Published: 08/31/66

Identification

Linear Frames
J. F. Ossanna, V. A. Vyssotsky, G. G. Ziegler

Purpose

The Multics I/O system provides capability for manipulating linear (unstructured) frames. This section describes the nature of linear frames and the constraints on linear frames.

Elements and Frames

A linear frame is intrinsically a single sequence of bits. It has a length, which is the number of bits in the frame, and which may be zero. If the length of a linear frame is greater than zero it has a first bit and a last bit. A linear frame has no intrinsic structure beyond that implied by the preceding statements, but a structure may be imposed on it to allow convenient manipulation of the data in the frame. In particular, referencing the data by bit number and number of bits would be inconvenient. To alleviate this problem, when a linear 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 which data, and 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 which bits are implied by a request for m elements starting with element number n. It is expected that 9 bits (1 element = 1 character) and 36 bits (1 element = 1 word) will be commonly used element sizes, but a user program may, for example, manipulate a frame as a sequence of 960 bit elements (1 element = one 80 column binary card image) if that is appropriate to the application. 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 the frame is property of the data in the frame, if the frame is read with element size different from that used in writing

the frame it is possible that the last element read from the frame may be incomplete. If this occurs, the last element will be transmitted with trailing binary zeros to form a complete element, and a status indication that this has occurred 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 Linear Frames

A linear frame may be attached as either a random frame or a sequential frame. (The primitives for handling sequential linear frames are discussed in section BF.1.12; those for random linear frames in section BF.1.13.) If it is attached as a random frame, a read or write call specifies the element to be read or written by element number. If, however, the frame is attached as a sequential frame, a read or write call specifies the element to be read or written by an increment to the current element number (e.g. "read starting with the tenth element after the current element"). The same frame may be attached as both a sequential and a random frame, at different times. Any frame which may be attached as a random linear frame could instead be attached as a sequential linear frame; however, the converse does not hold. In order for a frame attachable as a sequential linear frame to be attachable also as a random linear frame, the frame must either be a file system frame 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 linear frame, it contains no data, and its length is 0. After the first successful write call it contains as many elements as the element number of the last element written. This will be at least as many elements as the number which the write call requested to be written, but may be more than that. If, for example, the new frame was attached as a random linear frame, and suppose the first call after creation requested writing of six elements starting with element number 405. Then after return from the write call the frame will be 410 elements long, and the content of the frame will be 404 elements of binary zeros, followed

by the six elements that were written. A read call can successfully transmit any or all of elements 1 to 410, but a read cannot transmit elements with element numbers greater than 410; an attempt to read elements with higher numbers will cause an appropriate status return indicating that the elements were not transmitted. An end-of-frame condition is always considered to exist just beyond the last element written; this "running" end-of-frame can always be extended by subsequent write calls until the declared maximum length of the frame is reached. At that point an attempt to write beyond the declared maximum length will return an end-of-frame status. Write calls subsequent to the first may (provided the frame has been declared rewriteable) over-write elements already in the frame as well as extend the frame. If the frame is not rewriteable and the first successful write call places data in elements numbered 405 to 410, then subsequent write calls cannot write data into any of the elements numbered 1 to 410. In addition, if that first write statement wrote a zero element into element number 405, a subsequent read call could not distinguish the zero element in element number 405 from the zero element number 404. That is, the I/O system will not distinguish in any way on subsequent reads or writes between zero elements inserted to fill holes and zero elements transmitted to the frame as data.

Size of Linear Frames

Any linear frame attached to a process has a length (which may, of course, be zero). The maximum length may be explicitly declared by a bounds call after the frame is attached; if it is not so declared, a default declaration will apply. The default declared maximum length of a linear frame is the larger of

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

b) the number of elements of the declared element size required to contain the existing data in the frame.

If an explicit declaration of maximum length is given, the declared maximum length must be at least one element and must be at least enough elements of the declared element size to contain the existing data in the frame. The declared

maximum size may not be more than the number of entire elements which can be contained in $36x2^{24}$ bits, unless the frame is a non-insertable sequential frame, in which case the limit is $36x2^{30}$. If a write would result in increasing the length of the frame to more than the current declared maximum length, transmission will not occur, and an appropriate status return will occur.

The Bounds Call and the Sizes Call

Explicit declaration of the element size and maximum frame size of a linear frame 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 frame id. If name is a streamname, it refers to the frame to which the stream is attached. The arguments eltsiz, filsiz, recsiz and recnos are 35 bit signed integers. 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. Eltsiz may not be less than zero. If filsiz > 0, it is taken to be the declared maximum size of the frame, in elements. If filsiz = 0 or is null, the maximum frame size is not declared by the call. Filsiz may not be less than zero. The arguments recsiz and recnos are ignored in bounds calls for linear frames. The argument status is a bit string returned by the I/O system, and is described in section BF.1.21.

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

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

is equivalent to

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

and also to

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

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

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

In this case the argument eltsiz = 18 in the second call is ignored; the element size has already been declared to be 36. Conversely,

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

is not equivalent to the previous case; in this case alpha is declared to have element size 18 and maximum frame size 500.

The bounds call sets the maximum size of a frame but it is often desirable to know the current size of an existing frame. The actual size of a frame 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 number of elements of size eltsiz required to contain the data currently in the frame. The values of recsiz and norecs are unchanged by a call to sizes for a linear 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 linear frame as alpha, the sequence

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

declares the element size of alpha to be 36 bits, and the maximum frame size to be exactly enough 36 bit elements to contain the data already in the frame. Observe that sizes merely returned the current length, as some multiple of 36; the original maximum length might have been greater or in terms of some other element size.