MULTICS SYSTEM-PROGRAMMERS ' MANUAL

SECTION BF.1.16 PAGE 1

Published: 09/2/66

Identification

Random Logical Record I/O J. F. Ossanna, V. A. Vyssotsky, G. G. Ziegler

Purpose

The Multics I/O system provides capability for random access I/O. This section describes the I/O system calls for performing random logical record I/O.

Random Logical Record Frames

An existing logical record frame may be attached to a process (or a new logical record frame may be created and attached to a process) as a random frame by a call to the I/O system. A frame which already exists when it is so attached may have previously been attached to processes as either a sequential or a random frame or both, but not simultaneously. When attached as a random frame, a logical record frame may be regarded as a set of records identified by record number. A record number is represented in I/O calls as a 35 bit signed integer, whose value must be non-negative. Any two distinct records of a frame have different record numbers; however, if n is the record number of a record in the frame there is no requirement that the frame contain records with record numbers less than n. The I/O system calls for random logical record frames are formally identical to those described in section BF.1.15 for sequential logical record frames. They will be described in terms of the differences between their action on a random frame and their action on a sequential frame. One outstanding difference is that the argument recno of the <u>read, write</u>, seek and delete calls is interpreted differently. In each of these calls, recno is interpreted as follows. If $1 \leq \text{recno} \leq R$, where R is the declared maximum record number of the frame, then recno is the number of the record to be read, written, found or deleted. If recno = 0 or is null, then the record involved is the current record. If recno < 0 or if recno > R, the call is rejected, and the current record number becomes undefined. Another significant difference is that the normal mode for random logical record frames is the replacement mode. See section BF.1.04 for a detailed discussion of modes and their effect on write and delete calls.

MULTICS SYSTEM-PROGRAMMERS' MANUAL SECTION BF.1.16 PAGE 2

The Write Call.

The general form of the random <u>write</u> call is the same as that of the sequential <u>write</u> call

call write(name, recno, workspace, nelem[, status])

The arguments and actions of random <u>write</u> are identical to those of sequential <u>write</u>, except for the difference in interpretation of recno described above.

The Read Call

The general form of the random <u>read</u> call is the same as that of the sequential <u>read</u> call

call read(name, recno, workspace, nelem[, nelmt[, status]])

The arguments and actions of random <u>read</u> are identical to those of sequential <u>read</u>, except for the difference in interpretation of recno described above.

<u>The Tell Call</u>

The general form of the <u>tell</u> call for a random logical record frame is the same as for a sequential logical record frame

call tell(name,elemno[,status])

The arguments and actions of the <u>tell</u> call are exactly the same for random frames and for sequential frames.

The Seek Call

The general form of the random <u>seek</u> call is the same as that of the sequential <u>seek</u> call

call seek(name, recno[, status])

The argument recno is interpreted differently for sequentially and random frames, as described above. In addition, the action of <u>seek</u> is different for sequential and random frames. For random frames the I/O system assumes that a seek call will be followed eventually by a <u>read</u> or <u>write</u> call for the same record, and will attempt to perform look-ahead to retrieve records for which <u>seek</u> calls are issued. The look-ahead strategy depends on such factors as structure of the frame, core availability and I/O load on the system. Generally speaking, efficiency of accessing

MULTICS SYSTEM-PROGRAMMERS' MANUAL SECTION BF.1.16 PAGE 3

random frames will increase when look-ahead is employed, up to some level of look-ahead at which efficiency begins to decrease. A rule of thumb for planning is as follows: suppose record numbers are mentioned in <u>seek</u> calls in the same sequence as those record numbers are mentioned in <u>read</u> and <u>write</u> calls. Then efficiency can be expected to be near a peak if the amount of data look-ahead is of the order of 20,000 to 80,000 bits.

The Delete Call

The general form of the random <u>delete</u> call is the same as that of the sequential <u>delete</u> call:

cali delete(name, recno[, status])

The arguments and actions of random <u>delete</u> are identical to those of sequential <u>delete</u>, except for the difference in interpretation of recno described above.

The First and Tail Calls

The general form of the <u>first</u> and <u>tail</u> calls for random frames is the same as for sequential frames

call first(name[,status])
call tail(name[,status])

The arguments and actions of the <u>first</u> and <u>tail</u> calls for random frames are the same as for sequential frames.