4 FOR EVEN PARITY, CONVERTER OFF, TRANSLATOR OFF.	A 1750
*	5 FOR EVEN PARITY, CONVERTER OFF, TRANSLATOR ON.	A 1760
*	6 FOR ODD PARITY, CONVERTER OFF, TRANSLATOR OFF.	A 1770
*	7 FOR ODD PARITY, CONVERTER OFF, TRANSLATOR ON.	A 1780
*		A 1790
*	-----	A 1800
*		A 1810
*		A 1820
*	-----	A 1830
*		A 1840

*		A 1850
*	-----	A 1860
*		A 1870
*	GETNAM A ROUTINE TO OBTAIN THE VOLUME SERIAL NUMBER OF THE	A 1880
*	GETNAME TAPE BEING USED.	A 1890
*		A 1900
*	CALLING SEQUENCE: CALL GETNAM(NAME)	A 1910
*		A 1920
*	WHERE: NAME IS ANY VARIABLE OR ARRAY AT LEAST 8 BYTES LONG.	A 1930
*	(THE VOLUME SERIAL NUMBER IS OBTAINED FROM THE UCB AND THEREFORE	A 1940
*	GETNAM WILL NOT WORK ON COMPUTERS WITH FETCH PROTECT.	A 1950
*	-----	A 1960
*		A 1970
*		A 1980
*	ERROR MESSAGE OUTPUT.	A 1990
*		A 2000
*	GETBLK/GETBLOK ATTEMPTS NO ERROR MESSAGE OUTPUT.	A 2010
*	UPON DETECTION OF AN ERROR CONDITION, CONTROL IS PASSED TO	A 2020
*	THE SUBROUTINE/PROCEDURE MESOUT, WHICH IS WRITTEN IN FORTRAN.	A 2030
*		A 2040
*		A 2050
*	IF PL/1 IS USED THERE WILL BE NO ERROR OUTPUT.	A 2060
*		A 2070
*	THE CALLING SEQUENCE IS..	A 2080
*		A 2090
*	CALL MESOUT(MSGCOD, IOB, DCB, ECB, B, DEB, UCB)	A 2100
*		A 2110
*	-----MESSAGE CODE FOR MESOUT SUBROUTINE.-----	A 2120
*		A 2130
*	*MSGCODE=1 END-OF-FILE DETECTED. (READ OR FORWARD SPACE RECORD).	A 2140
*		A 2150
*	2 READ ERROR. (LATERAL OR LONGITUDINAL PARITY ERROR).	A 2160
*		A 2170
*	3 READ ERROR--POSSIBLE EOF. (DIFFERENT STATUS BYTE THAN	A 2180
*	USUAL READ ERROR.	A 2190
*		A 2200
*		A 2210

*	4	READ ERROR THAT WAS CORRECTED BY SUCCESSIVE RE-READS.	A 2220
*			A 2230
*	5	MAXIMUM NUMBER OF ERRORS EXCEEDED. (AS SET BY MAXERR).	A 2240
*		USER SHOULD TERMINATE JOB.	A 2250
*		A RETURN TO GETBLK WILL CAUSE A USER ABEND OF 2.	A 2260
*			A 2270
*	6	UNABLE TO OPEN FILE. (POSSIBLE MISSING OR MISPELLED	A 2280
*		DD CARD. (USER SHOULD TERMINATE JOB).	A 2290
*		A RETURN TO GETBLK WILL CAUSE A USER ABEND 1.	A 2300
*			A 2310
*	7	USER CALLED PUTBLK WITH L<=0.	A 2320
*			A 2330
*	8	PERMANENT OUTPUT ERROR..PUTBLK.	A 2340
*			A 2350
*	9	ATTEMPT TO CHANGE DD NAME WHEN FILE IS STILL OPEN.	A 2360
*			A 2370
*	10	I OR J ARE NEGATIVE IN CALL TO GETBLK.	A 2380
*			A 2390
*			A 2400
*			A 2410
*	IOB	INPUT/OUTPUT BLOCK	A 2420
*		INTEGER ARRAY 10 WORDS.	A 2430
*			A 2440
*	DCB	DATA CONTROL BLOCK	A 2450
*		INTEGER ARRAY 14 WORDS.	A 2460
*			A 2470
*	ECB	EVENT CONTROL BLOCK	A 2480
*		INTEGER VARIABLE.	A 2490
*			A 2500
*			A 2510
*	B IS	AN INTEGER ARRAY WITHIN GETBLK.	A 2520
*			A 2530
*	B(1)	=RECORD REQUESTED. (I).	A 2540
*	B(2)	=FILE REQUESTED. (J).	A 2550
*	B(3)	=RETURN CODE. (K).	A 2560
*	B(4)	=RECORD AT WHICH TAPE IS CURRENTLY POSITIONED.	A 2570
*	B(5)	=FILE AT WHICH TAPE IS CURRENTLY POSITIONED.	A 2580

*	B(6)=-----INTERNAL TO GETBLK-----	A 2590
*	B(7)=NUMBER OF RECORDS IN THIS FILE (WHEN EOF DETECTED).	A 2600
*	B(8)=FILE AT WHEN EOF DETECTED. =B(5)-1.	A 2610
*	B(9)=LENGTH IN BYTES OF LAST INPUT OR OUTPUT.	A 2620
*	B(10)=NO. OF REREADS TO USE FOR ERROR CORRECTION.	A 2630
*	AS SET BY SETERR.	A 2640
*	B(11)=MAXIMUM NO. OF ERRORS PERMITTED. AS SET BY MAXERR.	A 2650
*	B(12)=NUMBER OF READ ERROPS TO DATE.	A 2660
*	B(13)=-----INTERNAL TO GETBLK-----	A 2670
*	B(14)=MSGCOD	A 2680
*	B(15)=NUMBER OF READ TRYS USED. USEFUL WHEN ERROR WAS	A 2690
*	CORRECTED.. B(15) IS DECREMENTED FROM B(10) TO ZERO.	A 2700
*		A 2710
*	DEB DATA EXTENT BLOCK	A 2720
*	INTEGER*2 ARRAY 18 HALF-WORDS.	A 2730
*		A 2740
*	UCB UNIT CONTROL BLOCK	A 2750
*	INTEGER*2 ARRAY 22 HALF-WORDS.	A 2760
*		A 2770
***		A 2780
***	UCB(12), UCB(13), UCB(14) CONTAIN THE SIX SENSE BYTES FROM THE	A 2790
***	MAGNETIC TAPE CONTROL UNIT.	A 2800
***		A 2810
***	UCB(15), UCB(16), UCB(17) CONTAIN THE VOLUME SERIAL NUMBER OF	A 2820
***	THE TAPE.	A 2830
***		A 2840
*	MESOUT SHOULD NOT CHANGE ANY OF THE PARAMETERS PASSED TO IT.	A 2850
*	ITS ONLY PURPOSE IS TO DISPLAY ERROR CONDITIONS.	A 2860
*		A 2870
*	-----REGISTER USAGE WITHIN THIS PROGRAM-----	A 2880
*		A 2890
*	0 SUBROUTINE AND I/O LINKAGE.	A 2900
*	1 SUBROUTINE AND I/O LINKAGE.	A 2910
*	2 WORK REGISTER.	A 2920
*	3 WORK REGISTER.	A 2930
*	4 BASE REGISTER.	A 2940
		A 2950

*	5	MSGCODE FOR MESOUT ROUTINE.	A 2960
*	6	LINKAGE TO INTERNAL SUBROUTINES.	A 2970
*	7	RECORD (RECORD REQUESTED).	A 2980
*	8	FILE (FILE REQUESTED).	A 2990
*	9	RETURN CODE.	A 3000
*	10	PRECORD (PRESENT RECORD).	A 3010
*	11	PFILE (PRESENT FILE).	A 3020
*	12	=1	A 3030
*	13	SAVE AREA ADDRESS.	A 3040
*	14	RETURN ADDRESS. (SUBROUTINE LINKAGE).	A 3050
*	15	ADDRESS OF ROUTINE CALLED. (SUBROUTINE LINKAGE).	A 3060
*			A 3070
*			A 3080

```

TPCNTRL  START 0
          PRINT ON,NODATA,NOGEN
          ENTRY GETBLK
          ENTRY GETBLOK
          ENTRY GETPOS
          ENTRY GETPOSN
          ENTRY MAXERR
          ENTRY MAXERRS
          ENTRY SETLEN
          ENTRY SETERR
          ENTRY SETERRS
          ENTRY SETMOD
          ENTRY SETMODE
          ENTRY GETNAM
          ENTRY GETNAME
          ENTRY PUTBLK
          ENTRY PUTBLOK
          ENTRY PUTEOF
          ENTRY PUTEOFL
          ENTRY SETDDN
          ENTRY SETDDNM
          ENTRY CLOSEF
          ENTRY CLOSEFL

```

```

*
*

```

```

          DC      X'7',CL7'TPCNTRL'
          USING  TPCNTRL,4

```

```

*
*-----PL/1 ENTRY POINT.
*

```

```

GETBLOK  B      16(0,15)      PL/1 ENTRY POINT.
          DC      X'7',CL7'GETBLOK'
          DC      A(TPCNTRL)
          STM     14,12,12(13)  STORE REGISTERS USED.
          L       4,12(0,15)   LOAD THE BASE REGISTER.
          ST      13,SAVEAREA+4 CHAIN SAVEAREAS BACK.
          LA      12,SAVEAREA

```

```

A 3100
A 3110
A 3120
A 3130
A 3140
A 3150
A 3160
A 3170
A 3180
A 3190
A 3200
A 3210
A 3220
A 3230
A 3240
A 3250
A 3260
A 3270
A 3280
A 3290
A 3300
A 3310
A 3320
A 3330
A 3340
A 3350
A 3360
A 3370
A 3380
A 3390
A 3400
A 3410
A 3420
A 3430
A 3440
A 3450
A 3460

```

ST	12,8(0,13)	CHAIN SAVEAREAS AHEAD.	A 3470
LR	13,12	ADDRESS OF NEW SAVEAREA.	A 3480
MVI	PL1FLAG,X'FF'	TURN ON PL1 FLAG.	A 3490
L	2,0(0,1)	ADDRESS OF DOPE VECTOR FOR A.	A 3500
MVC	CCWIN+1(3),1(2)	MOVE ADDR OF A TO INPUT CCW.	A 3510
LH	3,4(0,2)	MAX. LENGTH OF CHARACTER STRING FROM DOPE VECTOR.	A 3520 A 3530
STH	3,CCWCNT	AS PER SETLEN.	A 3540
B	GETBLK1		A 3550

*

*			A 3570
*-----			A 3580
		FORTAN ENTRY POINT.	A 3590
*			A 3600
GETBLK	B	16(0,15)	A 3610
	DC	X'6',CL6'GETBLK'	A 3620
	DC	A(TPCNTRL) BASE ADDRESS.	A 3630
	STM	14,12,12(13) STORE REGISTERS USED.	A 3640
	L	4,12(0,15) LOAD THE BASE REGISTER.	A 3650
	ST	13,SAVEAREA+4 CHAIN BACK.	A 3660
	LA	12,SAVEAREA	A 3670
	ST	12,8(0,13) CHAIN AHEAD.	A 3680
	LR	13,12	A 3690
	MVI	PL1FLAG,X'00' TURN OFF PL1 FLAG.	A 3700
	MVC	CCWIN+1(3),1(1) MOVE INAREA TO INPUT CCW.	A 3710
GETBLK1	LM	6,8,4(1) L,RECORD,FILE FROM ARGUMENT LIST.	A 3720
	L	6,0(0,6) VALUE OF LENGTH.	A 3730
	ST	6,LENGTH	A 3740
	CLI	4(1),X'80' LAST PARAMETER?	A 3750
	BE	ARGEND YES.	A 3760
	L	7,0(0,7) VALUE OF RECORD.	A 3770
	CLI	8(1),X'80' WAS RECORD LAST ARGUMENT?	A 3780
	BE	ARGEND+2 YES.	A 3790
	L	8,0(0,8) VALUE OF FILE.	A 3800
	B	OPENCHEK	A 3810
ARGEND	SR	7,7 RECORD=0	A 3820
	SR	8,8 FILE=0	A 3830
OPENCHEK	LM	10,12,PRECORD	A 3840
	SR	9,9 RTCODE=0.	A 3850
	MVC	READTRYS(4),NERRTRYS READTRYS=25.	A 3860
	MVI	RERRFLAG,X'00' TURN OFF READ ERROR FLAG.	A 3870
	CLI	OPENFLAG,X'FF' CHECK IF FILE IS OPEN.	A 3880
	BE	*+8 YES.	A 3890
	BAL	6,OPENFILE NO..OPEN IT.	A 3900
	LTR	3,8	A 3910
	BZ	FILEOK IF FILE=0 (NOT SPECIFIED).	A 3920
	BM	CODE10 NEGATIVE FILE ARGUMENT IS ILLEGAL.	A 3930
	SR	3,11 FILE-PFILE.	

	BP	SKPFILE		A 3940
	BM	BSPFILE	POSITION TAPE TO PREVIOUS FILE.	A 3950
*				A 3960
FILEOK	LTR	3,7		A 3970
	BZ	RECOK	IF RECORD=0 (NOT SPECIFIED).	A 3980
	BM	CODE10	NEGATIVE RECORD ARGUMENT ILLEGAL.	A 3990
	SR	3,10	RECORD-PRECORD	A 4000
	BP	SKPRECD	POSITION TAPE BACKWARDS.	A 4010
	BM	BSPRECD	POSITION TAPE FORWARD.	A 4020
*				A 4030
RECOK	LR	7,10	RECORD=PRECORD /IN CASE THEY WEREN'T	A 4040
	LR	8,11	FILE=PFILE /SPECIFIED EXPLICITLY	A 4050
	CLI	LENGTH,X'FF'	CHECK IF LENGTH IS NEG. (L=-1)?	A 4060
	BE	RETURN	YES..NO INPUT..POSITION ONLY.	A 4070
	MVC	IOBIN+17(3),CCWMODEA	CCW ADDR TO IOB.	A 4080
	LH	3,CCWCNT		A 4090
	STH	3,CCWIN+6	SETUP MAXIMUM BUFFER SIZE IN CCW.	A 4100
	BAL	6,IORUTINE		A 4110
	AR	10,12	PRECORD=PRECORD+1	A 4120
*				A 4130
*-----	INPUT	IS COMPLETE..CHECK STATUS BYTE. IN CCW.		A 4140
*				A 4150
	CLI	CSW+4,X'0C'	NORMAL TERMINATION?	A 4160
	BE	READOKC	YES.	A 4170
	CLI	CSW+4,X'0D'	END OF FILE?	A 4180
	BE	EOFILE		A 4190
	CLI	CSW+4,X'0E'	READ ERROR?	A 4200
	BE	READERR	YES.	A 4210
	L	5,=F'3'	ANYTHING ELSE IS AN ERROR.	A 4220
	BAL	6,MESSAGE		A 4230
	B	READOK		A 4240
*				A 4250
READOKC	CLI	RERRFLAG,X'00'		A 4260
	BE	READOK	OFF.	A 4270
	L	5,=F'4'		A 4280
	BAL	6,MESSAGE	READ ERROR THAT WAS CORRECTED.	A 4290
*				A 4300

READOK	LH	3,CCWCNT	CALCULATE LENGTH.	A 4310
	SH	3,CSW+6		A 4320
	ST	3,LENGTH		A 4330
*				A 4340
RETURN	STM	7,11,RECORD	SAVE VARIABLES KEPT IN REGISTERS.	A 4350
	L	13,SAVEAREA+4		A 4360
	L	1,24(0,13)	RESTORE REGISTER 1.	A 4370
	CLI	PL1FLAG,X'00'	CHECK PL1 FLAG.	A 4380
	BE	RET1		A 4390
	L	2,0(0,1)		A 4400
	STH	3,6(0,2)	RETURN LENGTH TO DOPE VECTOR.	A 4410
RET1	L	2,4(0,1)		A 4420
	ST	3,0(0,2)	RETURN LENGTH TO L.	A 4430
	LTR	2,2		A 4440
	BM	NORTCDPM	LENGTH IS LAST PARAMETER.	A 4450
	L	2,8(0,1)		A 4460
	LTR	2,2		A 4470
	BM	NORTCDPM	BLOCK IS LAST PARAMETER.	A 4480
	L	2,12(0,1)		A 4490
	LTR	2,2		A 4500
	BM	NORTCDPM	FILE IS LAST PARAMETER.	A 4510
	L	2,16(0,1)		A 4520
	ST	9,0(0,2)		A 4530
RETURN12	EQU	*		A 4540
NORTCDPM	LM	14,12,12(13)	RESTORE REGISTERS.	A 4550
	BR	14	RETURN.	A 4560

*				A 4580
*-----END-OF-FILE DETECTED.				A 4590
EOFILE	S	10,=F'2'		A 4600
	STM	10,11,RCRDEOF	SAVE RECORD, FILE AT WHEN EOF DETECTED.	A 4610
	LR	10,12	PRECORD=1	A 4620
	AR	11,12	PFILE=PFILE+1	A 4630
	SR	3,3	LENGTH=0	A 4640
	ST	3,LENGTH		A 4650
	L	5,=F'1'		A 4660
	LR	9,5	RTCODE=1	A 4670
	BAL	6,MESSAGE		A 4680
	B	RETURN		A 4690
*				A 4700
*-----I OR J WERE NEGATIVE..CALL MESOUT OR ABEND 4.				A 4710
*				A 4720
CODE10	L	5,=F'10'	CODE 10 TO MESOUT.	A 4730
	BAL	6,MESSAGE		A 4740
	ABEND	4		A 4750
*				A 4760
*-----SKIP N FILES, N IN REG 3.				A 4770
*				A 4780
SKPFILE	MVC	IOBIN+17(3),CCWFSFA	MOVE FSF CCW TO IOB.	A 4790
	BAL	6,IORUTINE		A 4800
	SR	3,12		A 4810
	BP	SKPFILE+6		A 4820
	LR	11,8	PFILE=FILE	A 4830
	LR	10,12	PRECCRD=1	A 4840
	B	FILEOK		A 4850
*				A 4860
*-----BACKSPACE N FILES..-N IS IN REG. 3.				A 4870
*				A 4880
BSPFILE	LR	2,8		A 4890
	SR	2,12		A 4900
	BZ	REWIND	REWIND IF WANT FILE 1.	A 4910
	MVC	IOBIN+17(3),CCWBSFA	MOVE BSF CCW TO IOB.	A 4920
	SR	3,12	BACKSPACE AN EXTRA FILE.	A 4930
BSPFILE1	BAL	6,IORUTINE		A 4940

AR	3,12			A 4950
BM	BSPFILE1			A 4960
LR	3,12	SET R3=1		A 4970
B	SKPFILE	TO SKIP OVER LAST FILEMARK.		A 4980
*				A 4990
*	*-----SKIP N RECORDS, N IS IN REG 3.			A 5000
*				A 5010
SKPRECD	MVC	IOBIN+17(3),CCWFSRA		A 5020
SKPRECD1	BAL	6,IORUTINE		A 5030
	AR	10,12	PRECORD=PRECORD+1	A 5040
	CLI	CSW+4,X'0D'		A 5050
	BE	EOFILE		A 5060
	SR	3,12		A 5070
	BP	SKPRECD1		A 5080
	B	RECOK		A 5090
*				A 5100
*	*-----BACKSPACE N RECORDS, -N IS IN REG. 3.			A 5110
*				A 5120
BSPRECD	MVC	IOBIN+17(3),CCWBSRA		A 5130
BSPRECD1	BAL	6,IORUTINE		A 5140
	AR	3,12		A 5150
	BM	BSPRECD1		A 5160
	LR	10,7	PRECORD=RECORD	A 5170
	B	RECOK		A 5180
*				A 5190
*	*-----REWIND THE TAPE.			A 5200
*				A 5210
REWIND	MVC	IOBIN+17(3),CCWREWA		A 5220
	BAL	6,IORUTINE		A 5230
	LR	10,12	PRECORD=1	A 5240
	LR	11,12	PFILE=1	A 5250
	B	FILEOK		A 5260
*				A 5270
*	*-----READ ERROR DETECTED.			A 5280
*				A 5290
READERR	LH	3,CCWCNT	CALCULATE COUNT.	A 5300
	SH	3,CSW+6		A 5310

	SH	3,=H'12'		A 5320
	MVI	RERRFLAG,X'FF'	TURN ON READ ERROR FLAG.	A 5330
	BM	NOERRCOR	NO ERPOR CORRECTION IF < 12 BYTES.	A 5340
	L	3,READTRYS		A 5350
	SR	3,12		A 5360
	ST	3,READTRYS	READTRYS=READTRYS-1	A 5370
	BZ	NOERRCOR		A 5380
	L	3,=F'-1'	R3=-1	A 5390
	B	BSPRECD	BACKSPACE REC & TRY AGAIN.	A 5400
NOERRCOR	L	3,ERR2DATE	NUMBER OF ERRORS TO DATE.	A 5410
	AR	3,12		A 5420
	ST	3,ERR2DATE		A 5430
	LH	2,CCWCNT	CALCULATE LENGTH BEFORE CALL	A 5440
	SH	2,CSW+6	TO MESOUT ROUTINE.	A 5450
	ST	2,LENGTH		A 5460
	L	5,=F'2'	READ ERROR MESSAGE CODE.	A 5470
	S	3,MAXMERRS		A 5480
	BM	CONT		A 5490
	L	5,=F'5'		A 5500
	BAL	6,MESSAGE		A 5510
	ABEND	2 IF PL/1	OR MESOUT DOESN'T TERMINATE.	A 5520
CONT	BAL	6,MESSAGE		A 5530
	L	9,=F'-1'	RTCODE=-1	A 5540
	B	READOK		A 5550
	*			A 5560
	*-----	RESET BITS IN DCB AND CALL I/O SUPERVISOR WITH IOB ADDR.		A 5570
	*			A 5580
	DS	OF		A 5590
IORUTINE	MVI	TAPE+44,X'0C'	DON'T USE SYSTEM ERROR RECOVERY.	A 5600
	NI	TAPE+48,X'D2'		A 5610
	OI	TAPE+48,X'02'		A 5620
	LA	1,IOBIN		A 5630
	SVC	0		A 5640
	WAIT	ECB=ECB		A 5650
	BR	6		A 5660
	*			A 5670
	*-----	ROUTINE TO CALL MESOUT.		A 5680

*-----RETURN CURRENT POSITION ON THE TAPE.

*

```

      DS      OF
GETPOS    B      16(0,15)
GETPOSN   EQU    GETPOS
      DC      X'6',CL6'GETPOS'
      DC      A(TPCNTRL)
      STM     14,4,12(13)
      L       4,12(0,15)
      L       2,0(0,1)
      L       3,PRECORD
      ST      3,0(0,2)
      L       2,4(0,1)
      L       3,PFILE
      ST      3,0(0,2)
RETURN4   LM     2,4,28(13)
      BR      14

```

*

*-----SET A MAXIMUM TO THE NUMBER OF ERRORS PERMITTED.

*

```

      DS      OF
MAXERR    B      16(0,15)
MAXERRS   EQU    MAXERR
      DC      X'6',CL6'MAXERR'
      DC      A(TPCNTRL)
      STM     14,4,12(13)
      L       4,12(0,15)
      L       2,0(0,1)
      L       2,0(0,2)
      ST      2,MAXMERRS
      B       RETURN4

```

```

A 6050
A 6060
A 6070
A 6080
A 6090
A 6100
A 6110
A 6120
A 6130
A 6140
A 6150
A 6160
A 6170
A 6180
A 6190
A 6200
A 6210
A 6220
A 6230
A 6240
A 6250
A 6260
A 6270
A 6280
A 6290
A 6300
A 6310
A 6320
A 6330
A 6340
A 6350

```

*				A 6370
*-----	SET DD NAME INTO DCB BEFORE OPENING FILE.			A 6380
*				A 6390
*-----	FORTTRAN ENTRY POINT.			A 6400
*				A 6410
	DS	OF		A 6420
SETDDN	B	16(0,15)		A 6430
	DC	X'6',CL6'SETDDN'		A 6440
	DC	A(TPCNTRL)		A 6450
	STM	14,12,12(13)		A 6460
	L	4,12(0,15)		A 6470
	L	2,0(0,1)	ADDR OF DD NAME.	A 6480
SETDD1	CLI	OPENFLAG,X'00'	CHECK IF FILE IS OPEN.	A 6490
	BE	SETDD2	NO.	A 6500
	L	5,=F'9'	ERROR..CAN'T CHANGE	A 6510
	BAL	6,MESSAGE	DDNAME IF FILE IS OPEN.	A 6520
	ABEND	3	IN CASE MESOUT RETURNS.	A 6530
SETDD2	MVC	TAPE+40(8),0(2)	MOVE DDNAME INTO DCB.	A 6540
	B	RETURN12		A 6550
*				A 6560
*-----	SET DDNAME PL/1 ENTRY POINT			A 6570
*				A 6580
	DS	OF		A 6590
SETDDNM	B	16(0,15)		A 6600
	DC	X'7',CL7'SETDDNM'		A 6610
	DC	A(TPCNTRL)		A 6620
	STM	14,12,12(13)		A 6630
	L	4,12(0,15)	LOAD BASE REGISTER.	A 6640
	L	2,0(0,1)	ADDR OF DOPE VECTOR.	A 6650
	L	2,0(0,2)	ADDR OF CHAR STRING FROM DOPE VECTOR.	A 6660
	CLI	OPENFLAG,X'00'	CHECK IF FILE IS OPEN.	A 6670
	BE	SETDD3		A 6680
	ABEND	3	CAN'T CHANGE DDNAME IF FILE OPEN.	A 6690
SETDD3	MVC	TAPE+40(8),0(2)	MOVE DDNAME INTO DCB.	A 6700
	B	RETURN12		A 6710

*
 *-----ROUTINE TO CLOSE THE FILE.
 *

```

CLOSEF DS OF
        B 16(0,15)
CLOSEFL EQU CLOSEF
        DC X'6',CL6'CLOSEF'
        DC A(TPCNTRL)
        STM 14,12,12(13)
        L 4,12(0,15)
        CLI OPENFLAG,X'00'
        BE RETURN12
        CLOSE (TAPE,DISP)
        MVI OPENFLAG,X'00'
        B RETURN12
  
```

IF FILE NOT OPEN, DON'T CLOSE IT.
 PEWIND AND UNLOAD.
 RESET OPEN FLAG.

A 6730
 A 6740
 A 6750
 A 6760
 A 6770
 A 6780
 A 6790
 A 6800
 A 6810
 A 6820
 A 6830
 A 6840
 A 6850
 A 6860
 A 6870

	DS	OF	A	6890
SETLEN	B	16(0,15)	A	6900
	DC	X'6',CL6'SETLEN'	A	6910
	DC	A(TPCNTRL)	A	6920
	STM	14,4,12(13)	A	6930
	L	4,12(0,15)	A	6940
	L	2,0(0,1)	A	6950
	L	2,0(0,2)	A	6960
	STH	2,CCWCNT	A	6970
	B	RETURN4	A	6980
	DS	OF	A	6990
SETERR	B	16(0,15)	A	7000
SETERRS	EQU	SETERR	A	7010
	DC	X'6',CL6'SETERR'	A	7020
	DC	A(TPCNTRL)	A	7030
	STM	14,4,12(13)	A	7040
	L	4,12(0,15)	A	7050
	L	2,0(0,1)	A	7060
	L	2,0(0,2)	A	7070
	ST	2,NERRTRYS	A	7080
	B	RETURN4	A	7090

SETMOD	DS	OF		A 7110
	B	16(0,15)		A 7120
SETMODE	EQU	SETMOD		A 7130
	DC	X'6',CL6'SETMOD'		A 7140
	DC	A(TPCNTRL)		A 7150
	STM	14,4,12(13)		A 7160
	L	4,12(0,15)		A 7170
	LM	2,3,0(1)	ADDRESS OF DENSITY, MODE ARG.	A 7180
	L	2,0(0,2)	VALUE OF DEN.	A 7190
	L	3,0(0,3)	VALUE OF MODE.	A 7200
	N	2,=F'3'	STRIP DEN TO 3 BITS.	A 7210
	N	3,=F'7'	STRIP MODE TO 3 BITS.	A 7220
	SLL	2,6(0)		A 7230
	SLL	3,3(0)		A 7240
	AR	2,3		A 7250
	A	2,=F'3'	ADD IN MODE SET OPERATION.	A 7260
	STC	2,CCWMODE		A 7270
	STC	2,CCWMODE2		A 7280
	B	RETURN4		A 7290

GETNAM	DS	OF		A 7310
	B	16(0,15)		A 7320
	DC	X'6',CL6'GETNAM'		A 7330
	DC	A(TPCNTRL)		A 7340
	STM	14,12,12(13)		A 7350
	L	4,12(0,15)		A 7360
	CLI	OPENFLAG,X'FF'		A 7370
	BE	GETNAM1	FILE IS OPEN.	A 7380
	BAL	6,OPENFILE		A 7390
	L	1,24(0,13)	RESTORE REG. 1.	A 7400
GETNAM1	L	1,0(0,1)		A 7410
	L	2,TAPE+44		A 7420
	A	2,=F'32'		A 7430
	L	2,0(0,2)	UCB ADDR.	A 7440
	MVI	0(1),X'40'		A 7450
	MVC	1(6,1),28(2)	VOL SER NO FROM UCB	A 7460
	MVI	7(1),X'40'		A 7470
	B	RETURN12		A 7480
*				A 7490
*-----		ROUTINE TO OBTAIN VOLUME SERIAL NUMBER FROM UCB	PL/1.	A 7500
*				A 7510
GETNAME	DS	OF		A 7520
	B	16(0,15)		A 7530
	DC	X'7',CL7'GETNAME'		A 7540
	DC	A(TPCNTRL)		A 7550
	STM	14,12,12(13)		A 7560
	L	4,12(0,15)	LOAD THE BASE REGISTER.	A 7570
	MVI	PL1FLAG,X'FF'	TURN ON PL/1 FLAG.	A 7580
	CLI	OPENFLAG,X'FF'		A 7590
	BE	GETNAM2		A 7600
	BAL	6,OPENFILE	OPEN FILE IF NECESSARY.	A 7610
	L	1,24(0,13)	RESTORE REGISTER 1.	A 7620
GETNAM2	L	1,0(0,1)	ADDRESS OF DOPE VECTOR.	A 7630
	B	GETNAM1	SAME AS FORTRAN GETNAM.	A 7640

*			A 7660
*-----			A 7670
	FORTRAN OUTPUT.		A 7680
*			A 7690
PUTBLK	DS OF		A 7700
	B 16(0,15)		A 7710
	DC X'6',CL6'PUTBLK'		A 7720
	DC A(TPCNTRL)	BASE ADDRESS.	A 7730
	STM 14,12,12(13)	SAVE REGISTERS USED.	A 7740
	L 4,12(0,15)	LOAD BASE REG.	A 7750
	ST 13,SAVEAREA+4	CHAIN SAVEAREAS BACK.	A 7760
	LA 12,SAVEAREA		A 7770
	ST 12,8(0,13)	CHAIN AHEAD.	A 7780
	LR 13,12	NEW SAVEAREA.	A 7790
	MVI PL1FLAG,X'00'	TURN OFF PL1 FLAG.	A 7800
	L 3,4(0,1)	ADDRESS OF LENGTH (L).	A 7810
	L 3,0(0,3)	VALUE OF LENGTH.	A 7820
	MVC CCWOUT+1(3),1(1)	MOVE OUTAREA TO OUTPUT CCW.	A 7830
PUTBLK1	STH 3,CCWOUT+6	MOVE LENGTH TO OUTPUT CCW.	A 7840
	ST 3,LENGTH		A 7850
	LTR 3,3		A 7860
	BP *+12	MAKE SURE L>0.	A 7870
	L 5,=F'7'	L<=0, CALL MESOUT.	A 7880
	B WERR		A 7890
	CLI OPENFILE,X'FF'	CHECK IF FILE IS OPEN.	A 7900
	BE *+8		A 7910
	BAL 6,OPENFILE	OPEN FILE IF NOT ALREADY OPEN.	A 7920
	MVC IOBIN+17(3),CCWMD2A	MOVE CCW ADDR TO IOB.	A 7930
	MVI TAPE+44,X'00'	USE SYSTEM ERROR CORRECTION FOR	A 7940
*		OUTPUT.	A 7950
	BAL 6,IORUTINE	OUTPUT THE BLOCK.	A 7960
	CLI CSW+4,X'0C'	NORMAL TERMINATION?	A 7970
	BE OUTPUTOK	YES.	A 7980
	L 5,=F'8'	PERMANENT WRITE ERROR.	A 7990
WERR	BAL 6,MESSAGE		A 8000
	L 6,=F'-1'	L=-1 FOR ERROR RETURN.	A 8010
	ST 6,LENGTH		A 8020
OUTPUTOK	L 13,SAVEAREA+4	CLD SAVEAREA	

L 3,PRECORD
A 3,=F'1'
ST 3,PRECORD
L 1,24(0,13)
L 6,4(0,1)
L 5,LENGTH
ST 5,0(0,6)
B RETURN12

PRECORD=PRECORD+1
ARGUMENT LIST.
ADDRESS OF L.
RETURN L.

A 8030
A 8040
A 8050
A 8060
A 8070
A 8080
A 8090
A 8100

*				A 8120
*-----	PL/1	OUTPUT.		A 8130
*				A 8140
	DS	OF		A 8150
PUTBLOK	B	16(0,15)		A 8160
	DC	X'7',CL7'PUTBLOK'		A 8170
	DC	A(TPCNTRL)		A 8180
	STM	14,12,12(13)	SAVE REGISTERS.	A 8190
	L	4,12(0,15)	LOAD BASE REG.	A 8200
	ST	13,SAVEAREA+4		A 8210
	LA	12,SAVEAREA		A 8220
	ST	12,8(0,13)	CHAIN AHEAD.	A 8230
	LR	13,12		A 8240
	MVI	PL1FLAG,X'FF'	TURN ON PL1 FLAG.	A 8250
	L	2,0(0,1)	ADDRESS OF DOPE VECTOR FOR A.	A 8260
	MVC	CCWOUT+1(3),1(2)	MOVE OUTAREA ADDR TO CCW.	A 8270
	LH	3,6(0,2)	PICK UP LENGTH FROM DOPE VECTOR.	A 8280
	B	GETBLK1		A 8290

```

*
*-----WRITE AN END-OF-FILE.
*
      DS      OF
PUTEOF   B      16(0,15)
PUTEOF   EQU    PUTEOF
      DC      X'6',CL6'PUTEOF'
      DC      A(TPCNTRL)
      STM     14,12,12(13)
      L       4,12(15)
      ST      13,SAVEAREA+4
      LA      12,SAVEAREA
      ST      12,8(0,13)
      LR      13,12
      MVI     PL1FLAG,X'00'
      CLI     OPENFLAG,X'FF'
      BE      *+8
      BAL     6,OPENFILE
      MVC     IOBIN+17(3),CCWTMKA
      MVI     TAPE+44,X'00'
      BAL     6,IORUTINE
      L       3,=F'1'
      L       2,PFILE
      AR      2,3
      ST      2,PFILE
      ST      3,PRECORD
      L       13,SAVEAREA+4
      B       RETURN12
      EJECT

```

```

*
*-----PARAMETER LIST FOR MESOUT.
*

```

```

MSGPARM  DC      A(MSGCODE)
          DC      A(IOBIN)
          DC      A(TAPE)
          DC      A(ECB)
          DC      A(RECORD)

```

```

PFILE=PFILE+1
PRECORD=1

```

```

A 8310
A 8320
A 8330
A 8340
A 8350
A 8360
A 8370
A 8380
A 8390
A 8400
A 8410
A 8420
A 8430
A 8440
A 8450
A 8460
A 8470
A 8480
A 8490
A 8500
A 8510
A 8520
A 8530
A 8540
A 8550
A 8560
A 8570
A 8580
A 8590
A 8600
A 8610
A 8620
A 8630
A 8640
A 8650
A 8660
A 8670

```

DEBADDR	DS	F	A 8680
UCBADDR	DS	F	A 8690
*			A 8700
*			A 8710
RECORD	DS	F	A 8720
FILE	DS	F	A 8730
RTCODE	DS	F	A 8740
PRECORD	DS	F	A 8750
PFILE	DS	F	A 8760
ONE	DC	F'1'	A 8770
RCRDEOF	DS	F	A 8780
FILEEOF	DS	F	A 8790
LENGTH	DS	F	A 8800
NERRTRYS	DC	F'25'	A 8810
MAXMERRS	DC	F'25'	A 8820
ERR2DATE	DC	F'0'	A 8830
MESOUTA	DC	V(MESOUT)	A 8840
MSGCODE	DS	F	A 8850
READTRYS	DS	F	A 8860
SAVEAREA	DS	18F	A 8870
*			A 8880
*-----SINGLE BYTE FLAGS.....X'00'=OFF, AND X'FF'=ON.			A 8890
*			A 8900
PL1FLAG	DC	X'00'	A 8910
RERRFLAG	DC	X'00'	A 8920
OPENFLAG	DC	X'00'	A 8930
*			A 8940
* I N P U T / O U T P U T B L O C K .			A 8950
*			A 8960
IOBIN	DS	OD ALIGN ON DOUBLE WORD BOUNDARY.	A 8970
	DC	BL1'00000000' FLAGS 1.	A 8980
	DC	X'00' FLAGS2.	A 8990
	DC	X'00' SENSE BYTE 1.	A 9000
	DC	X'00' SENSE BYTE 2.	A 9010
	DC	X'00' ECB CODE.	A 9020
	DC	AL3(ECB) ECB ADDRESS.	A 9030
	DS	OF FLAGS 3.	A 9040

CSW	DC	D'0' CHANNEL STATUS WORD.	A 9050
	DC	X'00' SIO CODE.	A 9060
	DC	AL3(CCWIN) CHANNEL PROGRAM ADDRESS.	A 9070
	DC	X'00'	A 9080
	DC	AL3(TAPE) DCB ADDRESS.	A 9090
	DC	X'00' REPOSITION MODIFIER.	A 9100
	DC	X'000000' RESTART ADDRESS.	A 9110
	DC	X'0000' BLOCK COUNT INCREMENT.	A 9120
	DC	X'0000' ERROR COUNTS.	A 9130
*			A 9140
*			A 9150
*		E V E N T C O N T R O L B L O C K.	A 9160
*			A 9170
ECB	DC	F'0'	A 9180
ECBA	DC	AL3(ECB)	A 9190
*			A 9200

*
*
*
*

TAPE

DCB

D A T A C O N T R O L B L O C K .

DDNAME=INPUTAPE,
MACRF=(E),
EODAD=EOFILE,
DSORG=PS,
IOBAD=IOBIN,
RECFM=U,
DEV D=TA

*

A 9220
A 9230
A 9240
A 9250
X A 9260
X A 9270
X A 9280
X A 9290
X A 9300
X A 9310
A 9320
A 9330

*
*
*
*

C H A N N E L P R O G R A M S.

```

DS      OD ALIGN ON DOUBLE WORD BOUNDARY.
CCWMODE DC      X'0300000140000001'
CCWIN   DC      X'0200000020007D00'
CCWREW  DC      X'0700000100000001'
CCWBSR  DC      X'2700000100000001'
CCWBSF  DC      X'2F00000100000001'
CCWFSR  DC      X'3700000100000001'
CCWFSF  DC      X'3F00000100000001'
CCWMODE2 DC     X'0300000140000001'
CCWOUT  DC      X'0100000020000000'
CCWTMK  DC      X'1F00000000000001'
CCWERG  DC      X'1700000000000001'
CCWMODEA DC     AL3(CCWMODE)
CCWMD2A DC     AL3(CCWMODE2)
CCWINA  DC     AL3(CCWIN)
CCWREWA DC     AL3(CCWREW)
CCWBSRA DC     AL3(CCWBSR)
CCWBSFA DC     AL3(CCWBSF)
CCWFSRA DC     AL3(CCWFSR)
CCWFSFA DC     AL3(CCWFSF)
CCWOUTA DC     AL3(CCWOUT)
CCWTMKA DC     AL3(CCWTMK)
CCWERGA DC     AL3(CCWERG)
CCWCNT  DC     H'32000'

```

*

END

A 9350
A 9360
A 9370
A 9380
A 9390
A 9400
A 9410
A 9420
A 9430
A 9440
A 9450
A 9460
A 9470
A 9480
A 9490
A 9500
A 9510
A 9520
A 9530
A 9540
A 9550
A 9560
A 9570
A 9580
A 9590
A 9600
A 9610
A 9620
A 9630
A 9640

C	SUBROUTINE MESOUT(CODE, IOB, DCB, ECB, B, DEB, UCB)	B	0
C	INTEGER*2 IOB(20), DCB(28), DEB(18), UCB(22)	B	10
C	INTEGER ECB, B(30), CODE	B	20
C	INTEGER IOPRT/3/, MAXOUT/5/, ERROUT/0/	B	30
C		B	40
C	GO TO (100, 200, 300, 400, 500, 600, 700, 800, 900, 1000), CODE	B	50
C		B	60
C	C-----END-OF-FILE ENCOUNTERED.	B	70
C	100 WRITE(IOPRT,101) B(8), B(7)	B	80
	101 FORMAT('OEND OF FILE', I7, ' ENCOUNTERED. THERE WERE',	B	90
	1 I8, ' RECORDS IN THIS FILE.', /)	B	100
	120 RETURN	B	110
C		B	120
C	C-----UNCORRECTABLE PARITY ERROR ON INPUT.	B	130
C	200 WRITE(IOPRT,201) B(10)	B	140
	201 FORMAT('OPARITY ERROR DETECTED ON READ AFTER', I5, ' ATTEMPTS.')	B	150
	202 WRITE(IOPRT,203) B(2), B(1), B(9), IOB(5), IOB(6), IOB(7), IOB(8),	B	160
	1 UCB(12), UCB(13), UCB(14)	B	170
	203 FORMAT(' FILE', I5, ' RECORD', I6, ' LENGTH=',	B	180
	1 I6, ' BYTES. CSW=', Z4, 'X, Z4, ' SENSE BYTES=', Z4, /)	B	190
	GO TO 120	B	200
C		B	210
C	C-----ERROR DETECTED...NOTE STATUS BYTE.	B	220
C	300 WRITE(IOPRT,301)	B	230
	301 FORMAT('OERROR DETECTED..NO CORRECTION ATTEMPTED..NOTE STATUS BYTE	B	240
	1 IN CHANNEL STATUS WORD (CSW).')	B	250
	GO TO 202	B	260
C		B	270
C	C-----WARNING...READ ERROR THAT WAS CORRECTED.	B	280
C		B	290
		B	300
		B	310
		B	320
		B	330
		B	340
		B	350
		B	360

400 NREAD=B(10)-B(15)+1	B	370
WRITE(IOPRT,401) NREAD, R(2), B(1)	B	380
401 FORMAT('OWARNING..IT TOOK', I5, ' READ ATTEMPTS TO INPUT THIS REC	B	390
ORD CORRECTLY. FILE', I5, ', RECORD', I6, 1H., /)	B	400
WRITE(IOPRT,420)	B	410
420 FORMAT(1H , 130(1H-))	B	420
GO TO 120	B	430
C	B	440
C-----EXCESSIVE READ ERRORS...TERMINATE PROGRAM.	B	450
C	B	460
500 WRITE(IOPRT,501)	B	470
501 FORMAT('OJOB ABORTED BY GETBLK DUE TO EXCESSIVE READ ERRORS.')	B	480
WRITE(IOPRT,502)	B	490
502 FORMAT('OCHECK TRTCH AND DEN PARAMETERS ON DD CARD IF SEVEN TRACK	B	500
1 TAPE.')	B	510
503 CALL EXIT	B	520
C	B	530
C-----UNABLE TO OPEN FILE.....TERMINATE JOB.	B	540
C	B	550
600 WRITE(IOPRT,601)	B	560
601 FORMAT('OJOB ABORTED BY GETBLK BECAUSE UNABLE TO OPEN TAPE FILE.	B	570
1 POSSIBLE MISSING OR MISPELLED DD CARD.')	B	580
GO TO 503	B	590
C	B	600
C-----PUTBLK WAS CALLED WITH L<=0.	B	610
C	B	620
700 WRITE(IOPRT,701) B(9)	B	630
701 FORMAT('OPUTBLK WAS CALLED WITH L=', I12, /)	B	640
702 ERROUT=ERROUT+1	B	650
IF(ERROUT .LT. MAXOUT) GO TO 120	B	660
703 WRITE(IOPRT,704)	B	670
704 FORMAT('OJOB ABORTED BY PUTBLK DUE TO EXCESSIVE ERRORS.', /)	B	680
GO TO 503	B	690
C	B	700
C-----PERMANENT WRITE ERROR PUTBLK.	B	710
C	B	720
800 IREC=B(4)-1	B	730

WRITE(IOPRT,801) B(5), IREC	B	740
801 FORMAT('OPERMANENT WRITE ERROR (PUTBLK) , FILE', I5, ', RECORD',	B	750
1 I7, 1H., /)	B	760
GO TO 702	B	770
C	B	780
C-----ATTEMPT TO CHANGE DDNAME WHEN FILE STILL OPEN (SETDDN).	B	790
C	B	800
900 WRITE(IOPRT,901)	B	810
901 FORMAT('OATTEMPT TO CHANGE DDNAME BY SETDDN WHEN FILE IS STILL OPE	B	820
IN.')	B	830
GO TO 503	B	840
C	B	850
C-----USER ISSUED A CALL GETBLK WITH A NEGATIVE FILE OR BLOCK PARAMETE	B	860
C	B	870
1000 WRITE(IOPRT,1001)	B	880
1001 FORMAT('OGETBLK WAS INVOKED WITH INVALID PARAMETERS. RECORD REQUE	B	890
1STED=', I12, ', FILE REQUESTED=', I12, /)	B	900
WRITE(IOPRT,1002)	B	910
1002 FORMAT('OJOB TERMINATED.')	B	920
GO TO 503	B	930
END	B	940