Published: 08/31/66

Identification

Logical Record Frames
J. F. Ossanna, V. A. Vyssotsky, G. G. Ziegler

Purpose

The Multics I/O system provides capability for manipulating logical record frames. This section describes the structure of logical record frames, also called sectional frames.

Elements, Records and Frames

A logical record frame is a collection of data units called records, each record having a record number. Record numbers are positive integers, and any two distinct records in a frame have different record numbers. However, there is no requirement that a frame actually contain a data record for each record number; a frame may, for example, consist of one record, whose record number is 62. Each record is a sequence of bits, and at any given moment each particular record has a definite length, the length being the number of bits in the record. The various records of a frame need not, however, all have the same length; for example, record 11 may be 144 bits long although record 12 is only 9 bits long. If a record is rewritten its length may be increased or decreased. Consider the following examples. The content of a record 144 bits long may be replaced by writing with information 180 bits (or 36 bits) long; the length of the record then becomes 180 (or 36) bits. The content of the record may be replaced by information 0 bits long; the length of the record then becomes 0 bits. The record may be deleted, after which its length is zero, because it no longer exists. (The I/O system distinguishes between existing records of zero length and non-existent records; this distinction is available to user programs via status returns and status traps. A user program not concerned with differentiating these two cases may ignore the distinction.)

It is inconvenient to manipulate records using bit numbers and lengths using number of bits. Therefore, when a logical record frame is attached to a process, an element size in bits is declared. This element size is a positive integer which may be any integer in the range 1 to $36x2^{14}$, inclusive; if an element size is not explicitly declared, a default declared element size of 9 bits is assumed by

the I/O system. Read and write calls specify how much data is to be read or written in terms of number of elements, and the declared element size is used by the I/O system to determine the number of bits implied by a request for n elements. It is expected that the most commonly used element sizes will be 9 bits (1 element = 1 character) and 36 bits (1 element = 1 word), but a user program may, for instance, manipulate a frame as a collection of 23 bit elements if that is appropriate to the application. The size of any given record is determined by the number of elements in that record. It should be emphasized that the element size is not a property of the frame, nor of the data in the frame, but is rather a property of the way data is transmitted to and from the frame. Thus, it is possible to use a given frame today with element size 23 bits, and tomorrow with element size 41 bits. Because the length of a record is a property of the data in the record, if a record is read with element size different from that used for writing the record, the last element read from the record may be incomplete. If this occurs, the partial element will be transmitted with trailing binary zeros to form a complete element, and an appropriate status return indication will be given. However, it seems likely that in most applications a given frame will always be written and read with the same element size.

Random and Sequential Logical Record Frames

A logical record frame may be attached as either a random frame or a sequential frame. (The primitives for handling sequential logical record frames are discussed in section BF.1.15; those for random logical record frames in section BF.1.16.) If it is attached as a random frame, a read or write call specifies the record to be read or written by the record number of the record. If, however, the frame is attached as a sequential frame, a read or write call specifies the record to be read or written by an increment to the current record number (e.g. "read the second record after the current record"). The same frame may be attached as both a serial and a random frame, at different times, but not both at the same time. Any frame which may be attached as a random sectional frame may also be attached as a sequential sectional frame; however, the converse does not hold. In order for a frame attachable as a sequential sectional frame to be attachable also as a random logical frame, the frame must either be a file system file or else reside on a medium capable of being randomly accessed. In particular, a frame resident on a user-owned magnetic tape cannot be attached as a random frame. If however, the content of the tape frame

is copied into the file system, or onto a random-accessible medium, such as data disc, the copy may then be attached as a random frame.

When a new frame is created and attached as a random frame, it contains no records. After the first successful write call, it contains one record. This need not (and in general will not) be record number 1. If the first record written is record number 62, a read call immediately following for record 62 will read the record, but a read call for any other record number would fail. Each subsequent write call may either add a new record to the frame or (provided the frame has been declared rewriteable) may overwrite a record previously written.

If a frame is created as a random frame, it may subsequently be accessed as a serial frame. In case every record in the frame had not been written, read requests for the missing records will be rejected. Suppose, for example, that a random frame is created and only record 62 is written. If the frame is subsequently accessed as a serial frame, and a sequence of read calls is given, each call asking for the next record, the first 61 calls (for records 1-61) will be rejected with status returns showing that the records do not exist; the 62nd call will succeed, with status return showing that all records in the frame have been read, the 63rd request would give an end-of-frame return and the 64th and any subsequent requests would be rejected with a status return showing beyond-end-of-frame. See section BF.1.21 for details. If, still later, the frame were attached again as a random frame, a request to attach it as a frame of less than 62 records would fail, but it could be assigned as a random frame of any number of records greater than or equal to 62.

Element, Record and Frame Sizes

As has been observed above, the length of a record is a property of the data in the frame. Also properties of the data in the frame are:

a. The highest record number of an existing record in the frame, and

b. The total amount of data in the frame (i.e., the sum, over all existing records of the frame, of the number of bits in the record).

In order for the I/O system to be able to move data to and from a frame with acceptable efficiency, the I/O system

must know something about what the record sizes, maximum record numbers, and what the total amount of data in the frame will be during the course of the handling of the frame. Since all of these can increase during the handling of a frame, the I/O system requires explicit or default declaration, when the frame is attached, of the maximum values these quantities may be allowed to reach between attachment and detachment. Hence, when a frame is attached, the element size, the maximum record size (in elements), the maximum record number, and the maximum capacity of the frame (in elements) are fixed according to a set of rules which we will now describe.

The size of elements, records and frames are related by a set of constraints. The element size may be declared when the frame is attached, and may be any number of bits from 1 to $36x2^{14}$, inclusive. If no declaration of element size is given, the element size is given a default declared value of 9 bits. The record length of each record is determined each time that record is written, and is some integral multiple of the element size currently declared at the time the record is written. At the time the frame is attached, a maximum record size (in elements) may be declared; if no declaration is made, the maximum record size will be given a default declared value which is the greatest of

1. One element, or

2. The number of elements required to contain the largest existing record of the frame, or

3. The lesser of

   a. the largest integral number of elements which can be contained in $36x2^{12}$ bits, or

   b. 256 elements.

In other words, the default declared record size is always large enough to hold existing records, but may be larger; if the element size is no more than 576 bits, the default declared maximum record size is 256 elements, or enough elements to hold the largest existing record, whichever is greater. If the element size is greater than 576 bits but no more than $36x2^{12}$ bits the number of elements in the default declared maximum record size is as many elements as will fit into $36x2^{12}$ bits, or enough to hold the largest existing record, whichever is greater. If the element size is greater than $36x2^{12}$ bits, the maximum record size

is 1 element, or enough elements to hold the largest existing record, whichever is greater. The maximum record size may not be explicitly declared to be more than the lesser of

a. 2^{18} elements, or

b. the largest integral number of elements which can be contained in $36x2^{18}$ bits

nor may it be declared to be less than one element. If a write call specifies a number of elements larger than the current maximum declared record size, no output will occur, and an appropriate status return will occur (see section BF.1.21).

Each record in a frame has a positive integer record number, and no record may have a record number higher than some maximum record number. When a frame is assigned, a maximum record number may be declared. If no declaration is made, the maximum record number will be given a default declared value which is the greater of

a. 2^{14}, or

b. the highest record number of any existing record in the frame.

The maximum record number may not be declared to be more than 2^{24}, unless the frame is a non-insertable sequential frame, in which case the maximum number is 2^{30}.

If a read or write call specifies a record number greater than the current declared maximum record number, no transmission will occur, and an appropriate status return will occur (see section BF.1.21).

When a frame is attached, a maximum capacity (in elements) for the frame may be declared. If no declaration is made, the maximum capacity for the frame is given a default declared value which is the largest of

a. The maximum declared record size, or

b. The largest number of elements of the declared element size which can be contained in $36x2^{18}$ bits, or

c. The sum, over all existing records of the frame, of the number of elements of declared element size required to contain the data in the record.

The maximum capacity for the frame may not be declared to be more than the largest integral number of elements of the declared element size which can be contained in $36x2^{24}$ bits, unless the frame is a non-insertable sequential frame, in which case the limit is $36x2^{30}$. Nor may the maximum capacity be declared to be less than the declared maximum record size. If a write would result in increasing the actual content of the frame to more than the current declared maximum capacity, transmission will not occur, and an appropriate status return will occur (see section BF.1.21).

If a call is made to set bounds for an I/O frame, and either

a. The explicit declared record size is less than the number of elements of declared element size required to contain the largest existing record of the frame, or

b. The explicit declared maximum record number is less than the highest record number of an existing record of the frame, or

c. The explicit declared maximum capacity is less than the sum, over all existing records of the frame, of the number of elements of declared size required to contain the record

then the call will be rejected, and an appropriate status return will occur.

The Bounds Call and the Sizes Call

Explicit declaration of the element size, maximum record size, maximum record number, and maximum frame size of a logical record file is made by the bounds call, whose general form is:

call bounds(name,eltsiz,filsiz[,recsiz,recnos[,status]])

If bounds calls are to be effective for a frame, they must be given after the frame is attached, and before the first read, write, seek, delete, first or tail call for the frame is executed. The argument name is a character string of 1 to 31 characters. Its content is either a streamname or a device or frame id. If name is a streamname, it refers to the frame or device to which the stream is attached. The arguments eltsiz, filsiz, recsiz and recnos are 35 bit signed integers; their content must

be non-negative. If eltsiz > 0, it is taken to be the declared element size for the frame. If eltsiz = 0 or is null, the element size is not declared by the call. If filsiz > 0, it is taken to be the declared maximum frame size, in elements. If filsiz = 0 or is null, the maximum frame size is not declared by the call. If recsiz > 0, it is taken to be the declared maximum record size, in elements. If recsiz = 0 or is null, the maximum record size is not declared by the call. If recnos > 0, it is taken to be the declared maximum record number. If recnos = 0 or is null, the maximum record number is not declared by the call. The argument status is a 72 bit string returned by the I/O system, and is described in section BF.1.21.

Eltsiz, filsiz, recsiz and recnos for a given frame may be explicitly declared in the same bounds call, or in different bounds calls. However, for any one of these only the first explicit declaration will be effective. For example,

```
call bounds('alpha',36,500,10,100)
```

is equivalent to

```
call bounds('alpha',36,0,10,0)
call bounds('alpha',0,500,0,100)
```

and also to

```
call bounds('alpha',0,500,0,100)
call bounds('alpha',36,0,10,100)
```

All of these declare the element size to be 36 bits (1 word), the maximum size of the frame to be 500 elements (500 words), the maximum record size to be 10 elements (10 words) and the maximum record number to be 100. Another equivalent result is:

```
call bounds('alpha',36,0,10,0)
call bounds('alpha',18,500,20,100)
```

In this case the arguments eltsiz = 18 and recsiz = 20 in the second call are ignored; the element size has already been declared to be 36, and the maximum record size to be 10. Similarly, in the sequence

```
call bounds('alpha',18,500,20,100)
call bounds('alpha',36,0,10,0)
```

eltsiz and recsiz are ignored in the second call; in this case alpha is declared to have element size 18, maximum frame size 500, maximum record size 20 and maximum record number 100.

The bounds call establishes the limits of a frame but it is often desirable to be able to know the current size. The actual size of a frame, maximum size of an existing record of the frame, and maximum record number of an existing record, may be determined by the sizes call, whose general form is:

```
call sizes(name,eltsiz,filsiz[,recsiz,norecs[,status]])
```

The arguments name and status are the same as the corresponding arguments of the bounds call. The arguments eltsiz, filsiz, recsiz and norecs are 35 bit signed integers. The values of these arguments at call time are ignored by the I/O system. On return, eltsiz contains the explicitly declared element size, if an explicit declaration of element size has been given for the frame; if not, the value 9 is returned. On return, the value of filsiz is the sum, over all records of the frame, of the number of elements of size eltsiz required to contain the data currently in the record. The value of recsiz is the number of elements of size eltsiz required to contain the data currently in the currently largest record of the frame. The value of norecs is the record number of the highest numbered record currently in the frame.

A call to sizes for a given frame may be made at any time between attachment and detachment of the frame. For example, following attachment of an existing logical record frame as alpha, the sequence

```
call bounds('alpha',36,0)
call sizes('alpha',j,k,l,m)
call bounds('alpha',0,k,l,m)
```

declares the element size of alpha to be 36 bits, the maximum frame size to be exactly enough 36 bit elements to hold the data already in the frame, the maximum record size to be exactly enough 36 bit elements to hold the longest record already in the frame, and the maximum record number to be the number of the highest numbered record already in the frame. Observe that sizes merely returned the current filsiz, recsiz, and norecs in terms of some multiple of 36; the original maxima might have been greater or in terms of some other element size.