Subject: Re: Lots of files
Posted by Foldy Lajos on Mon, 19 Mar 2007 17:01:31 GMT
View Forum Message <> Reply to Message

On Mon, 19 Mar 2007, Lasse Clausen wrote:

> On 19 Mar, 14:31, David Fanning <n...@dfanning.com> wrote:
>> David Fanning writes:
>>> Use POINT_LUN, -fin, position & Print, position.
>>
>> Whoops! Should be:
>>
>> POINT_LUN, fin, -position & Print, position
>
> no, first solution was correct, using negative file units reads
> pointer location rather than setting it.
>
> aaaanyway...
>
> As David suspected (to be honest, I had that feeling, too), the readf
> command reads from a file unit until it reaches a newline, then
> formats the string according to the format code given (IDL help says:
> On input, read data from the file and format it according to the
> format code.). Therefore my code was slow, because it read 4MB of data
> and then chucked away ... well, lots of it.
>
> If one uses readu instead, one cannot use format code ("The READU
> procedure reads unformatted binary data") but it just reads as many
> bytes as fit in the variable provided. Hence
>
> line = bytarr(1440)
> openr, fin, filename, /get_lun
> while not(eof(fin)) do begin
> readu, fin, line
> hstat = String(line[12:14])
> tmp = where(stats eq hstat)
> printf, tmp[0]+1, string(line)
> endwhile
>
> works like a charm, too. and please note, finally I am able to use the
> not(eof(fin)) condition.
>
> Oh, and by the way, the above program runs in 0.05 seconds, whereas
> the readf with format takes 75 seconds but that doesn't surprise
> anybody, I guess.
>
> Right, that sorted, let's get some work done...
> Cheers

> Lasse
>
>

Just two little comments:

1. always use ~ (logical negation) instead of NOT (bitwise negation) for eof(). Here is an excerpt from the manual:

Using the NOT Operator

Due to the bitwise nature of the NOT operator, logical negation operations should always use ~ in preference to NOT, reserving NOT exclusively for bitwise computations. Consider a statement such as:

```
IF ((NOT EOF(lun)) && device_ready) THEN statement
```

which wants to execute statement if the file specified by the variable lun has data remaining, and the variable device_ready is non-zero. When EOF returns the value 1, the expression NOT EOF(lun) yields -2, due to the bitwise nature of the NOT operator. The && operator interprets the value -2 as true, and will therefore attempt to execute statement incorrectly in many cases. The proper way to write the above statement is:

```
IF ((~ EOF(lun)) && device_ready) THEN statement
```

2. Formatted I/O is line-oriented, as you have discovered. So your 'printf, tmp[0]+1, string(line)' command actually writes 1441 characters (adds a new line at the end). Use writeu instead to keep the original structure (single line file).

regards,
lajos
