Chapter 14

Auxiliary Input/Output

The auxiliary input/output statements provide a means to connect a file to a unit, inquire about the properties or status of a file or a unit, and change the position of a file.

An (auxiliary) *input-output-stmt* is one of: R14-1

- *open-stmt* ! file connection
- *close-stmt* ! file connection
- *inquire-stmt* ! file inquiry
- *backspace-stmt* ! file positioning
- *endfile-stmt* ! file positioning
- *rewind-stmt* ! file positioning

14.1 Connection of a Unit to an External File

An external unit has the property of being *connected* or *not connected*. If connected, it refers to a file. An external unit may be preconnected, or it may be connected by the execution of an `OPEN` statement. All I/O statements except an `OPEN`, a `CLOSE`, or an `INQUIRE` statement must refer to a unit that is connected to a file and thereby make use of or affect that file. A file may be connected even though it does not exist. An example is a preconnected new external file.

An external file that is connected to a unit has a property called *position*.

A unit must not be connected to more than one file at the same time, and a file must not be connected to more than one unit at the same time. The `CLOSE` statement allows a unit to be disconnected from a file, and the `OPEN` operation allows the unit to be reconnected to a different file.

If a unit is a character variable, it is always connected to the internal file designated by that variable.

Copyright © 1989–1994 Thinking Machines Corporation

A unit is *preconnected* if it is connected to a file at the beginning of execution of an executable program and may therefore be specified in I/O statements without the prior execution of an OPEN statement.

14.2 OPEN Statement

The OPEN statement (R14-2) is used to connect a unit to an existing file, to create a file that is preconnected, or to create a file and connect it to a unit.

An *open-stmt* is: R14-2

OPEN (*connect-spec(s)*)

A *connect-spec* is one of: R14-3

- *[* UNIT = *]* *unit-number*
- FILE = *scalar-char-expr*
- STATUS = *scalar-char-expr*
- ACCESS = *scalar-char-expr*
- FORM = *scalar-char-expr*
- RECL = scalar-integer-*expr*
- BLANK = *scalar-char-expr*
- READONLY
- ERR = *label*
- IOSTAT = scalar-integer-*variable*
- POSITION = *scalar-char-expr*
- ACTION = *scalar-char-expr*
- DELIM = *scalar-char-expr*
- PAD = *scalar-char-expr*

♦ A *connect-spec(s)* list must contain exactly one UNIT= specifier and may contain at most one of each of the other specifiers. If the optional phrase "UNIT =" is absent in a UNIT= specifier, then the *unit-number* must be the first specifier in the *connect-spec(s)* list.

♦ The *label* used in the ERR= specifier must be the label of an executable statement that appears in the same program unit as the OPEN statement.

An external unit may be connected by an OPEN statement in any program unit of an executable program; once connected, the unit may be referenced in any program unit of the executable program.

Version 2.2, October 1994
Copyright © 1989–1994 Thinking Machines Corporation

If a unit is connected to a file, execution of an OPEN statement for that unit is permitted. If the FILE= specifier is not included in such an OPEN statement, the file to be connected to the unit is the same as the file to which the unit is connected.

If the file to be connected to the unit is not the same as the file to which the unit is connected, the effect is as if a CLOSE statement without a STATUS= specifier had been executed for the unit immediately prior to the execution of an OPEN statement.

If the file to be connected to the unit does not exist but is the same as the file to which the unit is preconnected, the properties specified by an OPEN statement become part of the connection.

If the file to be connected to the unit is the same as the file to which the unit is connected, only the BLANK=, ERR=, IOSTAT=, DELIM=, and PAD= specifiers may have a value different from the one currently in effect. Execution of such an OPEN statement causes any new value of the BLANK= specifier to take effect, but does not cause any change in any of the unspecified specifiers, and the position of the file is unaffected. The ERR= and IOSTAT= specifiers from any previously executed OPEN statement have no effect on any currently executed OPEN statement.

Except for the FILE= specifier, a specifier that requires a *scalar-char-expr* may have a limited choice of possible character values. These values are listed for each such specifier. Trailing blanks are ignored, and the value is considered without regard to case. Some specifiers have a default value if the specifier is omitted.

The ERR= specifier is described in Section 13.4.1.6. The other specifiers for the OPEN statement are described in the following sections.

An example of an OPEN statement:

```
OPEN( 10, FILE = 'employee.dat' )
```

14.2.1 FILE= Specifier in the OPEN Statement

The FILE= specifier supplies the name of an external file or device being connected to a unit.

A (FILE) *connect-spec* is: R14-3
FILE = *scalar-char-expr*

The value of the FILE= specifier is a valid UNIX filename of the file or device to be connected. Any trailing blanks in the pathname are ignored, and the interpretation of characters is case sensitive.

Copyright © 1989–1994 Thinking Machines Corporation

If this specifier is omitted and the unit is not already connected to a file, the unit is connected to a file with a default name that is derived from the number of the unit being connected. (Refer to the *CM Fortran User's Guide* for information regarding the default association of units with filenames.)

14.2.2 STATUS= Specifier in the OPEN Statement

The `STATUS=` specifier provides information about the existence of a file to be connected.

A (STATUS) *connect-spec* is one of: R14-3

- `STATUS = "OLD"` ! file must exist
- `STATUS = "NEW"` ! file must not exist
- `STATUS = "SCRATCH"`
- `STATUS = "UNKNOWN"` ! default

If `"OLD"` is specified, the file must exist. If `"NEW"` is specified, the file must not exist. Successful execution of an `OPEN` statement with `"NEW"` specified creates the file. (A subsequent inquiry of the file status will return the value `"OLD"`.) If `"SCRATCH"` is specified, the file is created and connected to the specified unit for use by the executable program but is deleted at the execution of a `CLOSE` statement referring to the same unit, or at the termination of the program. `"SCRATCH"` and `"FILE"` cannot both be specified for the same file. If `"UNKNOWN"` is specified, the file will be opened if it exists, and will be created if it does not already exist. If this specifier is omitted, the default value is `"UNKNOWN"`.

14.2.3 ACCESS= Specifier in the OPEN Statement

The `ACCESS=` specifier indicates whether a file is being connected for sequential or direct access.

An (ACCESS) *connect-spec* is one of: R14-3

- `ACCESS = "SEQUENTIAL"` ! default
- `ACCESS = "DIRECT"`

If the specifier is omitted, the default value is `"SEQUENTIAL"`. An existing file must be accessible using the specified access method. For a new file, the run-time system creates a file that can be accessed using the specified access method.

Version 2.2, October 1994
Copyright © 1989–1994 Thinking Machines Corporation

14.2.4 FORM= Specifier in the OPEN Statement

The FORM= specifier indicates whether a file is being connected for formatted or unformatted I/O.

A (FORM) *connect-spec* is one of: R14-3

- FORM = "FORMATTED" ! default if ACCESS = "SEQUENTIAL"
- FORM = "UNFORMATTED" ! default if ACCESS = "DIRECT"
- FORM = "PHYSICAL"

If the specifier is omitted, the default value is "FORMATTED" if the file is being connected for sequential access and "UNFORMATTED" if the file is being connected for direct access.

Use FORM = "PHYSICAL" to speed up I/O operation when the following conditions apply:

- The order of array elements in the file is not important.
- Data is read into arrays of the same shape and layout as those from which it is written.
- Data is written and read using the same execution model and the same machine size and, if using SDA, the same SDA size.

An existing file must be accessible in the specified form. For a new file, the run-time system creates a file that can be accessed in the specified form.

14.2.5 RECL= Specifier in the OPEN Statement

The RECL= specifier designates the length of each record, in bytes, in a file being connected for direct access.

A (RECL) *connect-spec* is: R14-3

RECL = scalar-integer-*expr*

The value of the scalar-integer-*expr* must be positive. This specifier must be present when a file is being connected for direct access. For an existing file, the value of the RECL= specifier must correspond to the record length (or one of the allowed record lengths) for the file. For a new file, the run-time system creates a file with the specified record length.

Copyright © 1989–1994 Thinking Machines Corporation

14.2.6 BLANK= Specifier in the OPEN Statement

The BLANK= specifier indicates how blank characters are to be treated during formatted I/O.

A (BLANK) *connect-spec* is one of: R14-3

- BLANK = "NULL" ! default
- BLANK = "ZERO"

The BLANK= specifier is permitted only for a file being connected for formatted I/O. If "NULL" is specified, all blank characters in numeric formatted input fields on the specified unit are ignored, except that a field of all blanks has a value of zero. If "ZERO" is specified, all blanks other than leading blanks are treated as zeros.

On input, blanks are treated as if the file had been opened with a BLANK = "NULL" specifier in an OPEN statement.

14.2.7 READONLY Specifier in the OPEN Statement

The READONLY specifier indicates that a file is to be opened in read-only mode. A run-time error is reported if an attempt is made to WRITE to a unit associated with a file opened in this mode.

A (READONLY) *connect-spec* is: R14-3

READONLY

The example program below illustrates the opening of a file in read-only mode.

```
OPEN( UNIT=8, FILE="x42.dat", READONLY, STATUS="OLD" )
READ( 8, * ) X, Y
PRINT *, X, Y
WRITE( 8, * ) 123.   ! run-time error
END
```

The program is correct except for the erroneous WRITE statement.

NOTE: It is preferable to use ACTION="READ", described in Section 14.2.9.

Version 2.2, October 1994
Copyright © 1989–1994 Thinking Machines Corporation

14.2.8 POSITION= Specifier in the OPEN Statement

The POSITION= specifier indicates how the file being opened is to be positioned.

A (POSITION) *connect-spec* is one of: R14-3

- POSITION = "REWIND"
- POSITION = "APPEND"
- POSITION = "ASIS" ! default

The file must be connected for sequential access. A new file is positioned at its initial point. An existing file is positioned according to the keyword specified. If "REWIND" is specified, the file is positioned at its initial point. If "APPEND" is specified, the file is positioned so that the endfile record is the next record, if it has one. If the file does not have an endfile record, the file is positioned at its terminal point. If "ASIS" is specified and the file is already connected, the position is unchanged. If "ASIS" is specified and the file is not yet connected, the position is left unspecified.

14.2.9 ACTION= Specifier in the OPEN Statement

The ACTION= specifier indicates which actions on the file are permissible.

An (ACTION) *connect-spec* is one of: R14-3

- ACTION = "READ"
- ACTION = "WRITE"
- ACTION = "READWRITE" ! default

If "READ" is specified, no WRITE, PRINT, or ENDFILE statements may refer to this connection. If "WRITE" is specified, no READ statements may refer to this connection. If "READWRITE" is specified, any I/O statements may refer to this connection.

If READWRITE is included in the set of allowed actions for a file, both READ and WRITE must also be included in that set. If the file is new, the processor creates the file with the specified action. For an existing file, the specified action must be included in the set of allowed actions.

Copyright © 1989–1994 Thinking Machines Corporation

14.2.10 DELIM= Specifier in the OPEN Statement

The DELIM= specifier indicates which character will be used to delimit character constants written with list-directed or namelist formatting.

A (DELIM) *connect-spec* is one of: R14-3

- DELIM = "APOSTROPHE"
- DELIM = "QUOTE"
- DELIM = "NONE" ! default

The file must be connected for formatted I/O. If "APOSTROPHE" is specified, the apostrophe will be used to delimit character constants; if "QUOTE" is specified, the quote will be used. If "NONE" is specified, the character constant will not be delimited. The DELIM= specifier is ignored during input of a formatted record.

14.2.11 PAD= Specifier in the OPEN Statement

The PAD= specifier indicates whether an input record should be padded

A (PAD) *connect-spec* is one of: R14-3

- PAD = "YES"
- PAD = "NO" ! default

If "YES" is specified, a formatted input record is logically padded with blanks when an input list is specified and the format specification requires more data from a record than the record contains. If "NO" is specified, the input list and the format specification must not require more characters from a record than the record contains.

14.3 CLOSE Statement

The CLOSE statement terminates the connection of a unit to a file.

A *close-stmt* is: R14-4

CLOSE (*close-spec(s)*)

A *close-spec* is one of: R14-5

- *[* UNIT = *]* *unit-number*
- STATUS = *scalar-char-expr*
- ERR = *label*
- IOSTAT = scalar-integer-*variable*

Version 2.2, October 1994
Copyright © 1989–1994 Thinking Machines Corporation

- A *close-spec(s)* list must contain exactly one UNIT= specifier and may contain at most one of each of the other specifiers.
- If the optional phrase "UNIT =" is absent in a UNIT= specifier, then the *unit-number* must be the first specifier in the *connect-spec(s)* list.
- The *label* used in the ERR= specifier must be the label of an executable statement that appears in the same program unit as the CLOSE statement.

Execution of a CLOSE statement may occur in any program unit of a program and need not occur in the same program unit as the execution of the OPEN statement referring to that unit.

Execution of a CLOSE statement specifying a unit that does not exist or has no file connected to it is permitted and has no effect.

After a *unit* has been disconnected by execution of a CLOSE statement, it may be connected again, either to the same file or to a different file. After a *named file* has been disconnected by execution of a CLOSE statement, it may be connected again, either to the same unit or to a different unit, provided that the file still exists. There is no means of reconnecting an unnamed file once it is disconnected.

The IOSTAT= specifier and the ERR= specifier are described in Sections 13.4.1.7 and 13.4.1.6, respectively.

14.3.1 STATUS= Specifier in the CLOSE Statement

The STATUS= specifier determines the disposition of the file connected to the specified unit at completion of the CLOSE statement.

A (STATUS) *close-spec* is one of: R14-6
- STATUS = "KEEP"
- STATUS = "DELETE"

If "KEEP" is specified for a file that exists, the file continues to exist after the execution of the CLOSE statement. If "KEEP" is specified for a file that does not exist, the file will not exist after execution of the CLOSE statement. "KEEP" must not be specified for a file whose status prior to execution of a CLOSE statement is "SCRATCH".

If "DELETE" is specified, the file will not exist after the execution of the CLOSE statement.

Copyright © 1989–1994 Thinking Machines Corporation

If the STATUS= specifier is omitted, the default value is "KEEP", unless the file status prior to execution of the CLOSE statement is "SCRATCH", in which case the default value is "DELETE".

Examples of CLOSE statements:

```
CLOSE ( 10, STATUS = 'KEEP' )
CLOSE ( 5, STATUS = 'DELETE', ERR = 100 )
```

14.3.2 Implicit CLOSE at Program Termination

At termination of an executable program for reasons other than an error condition, all units that are connected are closed. Each unit is closed with status "KEEP" unless the file status prior to termination of execution was "SCRATCH", in which case the unit is closed with status "DELETE".

14.4 File Positioning Statements

The file positioning I/O statements provide the ability to manipulate the file position of an external file connected for sequential access and to write an endfile record. The file positioning I/O statements are BACKSPACE, ENDFILE, and REWIND.

A *file-positioning-stmt* is one of: R14-7

- BACKSPACE *unit-number*
- BACKSPACE (*position-spec(s)*)
- ENDFILE *unit-number*
- ENDFILE (*position-spec(s)*)
- REWIND *unit-number*
- REWIND (*position-spec(s)*)

A *position-spec* is one of: R14-8

- *[* UNIT = *]* *unit-number*
- ERR = *label*
- IOSTAT = scalar-integer-*variable*

- The *label* used in the ERR= specifier must be the statement label of a branch target statement that appears in the same program unit as the file positioning statement.
- A *position-spec(s)* list must contain exactly one UNIT= specifier and may contain at most one of each of the other specifiers.

Version 2.2, October 1994
Copyright © 1989–1994 Thinking Machines Corporation

- If the optional phrase "UNIT =" is absent in a UNIT= specifier, then the *unit-number* must be the first specifier in the *position-spec(s)* list.

A file that is not connected for sequential access must not be referred to by a BACKSPACE, an ENDFILE, or a REWIND statement.

Execution of a file positioning statement containing an IOSTAT= specifier causes the specified integer variable to become defined with a zero value if no error condition results, and with a positive error code otherwise.

14.4.1 BACKSPACE Statement

Execution of a BACKSPACE statement causes the file connected to the specified unit to be positioned before the preceding record. If there is no preceding record, the position of the file is not changed. Note that if the preceding record is an endfile record, the file becomes positioned before the endfile record.

A *backspace-stmt* is one of: R14-9
- BACKSPACE *unit-number*
- BACKSPACE (*position-spec(s)*)

Backspacing a file that is connected but does not exist is prohibited. Also, backspacing over records written using list-directed formatting is prohibited.

An example of a BACKSPACE statement:

```
BACKSPACE ( 10, ERR = 20 )
```

14.4.2 ENDFILE Statement

Execution of an ENDFILE statement writes an endfile record as the next record of the file when a file is connected for sequential access. The file is then positioned after the endfile record. If the file may also be connected for direct access, only those records before the endfile record are considered to have been written. Thus, only those records may be read during subsequent direct access connections to the file.

An *endfile-stmt* is one of: R14-10
- ENDFILE *unit-number*
- ENDFILE (*position-spec(s)*)

Copyright © 1989–1994 Thinking Machines Corporation

After execution of an ENDFILE statement, a BACKSPACE or REWIND statement must be used to reposition the file prior to execution of any file I/O statement.

Execution of an ENDFILE statement for a file that is connected but does not exist creates the file.

An example of an ENDFILE statement:

```
ENDFILE 11
ENDFILE ( 10, ERR = 20 )
```

14.4.3 REWIND Statement

Execution of a REWIND statement causes the specified file to be positioned at its initial point. If the file is already positioned at its initial point, execution of this statement has no effect on the position of the file.

A *rewind-stmt* is one of: R14-11

- REWIND *unit-number*
- REWIND (*position-spec(s)*)

Execution of a REWIND statement for a file that is connected but does not exist is permitted and has no effect.

Examples of REWIND statements:

```
REWIND 11   ! rewind unit 11
REWIND ( 10, ERR = 20, IOSTAT = IOS )        ! rewind unit 10
! place status in variable IOS; on error, branch to label 20
```

14.5 File Inquiry: INQUIRE Statement

The INQUIRE statement returns information about properties of a particular named file or of the connection to a particular unit.

An *inquire-stmt* is: R14-12

INQUIRE (*inquiry-spec(s)*)

Version 2.2, October 1994
Copyright © 1989–1994 Thinking Machines Corporation

An *inquiry-spec* is one of: R14-13

- *[* UNIT = *]* *unit-number*
- FILE = *scalar-char-expr*
- EXIST = *scalar-logical-variable*
- OPENED = *scalar-logical-variable*
- NUMBER = scalar-integer-*variable*
- NAMED = *scalar-logical-variable*
- NAME = *scalar-char-variable*
- ACCESS = *scalar-char-variable*
- SEQUENTIAL = *scalar-char-variable*
- DIRECT = *scalar-char-variable*
- FORM = *scalar-char-variable*
- FORMATTED = *scalar-char-variable*
- UNFORMATTED = *scalar-char-variable*
- RECL = scalar-integer-*variable*
- NEXTREC = scalar-integer-*variable*
- BLANK = *scalar-char-variable*
- ERR = *label*
- IOSTAT = scalar-integer-*variable*
- POSITION = *scalar-char-variable*
- ACTION = *scalar-char-variable*
- DELIM = *scalar-char-variable*
- PAD = *scalar-char-variable*

♦ An *inquiry-spec(s)* must contain exactly one FILE= specifier or one UNIT= specifier but not both, and may contain at most one of each of the other specifiers. If the optional phrase "UNIT =" is absent in a UNIT= specifier, then the *unit-number* must be the first specifier in the *inquiry-spec(s)* list.

There are two forms of the INQUIRE statement: the *by-unit* form uses a single UNIT= specifier, and the *by-file* form uses a single FILE= specifier. Except as noted in the sections below, the inquiry specifiers listed above may be used in either form of the INQUIRE statement.

The INQUIRE statement returns information about a file or a connection in the variables specified in the *inquiry-spec(s)* list of the statement. All assignments to these variables are performed according to the rules for assignment statements.

An INQUIRE statement may be executed before, while, or after a file is connected to a unit. All values assigned by an INQUIRE statement are those that are current at the time the statement is executed.

Copyright © 1989–1994 Thinking Machines Corporation

When a returned value of a specifier other than the NAME= specifier is of type character, the value returned is uppercase. The UNIX filename returned by the NAME= specifier may be in mixed case.

If an error condition occurs during execution of an INQUIRE statement, all inquiry specifier variables become undefined except for the variable designated by an IOSTAT= specifier.

A variable whose value may be changed as a result of its use in a specifier of an INQUIRE statement must not appear in another specifier in the same INQUIRE statement.

Table 11 on page 255 summarizes the values assigned to inquire specifier variables using both forms of the INQUIRE statement.

Examples of INQUIRE statements:

```
 INQUIRE ( "~/.emacs", EXIST = DOES_EXIST )
 INQUIRE ( UNIT = 10, OPENED = IS_OPEN, NAMED = IS_NAMED,
$  NAME = THE_NAME )
```

14.5.1 UNIT= Specifier in the INQUIRE Statement

A UNIT= specifier is required in a by-unit inquiry.

A (UNIT) *inquiry-spec* is: R14-13
[UNIT = *]* *unit-number*

The *unit-number* specifies the external unit about which the inquiry is being made. The unit need not exist or be connected to a file. If it is connected to a file, the inquiry is made with respect to both the connection and the connected file.

14.5.2 FILE= Specifier in the INQUIRE Statement

A FILE= specifier is required in a by-file inquiry.

A (FILE) *inquiry-spec* is: R14-13
FILE = *scalar-char-expr*

The *scalar-char-expr* specifies the file about which the inquiry is being made. The file must be a valid UNIX filename. The named file need not exist or be connected to a unit. The case of characters of the *scalar-char-expr* is significant.

Version 2.2, October 1994
Copyright © 1989–1994 Thinking Machines Corporation

14.5.3 EXIST= Specifier in the INQUIRE Statement

An EXIST= specifier may be used in a by-file or by-unit inquiry.

An (EXIST) *inquiry-spec* is: R14-13
EXIST = *scalar-logical-variable*

A by-file inquiry assigns the value true if a file exists with the name given in the FILE= specifier, false otherwise.

A by-unit inquiry assigns the value true if the unit specified by UNIT= exists, false otherwise.

14.5.4 OPENED= Specifier in the INQUIRE Statement

An OPENED= specifier may be used in a by-file or by-unit inquiry.

An (OPENED) *inquiry-spec* is: R14-13
OPENED = *scalar-logical-variable*

A by-file inquiry assigns the value true if the specified file is connected to a unit, false otherwise.

A by-unit inquiry assigns the value true if the specified unit is connected to a file, false otherwise.

14.5.5 NUMBER= Specifier in the INQUIRE Statement

A NUMBER= specifier may be used in a by-file or by-unit inquiry.

A (NUMBER) *inquiry-spec* is: R14-13
NUMBER = scalar-integer-*variable*

An inquiry assigns the number of the unit that is currently connected to the file. If there is no logical unit, the value of the variable is undefined.

Copyright © 1989–1994 Thinking Machines Corporation

14.5.6 NAMED= Specifier in the INQUIRE Statement

A NAMED= specifier may be used in a by-file or by-unit inquiry.

A (NAMED) *inquiry-spec* is: R14-13
NAMED = *scalar-logical-variable*

An inquiry assigns the value true if the file has a name, false otherwise.

14.5.7 NAME= Specifier in the INQUIRE Statement

A NAME= specifier may be used in a by-file or by-unit inquiry.

A (NAME) *inquiry-spec* is: R14-13
NAME = *scalar-char-variable*

An inquiry assigns the name of the file if it has a name; otherwise the specified character variable becomes undefined.

Note that the name returned by a by-file inquiry is not necessarily the same name given in the FILE= specifier of the OPEN statement, although the returned name is suitable for use in an OPEN statement.

14.5.8 ACCESS= Specifier in the INQUIRE Statement

An ACCESS= specifier may be used in a by-file or by-unit inquiry.

An (ACCESS) *inquiry-spec* is: R14-13
ACCESS = *scalar-char-variable*

An inquiry assigns the value "SEQUENTIAL" if the file is connected for sequential access, and the value "DIRECT" if the file is connected for direct access. The inquiry assigns the value "UNKNOWN" if the specified file or unit is not connected.

14.5.9 SEQUENTIAL= Specifier in the INQUIRE Statement

A SEQUENTIAL= specifier may be used in a by-file or by-unit inquiry.

A (SEQUENTIAL) *inquiry-spec* is: R14-13
SEQUENTIAL = *scalar-char-variable*

Copyright © 1989–1994 Thinking Machines Corporation

An inquiry assigns the value "YES" if the specified file supports sequential access, "NO" if the file does not support sequential access, and "UNKNOWN" if the fact is indeterminate.

14.5.10 DIRECT= Specifier in the INQUIRE Statement

A DIRECT= specifier may be used in a by-file or by-unit inquiry.

A (DIRECT) *inquiry-spec* is: R14-13
DIRECT = *scalar-char-variable*

An inquiry assigns the value "YES" if the specified file supports direct access, "NO" if the file does not support direct access, and "UNKNOWN" if the fact is indeterminate.

14.5.11 FORM= Specifier in the INQUIRE Statement

A FORM= specifier may be used in a by-file or by-unit inquiry.

A (FORM) *inquiry-spec* is: R14-13
FORM = *scalar-char-variable*

An inquiry assigns the value "PHYSICAL" if the keyword "PHYSICAL" was specified in the OPEN statement. Otherwise, an inquiry assigns the value "FORMATTED" if the specified file is connected for formatted I/O, "UNFORMATTED" if the file is connected for unformatted I/O, and "UNKNOWN" if the specified file or unit is not connected.

14.5.12 FORMATTED= Specifier in the INQUIRE Statement

A FORMATTED= specifier may be used in a by-file or by-unit inquiry.

A (FORMATTED) *inquiry-spec* is: R14-13
FORMATTED = *scalar-char-variable*

An inquiry assigns the value "YES" if the specified file may be accessed as a formatted file, "NO" if the file may not be accessed as a formatted file, and "UNKNOWN" if the fact is indeterminate.

Copyright © 1989–1994 Thinking Machines Corporation

14.5.13 UNFORMATTED= Specifier in the INQUIRE Statement

An UNFORMATTED= specifier may be used in a by-file or by-unit inquiry.

An (UNFORMATTED) *inquiry-spec* is: R14-13
UNFORMATTED = *scalar-char-variable*

An inquiry assigns the value "YES" if the specified file may be accessed as an unformatted file, "NO" if the file may not be accessed as an unformatted file, and "UNKNOWN" if the fact is indeterminate.

14.5.14 RECL= Specifier in the INQUIRE Statement

A RECL= specifier may be used in a by-file or by-unit inquiry about a file connected for direct access.

A (RECL) *inquiry-spec* is: R14-13
RECL = scalar-integer-*variable*

An inquiry assigns the record length, in bytes, of the specified file connected for direct access. If the specified file or unit is not connected for direct access, the scalar-integer-*variable* becomes undefined.

14.5.15 NEXTREC= Specifier in the INQUIRE Statement

A NEXTREC= specifier may be used in a by-file or by-unit inquiry about a file connected for direct access.

A (NEXTREC) *inquiry-spec* is: R14-13
NEXTREC = scalar-integer-*variable*

An inquiry assigns the value $n+1$ to the scalar-integer-*variable*, where n is the record number of the last record read from or written to the file connected for direct access. If the file is connected but no records have been read or written since the connection, the variable is assigned the value 1. If the file is not connected for direct access, or if the position of the file is indeterminate because of a previous error condition, the scalar-integer-*variable* becomes undefined.

Version 2.2, October 1994
Copyright © 1989–1994 Thinking Machines Corporation

14.5.16 BLANK= Specifier in the INQUIRE Statement

A BLANK= specifier may be used in a by-file or by-unit inquiry about a file connected for formatted I/O.

A (BLANK) *inquiry-spec* is: R14-13
BLANK = *scalar-char-variable*

An inquiry assigns the value "NULL" if null blank control is in effect for the file connected for formatted I/O, and is assigned the value "ZERO" if zero blank control is in effect for the file connected for formatted I/O. If the specified file or unit is not connected, or if the connection is not for formatted I/O, the variable is assigned the value "UNKNOWN".

14.5.17 ERR= Specifier in the INQUIRE Statement

An ERR= specifier may be used in a by-file or by-unit inquiry. The ERR= specifier is described in Section 13.4.1.

14.5.18 IOSTAT= Specifier in the INQUIRE Statement

An IOSTAT= specifier may be used in a by-file or by-unit inquiry.

An (IOSTAT) *inquiry-spec* is: R14-13
IOSTAT = scalar-integer-*variable*

Execution of an INQUIRE statement containing an IOSTAT= specifier causes the specified variable to become defined with a zero value if no error condition exists for the specified connection or with a positive integer value if an error condition exists.

If an error condition occurs during execution of an INQUIRE statement, all of the inquiry specifier variables except the one specified in the IOSTAT= specifier become undefined.

14.5.19 POSITION= Specifier in the INQUIRE Statement

A POSITION= specifier may be used in a by-file or by-unit inquiry.

A (POSITION) *inquiry-spec* is: R14-13
POSITION = *scalar-char-variable*

Copyright © 1989–1994 Thinking Machines Corporation

If the file has been opened but not repositioned, an inquiry assigns the value that was given in the OPEN statement ("REWIND", "APPEND", or "ASIS"). If the file has been repositioned and is at its initial point, "REWIND" is assigned; if the file has been repositioned but is not at its initial point, "UNDEFINED" is assigned. The value "UNDEFINED" is assigned if there is no connection or if the file is connected for direct access.

14.5.20 ACTION= Specifier in the INQUIRE Statement

An ACTION= specifier may be used in a by-file or by-unit inquiry.

A (ACTION) *inquiry-spec* is: R14-13
ACTION = *scalar-char-variable*

An inquiry assigns the value "READ" if the file is connected for input only, "WRITE" if it is connected for output only, "READWRITE" if it is connected for both input and output, and "UNDEFINED" if there is no connection.

14.5.21 DELIM= Specifier in the INQUIRE Statement

A DELIM= specifier may be used in a by-file or by-unit inquiry.

A (DELIM) *inquiry-spec* is: R14-13
DELIM = *scalar-char-variable*

An inquiry assigns the value "APOSTROPHE" if the apostrophe is used to delimit character data written by list-directed or namelist formatting, "QUOTE" if the quotation mark is used to delimit character data, "NONE" if character data will not be delimited, and "UNDEFINED" if there is no connection or if the connection is not for formatted I/O.

14.5.22 PAD= Specifier in the INQUIRE Statement

A PAD= specifier may be used in a by-file or by-unit inquiry.

A (PAD) *inquiry-spec* is: R14-13
PAD = *scalar-char-variable*

An inquiry returns the value "NO" if the connection of the file to the unit included the PAD= specifier and its value was NO, and "YES" if the value of the PAD= specifier was YES or the specifier was omitted.

Version 2.2, October 1994
Copyright © 1989–1994 Thinking Machines Corporation

Table 11. Values assigned to `INQUIRE` specifier variables.

Specifier	INQUIRE by File		INQUIRE by Unit	
	Unconnected	Connected	Connected	Unconnected
EXIST=	.TRUE. if file exists, .FALSE. otherwise		.TRUE. if unit exists, .FALSE. otherwise	
OPENED=	.FALSE.	.TRUE.		.FALSE.
NUMBER=	−1	*unit number*		−1
NAMED=	.TRUE.		.TRUE. if file exists, .FALSE. otherwise	.FALSE.
NAME=	filename (may not be same as FILE= value)		filename if named, else *undefined*	*undefined*
ACCESS=	"UNKNOWN"	"SEQUENTIAL" or "DIRECT"		"UNKNOWN"
SEQUENTIAL=	"YES", "NO", or "UNKNOWN"			"UNKNOWN"
DIRECT=	"YES", "NO", or "UNKNOWN"			"UNKNOWN"
FORM=	"UNKNOWN"	"FORMATTED" or "UNFORMATTED"		"UNKNOWN"
FORMATTED=	"YES", "NO", or "UNKNOWN"			"UNKNOWN"
UNFORMATTED=	"YES", "NO", or "UNKNOWN"			"UNKNOWN"
RECL=	*undefined*	if direct access, *record length*; else *undefined*		*undefined*
NEXTREC=	*undefined*	if direct access, next record #; else *undefined*		*undefined*
BLANK=	"UNKNOWN"	"NULL", "ZERO", or "UNKNOWN"		"UNKNOWN"
POSITION=	"UNDEFINED"	if direct access, "UNDEFINED", else "REWIND", "APPEND", or "ASIS"		"UNDEFINED"
ACTION=	"UNDEFINED"	"READ", "WRITE", or "READWRITE"		"UNDEFINED"
DELIM=	"UNDEFINED"	"APOSTROPHE", "QUOTE", "NONE", or "UNDEFINED"		"UNDEFINED"
PAD=	"UNDEFINED"	"YES" or "NO"		"UNDEFINED"

Copyright © 1989–1994 Thinking Machines Corporation