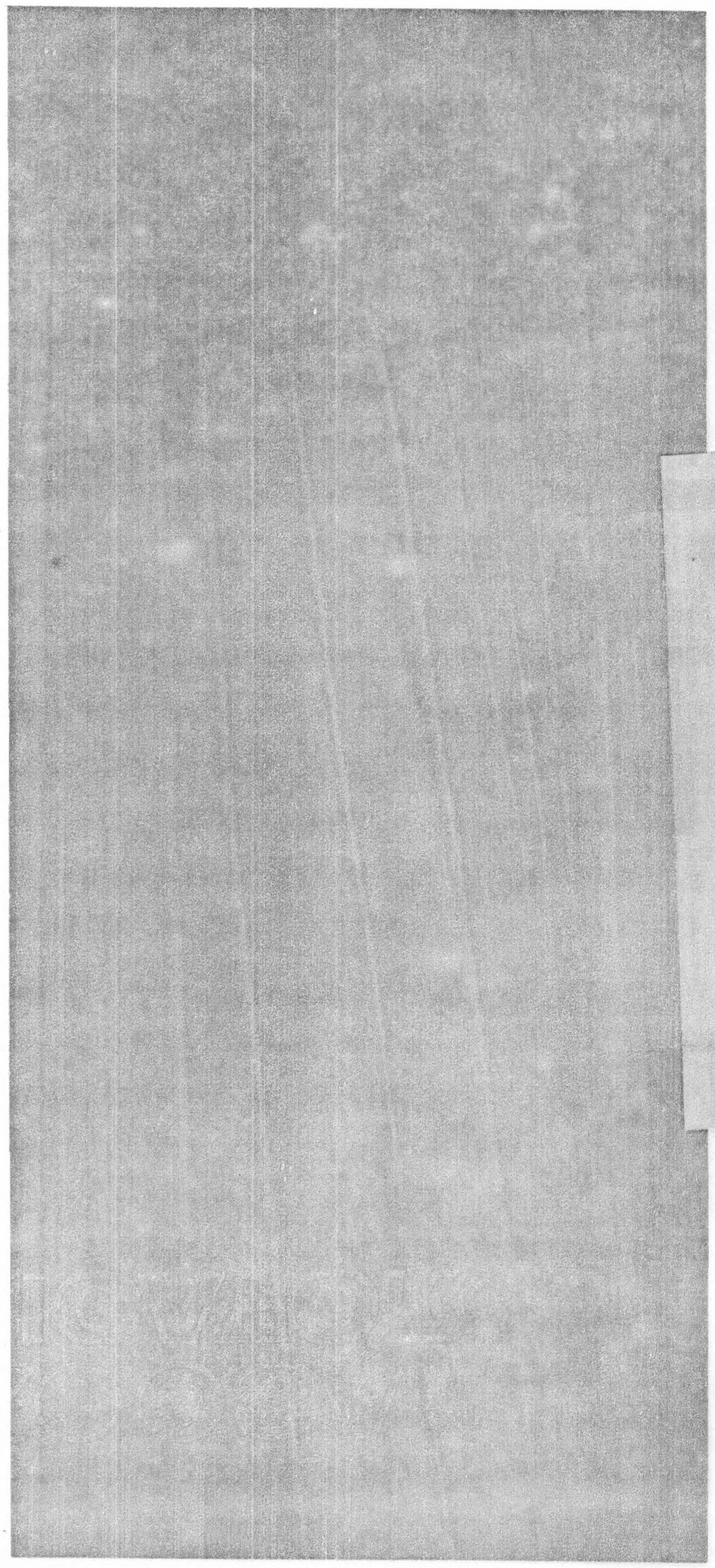
Honeywell

TSS TERMINAL/BATCH INTERFACE

SERIES 60 (LEVEL 66)/6000

GCOS

SOFTWARE



Honeywell

TSS TERMINAL/BATCH INTERFACE

SERIES 60 (LEVEL 66)/6000

GCOS

SUBJECT:

Description and Use of Time Sharing Subsystems and Batch-Type Programs.

SPECIAL INSTRUCTIONS:

This manual replaces Time Sharing System Terminal/Batch Interface Facility, Order No. BR99 for Series 6000 System users. Order No. BR99 must be used by Series 600 System users and by Series 6000 System users who are on prior software releases.

SOFTWARE SUPPORTED:

Series 60 Level 66 Software Release 1
Series 6000 Software Release G

DATE:

April 1974

ORDER NUMBER:

DD21, Rev. 0

PREFACE

This manual describes a group of special purpose subsystems and batch-type programs that are available to the users of Honeywell Series 6000 and Series 60 Level 66 Time Sharing Systems. The Series 60 Level 66 is hereafter referred to as the Series 60. The information in this manual refers to both the Series 6000 and the Series 60, unless otherwise specifically stated. Included in this manual are subsystem usage procedures, terminal operating procedures, and definitions of Time Sharing System error messages.

The following software documents are referenced in this manual:

Series 60 (Level 66)/6000 General Comprehensive Operating Supervisor (Order No. DD19)
Series 60 (Level 66)/6000 General Loader (Order No. DD10)
Series 60 (Level 66)/6000 File and Record Control (Order No. DD07)

File No,: 1713,1P13

DD21

CONTENTS

		Page
Section I	Terminal/Batch Interface Description.	1-1
Section II	CARDIN Subsystem.	2-1
	CARDIN Function.	2-1
	CARDIN Operation	2-1
	CARDIN Detailed Operation.	2-3
	CARDIN Command Language	2-3
	RUN or PRINT Question/Answer Sequence	2-7
	ASCBCD Question/Answer Sequence	2-12
	Tab Characters and Settings? Response. . . .	2-13
	BCDASC Question/Answer Sequence	2-14
	Tab Character and Settings? Responses. . . .	2-14
	BPUNCH or BPRINT Question/Answer Sequence . . .	2-14
	BPUNCH/BPRINT Output to Remote	2-14.1
	BPRINT/BPUNCH Multiple Copy Feature.	2-14.1
	COPY(nn) (C)	2-14.1
	APRINT Question/Answer Sequence	2-14.2
	Reformatting Information in the File.	2-15
	$ SELECTA Card Option	2-16
Section III	SCAN Subsystem.	3-1
	SCAN Function.	3-1
	SCAN Operation Description	3-2
	SCAN Verbs	3-3
Section IV	FDUMP Subsystem	4-1
	FDUMP Function	4-1
	FDUMP Operational Description.	4-1
	Short Form Usage of FDUMP Functions	4-3.1
	FDUMP Error Indication and Replies.	4-3.1
	FDUMP Copy Function Details	4-4
Section V	Conversational Debug.	5-1
	RBUG General Description	5-1
	RBUG Instructions.	5-2
Section VI	Conversational File and Record Input/Output	6-1
	Control Cards for File and Record Control I/O. . .	6-1
	Applicable I/O Calls	6-2
	Special File and Record Control I/O Considerations.	6-3
Section VII	JOUT Subsystem.	7-1
	JOUT Function.	7-1
	JOUT Operational Description	7-2
	Short Form Usage of JOUT Functions.	7-5
Section VIII	CONVERT Subsystem	8-1
	General.	8-1
	Functions.	8-1
	Commands	8-2
	Command Syntax	8-3
	Input Files Field	8-4

	Page
Output Files Field.	8-4
Options Field	8-4
Line Continuation	8-4
Control Characters and Blanks	8-5
Options.	8-5
Media Code Options (Default is ASCII)	8-5
Line Number Options (Default Is ASIS)	8-5
Character Manipulation Options (There Is No Default Option)	8-6
Miscellaneous Options (There Is No Default Option).	8-6
File Processing Option (Default Is Include for JRN)	8-6
File Processing Option (Default Is Exclude For All Other Commands).	8-6
Specialized Options (There Is No Default Option).	8-6
Media Code Options.	8-7
Line Number Options	8-7
Character Manipulation Options.	8-9
Miscellaneous Options	8-10
File Processing Options	8-10
Specialized Options	8-11
Parenthetical Line/Sequence Numbers.	8-12
Methods of Specifying Options.	8-13
Operations Involving PRINT and APRINT Records. . .	8-15
Conversion from APRINT (Media Code 7)	8-15
Conversion to APRINT (Media Code 7)	8-15
Conversion From PRINT (Media Code 3).	8-16
Conversion To PRINT (Media Code 3).	8-16
Responses to Errors.	8-17
Special Features of the JPRINT Command	8-18
Special Features of the JRN Command.	8-18
Special Control Characters Recognized by the ASCII Printer	8-18
Horizontal Tabulate Control Character	8-19
Horizontal Column Skip Control Character. . . .	8-19
Vertical Line Slew Control Character.	8-19
Slew to VFU Configuration Control Character . .	8-19
Shift Out Control Character	8-19
Shift In Control Character.	8-20
Shift Out/Shift In Combination.	8-20
Delete Character.	8-20
Space Character	8-20
Data and Control Characters, Bit Structure. . .	8-20
Examples of CONVERT Use.	8-21
Appendix A Interface Usage Examples.	A-1
Index	i-1

SECTION I

TERMINAL/BATCH INTERFACE DESCRIPTION

The Terminal/Batch Interface Facility of the Time Sharing System consists of an interrelated set of time sharing subsystems and batch programming facilities. These combine to provide the time sharing terminal user with full batch programming capabilities. This facility allows the user to submit a batch program, debug the program, and inspect the program output; all from a remote terminal. This interface also facilitates direct, interactive communication between the user's batch program and his (or another) terminal.

The elements making up this facility fall into four categories:

1. The CARDIN time sharing subsystem, which allows the actual submission of batch jobs to the System Input program and integrates the other elements into a functional whole.

2. Supporting subsystems, SCAN, JOUT, and FDUMP, which allow batch output scanning, batch output manipulation, and file dumping and modification (in octal), respectively. (The latter is primarily a debugging tool.)

3. Two batch dimension features, Conversational Debug Routine (RBUG), and Conversational-I/O extensions to the File and Record Control logical record processing routines, are for inclusion in the batch program itself. These allow direct access connection between the program and the remote terminal, where desired.

4. The CONVERT subsystem, which permits conversion of text information from any text file format to any text file format.

   NOTE: The CONVERT subsystem employs commands that duplicate commands now used with CARDIN. A future software release will remove these commands from the Time Sharing System. Files that make use of CONVERT commands must have characteristics as specified in Section VIII.

SECTION II

CARDIN SUBSYSTEM

CARDIN FUNCTION

CARDIN is the component subsystem of the Time Sharing System, that makes available to the time sharing user the wider facilities of batch processing. It permits a remote terminal user to perform such functions as developing and checking out a batch program written in macro assembler (GMAP) language. CARDIN also provides a convenient means of maintaining and initiating application programs such as APT, Linear Programming, and data reduction programs. Almost any other situation where a time sharing terminal user needs access to batch facilities can be accommodated by CARDIN.

Essentially, CARDIN allows the user to create at his terminal a card image input file, submit it for batch processing, and receive notice of the result (successful termination, abort, etc.). The user then may have the job output printed at the central site or a remote batch facility, or inspect the output file conversationally by selective listing at the terminal. (Refer to SCAN Subsystem, Section III.)

The TALK option of CARDIN allows a terminal user to talk to a program running in the central computer system. This option places the user's terminal in direct access, conversational connection with any program that was submitted through CARDIN and that contains remote terminal input/output. The TALK option facilitates the use of the Conversational Debug (RBUG) routine by allowing interactive, execution time debugging from the user's terminal.

Permission from the master user must exist in the Time Sharing System before a specific user can employ CARDIN. If master user's permission has been granted for use of TALK disposition, use of all CARDIN functions is implied.

CARDIN OPERATION

As in Time Sharing BASIC, the file-building facility provided by CARDIN requires line numbers. Also, as in BASIC, the subsystem edits keyboard input and updates the current file. The line numbers can be either stripped automatically from each line or moved to the card image sequence number field at the time the job input file is passed to the batch environment.

In converting the user's time sharing source ASCII file to a BCD card image job input file, one line of terminal input represents one card. The user must include in his source file all cards that would normally be required in the batch environment except the $ SNUMB, $ USERID, and ***EOF cards. A $ USERID card is generated by CARDIN when required for a permanent file, so that the user need not reveal his log-on password. This card does not show on a LIST request for the file. However, the user may supply his own $ USERID card if he wishes, and the System Input program uses this rather than the one supplied by CARDIN.

Tab characters may be used when building the file in order to avoid keying in the blanks required by a fixed-field language such as GMAP. The tab character can be assigned a sequence of character-position values--for example, tab,8,16,32,73. Each successive occurrence of the tab character within a line causes the next unused value in the sequence to be assigned to it. More than one tab character may be specified, each with its own sequence of tab settings.

ASCII-to-BCD conversion and reformatting of a file is obtained at the following times:

1. When the user gives a RUN command, causing conversion onto a scratch file and passage of the file to the System Input program for processing.

2. When the user gives an ASCBCD command, causing conversion onto a permanent file only.

In neither case is the original ASCII file itself affected.

The information needed by CARDIN for converting and reformatting a file may be supplied by the user in either of two ways:

- The tab characters, settings, and line number disposition may be supplied in response to questions asked by CARDIN.

- To minimize the user/subsystem interface, the same information may be supplied on the first line of the file itself.

The user may also convert a system format Binary Coded Decimal (BCD) file to a time sharing ASCII file by means of the BCDASC command.

The general operation of the subsystem is as follows:

1. The terminal user builds his file under CARDIN or calls for an OLD file previously prepared under CARDIN or other time sharing subsystem. If line numbered, the file can then be corrected, updated, resequenced, etc., using standard time sharing commands. The user can request that his current file be run, using RUN or RUN filedescr.

2. When the user requests a run of his current file, he is given the SNUMB sequence number of his batch run. He is then asked a series of questions concerning the format of his file and the desired disposition of the job, if this information did not appear on the first line of the file.

3. When the question/answer sequence is completed, the user's time sharing source file (which is in ASCII) is converted to BCD and put in the desired card format. It is then passed to the System Input program for processing.

4. Following a batch-job termination message, the user may inspect his output by giving the command SCAN filename, where filename is the name of a permanent file upon which the batch-job output has been placed. This placement is achieved by appropriate use of the $ PRMFL control card.

   The output scanning facilities of the SCAN subsystem are available through use of the SCAN command.

5. Instead of having the file converted and processed, the user can have it converted and then printed out at the terminal in the requested job input format by means of the PRINT command. This facilitates visual inspection before actually passing the file to the System Input program (with a subsequent RUN command). The PRINT command initiates a question/answer sequence similar to that for the RUN command.

CARDIN DETAILED OPERATION

CARDIN Comand Language

The CARDIN commands which follow are not all unique to CARDIN, but some have special meaning.

- RUN

  1. RUN

     Convert and pass the current file to batch processing, following the question/answer sequence that is initiated by this command if the file does not contain first line reformatting information. The SNUMB of the job is printed.

  2. RUN filedescr

     Convert and pass the file saved under filedescr to the batch world for processing, following the question/answer sequence that is initiated by this command if the file does not contain first line reformatting information. The current file, if one exists, is not disturbed. Filedescr is simply the file name for a quick access permanent file; for other permanent files it is the file description. The SNUMB of the job is printed.

- PRINT

  1. PRINT

     Reformat the contents of the current file and print it at the terminal, following the question/answer sequence that is initiated by the command if the file does not contain first line reformatting information.

  2. PRINT filedescr 1; filedescr 2;...;filedescr n

     Reformat the contents of the file saved under filedescr and print it at the terminal, following the question/answer sequence that is initiated by this command if the file does not contain first line reformatting information. The current file, if one exists, is not disturbed. (See definition of filedescr under RUN command.)

  3. PRINT n1,n2,n3,..., n

     Print only the line or lines specified by n to the terminal from the current file. This option does not function if ASIS or STRIP is in effect.

  4. PRINT a-b

     Print lines a through b of the current file. This option does not function if ASIS or STRIP is in effect.

  5. PRINT a-b,c,d-e

     Print lines a through b,c, and d through e of the current file. This option does not function if ASIS or STRIP is in effect.

  6. PRINT filedescr (operand list)

     Where the operand list can be any of the forms listed above: 3,4,or 5.

- JDAC

  JDAC name

  Allows time sharing terminal user to establish connection with a Direct Access (DAC) program, where name is the program's remote inquiry name already supplied to GCOS via a MME GEROUT. If name is not supplied with the JDAC request, the system returns a NAME? query.

  When the direct access program terminates, the system returns to the command entry (build mode) level within CARDIN.

- JSTS

  JSTS snumb (Job Status)

  Causes the current batch processing status of the job specified by snumb to be printed at the terminal in plain text.

With an OUTPUT WAITING or an OUTPUT COMPLETE status for the requested snumb, JSTS gives the termination status of the last job that reached termination. For example:

```
*JSTS 0003T

0003T OUTPUT WAITING
```

or

```
*JSTS 0003T

0003T OUTPUT COMPLETE
```

Note that when more than one job has been initiated, the termination status given may not be that of the requested snumb. The special designator "*" can be used to indicate that the desired snumb is to be that of the last job submitted.

- JABT

JABT snumb (Job Abort)

Causes the batch job specified by snumb to be aborted, with an abort code assigned. Only jobs containing a valid $ USERID control card for the requesting user will be aborted. The special designator "*" can be used to indicate that the desired snumb is to be that of the last job submitted.

- ASCBCD

ASCBCD ascfil;bcdfil (ASCII-to-BCD)

```
ASCBCD
INPUT FILES ascfil;bcdfil
```

If the response to "INPUT FILES" is null (carriage return), control is returned to the level from which the command was issued. The ASCII time sharing file specified by ascfil is converted to a standard system format BCD file on the permanent file specified by bcdfil. This follows the question/answer sequence that is initiated by this command if the ASCII file does not contain first line reformatting information. Both ascfil and bcdfil may be simply a file name or a full file description, as required. (See definition of filedesc under RUN command.) The ascfil field may specify the current file by an asterisk.

- ASCASC filedescr 1; filedescr 2

This command, issued under FORTRAN and other time sharing language systems, causes the translation of a time sharing format ASCII file to a standard system format ASCII file or vice versa. In both translations, file 1 is converted to the format required in file 2. If file 1 is in time sharing ASCII format (record media code 5), the file is read and converted to the word-oriented standard system ASCII format for file 2. File 2 may then be used as input data for the language system. If file 1 is in standard system ASCII format (record media code 6), the file is read and converted to the character-oriented time sharing ASCII format for file 2.

- APRINT

  APRINT filename

  The APRINT subsystem is designed to read either a standard time sharing ASCII file or a file that has been formatted through the RUNOFF/REFORM process; build a file of print-line images with the appropriate slew controls for the ASCII printer, and pass the print-line file to SYSOUT through the "backdoor" feature for printing on a high-speed printer.

- BCDASC

  BCDASC bcdfil;ascfil (BCD-to-ASCII)

  The standard system format BCD permanent file specified by bcdfil is converted to an ASCII time sharing file on the permanent file specified by ascfil. This follows the question/answer sequence that is initiated by this command. Both bcdfil and ascfil may be simply a file name or a full file description, as required. (See definition of filedescr under RUN Command.) The ascfil field may specify the current file by an asterisk.

- BPUNCH and BPRINT

  1. BPUNCH ascfil (Batch Punch)

     BPRINT ascfil (Batch Print)

     The contents of the ASCII time sharing file specified by ascfil is converted to BCD and is punched or printed, respectively, at the central computer site or a remote site. This follows the question/answer sequence that is initiated by these commands, if the file does not contain first line reformatting information. These commands allow the user to create card backup for his time sharing files or to list long files on a high speed printer. The ascfil field may be simply a file name or a full file description, as required. (See definition of filedescr under RUN Command.) The ascfil field may specify the current file by an asterisk.

     NOTE: The label disposition and tab settings are either punched on cards (BPUNCH) or printed (BPRINT) as $ NOTE cards.

     Since a batch dimension bulk media conversion (BMC) job is implied by these commands, batch $ IDENT card information is requested.

  2. BPUNCH F1;*;F2;F3; ....;Fn

     BPRINT F1;*;F2;F3;.....;Fn

     Concatenates the named files (* is used to denote the current file) and punches or prints them as one file. For maximum efficiency all the files should contain a first-line format record.

- FDUMP

  Permits inspection and maintenance of permanent files at remote terminals. Refer to the description of the FDUMP subsystem in Section IV.

- SCAN

  Allows the scanning of batch output stored on a permanent file. Refer to the description of the SCAN subsystem in Section III.

- JOUT

  JOUT snumb

  Permits the inspection from a time sharing terminal of the output of certain types of batch jobs. The special designator "*" can be used to indicate that the desired snumb is to be that of the last job submitted. Refer to the description of the JOUT subsystem in Section VII.

RUN or PRINT Question/Answer Sequence

If the file to be converted does not contain first line reformatting information immediately following the RUN or PRINT command, a fixed sequence of questions is asked by the system concerning the format and desired disposition of the file.

Response to CARD FORMAT,.....?

The user response to the first part of the question:

CARD FORMAT,DISPOSITION?

may be ASIS, STRIP, MOVE, or NORM. This response may be followed by a comma and the disposition(s) desired (WAIT, TALK, URGC, JOUT, or ROUT) or just a carriage return. The short form defined in parentheses may also be used.

ASIS (A)

The user wants the file passed to batch just as it exists. In this case, each line of terminal input is converted to BCD and put in card format. The content of the first character position of the terminal line goes in the BCD record, unless it is a tab character. Use of ASIS implies either that the ASCII file does not contain initial line numbers or that the user desires them to appear in the BCD output file.

STRIP (S)

This specifies that the user has employed line numbers when preparing his file and wants the line number stripped from each line before converting to the BCD record. When this option is used, the first nonnumeric character of a terminal line, if not a tab, goes in column 1 of the BCD record. (Line numbers in the source file are unaffected.)

MOVE (M)

This option states that the user has employed line numbers when preparing his file and wants the line numbers moved to the sequence field (columns 73-80) of the card. When this option is used, the first nonnumeric character of a terminal line, if not a tab, goes in column 1 of the card.

NORM (N)

This option implies the MOVE option and specifies that the normal tab character (the colon) and tab settings (8,16,32,73) have been employed in building the file. Following a response of NORM, the TAB CHARACTER AND SETTINGS? query is omitted.

The NORM format may also be used to define a character other than the colon to be used as a tab and may permit multicard lines by means of a character used to indicate the end of a card. These options are indicated in a parentheses enclosed field following NORM. Either character may be omitted, as shown in the examples below.

Examples:

NORM(;,/)
The semicolon is the tab character and end-of-card is indicated by /. That is, a line such as the following has been employed in the file:

70;LDA;A/;STA;B/;TRA;C

NORM(,;)
The colon is the tab character by default, and the semicolon is the end-of-card character.

NORM(;)
The semicolon is the tab character.

A line number is from one to eight numeric characters immediately followed by a nonnumeric character (including blank), the numerics being the first nonblank characters in the line:

ϕ...ϕnnnnnnnnxcc.....c

Where: ϕ...ϕ - Optional initial blank.

nnnnnnnn - Numeric characters.

x - Nonnumeric character.

cc.....c - Any characters.

For STRIP, MOVE, and NORM, if the user wants a numeric in column 1 of the card image and line numbers exist in the source file, a pound sign (#) character immediately following the line number causes the character following it to go into column 1.

For example, 220#123 results, upon a MOVE conversion, in the card numbered 220 having 123 in columns 1, 2, and 3. If the user desires that a pound sign go into column 1 of the card image, two pound signs following the line number achieves that effect. For example, 220## results, upon a MOVE conversion, in the card numbered 220 having a pound sign in column 1.

Response to the DISPOSITION Portion

Following a response to the CARD FORMAT portion of this question, the user answers the DISPOSITION portion with one of the following entries (note that each is preceded by a comma).

...,WAIT	-- Wait for job termination.
...,TALK	-- Enter conversational mode. (Use may be restricted by master user. See discussion of TALK.)
...,URGC(xx)	-- Assign specified initial urgency to this job.
...,JOUT	-- Save all implied files for examination by JOUT subsystem.
...,ROUT(xx)	-- Direct output to station xx. Only one ROUT entry is permitted.

If, instead of entering a reply to DISPOSITION, the user follows his card format entry with a carriage return, the job is initiated, and the system returns to build mode for further CARDIN command entries.

NOTE: If an error occurs in the DISPOSITION portion of the response, the user should respond to the CARD FORMAT, DISPOSITION? question that then reappears with the desired format, a correction of the disposition error, and any disposition entry that followed the one in error.

WAIT (W)

This response indicates that the user desires to wait at this point for a status indication sent upon completion of the job in the batch environment. Normal termination is indicated by the message

NORMAL TERMINATION

Any abnormal termination is indicated by an appropriate plain text message. (See abort/delete messages in the General Comprehensive Operating Supervisor reference manual.)

Following the termination message, the system returns to build mode for further CARDIN command entries.

TALK (T)

This response implies that the batch job includes execution of a program containing conversational (direct access) input/output -- either user-implemented direct access I/O or, for example, the RBUG routine (see Section V). Following the actual submission of the job by CARDIN, the user's terminal is placed in direct access connection with the submitted program (by SNUMB). When the job terminates, the user receives a normal or abnormal termination message and returns to build mode, as with WAIT.

Permission from the master user must exist in the Time Sharing System before a specific user can employ the TALK disposition. (Granting of permission to use TALK implies permission to use all CARDIN functions.)

URGC(xx) (U)

This response may be used only when the RUN command has been previously entered. The response indicates that the user wants to assign initial urgency xx to the job. The assigned urgency must not be greater than the maximum allowed, or the system will reply with the message ILLEGAL URGENCY after the next carriage return and follow it with the CARD FORMAT, DISPOSITION? question.

If xx is not specified, maximum allowable urgency is automatically assigned.

This disposition response may be specified by itself or along with JOUT and/or WAIT or TALK.

JOUT (J)

This response may be used only when the RUN command has been previously entered. The response requests that all implied files be saved so that they may be examined using the JOUT subsystem.

This disposition response may be specified by itself or along with URGC and/or WAIT or TALK.

ROUT(xx) (R)

This response may be used only when the RUN command has been previously entered. The response indicates that the user wishes the implied files generated by the program execution to be directed to the specified two character remote station. This response to DISPOSITION may be specified by itself or along with any other disposition. Only one ROUT entry permitted.

Following are examples of responses to the CARD FORMAT, DISPOSITION? question. Note these points:

1. A response must be entered for CARD FORMAT, and this response must be the first item.

2. The responses may be represented by their initial letters.

3. The DISPOSITION responses may be in any order.

4. The responses are separated by commas, with no embedded blanks.

Examples:

```
MOVE
ROUT(BC)
MOVE,TALK
R(AD)                    (same as as ROUT(AD))
M,T                      (same as MOVE,TALK)
NORM(;,/),W              (W = WAIT)
N(;,/),URGC(35),J        (N = NORM, J = JOUT)
N,U(35),J,T              (same as NORM, URGC(35),JOUT,TALK)
A,J                      (same as ASIS,JOUT)
S,U,JOUT                 (same as STRIP,URGC,JOUT)
A,R(DD)                  (same as ASIS,ROUT(DD))
```

Response to TAB CHARACTERS AND SETTINGS?

This question is asked only if NORM has not been specified in response to CARD FORMAT, DISPOSITION?

When the file is being prepared, the user can employ tab characters to format his input instead of the required blanks. Any character except the pound sign (#) can be used to indicate a tab, or the regular tab key can be used. Delimiters used for tab should be unique. When responding to the above question, the user may specify a tab character and up to 20 tab settings. All fields are separated by commas. For example, if the user employed an ampersand (&) as a tab character and desired successive tab values of 8,16, and 40 in his card-image input file, he would respond with &,8,16,40.

If the user employed the standard character and settings (:,8,16,32,73), he may respond with NORM or N.

If the user did not use any tabs in preparing his input, he responds to the above question with a carriage return.

The user may employ multiple tab characters (tab entries separated by a semicolon) in his file--one for language statements and another for control cards, for example.

&,8,16;+,12,32,40

Where: the tab characters are & and +.

A maximum of seven different tab characters and a total of 35 individual tab settings can be specified within the limits of one 80-character input line of the file.

ASCBCD Question/Answer Sequence

The questions LABELS? and, conditionally, TAB CHARACTERS AND SETTINGS? are asked when the ASCBCD command is given, if the file ascfil does not contain first line reformatting information. The responses are for the most part identical with those given in the RUN and PRINT question/answer sequence. Where the responses are identical, the explanation of them is not repeated, as the meaning is the same.

The user can give both the label and tab responses when the LABEL? question is asked. A blank delimiter is used to separate the format directive from the tab character and setting information, thus corresponding to the structure of a first-line reformatting record.

The absolute maximum size of an ASCII record that can be processed by ASCBCD is 1272 data characters. The resulting BCD records may be as large as 1908 data characters if the tabbing feature is used to provide the additional expansion.

NOTE: Any response to either the LABEL? or the TAB CHARACTER AND SETTING? questions that implies a standard format (columns 8,16,32,73 or the use of columns 73-80) results in the output being 80 character card image records regardless of the length of the ASCII input record.

Response to LABELS?

- MOVE
- STRIP
- ASIS
- NORM
- abcde$(i,j)_1$; abcde$(i,j)_2$; ...; abcde$(i,j)_n$

  This prefix option implies that the line numbers in the file are to be moved to the BCD label field (columns 73-80) and that the alphanumeric prefix abcde$_i$ is to be inserted -- left justified -- in the label field as specified by the line numbers $(i,j)_j$.

abcde represents any alphanumeric prefix, to a maximum of five characters

i represents the initial line number

j represents the final line number to which abcde is to be prefixed

The (i,j) field is mandatory; the conventions on its use are as follows:

(i,j) = (i,j)

(,j) = (0,j)

(i,) = (i,99...9)

Multiple sets of prefix specifications may be given, semicolon separated, as shown in the general form above. If an interval of line numbers is found in the file that has not been specified by the user, only the line number will appear in the label field.

The prefix will overlay leading significant digits of the line number, if the line number becomes too large to be accommodated.

TAB CHARACTERS AND SETTINGS? RESPONSE

If the response to LABELS? was not NORM, the question TAB CHARACTERS AND SETTINGS? is asked. The possible responses are:

- carriage return (null response) -- Indicates that no tab characters are employed in the file.
- $\underline{t},\underline{s}_1,\underline{s}_2,..,\underline{s}_n$ -- Indicates a single set of tab character ($\underline{t}$) and settings ($\underline{s}_n$) have been used, as in the RUN or PRINT responses. The value of $\underline{s}_n$ must be less than 1273.
- $\underline{\text{tab-set}}_1;\underline{\text{tab-set}}_2;...;\underline{\text{tab-set}}_n$ -- Indicates that multiple sets of tab characters and settings have been used in the file -- as in the RUN or PRINT responses--where tab-set has the same form as the single-set specification shown above.

A maximum of seven different tab characters and a total of 35 individual tab settings can be specified within the limits of one 1272-character input line of the file.

BCDASC Question/Answer Sequence

The questions LINE NUMBERS? and TAB CHARACTER AND SETTINGS? are asked when the BCDASC command is given. The first question requests information concerning what line numbers, if any, are desired in the converted ASCII file (ascfil). The second question requests information concerning a single tab character. The user may give both the line number and tab responses when the LINE NUMBERS question is asked. A blank delimiter is used to separate the line number directive and tab character and setting information, thus corresponding to data portion structure of the CARDIN first-line reformatting record.

The responses to LINE NUMBERS are:

- carriage return (null response) -- Do not create line numbers for the ASCII file.
- MOVE -- Use the sequence-number digits of the label field in the BCD file as numbers for the ASCII file. Implies the BCD file is 80 column card images. Characters exceeding 80 will be ignored.
- NORM -- Uses the sequence number as in the MOVE option and also a colon(:) is used for the tab character and the settings are 8, 16, 32 and 73.
- AUTO -- Create line numbers for the ASCII file, starting with 0010 and incrementing by 10.
- AUTO n,m -- Create line numbers for the ASCII file, starting with n and incrementing by m. If either n or m is null (AUTO n, or AUTO, m), the appropriate value, as shown under AUTO above, is applied.
- ASIS,n -- Convert from BCD to ASCII n columns of the BCD record.

TAB CHARACTER AND SETTINGS? RESPONSES

The response is similar to the response to the same question under ASCBCD. However, only one tab character and its settings may be entered. Blanks in the ASCII file are eliminated and the tab character inserted as directed in the response. Thus, the created ASCII file appears as though it had been entered from a terminal with the use of tab characters.

BPUNCH or BPRINT Question/Answer Sequence

Both the BPUNCH and BPRINT commands ask the same series of questions:

$ IDENT?

LABELS?

TAB CHARACTERS AND SETTINGS?

The responses to LABELS? and TAB CHARACTERS AND SETTINGS? are the same as those for the ASCBCD command. The response to $ IDENT is identical in form and content to the variable field information that is required on the $ IDENT control card for a comparable batch job. The resulting BMC job utilizes SYSOUT to produce the listing of punched cards unless the output file size is greater than 300 llinks. When this size limitation is exceeded, a dedicated device is requested with the appropriate $ PRINT or $ PUNCH control card.

NOTE: BPRINT maximum record size is 132 print characters plus two slew control characters.

BPRINT/BPUNCH OUTPUT TO REMOTE

The ROUT disposition code used by CARDIN users to direct output to a remote station is also functional for BPRINT/BPUNCH users.

A complete description of the ROUT disposition can be found in the CARDIN-RUN or PRINT question/answer section.

NOTE: When files are concatenated for a BPRINT or BPUNCH job, the ROUT disposition on the <u>first</u> file determines whether the output is directed to a remote station or processed at the central site.

BPRINT/BPUNCH MULTIPLE COPY FEATURE

BPRINT/BPUNCH users may produce up to 13 copies of a listing or punched deck from a single BPRINT or BPUNCH command through the use of the COPY(nn) or C(nn) disposition code.

COPY(nn) (C)

This response will cause the generation of nn multiple copies of the listing or punched deck. The maximum number of copies that can be produced from a single BPRINT/BPUNCH job is 13. This response may be specified by itself or along with other disposition codes.

The same rules that govern the use of the ROUT disposition apply to the COPY option.

NOTE: When files are concatenated for a BPRINT or BPUNCH job, the COPY disposition on the <u>first</u> file determines the number of copies that will be generated.

APRINT Question/Answer Sequence

APRINT is invoked by typing:

APRINT filename

Where: filename is either a standard time sharing ASCII file or the output file specified with a REFORM command.

APRINT then asks the question:

$ IDENT?

to which the user responds with the same information as is entered in a batch job $ IDENT card.

APRINT then asks:

RUNOFF FORMAT?

The user response may be either:
YES or
NO

depending on which file type is to be printed.

The NO or N response, indicating a standard time sharing ASCII file, results in one more question being asked by the APRINT subsystem:

LINE COUNT 55?

to which user responds:

YES
CR (carriage return) or
n

Where: YES, Y or CR indicate that paging is done on a 55 line-per-page basis

n is a one- to three-digit integer to be used by APRINT for the lines-per-page count.

The file is then reformatted by the APRINT subsystem and passed along to "backdoor" SYSOUT for printing.

NOTE: The APRINT subsystem does not recognize the first line reformatting convention.

First Line Reformatting Information in the File

To minimize the subsystem/user interface involved in the use of the RUN, PRINT, BRUNCH, BPRINT, and ASCBCD commands, the required reformatting and disposition information may be contained within the file to be converted as its first line and identified by the special marker characters ##. If this information exists in the file submitted, the normal question/answer sequence is bypassed.

The general format of the first line reformatting information is as follows:

line-number##first-response second-response

Where:

line-number - Optional line number.

first-response - An appropriate full response to CARD FORMAT, DISPOSITION? in the case of RUN; to CARD FORMAT? in the case of PRINT; or to LABELS? in the case of ASCBCD.

second-response - An appropriate response to TAB CHARACTER AND SETTINGS? -- if required -- in the case of all three commands, separated from the preceding first-response by one blank.

The form of the information represented by first-response and second-response is precisely that which would be required by the question/answer sequence of the respective commands.

An example of first line control information for the RUN command is as follows:

```
10##MOVE,WAIT ;,8,16,32
```

Note that the semicolon following the single blank specifies the tab character and is not a separator.

Another example:

```
10##NORM
```

Note that the second-response portion is null in this example because of the use of NORM as the first-response.

$ SELECTA Card Option

With $ SELECTA cards the user may merge two or more time sharing input files into one batch file. The select function is initiated when a $ SELECTA card is detected in the input stream. The format is:

```
1       8       16
$       SELECTA filedescr
```

Where: The variable field (filedescr in columns 16 through 72) contains only the file selected for merging with the current file.

NOTE: Do not attempt to include comments on this card.

Upon encountering this card, CARDIN stops processing input from the current file and begins processing the selected file. When an EOF is encountered on the selected file, processing reverts to the previous file. The $ SELECTA card is deleted from the input stream by CARDIN and does not become part of the batch input.

Restrictions on the $ SELECTA card option are as follows:

- Nesting of a select activity within another select activity is permitted to a depth of 10.
- The contents of the selected file must be in time sharing format.
- No test will be made for first line reformatting information on selected files.
- A $ ENDJOB card detected in a selected file will be bypassed.

The following terminal printout is an example of the use of the CARDIN subsystem, utilizing the LIST and PRINT control commands and the SCAN subsystem. The text enclosed within brackets is not part of the printout but has been added to explain salient features.

```
*CARDIN
*OLD PASSRC
*LIST
```

List the input file.

```
010$;IDENT;VXA,JANEDOE
020$;NOLIB
030$;OPTION;NOSETU,SAVE/PX,NOGO
040$;LOWLOAD
050$;ENTRY;BEGIN
060$;GMAP;NDECK
070;TTL;SUBSYSTEM TO LIST AVAILABLE FILE TABLE
080;LODM;.G3TSS
090;.SSDRL
095;BSS;36;REQUIRED RESERVE SPACE FOR TS
100;SYMDEF;BEGIN
110BEGIN;DRL;PASAFT;READ THE AFT
120;ZERO;LOCA,0
130;LXL2;LOCA;GET FILE COUNT
140;TZE;END;NO FILES...
150;EAX1;LOCA+1
160AGAIN;LDAQ;0,1;PUT NAME IN LINE
170;STAQ;NAME
180;LDAQ;TAL1;REFRESH THE TALLIES.
190;STAQ;TEMP
200;DRL;KOUT;WRITE THE FILENAME.
210;ZERO;TEMP,CHAR
220;EAX1;2,1;INCREMENT LOCA POINTER
230;EAX2;-1,2;DECREMENT COUNT
240;TNZ;AGAIN;GET NEXT NAME
250END;DRL;RETURN
260;REM
270LOCA;BSS;25;AFT BUFFER
280TAL1;TALLY;TEMP+1,1;OUTPUT TALLY
290;TALLYB;NAME,8
310NAME;ASCII;2,
320;END
```

Place program in system loadable format on permanent file PASFIL.

```
330$;PRMFL;H*,R/W,R,JANEDOE/PASFIL
```

Place system output file on permanent file PASLST for use by SCAN.

```
340$;PRMFL;P*,R/W,L,JANEDOE/PASLST
350$;ENDJOB
READY
*
```

Indicates BUILD mode.

```
*PRINT◄--------  Print the file in its extended format.
CARD FORMAT ?
MOVE
TAB CHARACTER AND SETTINGS?
;,8,16,32

08/02/68     12.92

010 $       IDENT    VXA,JANEDOE
020 $       NOLIB
030 $       OPTION   NOSETU,SAVE/PX,NOGO
040 $       LOWLOAD
050 $       ENTRY    BEGIN
060 $       GMAP     NDECK
070         TTL      SUBSYSTEM TO LIST AVAILABLE FILE TABLE
080         LODM     .G3TSS
090         .SSDRL
095         BSS      36          REQUIRED RESERVE SPACE FOR TS
100         SYMDEF   BEGIN
110 BEGIN   DRL      PASAFT           READ THE AFT
120         ZERO     LOCA,0
130         LXL2     LOCA             GET FILE COUNT
140         TZE      END              NO FILES...
150         EAX1     LOCA+1
160 AGAIN   LDAQ     0,1              PUT NAME IN LINE
170         STAQ     NAME
180         LDAQ     TAL1             REFRESH THE TALLIES.
190         STAQ     TEMP
200         DRL      KOUT             WRITE THE FILENAME.
210         ZERO     TEMP,CHAR
220         EAX1     2,1              INCREMENT LOCA POINTER
230         EAX2     -1,2             DECREMENT COUNT
240         TNZ      AGAIN            GET NEXT NAME
250 END     DRL      RETURN
260         REM
270 LOCA    OBSS     50               AFT BUFFER
280 TAL1    ETALLY   TEMP+1,1         OUTPUT TALLY
290          TALLYB  NAME,8
300 CHAR     OCT     015012177177     CR,LF,NULL,NULL
310 NAME    EASCII   2,
320          END
330 $       PRMFL    H*,R/W,R,JANEDOE/PASFIL
340 $       PRMFL    P*,R/W,L,JANEDOE/PASLST
350 $       ENDJOB

*
```

```
*RUN◄---- Prepare to send input file to batch.
  SNUMB # 0122T◄--- SNUMB Assigned to this job.
CARD FORMAT,DISPOSITION ?
M,W◄--- CARDIN options MOVE and WAIT
TAB CHARACTER AND SETTINGS?
;,8,16,32◄---------- As used in input file.
  NORMAL TERMINATION.◄--- Job termination from batch.
*SCAN PASLST◄------- Call SCAN subsystem.
FORM?GMAP◄---------- Identify job type
006 ERRORS
EDIT?Y
?FLAG◄------------- List assembly errors.
M 2 LODM .G3TSS 00000080 #0013
O 000051 000000 0000 11 000 11 AGAIN LDA 0,1
  PUT NAME IN LINE 0000016
O #0023
A 000052 000117 7570 00 010 12 STAQ NAME 00000170 #0024
U 000054 000000 7570 00 000 14 STAQ TEMP 00000190 #0026
U 000056 000000 000116 001 16 ZERO TEMP,CHAR 00000210 #0028
U 000114 000000 0001 00 000 23 TAL1 TALLY TEMP+1,1
  OUTPUT TALLY 000002
80 #0035
TEMP #0061

?DONE◄--------
```

Leave. At this point the assembly errors should be corrected and the job resubmitted.

To place the object program (H*) on file PASFIL, a $ EXECUTE card would be added to the deck setup of the source program.

SECTION III

SCAN SUBSYSTEM

SCAN FUNCTION

The SCAN subsystem permits scanning output from the following types of batch jobs at a Time Sharing System terminal:

- Those submitted via CARDIN.
- Those submitted via remote batch facility.
- Those submitted at the central site.

The output to be scanned must have been saved on a permanent file.

The output file to be scanned falls into one or more of the following format categories:

GMAP assembly listing (GMAP)

FORTRAN compilation listing (FORT)

COBOL compilation listing (COBOL)

General Loader output: load map, etc. (LOAD)

User-generated output (USER) -- that is, not in any standard format

Special command verbs and other facilities are provided for operating on these forms of output.

Essentially, the SCAN subsystem allows perusal, by means of highly selective listing, of any type of batch output, with special features that facilitate efficient inspection of commonly desired items of information -- for example, flagged lines in a GMAP assembly.

SCAN OPERATION DESCRIPTION

The SCAN question/answer sequence is as follows:

Subsystem selection

Response: SCAN

Question: FILE?

Response: filedescr or filedescr;n

where: filedescr is the file description and n is the number of segments to be scanned in sequence when the file is segmented by end-of-file marks.

Question: FORM?

Response: One of the following, indicating the format of the file:

GMAP - subsystem initially responds with the number of assembly errors detected.

FORT - subsystem initially responds with the number of compilation errors detected.

COBOL - subsystem initially responds with the number of compilation errors detected.

LOAD - subsystem initially responds with the number of errors detected by the loader.

USER - subsystem initially responds with the question, CODE? The user's answer to this question is from one to five characters to be used by the subsystem as a line code in searching the user's output text, or the answer may be a null response (carriage return only). The matching line code in the file is assumed to be found left-justified in character positions 2-6 of each line; the FIND and PRINT verbs ignore all lines in user format output that do not contain the current line code (if line code is given). The initial line code given may be replaced by means of the CODE verb (see the following section). The null response is equivalent to ignore line codes, if any.

Line codes can serve to mark lines of the same type or category in the output text. Use of this feature implies that the user has prepared his output with this feature in mind. It is not an essential part of the system, however; and SCAN may be used with output files not having line codes. But if the user has control of his output format, line codes can facilitate the use of SCAN.

A matching mask is created automatically, so that a hierarchy of line codes can be established. For example, given lines with E1, E2, E1.5 as line codes, a code E will find all lines, code E1 will find lines E1 and E1.5, etc.

Question: EDIT?

Response:

*Y	compress blanks, line number will not be printed
Y*	compress blanks, line number will not be printed
*N	type blanks, line number will not be printed
N*	type blanks, line number will not be printed
*	type blanks, line number will not be printed
carriage return	type blanks, line number will print
Y	compress blanks, line number will print
N	type blanks, line number will print

To shift modes, type EDIT and enter the appropriate reply when the subsystem issues the edit question.

Question: ?

Response: a SCAN verb

Note the four-level structure in the question/answer sequence:

1. Subsystem selection
2. FILE?
3. Format (FORM?)
4. Verbs(?)

The transition from a lower level to the next higher is made by responding to one of these questions with a null line (carriage return only). For example, if a user is scanning a file created by a FORTRAN compile and execute, he may wish to use the formats FORTRAN and LOAD alternately. When he has scanned the FORTRAN listing and wishes to proceed to the load map, or vice versa, he responds to the initial question mark (verb level) with a null line; SCAN then asks for a new format (FORM?).

SCAN VERBS

The SCAN verbs are given below, with their arguments. Only the first four characters are significant; that is, PRINT can be abbreviated as PRIN, and BATCH can be extended to BATCHJOB. A blank must separate the verb from its argument.

First letter abbreviations of the following SCAN verbs are also acceptable: PRINT, FIND, SPACE, BACK, ERROR, UNDE, and CODE.

FIND

FIND /literal string/;n

/ represents any desired delimiter chosen by the user. The literal string is a pattern of characters to be searched for; n represents the nth occurrence.

The FIND verb positions an implied pointer to the nth line containing the literal string (beginning with the line currently pointed to). If n is not given, 1 is assumed. If no literal string is given, all lines are assumed to match; for example, FIND; 2 is equivalent to SPACE 1 (see SPACE verb).

The FIND verb also accepts all standard Text Editor argument forms.

PRINT

PRINT n

n is the number of lines to be printed.

PRINT causes a printout at the terminal of the next n lines, beginning with the current value of the pointer. If n is not specified, only one line is listed. Lines are listed with their automatically generated line numbers for future reference in LINE verb commands. Line codes are effective, if in force.

Note that the PRINT verb moves the pointer, so that a subsequent FIND verb begins its search with the last line that is printed.

If n consists of the literal ALL, all lines with a matching code are printed, from the current line to end-of-file.

The appearance of only a number in the listing of a line indicates that the corresponding record is a slew record.

The PRINT verb also accepts all standard Text Editor argument forms.

LIST

LIST n

LIST is synonymous with PRINT in all respects.

BATCH

BATCH ab

The system asks STATION CODE? The user replies ab or simply a carriage return, where ab is the station code of a remote-batch terminal.

The system then asks, $ IDENT?, to which the user replies with the variable field of his batch $ IDENT card.

The BATCH verb initiates a Bulk Media Conversion (BMC) job which transfers the entire contents of the file to remote printer ab. If the station-code reply is null, the output is printed at the central site.

SPACE

SPACE n

Spaces the pointer ahead n lines. If n is not specified, the pointer advances one line. An attempt to position the file beyond its end results in the file being positioned at its beginning, with a warning message to the terminal (EOF). The SPACE verb operates independently of line codes.

BACK

BACK n

Spaces the pointer back n lines. If n is not given, the file is rewound (pointer moved to line 1). The BACK verb operates independently of line codes.

LINE

LINE n

As each line is listed, an automatically generated line number is typed with it. The LINE verb repositions the pointer to the specified line number, n. (The line number used need not have been printed prior to being referred to.) Line codes are not effective.

The above are general purpose verbs. The following special verbs are useful for manipulating specific formats:

ERROR

ERROR n

Requests a list of the next n error printouts of the form corresponding to the output format in question. (For example, if the format is GMAP, flagged instructions. If USER, .FXEM messages are listed.) The absence of n implies all such messages.

UNDEFINED

UNDE

This command (no argument) is used while scanning GMAP assemblies to list all undefined symbols.

FLAG

FLAG x

Lists all lines of a GMAP assembly having the error flag specified by x (A, U, M, O, etc.). The absence of a specific error tag implies that the user wishes a list of all flagged instructions.

LOAD MAP

LOAD

Prints out an abbreviated load map. Only primary SYMDEFs are listed, and library routines are omitted.

CODE

CODE abcde

Employed with the user format to change the line code. The argument abcde is a one- to five-character code from the BCD character set. A null argument turns off line codes. That is, all line codes are accepted until the CODE verb is used to resume with a valid line code.

EDIT

Returns the subsystem to EDIT?

DONE

Returns the subsystem to subsystem selection level.

BYE

Terminates user's current session with the Time Sharing System.

REM

REM text

Produces the text as remarks line on printer log; otherwise ignored by SCAN.

SECTION IV

FDUMP SUBSYSTEM

FDUMP FUNCTION

The FDUMP subsystem (file dump and correction) is a remote terminal file inspection and maintenance facility for permanent files in the GCOS file system. These files reside on a shared-file type of device such as a disk storage unit. The files may be generated in either batch, remote/batch, or time sharing environments.

With the FDUMP subsystem, the user at a remote terminal can manipulate the content of any permanent file (to which he has access) as follows:

- Read into memory any given 320-word llink of the file.
- Scan the llink content, by dumping all or nonzero portions of the block at his terminal, using the snapshot function.
- Patch selected locations in the llink, using the patch function.
- Rewrite the corrected llink back onto the file.
- Copy the file.

NOTE: The maximum number of llinks in a file that can be accessed through FDUMP is 16,343.

This facility is provided primarily for those situations where the user needs or wishes to inspect the actual binary content of a file. It is, therefore, mainly a debugging tool. The file is dumped and patches are specified in octal form. Word locations within a block are also specified in octal, beginning with zero.

FDUMP OPERATIONAL DESCRIPTION

The user selects FDUMP at the subsystem selection level or as a command under the CARDIN subsystem.

The first-level question asked after the subsystem is called is:

FILE NAME?

The permissible responses are:

carriage return - return to the subsystem selection level.

filedescr - Specified file is accessed, if possible, and treated as a linked or random file, depending on how it is defined.

filedescr;L - Specified file is accessed, if possible, and treated as a linked file.

filedescr;R - Specified file is accessed, if possible, and treated as a random file.

Caution: filedescr must not be the same name as another even if the duplicate filedescr is in another catalog when the FUNCTION is copy (C).

The second-level question is:

BLOCK TO BE READ -

The permissible responses are:

carriage return - return to the FILE NAME? level.

n - the block specified by the block serial number n will be read into an internal buffer. The Copy function (see following function description) requires a dummy response of 1.

If the block serial number is outside the limits of the file, the error message BSN OUTSIDE FILE LIMITS is given, and BLOCK TO BE READ will be repeated. If the block serial number is within the current file size but the implied block was not written on the file, it is read but it contains garbage data not pertaining to that file.

The third level (and final) question is:

FUNCTION?- (this question is repeated upon return from any of the FDUMP functions.)

The permissible responses are:

carriage return - return to BLOCK TO BE READ.

Sloc - Snap the specified (octal) location.

Sloc-loc - Snap the field specified by (octal) location-location (from-to).

Sloc,n - Snap n (octal) words starting with the specified (octal) location.

Ploc data - Patch the specified (octal) location with the specified (octal) data. The data entry may be a multiple entry. For example:

```
Ploc data1, data2, data3
    data1 goes to loc
    data2 to loc+1
    data3 to loc+2
```

W - Write the corrected block back into the permanent file.

C filedescr - Copy the complete file onto another file specified by filedescr. (Refer to FDUMP Copy Function Details paragraph.)

D - Done. Return to SYSTEM? or to CARDIN.

FDUMP Error Indication and Replies

1. When the named file cannot be accessed, FDUMP replies

   CANNOT ACCESS FILE filename

   and returns control to the calling level (SYSTEM? or CARDIN).

2. When the block serial number given is either zero or is a number larger than the possible number of blocks in the file, the error message is:

   BSN OUTSIDE FILE LIMITS

   BLOCK TO BE READ is then repeated.

   For linked files, block size is assumed to be 320 words; the first block serial number is 1. Random files are positioned by multiples of 64 words, beginning with block 0. However, they are read in blocks of 320. Therefore, one read makes available five contiguous blocks of 64 words.

3. When the system receives a bad hardware status, FDUMP replies:

   51 FILE filename -- I/O STATUS xx

   FUNCTION?

   A partial block may have been Read and may be correctable by use of the S, P, and W functions. If none of the block appears to have been Read, answer with a carriage return to repeat the BLOCK TO BE READ question. Also, verify the block serial number that was specified.

4. When parameters are incorrect in form for the S, P, or W functions, FDUMP will reply

   INVALID INPUT-RETYPE

Short Form Usage of FDUMP Functions

Once the user has become familiar with the conversational, or question/response sequence, form of FDUMP, a short form of function specification which effectively eliminates questions normally asked can be used. Multiple responses can be supplied on a single line, separated from one another by a space character. For example:

```
*FDUMP * 1 S50,100 DONE
```

NOTE: n consecutive spaces represent the equivalent of n-1 null responses.

FDUMP Error Indication and Replies

1. When the named file cannot be accessed, FDUMP replies

   CANNOT ACCESS FILE filename

   and returns control to the calling level (subsystem selection or CARDIN).

2. When the block serial number given is either zero or is a number larger than the possible number of blocks in the file, the error message is:

   CSN OUTSIDE FILE LIMITS

   CLOCK TO CE READ is then repeated.

   For linked files, block size is assumed to be 320 words; the first block serial number is 1. Random files are positioned by multiples of 64 words, beginning with block 0. However, they are read in blocks of 320. Therefore, one read makes available five contiguous blocks of 64 words.

3. When the system receives a bad hardware status, FDUMP replies:

   ```
   <51>FILE filename -- I/O STATUS xx

    FUNCTION?
   ```

   A partial block may have been read and may be correctable by use of the S, P, and W functions. If none of the block appears to have been read, answer with a carriage return to repeat the CLOCK TO CE READ question. Also, verify the block serial number that was specified.

4. When parameters are incorrect in form for the S, P, or W functions, FDUMP will reply

   INVALID INPUT-RETYPE

FDUMP Copy Function Details

The copy is a physical copy; that is, it does not stop at logical end-of-file but continues to the file length defined in the file system as current size.

If the copy file is smaller than the size defined for the file to be copied, FDUMP grows the copy file to the necessary size.

Note that filedescr, specifying the copy file, may be simply a file name or may be a catalog/file string, but must not have the same filename even if duplicate filename is under a separate catalog.

At completion of the copy, the user is returned to the FILE NAME? level.

SECTION V

CONVERSATIONAL DEBUG

RBUG GENERAL DESCRIPTION

The Conversational Debug Routine (RBUG) is a dynamic debugging tool for batch programs initiated through the CARDIN subsystem. The user of RBUG should be familiar with the normal DEBUG routine and its associated control cards as described in the General Loader reference manual.

When the control card $ USE RBUG are used, the normal DEBUG routines are superseded by RBUG; control and information are then obtained in exactly the same manner as in the standard routine. By the use of the BREAKPOINT pseudo-variable on the General Loader DEBUG control card, the user may gain control at specified points in his program. At those points, the user may:

1. Obtain snapshots of locations and registers.

2. Replace the contents of locations and registers.

3. Insert further breakpoint locations or delete breakpoints he has placed earlier (breakpoints specified in the control card or FORTRAN statement may not be deleted).

4. Continue the run at the interrupt point or another point; or stop the run, either by normal termination or abort.

Because of the way the loader builds DEBUG tables, variables are printed for each debug location in reverse order. Therefore, if a breakpoint is desired, the BREAKPOINT pseudo-variable should be the first in the list, so that a typeout of the requested variables occur before the breakpoint is executed. When a point is reached in the execution of the program for which conditions specified by the control card or DEBUG statement are met, output (if any) is typed at the user's terminal. In addition, if there has been a breakpoint specified, execution of the program is suspended and a question mark printed, asking for instructions. After each instruction has been acted upon, a question mark is typed, requesting further instructions until the user asks to run, terminate, or abort.

RBUG instructions, listed in the following section, are all composed of a single letter command followed in most cases by an argument. The argument follows the standard DEBUG convention that all variables are considered to be single dimension arrays, with the first element being element number 1. Therefore the general form of an argument is:

Name (i, j, k)

where the subscripts have the meanings from, to, and increments of. For example A(3,9,2) has the meaning from A(3) to A(9) in steps of 2 -- that is, A(3), A(5), A(7), A(9).

The name is that of a variable within the subroutine at which the program was interrupted or of a variable in common. If the name contains only numerics between 0-7, it is considered an octal location. Therefore 9(3) would be interpreted as the location of statement 9 plus 2, but 7(9) would be interpreted as octal location 17 (7 plus 8 = decimal 15 = octal 17). In order to enable reference to FORTRAN statements not containing an 8 or 9, the pound sign (#) is treated as an invisible alphabetic. For example, statement 23 can be referred to as #23, 23#, or 2#3.

Subscripts, where applicable, are optional.

RBUG INSTRUCTIONS

The following RBUG instructions are available to the user:

A (ASCII) prints out in ASCII format the contents of the requested locations.

B (Breakpoint) inserts a dynamic breakpoint at the desired location. When this point in the program is later reached, the user is notified, and has then the same instruction capability as he would have at a normal breakpoint. However, if the breakpoint location is specified octally, no symbol table is available; and therefore at this breakpoint only octal addresses may be used.

C (Complex) prints out the contents of pairs of cells in complex format. Subscripts refer to pairs, not to individual locations.

D (Double precision) prints out a double precision, floating point representation of the word pairs referenced. Subscripts refer to word pairs, as above.

E (Erase breakpoint) eliminates a previously inserted dynamic breakpoint.

F (Floating) prints out a floating point representation of the locations referenced.

H (Hollerith) prints out a Hollerith (BCI) representation of the locations referenced.

I (Integer) prints out a decimal representation of the locations referenced.

L (Logical) prints out a logical representation (T or F) of the locations referenced.

M (Modify) is used to change the contents of a register. The form of the instruction is Mr xxxx, where r represents the register to be modified, and xxxx is an octal representation of the new contents - right-justified and zero filled in the register. The permissible values of r are A, Q, E, I, or X0 through X7.

O (Octal) prints out the contents of the referenced locations in octal notation.

P (Patch) replaces the contents of the given location with a new value. General form of the instruction is Pname(i) xxxxxxx. The value xxx... is right-justified and zero-filled.

Q (Quit) - no argument. Causes the program being monitored to abort with reason code X1.

R (Run) causes the program to resume at the instruction which was replaced with the breakpoint. If an argument is given (single location), the program is given control at that point.

S (Starting address) is a single argument instruction. The location given as the argument is added to all subsequent locations given in octal. For example, having given SABC as an instruction, 023 prints out (in octal) cell 23(8) relative to location ABC. Locations specified symbolically are not affected by the starting address. The address may be changed at will by the user and may be nullified by a simple S (equivalent to S0, start at zero).

T (Terminate) causes the program to terminate normally, by calling .FEXIT.

W (Where) is used with a single argument. It prints out in octal the effective address of that argument.

X (Display registers) is used without argument. It yields a panel dump of all registers. It may also be used selectively with the following forms: XA, XQ, XE, XI, or X0 through X7.

The user establishes direct access connection with his program either by the TALK option of CARDIN or by logging off from time sharing and calling a GRTS or NPS phone number. He then requests to be connected with his SNUMB. For example, when submitting from CARDIN, the user is notified that he has been assigned SNUMB #0042T. He then answers BYE to the system question, calls GRTS or NPS, and requests $*$DAC0042T.

Appendix A contains a sample program in which use of RBUG is illustrated.

SECTION VI

CONVERSATIONAL FILE AND RECORD INPUT/OUTPUT

The File and Record Control routines include a conversational, or direct access, input/output capability, using the File and Record Control OPEN, GET, PUT, and CLOSE calls, and a direct access file control card, $ DAC. The $ DAC control card allows the user to specify one or more of his program files as conversational, to be connected directly to a remote terminal device.

An additional control card, $ USE .RTYP, must be used in order to activate the required File and Record Control routines (.GOPNR, .GCLSR, .GGETR, .GPUTR).

Appendix A contains a sample program in which use of File and Record Control conversational routines are illustrated.

CONTROL CARDS FOR FILE AND RECORD CONTROL I/O

- $ DAC

The Direct Access (DAC) file control card has the following format:

```
$ DAC fc (,d)
```

Where: fc - A two-character alphanumeric file code identifying a direct access file referred to in the program (or FCB).

d - An optional one-character logical unit designator. If the ,d field is omitted, blank is assumed.

The logical unit designator is provided primarily for future system implementations. This field normally is omitted; if the associated terminal device is to be connected via the TALK option in CARDIN -- the standard case -- this field must be omitted (or d must be blank).

- $ USE .RTYP

The $ USE .RTYP control card must appear in the job control deck to direct loading of the correct File and Record Control routines. If this card does not appear, any reference to a file defined as direct access causes a program abort with a code of NT.

APPLICABLE I/O CALLS

The following File and Record Control calls are applicable to a direct access, or conversational, file:

- OPEN -- Connect a remote terminal device and open file.
- GET -- Read logical record from a remote terminal.
- PUT -- Write logical record to a remote terminal.
- CLOSE -- Close file and Disconnect a remote terminal device.

The calling sequences are standard, as described in the File and Record Control reference manual. The usage of the calls is also standard except as noted in the following functional descriptions.

- OPEN

  This function attempts to connect a remote terminal device identifying itself by snumbd, where snumb is the sequence number of the calling program (job), and d is the logical unit designator (normally blank). If no such terminal requests a direct access connection within a reasonable time (on the order of several minutes), the FCB is marked file-not-present; and control is returned to the program. If a connection is made, the station code (line ID) of the terminal is placed in LOCSYM -4 (upper); and the file is marked present and open.

- GET

  The terminal is sent the input-request message:

  fc?

  where fc is the current file code. When input is received from the terminal, it is translated to BCD and stored as a standard system record in the user's buffer. If the record type is fixed, short input lines are blank filled to correspond to the record size specified in the FCB. Variable length records are blank filled to a minimum of 14 words.

  A null input line (carriage return only) results in an end-of-file return, as will also a break or disconnect. On an end-of-file return, the user may determine the reason from the first status return word (LOCSYM -3, bits 6-11); the code is:

  0 = null line

  1 = break signal

  2 = terminal disconnect

  The carriage return null input does not set the end-of-file bit, thus communication with the terminal may continue.

- PUT

  The output in the user's buffer is translated from BCI to ASCII and sent to the terminal. The use of this call carries the same restriction as is imposed on the WTREC call -- data must be present prior to the call. (This restriction does not normally apply to PUT for nonconversational files, where it can be used to reserve buffer space without moving data.)

  A break signal or a disconnect during the call will cause the end-of-file status to be posted in the FCB, with status code 1 or 2. (See GET, above.) (End-of-file during a PUT on nonconversational files normally causes an abort.)

- CLOSE

  The connected terminal is notified that the file is being closed, and the corresponding file is marked closed.

SPECIAL FILE AND RECORD CONTROL I/O CONSIDERATIONS

All required ASCII-to-BCD and BCD-to-ASCII translations are performed by the GET and PUT subroutines. The user's output buffer must therefore contain BCD data; his input buffer contains BCD data following the GET call.

During a PUT call, a break or attention signal (normal means of interrupt from a terminal) or a user-initiated disconnect results simply in the posting of an end-of-file status in the FCB, rather than a program abort. It is the user's responsibility to check periodically for such occurrences (code 1 or 2 in LOCSYM -3, bits 6-11).

The optional logical unit designator in the $ DAC control card allows multiple terminals to be connected simultaneously to one program (one terminal device per file). The terminals requesting connection are distinguished by $snumbd_1$, $snumbd_2$,..., $snumbd_n$. A terminal connected via the TALK option of CARDIN implicitly carries a logical unit designator of blank (040 octal); the corresponding $ DAC control card d field must be null. Any terminals to be connected to nonblank designated program files must be connected through the normal GRTS procedure: (1) dialing up a GRTS line and (2) a direct access connection request, $*$ DAC snumbd.

One terminal device may be connected to more than one program file (though the converse is not true). In this case the several $ DAC file cards specify or imply the same logical unit designator (normally blank).

SECTION VII

JOUT SUBSYSTEM

JOUT FUNCTION

The JOUT subsystem permits manipulating output from the following types of batch jobs at a time sharing terminal:

- Those submitted via CARDIN with a DISPOSITION response (after a RUN command) that includes J or JOUT.
- Those submitted via remote batch.
- Those submitted at the central site.

The output to be manipulated must be designated for a remote destination and must contain a $ USERID control card.

The output file to be scanned will fall into one or more of the following format categories:

GMAP assembly listing (GMAP)

FORTRAN compilation listing (FORT)

COBOL compilation listing (COBOL)

PL/I compilation listing (PL/I)

MEMORY dump (DUMP)

General Loader output: load map, etc. (LOAD)

User-generated output -- that is, not in any standard format

JOUT OPERATIONAL DESCRIPTION

The JOUT question/answer sequence is as follows:

Subsystem selection

Response: JOUT or JOUT sssss

where: sssss - SNUMB for the job whose output is to be manipulated. If only JOUT is entered, the system requests SNUMB?

A null response causes a return to subsystem selection level.

If the job output is not available for manipulation, the system transmits one of the following messages to the terminal and returns to the subsystem selection level (sssss is the job SNUMB):

sssss OUTPUT IS BUSY
The job was found but is being printed at the central site, at a remote batch station, or at another time sharing terminal.

sssss OUTPUT NOT FOUND
The desired job was not found in the system.

sssss NOT YOUR JOB
The log-on USERID does not match the USERID found in the job stream.

BATCH SYSTEM FULL--TRY LATER
Time sharing cannot communicate with SYSOUT at this time, or not enough room exists to read the file.

sssss READING-CR
The job is being input through the central site card reader.

sssss READING-MT
The job is being read from magnetic tape.

sssss READING-RMT
The job is being read from a remote station.

sssss WAITING-ALLOC
The job is not yet a candidate for peripheral allocation.

sssss WAIT-PERIF
The job is waiting for peripheral allocation.

sssss WAIT-CORE
The job is waiting for memory allocation.

sssss IN HOLD
The central site operator has placed a hold on this job.

sssss IN LIMBO
The job is waiting for a RUN command from the central site operator.

sssss EXECUTING
The job is in execution.

sssss WAIT-TAPE
The job is waiting for the operator to mount or ready a specified tape.

sssss TOO BIG
This job's requirements exceed the sieve limits set by the central site.

sssss OVERDUE
The allocation of this job is overdue.

sssss IN RESTART
The job is waiting to be restarted by the operator after a system failure.

sssss TERMINATING
This job is engaged in the termination procedure.

QUESTION: FUNCTION?

Response: One of the following commands. When the function has been completed, the system returns to FUNCTION?

- ACTIVITY

  ACTIVITY n

  JOUT prepares to read the activity specified by n where n cannot exceed 17.

- DIRECT

  a. DIRECT id

     Direct the output to the remote station specified by id.

  b. DIRECT ONL

     Print the output at the central site.

- EPRINT

  EPRINT rc

  Simulate printer report output. The report code (rc) may be any of the codes received from the LIST command, or $$ may be substituted for a report code. The $$ causes the printing of the J* file (control card list and execution report) at the terminal. Trailing blanks and blank lines are suppressed.

- HOLD

  HOLD

  If the user responds HOLD the subsystem deaccesses the SYSOUT data and returns the user to either the build mode or the system level and the output is not processed by SYSOUT Report Writer.

- LIST

  LIST

  List the report codes associated with the current activity.

- PRINT

  PRINT rc

  Simulate printer report output. The report code rc may be any of the codes received from the LIST command, or $$ may be substituted for a report code. $$ causes the printing of the J* file (control card list and execution report) at the terminal. Multiple blanks are suppressed by the PRINT command.

- RELEASE

  RELEASE

  Remove the output from the system.

- SCAN

  SCAN rc

  Scan the job output with report code rc. The system requests FORM? From this point, the question/answer sequence and the facilities available are the same as for the SCAN subsystem (see Section III), with the exceptions noted below:

  1. The following SCAN verbs are not available: BATCH, REM, REM text, and BYE.

  2. Output in Memory dump format may be scanned. (Answer DUMP to the FORM? question. There is no initial subsystem response to this answer; the EDIT? question appears immediately.)

  3. DONE returns the user to the FUNCTION level.

SECTION VIII

CONVERT SUBSYSTEM

GENERAL

The CONVERT subsystem converts textual information from any text file format to any other text file format, thus eliminating the need for special-purpose conversion programs.

FUNCTIONS

The CONVERT subsystem has three primary functions, as follows:

1. The conversion of textual information between any two of the following physical file formats.

Media Code	File Format
0	variable-length BCD
1	BCD compressed deck (COMDK) card image
2	BCD 14-word card image
3	BCD variable-length print line image
5	obsolete TSS ASCII
6	standard system format ASCII
7	ASCII print line image
10	ASCII card image

2. The reformatting of files, accomplished by manipulating line numbers and recognizing special option characters and manipulating records accordingly.

3. Initiation of batch jobs. These batch jobs can be user-submitted or they can be submitted by the CONVERT subsystem in order to provide a printer listing or punched deck.

COMMANDS

A user determines which functions are to be performed by means of the following commands.

1. CONVERT (convert text)
2. JRN (batch job initiation)
3. JPRINT (print on a high-speed printer)
4. JPUNCH (punch a card deck)
5. DISPLAY (print at a terminal)

Each of these commands can be used in place of one or more TSS commands presently in use but designed to be removed in a future software release. TSS users should find it advantageous to use the CONVERT commands for the following reasons:

1. The CONVERT commands have much greater power and flexibility.
2. The CONVERT commands are truly a family of commands in that they all share the same user interface. Once a user becomes conversant with one CONVERT command, the user is essentially conversant with all CONVERT commands.

The CONVERT commands and their corresponding TSS commands are as follows:

CONVERT Command	TSS Command
CONVERT	ASCBCD,BCDASC,ASCASC
JRN	RUN (CARDIN)
JPRINT	BPRINT
JPUNCH	BPUNCH
DISPLAY	PRINT

Note that the user cannot give a CONVERT command and expect that it give identical results as from use of its corresponding TSS command. This follows from the fact that the user interface for CONVERT commands is not compatible with that of the corresponding TSS commands.

User interface changes have taken place in the following areas:

1. CONVERT commands do not recognize embedded # control lines but instead recognize $$ control lines. Ambiquity results from the use of ## to delimit special lines for both command file processing and card processing. Furthermore, the rigid, fixed format of the ## control line does not lend itself to use with the greatly increased number and complexity of options being made available.

2. CONVERT commands do not recognize $ SELECT A lines but instead rely upon the SELECT option. The SELECT option offers these capabilities, unavailable with the $ SELECTA line:

   a. If desired, only specific portions or segments of a selected file are included.

   b. If desired, options that apply specifically to the selected file can be given.

   c. The SELECT option is given on $$ control lines, as are other options. This means that other options can be given on the same line as the SELECT option, eliminating the artificial and misleading distinction between the selection function (option) and all other functions (options).

3. CONVERT commands do not prompt the user with question/answer sequences, such as

   CARD FORMAT, DISPOSITION?

   LABELS?

   TAB CHARACTERS AND SETTING?

   The CONVERT commands assume appropriate default values when such information is not given.

COMMAND SYNTAX

The command syntax for all commands is:

command [input file(s)] [=output file] [:options]

Brackets indicate fields that can be omitted from the command string; they are not actually typed. The three fields do not have to be ordered as shown; they can be given in any order. If none of the three fields follows the CONVERT command, the user is prompted for input. Command string characters can either be upper or lowercase.

CONVERT determines the formats of the input files by inspection. The user need not specify the input formats. If the VERIFY option is exercised, CONVERT tells the user what the input file formats are. If there are two or more input files, they need not all have the same format. CONVERT accepts all text file formats and translates the text internally to ASCII before reformating and writing to the output file.

The textual information from the input files is read, reformatted, and written to the output file. Thus the output file is a concatenation of the reformatted input files. If the output file is also one of the input files, a temporary output file, CNVRT*, is used and its contents are copied back to the declared output file as the final step.

Input Files Field

The [input files] field is a string of generalized catalog/file descriptions separated by semicolons. The input field must be preceded by a semicolon unless it is the first field following the CONVERT command. In that case, no preceding semicolon is required. Each file description may be either a file name or a full catalog/file description. Alternate names and permissions are allowed. If permissions are given, they are used, but they must include the required permission, query, as a subset. Line numbers or sequence numbers can be specified within parentheses after each catalog/file description. The current file (*SRC) is specified by the character "*" or by "*SRC". If the input file description is null, the current file (*SRC) is assumed by default. An input file described by ** is assumed to be the terminal.

Input files not initially in the AFT are removed from the AFT when processing is complete. Input files are accessed and deaccessed in sequence as processing continues; there is no limit on the number of input files.

Output File Field

The [=output file] field is a single, generalized catalog/file description preceded by an equal sign. Alternate names and permissions are allowed. If permissions are given, they are used, but they must include the required permission, write, as a subset. Parenthetical line numbers or sequence numbers are forbidden. The current file (*SRC) is specified by the character "*" or by "*SRC". If the output file description is null, the current file (*SRC) is assumed by default. An output file described by "**" is assumed to be the terminal.

The output file field is not meaningful except to the CONVERT command. Therefore, for all other commands, the specification of an output file is treated as an error.

Options Field

The [:options] field is a string of mnemonics preceded by a colon. With the exception of the DEFAULT option, options may be entered in any order. Options must be separated from each other by a comma.

Some options require arguments and some options accept but do not require arguments. When arguments are given, they must immediately follow the option and must be enclosed in parentheses.

Line Continuation

Line continuations can occur at the delimiters "=", ":", "(", ";", or ",". They can also occur at the file descriptor delimiters "/", "$", and quote. Continuation input is also requested whenever a line is terminated while giving parenthetical input.

Control Characters and Blanks

In general, blanks and control characters on the command line or on embedded $$ control lines are ignored. However, CONVERT does recognize and utilize a control character when:

1. It is specified as the tab character.
2. It is specified as the untab character.
3. It is specified as the begin character.

CONVERT recognizes and utilizes blanks and control characters when they are part of the IDENT option's argument.

OPTIONS

A listing of the available options follows.

Media Code Options (Default Is ASCII)

BCD	records are converted to media code 0
COMDK	records are converted to media code 1
CARD	records are converted to media code 2
PRINT	records are converted to media code 3
OLDASC	records are converted to media code 5
ASCII	records are converted to media code 6
APRINT	records are converted to media code 7
ACARD	records are converted to media code 10
SAME	records are written without a media code change

Line Number Options (Default Is ASIS)

ASIS	or A	leave line numbers as is
INSERT (i,j)	or I (i,j)	insert line numbers
MOVE	or M	move line numbers from one location to another
NORM (ch)	or N (ch)	move option with tab character expansion
RESE (i, j)	or R (i,j)	resequence line numbers
STRIP	or S	strip line numbers from a line
LABEL (abcde (i-j)) or L (abcde (i-j))		label for cols. 73-77 of COMDK, CARD, and ACARD

Character Manipulation Options (There Is No Default Option)

BEGIN (ch)	or B (ch)	begin a new line
COLUMNS (i-j)	or C (i-j)	delete all chars. except those in cols. i through j
LOWER		convert all characters to lowercase
UPPER		convert all characters to uppercase
SQUEEZE		replace multiple blanks with a single blank
TRAIL		delete all trailing blanks
TAB (ch,i,j---;ch,i,j---;---)		
or T (ch,i,j---;ch,i,j---;---)		expand tab chars. into blanks
UNTAB (ch,i,j---;ch,i,j---;---)		
or U (ch,i,j---;ch,i,j---;---)		insert tab chars. replacing blanks

Miscellaneous Options (There Is No Default Option)

DISCARD	or D	discard non-text records
VERIFY	or V	report input file formats
IGNORE		do not process subsequent $$ control lines
DEFAULT		use the default value for all options
TIME		print the date and time at the terminal

File Processing Option (Default Is Include For JRN)

(Default Is Exclude For All Other Commands)

SELECT (file)	use the selected file as input
EXCLUDE	do not expand any selected file
INCLUDE	expand all selected files

Specialized Options (There Is No Default Option)

JOUT	or J	save implied files for examination by JOUT
WAIT	or W	wait for job termination
ROUT (xx)		direct output to station xx
TALK		converse with a DAC program
URGC (i)		assign an initial urgency to a job
COPY (i)		generate multiple copies of a listing
IDENT (info)		$ ident line info
MONITOR		print status of spawned batch job at terminal
DIRECT		ignore the JOUT and ROUT options
DISMISS		ignore the WAIT, TALK, and MONITOR options

Media Code Options

The output record format options specify the physical format of the output record. These options are meaningful only to the CONVERT command. For all other commands the format of the output record is implied by the function of the command. The default option for the CONVERT command is "ASCII". A list of the options and their meanings is as follows:

BCD - variable-length BCD

COMDK - BCD compressed deck card image (COMDK)

CARD - BCD 14-word card image

PRINT - BCD variable-length print line image

OLDASC - obsolete TSS ASCII

ASCII - standard system format ASCII

APRINT - ASCII print line image

ACARD - ASCII card image

SAME - a record output media code is the same as its input media code

Line Number Options

Line numbers can exist with COMDK, CARD, ACARD, OLDASC, and ASCII records. All BCD, PRINT, and APRINT records cannot possess line numbers. The line number for an ASCII or OLDASC record consists of one to eight numeric characters. These numeric characters must be among the first eight characters in a line. A line number is defined to include any leading blanks. A line number is terminated by a non-numeric character, including blank. If the "#" character terminates a line number and if it is one of the first eight characters of a line, it is considered to be a delimiter. It is treated as neither part of the line number nor part of the text. The line number for COMDK, CARD, and ACARD records is defined to be all the trailing digits in columns 73-80. This field may begin with non-numerics; these also are considered neither part of the line number nor part of the text.

The line number options may specify:

1. Whether line numbers are to appear in the output text.

2. The actual line number values.

The default line number option is "ASIS". A description of each of the options follows:

ASIS — Line numbers are assumed not to be present in the input file. Text, including leading/trailing numeric characters and "#"'s are left as is.

STRIP — Strip line numbers from the input text before reformatting and writing the output text. Input COMDK, CARD, and ACARD records are truncated at column 72. Line numbers on ASCII and OLDASC records, when present, are discarded and the first character following the line number is treated as the first character of the line.

MOVE — Move line numbers. The input records have the line numbers detached from the text string, either from the front (ASCII or OLDASC) from columns 73-80 (COMDK, CARD, or ACARD). The output records have the line numbers re-attached to the text string, either at the front (ASCII or OLDASC) or in columns 73-80 (COMDK, CARD, or ACARD). If the output records are BCD, PRINT, or APRINT, the line numbers are not re-attached and the M option acts similar to the S option.

I(i,j) — Insert line numbers beginning with line number i and incrementing by j. The arguments i and j are optional. If they are not given, the defaults are i=10 and j=10. The input file is assumed not to be line-numbered. If the output records are BCD, PRINT, or APRINT, line numbers are not inserted and the I option is ignored.

R(i,j) — Resequence line numbers. Strip any existing line numbers from the input text and insert new line numbers in the output text, beginning with i and incrementing by j. The arguments i and j are optional. If they are not given, the defaults are i=10 and j=10. If the output records are BCD, PRINT, or APRINT, line numbers are not inserted and the R option behaves as the S option.

N(ch) — Implies the M option and specifies that the normal tab character (the colon) and tab settings (8, 16, 32, 73) have been employed in building the input file(s). The (ch) argument may be used to define a character which replaces the colon as the tab character.

LABEL (abcde(i-j)fghij(i-j)---) — If the output records are COMDK, CARD, or ACARD, then a label is placed left-justified in columns 73-77. The label is specified as 1 to 5 non-blank characters. The fields "abcde" and "fghij" represent the labels. The label is placed on only those lines with line numbers between i and j inclusive. Up to 10 distinct labels may be given. If more than one label is given though, the (i-j) specifications may not overlap.

The LABEL option is meaningful only if line numbers are attached to output records. Therefore, the label option is completely ignored unless it is accompanied by either the insert, resequence, or move option.

For the I and R options, output line numbers for ASCII and OLDASC records will have at least the number of digits specified for i in I(i,j) or R(i,j). Thus R(0010,10) will result in line numbers 0010, 0020, 0030,---.

Input records are assumed to have line numbers when the STRIP, MOVE, and RESEQUENCE options are specified. Otherwise, line numbers are assumed to be absent and leading numerics in ASCII format are treated as real text. When line numbers are assumed present, tabbing and columnizing are performed relative to the start of the real text.

The user must be careful not to alter the line number values of a BASIC file.

Character Manipulation Options

A description of each of the character manipulation options follows.

TAB(ch,i,j---;ch,i,j---;----) Expand tab characters into blanks. Use "ch" as a tab character with settings i,j,k,etc. Usually, any occurrence of the tab character in the input file(s) results in the replacement of the tab character with a string of blanks up to the next tab setting. However, if a tab character is encountered beyond the last tab setting specified for that tab character, it is treated as a normal non-tab character.

If a tab character is specified without specifying any tab settings, default settings of 8, 16, 32, and 73 are assumed. If the tab option is given without any arguments, the normal tab character, colon, and the default settings are assumed. There is no limit to the number of tab characters or settings allowed.

UNTAB(ch,i,j---;ch,i,j---;----) Insert tab characters, replacing blanks. Use "ch" as a tab character with settings i, j, k, etc. Any occurrence of a string of blanks terminating on an "untab" tab stop is replaced by the character "ch".

If a tab character is specified without specifying any tab settings, default settings of 8, 16, 32, and 73 are assumed. If the untab option is given without any arguments, the normal tab character, colon, and the default settings are assumed. There is no limit to the number of tab characters or settings allowed.

LOWER Convert all alphabetic characters to lowercase. This option is meaningful only if the output records are ASCII, OLDASC, or APRINT.

UPPER Convert all alphabetic characters to uppercase. This option is meaningful only if the output records are ASCII, OLDASC, or APRINT.

BEGIN(ch) Begin a new line (record) immediately after the character "ch". The character "ch" is treated as a delimiter and not part of the text. It is not placed in the output text. When the "ch" character is located at the beginning or end of a line, it is simply deleted. Strings of the "ch" character are treated as a single "ch" character.

COLUMNS(i-j) Delete all of the characters in a line except those which are located within columns i through j inclusively. The options BEGIN and TAB are both completed before COLUMNS takes effect. If a record does not extend through column j prior to the COLUMNS option execution, it is blank filled to column j. Thus, when the COLUMNS options is in effect, the length of all generated output records is j-i+1 characters.

SQUEEZE Replace any string of two or more blanks by a single blank. The options BEGIN, TAB, COLUMNS, and UNTAB are all performed before SQUEEZE is executed.

TRAIL Delete all trailing blanks on a line. The TRAIL option is performed immediately after the SQUEEZE option.

A number of options affect the length of an output text line. It is important that the user understand the order in which these options are performed. The order (from first to last) in which the options are executed is:

BEGIN
TAB
COLUMNS
UNTAB
SQUEEZE
TRAIL

Miscellaneous Options

VERIFY — If the VERIFY option is in effect when CONVERT completes the processing of an input file, then CONVERT gives a brief summary of the number of records obtained from the file. This summary gives, for each media code, the number of records which had that media code.

IGNORE — Ignore all embedded $$ control lines. Treat them as text.

DISCARD — Discard all non-text records. Non-text records are those records whose media code is not one recognized and interpreted by CONVERT. The JRN, JPRINT, JPUNCH, and DISPLAY commands require that non-text records be discarded. The CONVERT command normally does not require that non-text records be discarded. When non-text records are encountered during the execution of the CONVERT command, they are written to the output file, but no reformatting or media conversion is performed.

TIME — When the TIME option is invoked, the date and time of day are printed at the user's terminal.

DEFAULT — The DEFAULT option is used to nullify all options which the user has specified either on the command line or embedded $$ control lines. The default option has no affect on any of the "specialized" options. Because of the nature of the DEFAULT option, it is meaningless for it to be located in the options field of the command line. Therefore, if the DEFAULT option is encountered in the options field, an error message is issued. The same reasoning applies to the placement of the DEFAULT option anywhere other than the beginning of a $$ control line.

File Processing Options

SELECT (file) — The SELECT option is analogous to the $ SELECTA card. The select option allows an input file to specify other input files. Upon encountering the SELECT option, the selected file is obtained and is used in place of the $$ control line. Nesting of selects is permitted up to 17 levels. The SELECT option is meaningful and valid only on a $$ control line. Only one SELECT option may be specified on a $$ control line.

INCLUDE — If the INCLUDE option is in effect, CONVERT, upon encountering the SELECT option, uses the selected file as an input file.

EXCLUDE — If the EXCLUDE option is in effect, CONVERT ignores the SELECT option.

The purpose of the INCLUDE and EXCLUDE options is to allow the user to control the performance of the select options while not forcing him to disregard:

1. Other options on the same $$ control line.

2. All $$ control lines.

The INCLUDE option is the default option for the JRN command. The EXCLUDE option is the default option for the JPRINT, JPUNCH, DISPLAY, and CONVERT commands.

Specialized Options

The "specialized" options are a class of options completely distinct and separate from all preceding options. The "specialized" options are unlike other options in that they take effect only when all input files have been read, converted, and closed; i.e., after the output file has been completely generated. All other options, of course, are meant to be used when the output file is in the process of being generated.

As a result of the inherent differences between these classes of options, they are handled differently by the CONVERT subsystem. These differences are discussed under "Methods of Specifying Options", below.

JOUT	The JOUT option is applicable only to the JRN command. This option results in all implied files being saved so that they may be examined using the JOUT subsystem.
ROUT(xx)	The ROUT option is applicable to the JRN, JPRINT, and JPUNCH commands. This option causes the implied files generated by the program execution to be directed to the specified two-character remote station. Only one ROUT entry is permitted.
WAIT	The WAIT option is applicable to the JRN, JPRINT,and JPUNCH commands. This option causes the user to wait until the completion of the spawned job in the batch environment. The wait period may be broken out of by hitting the break key. When the job completes execution, the user is informed of the job's termination status and, if the JOUT option is in effect, the JOUT subsystem is invoked.
TALK	The TALK option is applicable only to the JRN command. This option implies that the batch job includes execution of a program containing conversational (direct access) input/output. This option causes the user's terminal to be placed in direct access connection with the submitted program (by SNUMB) following its submission to the batch environment. When the job completes execution, the user is informed of the job's termination status and, if the JOUT option is in effect, the JOUT subsystem is invoked.
URGC(xx)	The URGENCY option is applicable only to the JRN command. This option indicates that the user wishes to assign initial urgency xx to the spawned batch job. If the assigned urgency is greater than the maximum allowed for the user, he is given the message ILLEGAL URGENCY and the batch job is not spawned. If xx is not specified, maximum allowable urgency is automatically assigned.

COPY(nn) The COPY option is applicable only to the JPRINT and JPUNCH commands. This option causes the generation of nn multiple copies of the listing or punched deck. The maximum number of copies that can be produced from a single JPRINT/JPUNCH job is 13.

IDENT(info) The IDENT option is applicable to the JPRINT and JPUNCH commands. This option allows the user to minimize the subsystem/user interface involved in the use of the JPRINT/JPUNCH commands. When the IDENT option is present, the normal question/answer sequence of

```
$ IDENT? response
```

is bypassed. The information presented as the IDENT option argument is used instead of the user-response to the question.

MONITOR The MONITOR option is applicable to the JPRINT, JPUNCH, and JRN commands. This option allows the user to monitor or track the status of his spawned job as it is executed in the batch environment. When the job completes execution, the user is informed of the job's termination status and, if the JOUT option is in effect, the JOUT subsystem is invoked.

DIRECT The DIRECT option is applicable to the JRN, JPRINT, and JPUNCH commands. If the DIRECT option is given on the command line, it overrides any JOUT or ROUT option which the user has placed on a $$ control line. This option allows the user who, for instance, usually specifies the JOUT option to place it on a $$ control line. He can then override it without being required to change his $$ control line.

DISMISS The DISMISS option is applicable only to the JRN command. If the DISMISS option is given on the command line, it overrides any TALK, WAIT, or MONITOR option which the user has placed on a $$ control line. This option allows the user who, for instance, usually specifies the MONITOR option to place it on a $$ control line. He can then override it without being required to change his $$ control line.

The ROUT, JOUT, and DIRECT options are mutually exclusive. The MONITOR, TALK, WAIT, and DISMISS options are also mutually exclusive. Mutually exclusive options are a group of options for which only one member of the group of options may be in effect. If the user attempts to give two mutually exclusive options in the options field of the command line or on a $$ control line, an error message is given. The handling of these options when they are given sequentially is discussed under "Methods of Specifying Options", below.

PARENTHETICAL LINE/SEQUENCE NUMBERS

Only certain portions of an input file are copied to the output file if parenthetical line numbers or sequence numbers are appended to the file descriptor. When parenthetical line numbers are given, CONVERT copies only those records whose line numbers are specified by the parenthetical input. Parenthetical line numbers should be given only when a file is line-numbered and all line numbers are in ascending order. If parenthetical line numbers are given for an input file and the file is found to contain records without line numbers, then an error message is issued and processing of the input file is terminated.

All records are implicitly assigned sequence numbers. A record's sequence number identifies a record's position within a file. For instance, the first record of a file has a sequence number of 1 while the fifth record of a file has the sequence number 5. The special character "#" is used to indicate that the parenthetical numbers are to be interpreted as sequence numbers rather than line numbers. The "#" character may be placed before any or all of the numbers in a parenthetical expression. Line and sequence numbers cannot be mixed.

A hyphen between two numbers indicates a range of numbers whereas a comma indicates two distinct numbers and/or ranges. Only specified ranges and numbers are copied to the output file. If the first number of a range is empty, zero is assumed. If the second number of a range is empty, 99999999 is assumed.

Parenthetical line and sequence numbers can be given in any order. They do not need to be in ascending order. The order in which records are copied is the order in which they are given in the parenthetical input.

Examples

```
filedescriptor(90,310)          line numbers 90 and 310
filedescriptor(90-310)          line numbers 90 through 310
filedescriptor(70,90-310,50)    line numbers 70, 90 through 310, and 50
filedescriptor(#90-310)         sequence numbers 90 through 310
```

METHODS OF SPECIFYING OPTIONS

All of the options can be given by the user in any or all of four different ways. The four methods of specifying options are as follows:

Method 1 — Options may be given in the options field of the command line.

Example: CONVERT INFILE:MOVE,VERIFY

Method 2 — Options can be given within the parenthetical input field following input file description. The options must follow the parenthetical line/sequence numbers, if there are any, and must be separated from them by a colon.

Example:
```
CONVERT INFILE(10-100:MOVE,VERIFY)
CONVERT INFILE(:MOVE,VERIFY)
```

Method 3 — Options can be given on $$ control lines embedded in the input files.

Example:
```
$$ MOVE,VERIFY
$$ SELECT(INFILE),TAB(#),RESE
```

Method 4 — Options can be given within the parenthetical input field following an input file description that is the argument for the select option.

Example:
```
$$ SELECT(INFILE(10-100:MOVE,VERIFY))
$$ SELECT(INFILE:MOVE,VERIFY)
```

The method used to specify the "specialized" options is unimportant. Once a "specialized" option is specified, it overrides all subsequent declarations of the same or conflicting options. Thus, a declaration of a "specialized" option on the command line overrides similar declarations on $$ control lines.

The method used to specify all other options is very important. In general, the options that are given in the options field (Method 1) are construed to be the default options that are initially in effect for every input and selected file.

Options that are given on an embedded $$ control line (Method 3) but not within a select option argument (Method 4) apply to the input field containing the $$ control line. Thus, the options given on a $$ control line of one input file cannot affect the processing of subsequent input files.

Method 3 options override any previously set conflicting options for the file and add to any previously set non-conflicting options for the file. The options in effect for a file can be changed as many times as desired by simply inserting in the file the necessary $$ control lines. There in no limit to the number of $$ control card that can be placed in a file. There are also no restrictions as to where $$ control lines can be placed within a file.

In some cases it is desirable to tailor options to the requirements of a specific file but there is no means of placing a $$ control line within the file. It is for this case that options are allowed within the parenthetical field following a file description (Method II and Method IV options). Placement of options in the parenthetical field following a file description is equivalent to placing the options on a $$ control line at the beginning of the file.

The method of interpreting and handling options that was described above is considered to be ideal for all options except the INSE and RESE options. These two options are handled by the same rules as the other options, but application of the rules produce results that can be difficult to decipher.

Whenever the INSE or RESE options are encountered, line numbering of the output records begins with the i and j values specified by the most recent INSE or RESE. This means that when the user concatenates several input files using CONVERT, the output file will not usually be line numbered in strictly ascending order.

Instead, there will be breaks in the line sequencing of the output file that correspond to the points of concatenation of the input files. This happens because INSE and RESE options are construed, like all other options, to take effect anew each time a new input file is processed. Also, this result occurs regardless of whether the options are specified on the command line or on a $$ control line in each file.

When the user desires to concatenate several files and produce an output file with line numbers in strictly ascending order, the user should issue the CONVERT command a second time or, if the output file is ASCII, issue the RESEQUENCE command. An example of the proper format for the second CONVERT command is as follows:

```
CONVERT OUT-FILE = OUT-FILE :SAME,RESE
```

Option conflicts within the same line (either command or $$ control) result in an error message being given. Special options that are meaningful to some commands but not others are ignored by those commands to which they are meaningless. However, if an option is specified which has no meaning to any of the commands, an error message is issued.

Conversion From APRINT (Media Code 7)

When reading an APRINT record, CONVERT recognizes the special control characters described under "Special Control Characters Recognized by the ASCII Printer", below.

Upon encountering any of the special characters, the CONVERT subsystem performs the editing functions equivalent to those performed by an ASCII printer. For instance, when CONVERT encounters the horizontal tab sequence, it replaces it with the appropriate number of blanks. When CONVERT encounters the vertical tab sequence, it replaces it with a carriage return or a number of line feeds, whichever is appropriate.

The only instance where CONVERT cannot simulate the printer is when it encounters the character sequence calling for a slew to a VFU tape position. When CONVERT encounters such a sequence, it interprets it as a slew to top of page. This sequence is replaced by the appropriate number of line feeds. The number of line feeds is equal to the difference between the present page position and the maximum number of lines to a page for the particular site.

Conversion To APRINT (Media Code 7)

When converting a record to APRINT format, CONVERT recognizes and interprets:

1. The special control characters line feed, carriage return, and back space.

2. Special control characters. CONVERT replaces line feeds and carriage returns with the appropriate vertical tab character sequences. CONVERT replaces all of the back spaces in a line with sequences of carriage returns and overwrite lines. CONVERT attempts to minimize the number of overwrite lines and the length of overwrite lines. When generating overwrite lines, CONVERT takes into account and compensates for the effects on the printer of all of the special control characters as well as its own generated carriage return and line feed equivalences.

An example of what CONVERT would do with a line follows:

INPUT LINE: ABC[bs][bs][bs]___ lf DEF[bs]_GHIJ

OUTPUT LINE: ABC[vt][null]___[vt][soh]sssDEFGHIJ[vt][null]sssss_

PRINTED LINES: ABC
DEFGHIJ

where bs = the ASCII back space character (ASCII 010)

lf = the ASCII line feed character (ASCII 012)

vt = the ASCII vertical tab character (ASCII 013)

null = the ASCII character 000

soh = the ASCII character 001

s = the ASCII space character (ASCII 040)

Conversion From PRINT (Media Code 3)

When reading a PRINT record, CONVERT recognizes the following special control characters.

Control Char. Sequence			Action
Char. 1	Char. 2	Char. 3	
77	11xxxx		None, unless char. 2 is octal 77
77	17	xx	Case shift. Prints char. 3
77	77	17	No case shift. Prints a ?
77	77	20	No case shift. Prints a blank
77	77	77	No case shift. Prints !
77	00xxxx		Feeds xxxx lines by countdown
77	01xxxx		Feeds to xxxx on a VFU tape
77	20		Feeds to top of page by VFU tape
77	10xxxx		Insert xxxx spaces in the print line. Insert these spaces in multiples of 8.
17			Ignore this character. Its position should be occupied by the char. immediately following it.

NOTE: The six-character strings in the "Char. 2" column represent binary numbers.

Upon encountering these special characters, CONVERT performs editing functions equivalent to those performed by a typical printer. The only instances where CONVERT cannot simulate the printer is when it encounters the character sequence calling for a slew to a VFU tape position. When CONVERT encounters such a sequence it interprets it as a slew to top of page. This sequence is replaced by the appropriate number of line feeds. The number of line feeds is equal to the difference between the present page position and the maximum number of lines to a page for the particular site.

Conversion To PRINT (Media Code 3)

When converting a record to PRINT format, CONVERT first converts the record to APRINT format (see "Conversion To APRINT") above. Once it has the APRINT format, CONVERT replaces all of the editing characters for ASCII printers with the equivalent editing characters for a BCD printer.

RESPONSE TO ERRORS

The action taken by CONVERT when it detects an error depends upon:

1. The nature of the error.

2. When/where CONVERT encounters the error.

For example, if CONVERT detects duplicate options in the options field of the command line, it immediately terminates execution. If CONVERT detects duplicate options on a $$ control line, it immediately terminates execution of the file containing the line but continues to process other input files. The one constant in all of CONVERT error handling is that whenever an error is encountered an error message is issued. The manner in which CONVERT handles various classes of errors is enumerated below.

1. If CONVERT detects an error on the command line, it terminates execution regardless of the nature of the error.

2. If CONVERT detects an error in processing a record, it discards the record but continues to process the input file containing it. Examples of errors detected in processing records are:

   a. Records which are too large.

   b. Records containing non-ASCII characters.

3. If CONVERT detects an error, which by its nature is applicable to an entire input file, it discontinues the processing of that input file but it continues to process subsequent input files. Examples of such an error are:

   a. An attempt to read from a random file.

   b. An attempt to read from a null file.

   c. An attempt to read from a non-existent file.

4. If CONVERT detects an error in writing records to the output file, it terminates execution under all circumstances. Such as error occurs, for example, when a user attempts to store too much data in a file with insufficient file space.

If an error of any sort is detected, JRN will not spawn a batch job. The nature of the error is immaterial. JRN, upon encountering a non-fatal error, continues to process input files, but it terminates just prior to the point where the batch spawn takes place. JRN is the only CONVERT command which does not go on to a normal termination upon encountering a non-fatal error.

SPECIAL FEATURES OF THE JPRINT COMMAND

The JPRINT command begins the listing of each input file at the top of a printer page. On the page prior to that page JPRINT gives:

1. The complete catalog/file description of the file.

2. The tab characters and settings that are initially in effect for the file.

   Because embedded $$ control lines can change the tab characters and setting for a file, the initial tab characters and settings can quickly become obsolete. Since it is felt that most JPRINT users wish to see the embedded $$ control lines that change tab characters or otherwise affect the processing of files, all $$ control lines are printed out in the listing as they occur.

If the MOVE, RESEQUENCE, or INSERT options are in effect, JPRINT produces card image records with line numbers and labels, if appropriate, in columns 73-80. This feature is provided only to make JPRINT functionally similar to the old BPRINT command.

SPECIAL FEATURES OF THE JRN COMMAND

The JRN command checks all $ control card records to see if the following cards are present:

1. $ ASCII
2. $ ENX
3. $ PRMFL
4. $ SELECT
5. $ SELECTD

From the time JRN encounters the $ ASCII card until it encounters the $ ENX card, all records except $ control cards are written as ACARD records. JRN always writes $ control cards as CARD records. The $ ASCII and $ ENX cards are deleted by JRN.

When JRN encounters the $ PRMFL, $ SELECT, and $ SELECTD cards, it removes the referenced file from the AFT if:

1. The referenced file is in the AFT.

2. The file in the AFT is not a temporary file.

3. The file is not being used by JRN.

SPECIAL CONTROL CHARACTERS RECOGNIZED BY THE ASCII PRINTER

Upon receiving a print command, the printer subsystem begins requesting a print line of ASCII oriented 9-bit data characters from memory. There are several characters that have unique meaning when sent to the printer subsystem as part of the print line of data, and are treated as described in the following paragraph.

Horizontal Tabulate Control Character

The HT character does not occupy a position on the print line. The character following this control character identifies the address of the last column to be tabbed over.

The number is contained in the low-order bits and must be equivalent to 127 or less.

If N is the column to be skipped to, then:

128 >N> current column position.

In other words, the tab function is valid from left to right on the print line only, and an address of zero designates column 1, an address of 1 designates column 2, an address of 127 designates column 128.

Horizontal Column Skip Control Character

The US character does not occupy a position on the print line. The character following this control character identifies the number (in binary) of columns to be skipped and left blank.

The number is contained in the seven low-order bits and may have any value from 0 to 127.

Vertical Line Slew Control Character

The VT character does not occupy a position on the print line. The character following this control character identifies the number of lines paper is to advance (in binary). The printer also interprets the VT character as an implied carriage return. The number is contained in the seven low-order bits and may have any value from 0 to 127.

Slew to VFU Configuration Control Character

The FF character does not occupy a position on the print line. The character following this control character identifies the code on the VFU loop to be matched. Only the four low-order bits are significant. If the four low-order bits are 0000, paper is advanced to "Top of Page".

Shift Out Control Character

The SO control character does not occupy a position on the print line. All following control characters, except the "Shift In" control character, are printed regardless of normal interpretation.

Shift In Control Character

The SI control character does not occupy a position on the print line. All following characters are interpreted in the normal manner until a shift out is received again.

A "Shift In" received without a prior "Shift Out" does not effect normal operation, and is treated as an ignore character.

Shift Out/Shift In Combination

The combination of these two functions provides the ability to print normally unprintable characters.

Delete Character

The Delete character does not occupy a position on the print line and is ignored by the printer. The remaining print line characters are left justified.

The Delete character retains its function even after a "Shift Out" is received.

Space Character

The Space character occupies a character space on the print line and is not printable.

The Space character remains unprintable even after a "Shift Out" is received.

Data and Control Characters, Bit Structure

1. Printable data characters fall within the range of characters bounded by the Space and Delete characters; i.e., octal 41 through octal 176, inclusive.

2. If the subsystem is in the "Shift Out" ASCII edit mode, all characters within the range (000-037) are considered printable, except "Shift In". However, if a character code is received which is not represented in the train image buffer, the character is treated as a space character.

3. If the subsystem is in the normal ASCII edit mode, all characters within the range (000-037) are treated as ignore characters, except those previously defined as control characters for this subsystem.

EXAMPLES OF CONVERT USE

The following examples illustrate the use of CONVERT.

1. Retrieve the contents of saved file named FILEA and write contents onto the current file.

   CONVERT FILEA

   The text of FILEA is copied as is onto the current file in ASCII format. Line numbers, if present, are treated as text. There is no case shifting, tabulation, or reporting. This usage is equivalent to OLD FILEA except that non-ASCII files are accepted.

2. Save the contents of the current file on file named FILEA.

   CONVERT = FILEA

   The output file format is ASCII. Line numbers, if present, are copied as text. This usage is equivalent to RESAVE FILEA, except that if FILEA does not exist, CONVERT will create a temporary file of that name.

3. Save the current file on FILEA in card format.

   CONVERT = /FILEA: CARD

   If FILEA does not exist, CONVERT will create it as a permanent file.

4. List the contents of FILEA on the terminal.

   CONVERT FILEA = **

   This usage is identical to LIST FILEA except that non-ASCII files are accepted.

5. Resequence line numbers on the current file.

   CONVERT: R(0020,20)

   This usage is equivalent to RESE 20,20 (except for BASIC files). Minimum width of each line number is four digits.

6. Insert line numbers on the current file.

   CONVERT: I

   This usage is equivalent to RESE#.

7. Convert alphabetics on the current file to lowercase.

   CONVERT: LOWER

8. Replace tab characters : and > on the current file by the indicated number of blanks.

   CONVERT T(:;>,7,13),M

   The M option is required to cause tab stops to be calculated relative to the text and not relative to the beginning of the line numbers. Because no tab settings are specified for the : character, 8,16,32,73 are assumed.

9. Find out how many lines are on the current file.

```
CONVERT: V
```

10. Retrieve all lines between the 15th and 44th inclusively on file named SOURCE and insert the : tab character, with default settings of 8,16,32,73. Copy to the current file (the assumption is that SOURCE has no line numbers).

```
CONVERT SOURCE (#15-44):U
```

11. Save the lines numbered 10 to 100 from CATA/FILEA and lines 150 to 200 from the current file on file named SAVEFILE in card format, without line numbers.

```
CONVERT CATA/FILEA(10-100);*(150-200)=SAVEFILE:CARD,S
```

12. Save the current file on FILEA in card format with line numbers moved to columns 73 to 80 and with the label ABC on those lines with line numbers between 10 and 100 inclusive.

```
CONVERT = FILEA: CARD, M, L (ABC(10-100))
```

13. Concatenate the contents of files FILEA, FILEB, and FILEC, strip all line numbers, expand tabs, save in BCD variable length records on SAVEFILE, and report the results.

```
CONVERT FILEA,FILEB,FILEC=SAVEFILE:S,V,BCD,

MORE? T(:,10,20,30,40,50)
```

14. The program contained in FILEA is passed to the batch system. Tabs are expanded and line numbers are stripped. The tab character is : and the settings are 8,16,32, and 73.

```
JRN   FILEA: T,S
```

15. The contents of FILEA are printed at a high-speed printer. Tabs are expanded and line numbers are stripped. The tab character is : and the settings are 8,16,32, and 73.

```
JPRINT   FILEA(: T,S)
```

16. The contents of FILEA are punched into a card deck. Tabs are expanded and line numbers are stripped. The tab character is : and the settings are 8,16,32, and 73.

```
JRN   FILEA   and   FILEA =   $$ S,T
                              .  .  .
                              .  .  .
                              .  .  .
```

17. The program contained in INPUT is passed to the batch system along with the contents of the files HAGAN/SCHD and HAGAN/ALTER. INPUT has its line numbers stripped and its tabs expanded where the tab characters is : and the settings are 8,16,32, and 73. HAGAN/SCHD has its line numbers left ASIS and its tabs expanded where the tab character is ; and the settings are 8,16,32, and 73. HAGAN/ALTER has its line numbers stripped and no tabs are expanded. A report is given for all three files which gives the number and type of records obtained from each file. The program is given JOUT disposition and the execution of the program is monitored.

```
JRN   INPUT

INPUT =    10$$S,T,J,V,MONI
           20$:IDENT:M24GPCX13, H HAGAN , STATION G
           30$:OPTION:NOSETU,NOGO
           40$:LOWLOAD
           50$:GMAP:COMDK
           60$:PRMFL:**,R,R,FRICK/SR5KMAC
           70$$ SELECT(HAGAN/SCHD(:T(;),V))
           80$$ SELECT (HAGAN/ALTER(:V,S))
           90$:LIMITS:,,,25000
           100$:PRMFL:K*,R/W,S,HAGAN/SCHD-COM
           110$:PRMFL:C*,R/W,S,HAGAN/SCHD-OBJ
           120$:ENDJOB
```

18. The contents of FILEA are printed at the user's terminal. Line numbers are shown in colums 1 through 8 and text begins in column 9. Tabs are expanded. The tab characters are % and \. The settings for % are 10 and 20. The settings for \ are 30, 40, and 50.

```
DISPLAY   FILEA :M, T(%,10,20;\,30,40,50)
```

APPENDIX A

INTERFACE USAGE EXAMPLES

The following sample program illustrates the use of interrelated time sharing subsystems and batch programming features. The program is submitted by means of the time sharing CARDIN subsystem. Direct conversation between the program and the user's terminal is then initiated, and use is made of two conversational batch dimension features--Conversational Debug Routine (RBUG) and conversational I/O extensions to File and Record Control. Text within brackets is not part of the program but has been added to illustrate particular features.

The program, submitted under CARDIN, makes use of conversational I/O extensions to File and Record Control.

```
0100$;IDENT;VXEOO,JDOE
0200$;OPTION;FORTRAN
201$,USE;.RTYP [Required when $ DAC cards are present.]
0300$,FORTRAN;NDECK
400#2;WRITE(6,3)
0500#3;FORMAT(27HPROGRAM TO CALCULATE RECOIL)
0600#1;WRITE(6,4)
0700#4;FORMAT(9HRIFLE WT.)
0800;READ(5,5)WR
0900#5;FORMAT(F6.2)
1000;WRITE(6,7)
1100#7;FORMAT(10HBULLET WT.)
1200;READ(5,5)WB
1300#9;WRITE(6,10)
1400#10;FORMAT(8HVELOCITY)
1500#11;READ(5,5)VB
1600;WRITE(6,13)
1700#13;FORMAT(10HPOWDER WT.)
1800;READ(5,5)WP
1900;X=WB*VB+4700.*WP
2000;Y=7000.*WR
2100;Z=WR/64.4
2200;E=Z*(X/Y)**2
2300#15;WRITE(6,16)E
2400#16;FORMAT(8HENERGY= ,F6.2,9H FT. LBS.)
2500;GO TO 1
2600;END
2700$;EXECUTE
2800$;DAC;05
2900$;DAC;06
3000$;ENDJOB
```

The program is then formatted for legibility.

```
*PRINT
CARD FORMAT ?
NORM(;)

09/10/69    09.69

0100    $     IDENT   VXE00,JDOE
0200    $     OPTION  FORTRAN
201     $     USE     .RTYP
0300    $     FORTRAN NDECK
400     2     WRITE(6,3)
0500    3     FORMAT(27HPROGRAM TO CALCULATE RECOIL)
0600    1     WRITE(6,4)
0700    4     FORMAT(9HRIFLE WT.)
0800          READ(5,5)WR
0900    5     FORMAT(F6.2)
1000          WRITE(6,7)
1100    7     FORMAT(10HBULLET WT.)
1200          READ(5,5)WB
1300    9     WRITE(6,10)
1400    10    FORMAT(8HVELOCITY)
1500    11    READ(5,5)VB
1600          WRITE(6,13)
1700    13    FORMAT(10HPOWDER WT.)
1800          READ(5,5)WP
1900          X=WB*VB+4700.*WP
2000          Y=7000.*WR
2100          Z=WR/64.4
2200          E=Z*(X/Y)**2
2300    15    WRITE(6,16)E
2400    16    FORMAT(8HENERGY= ,F6.2,9H FT. LBS.)
2500          GO TO 1
2600          END
2700    $     EXECUTE
2800    $     DAC       05
2900    $     DAC       06
3000    $     ENDJOB
```

[The program is then passed to the batch system for processing; the TALK option permits direct-access connection.]

```
*RUN
  SNUMB # 0165T
CARD FORMAT, DISPOSITION ?
NORM(;),TALK
PROGRAM TO CALCULATE RECOIL
RIFLE WT.
05?8.5
BULLET WT.
05?150
VELOCITY
05?3200
POWDER WT.
05?58
ENERGY=320.06 FT. LBS.
RIFLE WT.
05? <-- Carriage return; null response.
**************************************************
**************************************************ERROR
  TRACE OF CALLS IN REVERSE ORDER
  CALLING      ID   ABSOLUTE     ARGUMENT          ARGUMENT
  ARGUMENT          ARGUMENT     ARGUMENT
  ROUTINE      #    LOCATION        #1                #2
     #3               #4           #5
  .FEOF.       19   035114    000000000042
                                            ERROR
  .FRDD.      954   036771
                                            ERROR
  ......        5   037556    000000000005       352606330255
                                             ERROR
  END OF FILE READING    FILE CODE 05 OPTIO
  OPTIONAL RETURN NOT REQUESTED013068
**************************************************
**************************************************ERROR
CLOSING FILE 05
CLOSING FILE 06

ACTIVITY TERMINATED
NORMAL TERMINATION
```

A $ USE RBUG card is substituted for $ USE .RTYP to initiate the RBUG subroutine, and breakpoints are inserted. The program is then formatted in its new version.

```
*201$;USE;RBUG
*202$;DUMP;......
*203;DEBUG;2/(BREAKPOINT)
*204;DEBUG;15/(BREAKPOINT)
*PRINT
CARD FORMAT ?
NORM(;)

09/10/69   09.85

0100    $   IDENT   VXE00,JDOE
0200    $   OPTION  FORTRAN
201     $   USE     RBUG
202     $   DUMP    ......
203         DEBUG   2/(BREAKPOINT)
204         DEBUG   15/(BREAKPOINT)
0300    $   FORTRAN NDECK
400     2   WRITE(6,3)
0500    3   FORMAT(27HPROGRAM TO CALCULATE RECOIL)
0600    1   WRITE(6,4)
0700    4   FORMAT(9HRIFLE WT.)
0800        READ(5,5)WR
0900    5   FORMAT(F6.2)
1000        WRITE(6,7)
1100    7   FORMAT(10HBULLET WT.)
1200        READ(5,5)WB
1300    9   WRITE(6,10)
1400    10  FORMAT(8HVELOCITY)
1500    11  READ(5,5)VB
1600        WRITE(6,13)
1700    13  FORMAT(10HPOWDER WT.)
1800        READ(5,5)WP
1900        X=WB*VB+4700.*WP
2000        Y=7000.*WR
2100        Z=WR/64.4
2200        E=Z*(X/Y)**2
2300    15  WRITE(6,16)E
2400    16  FORMAT(8HENERGY= ,F6.2,9H FT. LBS.)
2500        GO TO 1
2600        END
2700    $   EXECUTE
2800    $   DAC     05
2900    $   DAC     06
3000    $   ENDJOB
```

[The program is again passed to the batch system, along with the TALK option. Control of the program is obtained at breakpoints, interrogations are made, and the program is then permitted to continue and run to termination.]

```
*RUN
  SNUMB #0166T
CARD FORMAT, DISPOSITION ?
NORM(;),TALK
***ROUTINE  ......  LOC 2          COUNT    000001
???R
PROGRAM TO CALCULATE RECOIL
RIFLE WT.
05?8.5
BULLET WT.
05?150
VELOCITY
05?3200
POWDER WT.
05?58
***ROUTINE  ......  LOC 15         COUNT    000001
???FWR  CHECK ANSWER
WR        0.85000000E 01
???FE  I CHECKED THE WRONG THING
E         0.38320059E 05
???FWB  CHECK BULLET WEIGHT
WB        0.15000000E 04
???R#2 BULLET WEIGHT FUNNY, TRY AGAIN WITH DECIMAL POINTS
***ROUTINE  ......  LOC 2          COUNT    000002
???R    BEGIN NORMAL RUN
PROGRAM TO CALCULATE RECOIL
RIFLE WT.
05?8.5  8 AND 1/2 POUNDS
BULLET WT.
05?150.0  150 GRAINS
VELOCITY
05?3200.0   FEET PER SECOND
POWDER WT.
05?58.0  GRAINS
***ROUTINE  ......  LOC 15         COUNT    000002
???FE  GET A PEEK AT ANSWER
E
???R  LOOKS GOOD
ENERGY= 21.11 FT. LBS.
RIFLE WT.
05?7.5
BULLET WT.
05?150.
VELOCITY
05?2175
POWDER WT.
05?31.0  I MADE A MISTAKE.  NO DECIMAL ON VELOCITY
***ROUTINE  ......  LOC 15         COUNT    000003
???R#2  TRY AGAIN.  THIS TIME WITH DECIMAL POINT.
***ROUTINE  ......  LOC 2          COUNT    000003
???R
PROGRAM TO CALCULATE RECOIL
```

```
RIFLE WT.
05?7.5
BULLET WT.
05?150.
VELOCITY
05?2175
POWDER WT.
05?0.  AGAIN FORGOT THE DECIMAL POINT
***ROUTINE  ......  LOC 15          COUNT    000004
???R#2  TRY AGAIN
***ROUTINE  ......  LOC 2           COUNT    000004
???R    TRY A "NORMAL" RUN
PROGRAM TO CALCULATE RECOIL
RIFLE WT.
05?7.5
BULLET WT.
05?150.0
VELOCITY
05?2175.0
POWDER WT.
05?31.0
***ROUTINE  ......  LOC 15          COUNT    000005
???FE   PEEK AT ANSWER
E        0.94112817E 01
???R  ANSWER SEEMS ABOUT RIGHT FOR 30/30 WITH LIGHT LOAD
ENERGY=   9.41 FT. LBS.
RIFLE WT.
05?7.5          TRY 30.06 TYPICAL LOAD
BULLET WT.
05?180.0
VELOCITY
05?2505.0
POWDER WT.
05?45.0
***ROUTINE  ......  LOC 15          COUNT    000006
???R     LET IT GO NORMALLY
ENERGY=  18.54 FT. LBS.
RIFLE WT.
05?0.
BULLET WT.
05?0.
POWDER WT.
05?0.
       DIV CHECK    AT LOCATION   037651
      EXP OVERFLO   AT LOCATION   037400
***ROUTINE  ......  LOC 15          COUNT    000007
???T LET THE PROGRAM QUIT NORMALLY
**EXIT
CLOSING FILE 05
CLOSING FILE 06

ACTIVITY TERMINATED
NORMAL TERMINATION

*BYE
```

INDEX

$ DAC
$ DAC 6-3
$ DAC 6-1

$ IDENT
$ IDENT card 2-6

$ IDENT?
$ IDENT? 3-4
$ IDENT? 2-14

$ SELECTA
$ SELECTA Card Option 2-16

$ USE
$ USE .RTYP 6-1
$ USE .RTYP A-4
$ USE RBUG A-4
$ USE RBUG 5-1

$ USERID
$ USERID control card 7-1

.RTYP
$ USE .RTYP 6-1
$ USE .RTYP A-4

ACTIVITY
ACTIVITY 7-3

APRINT
APRINT 2-14
APRINT 2-6

ASCBCD
ASCBCD 2-5
ASCBCD Question/Answer Sequence 2-12

ASCII-TO-BCD
ASCII-to-BCD conversion 2-2

ASIS
ASIS 2-7
ASIS 2-12

BACK
BACK 3-5

BATCH
BATCH 3-4

BCDASC
BCDASC 2-6
BCDASC Question/Answer Sequence 2-13

BLOCK TO BE READ
BLOCK TO BE READ 4-2

BPRINT
BPUNCH and BPRINT 2-6
BPUNCH or BPRINT Question/Answer Sequence 2-14

BPUNCH
BPUNCH and BPRINT 2-6
BPUNCH or BPRINT Question/Answer Sequence 2-14

BREAKPOINT
breakpoint 5-1

BYE
BYE 3-6

CALLS
I/O CALLS 6-2

CARD
$ IDENT card 2-6
$ SELECTA Card Option 2-16
$ USERID control card 7-1

CARDIN
CARDIN Command Language 2-3
CARDIN DETAILED OPERATION 2-3
CARDIN FUNCTION 2-1
CARDIN OPERATION 2-1
CARDIN SUBSYSTEM 2-1
TALK option of CARDIN 2-1

CHARACTER
standard character 2-12
multiple tab characters 2-12
Tab characters 2-2
tab characters 2-12

CLOSE
CLOSE 6-3

CODE
CODE 3-6
Line codes 3-2

COMMAND
CARDIN Command Language 2-3
Special command verbs 3-1

CONTROL
$ USERID control card 7-1
FILE AND RECORD CONTROL I/O CONSIDERATIONS 6-3

CONTROL CARDS
CONTROL CARDS FOR FILE AND RECORD CONTROL I/O 6-1

CONVERSATIONAL
CONVERSATIONAL DEBUG 5-1
CONVERSATIONAL FILE AND RECORD INPUT/OUTPUT 6-1

CONVERSION
ASCII-to-BCD conversion 2-2

COPY
copy file 4-4

COPY FUNCTION
FDUMP Copy Function Details 4-4

DEBUG
CONVERSATIONAL DEBUG 5-1

DESCRIPTION
FDUMP OPERATIONAL DESCRIPTION 4-1
JOUT OPERATIONAL DESCRIPTION 7-2
RBUG GENERAL DESCRIPTION 5-1
SCAN OPERATION DESCRIPTION 3-2

DIRECT
DIRECT 7-3

DISPOSITION
DISPOSITION Portion 2-9

DONE
DONE 3-6

DUMP
file dump 4-1

EDIT
EDIT 3-6
EDIT? 3-3

EPRINT
EPRINT 7-3

ERROR
ERROR 3-5
FDUMP Error Indication and Replies 4-3

EXAMPLES
INTERFACE USAGE EXAMPLES A-1

FDUMP
FDUMP 2-7
FDUMP Copy Function Details 4-4
FDUMP Error Indication and Replies 4-3
FDUMP FUNCTION 4-1
FDUMP OPERATIONAL DESCRIPTION 4-1
FDUMP SUBSYSTEM 4-1
FDUMP functions 4-2

FILE
CONVERSATIONAL FILE AND RECORD INPUT/OUTPUT 6-1
copy file 4-4
FILE AND RECORD CONTROL I/O CONSIDERATIONS 6-3
FILE NAME? 4-1
file dump 4-1

FILE AND RECORD CONTROL
File and Record Control routines 6-1

FILE-BUILDING
file-building 2-1

FILES
permanent files 4-1

FIND
FIND 3-3

FIRST
First Line Reformatting Information 2-15

FLAG
FLAG 3-5

FORM?
FORM? 3-2

FUNCTION
CARDIN FUNCTION 2-1
FDUMP FUNCTION 4-1
FUNCTION? 4-2
JOUT FUNCTION 7-1
SCAN FUNCTION 3-1

GET
GET 6-3
GET 6-2

I/O
FILE AND RECORD CONTROL I/O CONSIDERATIONS 6-3
I/O CALLS 6-2

INPUT/OUTPUT
CONVERSATIONAL FILE AND RECORD INPUT/OUTPUT 6-1

INSTRUCTIONS
RBUG INSTRUCTIONS 5-2

INTERFACE
INTERFACE USAGE EXAMPLES A-1

JABT
JABT 2-5

JDAC
JDAC 2-4

JOUT
JOUT 2-7
JOUT 2-10
JOUT FUNCTION 7-1
JOUT OPERATIONAL DESCRIPTION 7-2
JOUT SUBSYSTEM 7-1

JSTS
JSTS 2-4

LABELS?
LABELS? 2-12
LABELS? 2-14

LANGUAGE
CARDIN Command Language 2-3

LINE
First Line Reformatting Information 2-15
LINE 3-5
Line codes 3-2
line numbers 2-8

LINE NUMBERS?
LINE NUMBERS? 2-13

LIST
LIST 3-4
LIST 7-3

LITERAL
literal string 3-3

LOAD
LOAD MAP 3-6

MOVE
MOVE 2-8
MOVE 2-12
MOVE 2-13

NAME?
FILE NAME? 4-1

NORM
NORM 2-12
NORM 2-8

OPEN
OPEN 6-2

OPERATIONAL
FDUMP OPERATIONAL DESCRIPTION 4-1
JOUT OPERATIONAL DESCRIPTION 7-2

OPTION
$ SELECTA Card Option 2-16
STAB option A-1
TALK option of CARDIN 2-1

PERMANENT
permanent files 4-1

PRINT
PRINT 3-4
PRINT 2-4
PRINT 7-4
RUN or PRINT Question/Answer Sequence 2-7

PUT
PUT 6-3

QUESTION/ANSWER
ASCBCD Question/Answer Sequence 2-12
BCDASC Question/Answer Sequence 2-13
BPUNCH or BPRINT Question/Answer Sequence 2-14
RUN or PRINT Question/Answer Sequence 2-7

RBUG
$ USE RBUG A-4
$ USE RBUG 5-1
RBUG 5-1
RBUG A-1
RBUG GENERAL DESCRIPTION 5-1
RBUG INSTRUCTIONS 5-2

RECORD
CONVERSATIONAL FILE AND RECORD INPUT/OUTPUT 6-1
FILE AND RECORD CONTROL I/O CONSIDERATIONS 6-3

REFORMATTING INFORMATION
First Line Reformatting Information 2-15

RELEASE
RELEASE 7-4

REM
REM 3-6

ROUTINES
File and Record Control routines 6-1

ROUTXX
ROUT(xx) 2-10

RUN
RUN 2-3
RUN or PRINT Question/Answer Sequence 2-7

SCAN
SCAN 2-7
SCAN 7-4
SCAN FUNCTION 3-1
SCAN OPERATION DESCRIPTION 3-2
SCAN SUBSYSTEM 3-1
SCAN VERBS 3-3

SETTINGS
tab settings 2-12
tab settings 2-2

SNAPSHOTS
snapshots 5-1

SPACE
SPACE 3-4

STAB
STAB option A-1

STANDARD
standard character 2-12

STATION CODE?
STATION CODE? 3-4

STRING
literal string 3-3

STRIP
STRIP 2-12
STRIP 2-8

SUBSYSTEM
CARDIN SUBSYSTEM 2-1
FDUMP SUBSYSTEM 4-1
JOUT SUBSYSTEM 7-1
SCAN SUBSYSTEM 3-1

TAB
multiple tab characters 2-12
Tab characters 2-2
tab characters 2-12
tab settings 2-12
tab settings 2-2

TAB CHARACTERS AND SETTINGS?
TAB CHARACTERS AND SETTINGS? 2-11
TAB CHARACTERS AND SETTINGS? 2-14
TAB CHARACTERS AND SETTINGS? 2-13

TALK
TALK 2-10
TALK A-3
TALK A-5
TALK option of CARDIN 2-1

UNDEFINED
UNDEFINED 3-5

URGCXX
URGC(xx) 2-10

VERBS
SCAN VERBS 3-3
Special command verbs 3-1

WAIT
WAIT 2-9