Subject: Re: binary data files with ascii header
Posted by tisone on Tue, 08 Mar 1994 22:27:49 GMT
View Forum Message <> Reply to Message

In article <2lifom$ak3@tali.hsc.colorado.edu>, mtadams@columbine.hsc.Colorado.EDU (Matthew T. Adams) writes:
|> Hello, all.
|>
|> I'm trying to create binary data files that have an ascii header followed by a
|> string(12b) (a control-L, so that I can use 'more' to view just the headers), a
|> string(10b) (a return character), and then the binary data. For example,
|>
|> pro test_write
|> arr=bytarr(256,256,36) ; thirty-six 256x256 images
|> openw,unit,'test.dat',/get_lun
|> printf,unit,'header'
|> printf,unit,'information'
|> printf,unit,'here...'
|> printf,unit,string(12b)
|> writeu,unit,arr
|> close,unit
|> free_lun,unit
|> return
|> end
|>
|> Creating them seems to present no problem; it's in reading them that I get hosed:
|>
|> function test_read
|> openr,unit,'test.dat',/get_lun
|> str=''
|> readf,unit,str ; read string 'header'
|> readf,unit,str ; read string 'information'
|> readf,unit,str ; read string 'here...'
|>
|> ; now skip over the ^L and return characters...
|> char=0b
|> if not (eof(unit)) then repeat readu,unit,char until ((char eq 10b) or (eof(unit))
|>
|> ; file pointer should now be pointing to first byte of binary data...
|> if (eof(unit)) then begin
|>   print,'premature eof.'
|>   close,unit
|>   free_lun,unit
|>   return,-1
|> endif else begin
|>   arr=bytarr(256,256,36)
|>   readu,unit,arr
|>   close,unit

|>   free_lun,unit
|>   return,arr
|> endelse
|> end
|>
|> There is often no eof() error; all data seems to be read ok, until I step through
|> my 36 images. Each image is column shifted by some amount (the same for each
|> image).
|>
|> Any help? Is my error in combining printf & writeu or readf & readu?
|>
 I think one problem if I am not incorrect is that when you
write using a printf statement--It not only has written the above
string out plus a line feed character--When you read it back in--
A readf takes the line feed character into account while a readu
does not and reads only the number bytes that you tell it. You
write out the control-L with a printf which really consists of
two bytes being written out--not just one that you told the
following readu to do. To test this out--Look at an octal dump
in order to see what the real file that you are writing really
looks like in order to see what you are really trying to read.
My guess at this point is that you need to read an extra byte
in with the readu line that you have for reading the control-L
in your file--A thing like that can cause an alignment problem
in reading the file. Anyhow this is something to take a look at

Vic Tisone tisone@hao.ucar.edu
