SECTION XII
Non-Interactive Programming

BASIC/3000 has the capability to enter programs in a non-interactive manner. This section describes how the user may input from a card reader or paper tape, and how he may print output on a line printer or punch it on paper tape. It also describes how to input commands or programs stored on an ASCII file.

Card Reader/Line Printer

If the user has access to a card reader and a line printer, he may punch his BASIC program on cards and input it through the card reader and receive output on the line printer. In addition to the BASIC program cards, he will need a :BASIC command card preceding his program deck. The deck may include any BASIC commands as well as statements.

:BASIC Command

This command causes the MPE/3000 Operating System to invoke the BASIC Interpreter. The :BASIC command has the form:

:BASIC commandfile,inputfile,listfile

Any or all of the parameters may be omitted. Position of parameters is significant so commas must be included when leading parameters are omitted. Each parameter must be an existing ASCII file with the following meaning:

commandfile	BASIC subsystem input; original source of all commands and statements of BASIC program. Default is $STDINX.
inputfile	BASIC program input; contains data input to BASIC through INPUT, ENTER, and LINPUT statements. Default is $STDINX.
listfile	BASIC program output; receives data output from BASIC program. Default is $STDLIST.

$STDINX is the job/session input device. For example, when input is from the card reader, $STDINX is the card reader; when input is from the terminal, $STDINX is the terminal.

$STDLIST is the job/session output device. When the card reader is used for input, the job/session output device is normally the line printer although this may vary between HP 3000 installations. When the terminal is used, the output device is also the terminal.

Examples:

1. Suppose the user inputs his program from the card reader with output to the line printer, but all the data used by the program is stored on an ASCII disc file called IN. He uses the :BASIC command:

   ```
   :BASIC ,IN
   ```

   The comma signals that the commandfile is the default file $STDINX. The default $STDLIST is used for output but since this is at the end, no comma is needed.

Note that this card, if entered as a command from a terminal, would perform the same function if program input and output were both at the terminal with data input from IN.

2. Suppose the user has a disc file named COMMAND which contains a set of BASIC commands, and he wants output on the line printer:

```
:FILE PRINTER;DEV=LP
:BASIC COMMAND,,*PRINTER
```

The file named PRINTER is associated with the line printer by the :FILE command. It is named in the :BASIC command to replace the $STDLIST which, in this case, would have been the terminal. COMMAND is a disc file; the commands and statements of the BASIC program are read from this file.

> EOD Command

If a BASIC card deck contains a RUN command, a >EOD command must follow RUN. When the BASIC program contains an INPUT, ENTER, or LINPUT statement input, the input data is punched on cards placed between the RUN card and the >EOD card.

The >EOD command does not produce the same effect as an End-of-File (EOF) command. For instance, when an INPUT, ENTER, or LINPUT statement encounters an >EOD in columns 1 through 4, a run-time error will occur. Furthermore, READ# or LINPUT# statements are not affected by >EOD. If the user wants to use >EOD command as an EOF command with the READ# and LINPUT# statements, the >EOD must be tested programmatically as follows:

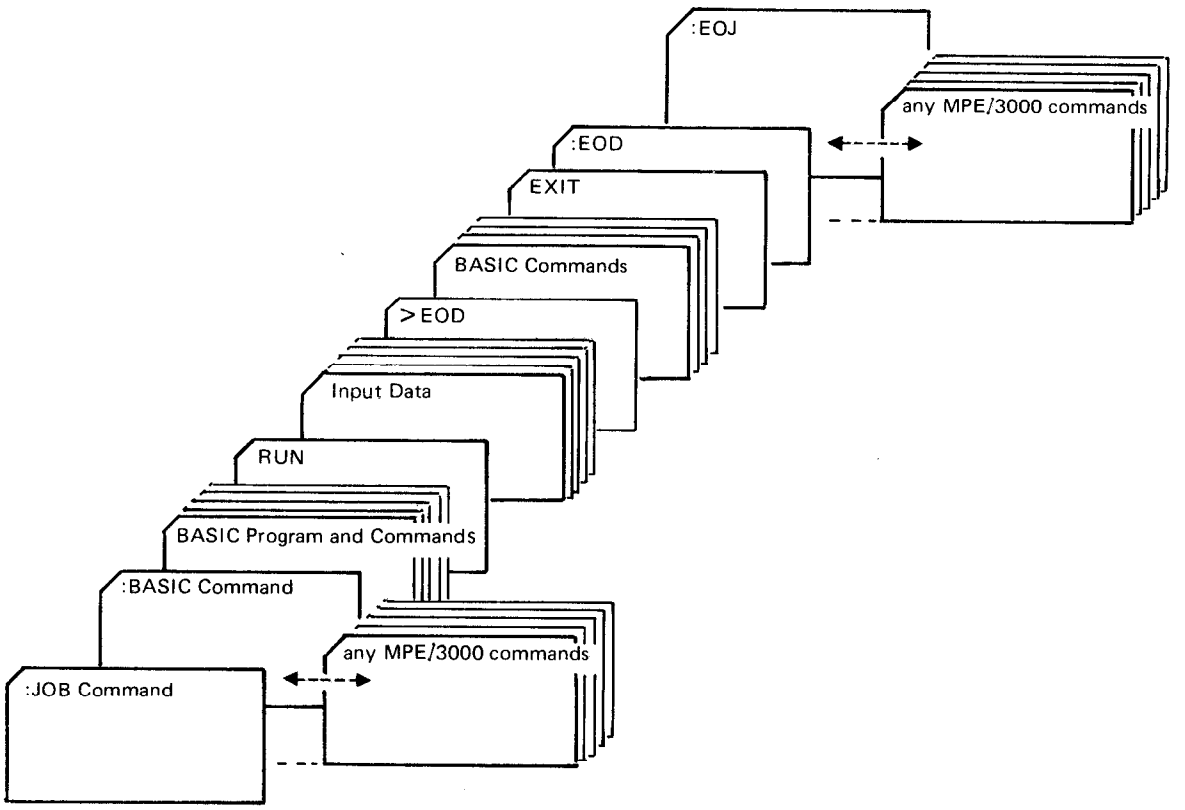
```
READ #1; A$
IF A$(1,4) = ">EOD" THEN . . .
```

Deck Structure

In addition to the :BASIC command and the program deck, the standard MPE/3000 command cards for running a job must be used. These are the :JOB card preceding all other cards in the job, and the :EOJ card that terminates the job. Also, an EXIT card must terminate the BASIC program deck followed by an :EOD card.

Besides these cards, any other MPE/3000 cards needed for the job may be included.

A deck structure for a BASIC program that requests input with INPUT, ENTER, or LINPUT:



If no input data is required, the input data cards are omitted, but >EOD is left. The >EOD card must follow the input data of each run.

Paper Tape

At a terminal with a paper tape punch and reader, the user may write his programs on paper tape and read them from paper tape. Commands are provided by BASIC/3000 that enable the terminal user to prepare and use paper tapes.

PREPARING A PAPER TAPE

A paper tape may be punched on-line using the PUNCH command, or it may be punched off-line.

PUNCH Command

The PUNCH command allows the user to punch a program on paper tape while operating in interactive (on-line) mode at the terminal. Following each line, PUNCH automatically inserts an X-OFF character preceding the carriage return and linefeed. A paper tape prepared by PUNCH should be read with the TAPE command.

Form

PUNCH [first [-last]] [, OUT = asciifile] [,RECSIZE = number] [,NONAME]

where *first* and *last* specify the range of statements to punch, and *asciifile* specifies an output file. Normally, the output file is the standard list file. If neither *first* nor *last* is specified, the entire program is punched. If only *first* is specified, just that statement is punched. Control characters are inserted, as required, to allow reading the punched program back through a tape reader.

Explanation

The PUNCH command is identical to LIST except that the output is preceded and followed by headers and trailers of null characters. If the OUT parameter is omitted, the punched program is listed at the terminal. Otherwise it is output to the specified ASCII file.

If only a portion of the program is to be punched, the first and, optionally, the last lines to be punched are specified. If a maximum record size other than 72 is desired, it can be specified with the RECSIZE parameter. If *NONAME* is specified, the program name is not punched; this is useful when punching programs to be read back with the XEQ command.

Each punched record contains a program statement. PUNCH automatically terminates each record with an X-OFF character. The X-OFF precedes the carriage return on the tape, and linefeed follows the carriage return. The form is:

output record X-OFF carriage return linefeed

Examples:

```
PUNCH 10-200
```

Lines 10 through 200 of the current BASIC/3000 program are punched on paper tape.

```
PUNCH,OUT=AA
```

The entire current program is punched on the ASCII file AA.

```
PUNCH 500,RECSIZE=132
```

Line 500 of the program is punched. A record size of 132 characters is used.

PUNCHING PAPER TAPE OFF-LINE

To prepare a BASIC program on paper tape, the user must:

1. Turn teleprinter control knob to "LOCAL".
2. Turn on the tape punch by pressing the "ON" button on the punch.
3. Type a series of null characters using the "HERE IS" key or control shift P ($@^C$). This punches leading holes on the tape.
4. Type the program as usual following each line with a carriage return.
5. Type a series of null characters using the "HERE IS" key or control shift P ($@^C$). This punches trailing holes on the tape.
6. Turn off the tape punch by pressing the "OFF" button on the punch.

When programs are punched off-line, the H^C and X^C keys may be used for corrections. If X^C is used to delete a line, it must be followed by X-OFF and a carriage return and linefeed.

If the punched tape is to be read by the TAPE command, the user must press the X-OFF character following each line *before* he presses the carriage return. X-OFF (S^C) is a key on the teleprinter keyboard.

READING A PAPER TAPE

Two commands are provided to read paper tapes at a terminal equipped with a paper tape reader; they are TAPE and SPOOL.

TAPE Command

The TAPE command allows the user to read commands, programs, and data through a paper tape reader connected to the terminal. Only tapes that contain an X-OFF character after each record can be read with TAPE. Also, in order to read with TAPE, the terminal must be equipped with a reader that recognizes X-OFF.

Form

TAPE

Explanation

After typing TAPE, turn on the tape reader. The tape will be read until the end of the program or programs on the tape. If the tape contains data to be input, as much data is read as was requested by INPUT, ENTER, or LINPUT, or until the end of the tape.

When through reading from tape, the user returns to terminal mode with the KEY command.

KEY Command

The KEY command returns the terminal user to terminal mode following completion of a tape read using TAPE.

Form

KEY

Explanation

When KEY is typed, the input mode entered with TAPE is terminated and the user is returned to the terminal for further interactive execution or to log off.

SPOOL Command

The SPOOL command is used to read paper tapes that have not been punched with X-OFFs preceding the carriage return and linefeed.

Form

SPOOL

Explanation

After typing SPOOL, turn on the tape reader. When the tape is through, the user types the control key Y^c to terminate the tape. At this point, any error messages are printed at the terminal.

Command Input from Files

If commands or programs are stored on an ASCII file, the XEQ command may be used to cause the BASIC/3000 Interpreter to read this file.

Form

XEQ asciifile ,ECHO

The *asciifile* has been created and contains commands or programs the user needs. If ECHO is specified, the records from the ASCII file are listed on the terminal as they are input.

Explanation

When the XEQ command is entered, the specified file is read and executed until it reaches an end-of-file. Any program input is still read from the *inputfile*. When the end-of-file of the *asciifile* is reached, control returns to the original command file. For instance, if the job was entered on cards with the :BASIC command, control returns to the *commandfile* specified in that command; if the user entered XEQ from the terminal, control resumes at the terminal.

An XEQ command within the XEQ file will close the first file and open a new one.

Examples:

An ASCII file BATCHJOB contains the commands to GET and RUN three programs (AA, BB, and CCC) stored in the user's library. In order to run all three programs at a terminal or in a BASIC card deck, only the command XEQ BATCHJOB is required.

This is illustrated in the examples below, one with the ECHO parameter, one without:

```
>XEQ BATCHJOB,ECHO
GET AA
RUN
AA
END OF AA

GET BB
RUN
BB
END OF BB

GET CCC
RUN
CCC
END OF CCC
```

ECHO causes a list of the commands in BATCHJOB. When each program is run, it outputs a message that it is through. This is part of the individual programs, not BATCHJOB.

The example below without ECHO runs each program consecutively in the same way as the previous example, but it does not list the commands contained in the file BATCHJOB:

```
>XEQ BATCHJOB
AA
END OF AA

BB
END OF BB

CCC
END OF CCC
```