Subject: Re: point_lun question
Posted by Pavel Romashkin on Tue, 02 Nov 1999 08:00:00 GMT
View Forum Message <> Reply to Message

Frank,

I have not found POINT_LUN in the code you sent and therefore see no problem with it. I wrote several file reading procedures for very convoluted file formats and had to liberally use cursor positioning information. I never did it the way you are doing it and don't know why is FSTAT field not working for you. I also missed the point of attempting to store positions in the code you sent, to start with. I guess you have a reason for that, which is not clear to me from the code clip. I also anticipate that the code you sent is undue slow due to re-creation of arrays and reading variables throug a string.

I know that using POINT_LUN worked perfectly fine for me for both setting and retrieving cursor position. I checked the code on Macintosh, Windows and UNIX and never had failures. You may want to try this approach if you really want to get positions, although I see you make no use of them in the code.

I tried to see what is your code doing here:

```
function rd_data_header
current_line=""
count=0

filen=pickfile()
openr,unit,filen,/get_lun

while not (EOF(unit)) do begin

    ; Read one line from the file
readf,unit,current_line

    ; If it is a comment line, remember the position at its end.
if (strpos(current_line,'#S ') eq 0) then begin
current_pos=fstat(unit) ; Try replacing with point_lun, -unit, current_pos

    ; For the very first time, do the following:
if (count=0) then begin
 result_header=current_line ; This is a string!
 result_pos=current_pos.cur_ptr
 count=1
endif

    ; All other times, do this:
```

```
if (count=1) then begin
    ; Produce array of size +1
 tmp=strarr(size(result_header,/n_elements)+1)
    ; Produce array of size +1
 tmp_pos=lon64arr(size(result_pos,/n_elements)+1)

    ; Place result_header in up to -1 elements of tmp.
tmp[0:size(result_header,/n_elements)-1]=result_header
    ; Place the position in up to -1 elements of tmp_pos
 tmp_pos[0:size(result_pos,/n_elements)-1]=result_pos

    ; record the same line in [1] position of tmp
 tmp[size(result_header,/n_elements)]=current_line
    ; Do the same with position
 tmp_pos[size(result_pos,/n_elements)]=current_pos.cur_ptr

 result_header=tmp
 result_pos=tmp_pos ; set tmp vars to current result

; Positions are stored and never used.
; Data not placed anywhere for future use. The result contains byte
;locations in file.
endif

endif

endwhile
free_lun,unit
result={header:result_header, position:result_pos}
return,result
end

************
```

It looks to me that the code needs to be heavily optimized.
Good luck,
Pavel

Subject: Re: point_lun question
Posted by Wayne Landsman on Tue, 02 Nov 1999 08:00:00 GMT
View Forum Message <> Reply to Message

Frank wrote:

> Hi,
> I have a data file with several measurements separated by some header
> lines followed by data points. The whole file containes many

> measurements. So I want to write a procedure, that reads the header and
> the position of the file pointer at the end of the header and returns
> this in a structure for further data processing.
> Actually the routine finds the header, however, point_lun returns the
> 'correct' value only for the first header. If I check the second
> position of the pointer using an hexditor the positions are not correct.
> Whats wrong ? Do I use the wrong data type
>

Are you running under Windows? If so, then you may need to add a /BINARY (or /NOAUTOMODE) switch to your openr statement.-- otherwise your use of the EOF() function can corrupt the file pointer. It has something to do with whether the file is being read in text mode or binary mode, as discussed in the Windows-specific documentation for openr. (However, the way that the EOF() function can corrupt the file pointer without these keywords present still feels like a bug to me.)

Peter Mason wrote a note on this problem to comp.lang.idl-pvwave last March.

--Wayne
Landsman
landsman@mpb.gsfc.nasa.gov
