Subject: Re: openr and /get_lun
Posted by Craig Markwardt on Mon, 17 Apr 2000 07:00:00 GMT
View Forum Message <> Reply to Message

mallors@ips1.msfc.nasa.gov (Robert S. Mallozzi) writes:

> In article <onitxkz7p7.fsf@cow.physics.wisc.edu>,
> Craig Markwardt <craigmnet@cow.physics.wisc.edu> writes:
>>
>> I have noticed that the use of /GET_LUN and ERROR keywords to openr is
>> not as helpful as I would have hoped. Do other have this experience?
>> The problem is that when an error occurs, it is hard to know whether
>> the file unit was "gotten" or not.
...
>
>
> I guess I never thought about it too much, because if
> there is an error with the OPEN, then I should handle
> it somehow:
>
> OPENR, fl, 'nofile', /GET_LUN, ERROR = error
> IF (error NE 0) THEN BEGIN
> MESSAGE, /CONTINUE, 'Could not open file.'
> RETURN
> ENDIF
> .
> .
> .
> FREE_LUN, fl
>
>
> Otherwise, if you don't want to handle the error, you can just
> free the unit number conditionally, as I am sure you know:
>
> IF (error EQ 0) THEN FREE_LUN, fl

I totally agree that the error condition must be handled. What I was getting at is that OPENR, ..., /GET_LUN does two things: allocate a LUN, and open a file. If you get an ERROR condition back, it's impossible to know which of these two things failed. In fact in the above example you gave, the unit FL may be undefined, so FREE_LUN will fail.

David suggests using N_ELEMENTS(FL) to see if it's defined. That works, but only if that's the first time I use FL, something I didn't point out in my original example.

As the error checking got more detailed, I realized that it's easier

and takes less code to decouple the GET_LUN from the OPEN. Hence,

```
GET_LUN, fl
OPENR, fl, file, ERROR=err
IF error NE 0 then <Handle error>
FREE_LUN, fl
```

is guaranteed to work since FL is always defined.

> I sure wish we had a boolean datatype - the mistake of
> using something like "IF (NOT error) THEN" is one that
> is really a pain to find, although it certainly makes
> your code much more readable.

I agree with you there. OR, do we need boolean operators instead?
For example, a BNOT operator which takes the "logical" NOT instead of
the bitwise NOT,

```
NOT 0   -> 255      BNOT 0 -> 1
NOT 1   -> 254      BNOT 1 -> 0
```

Craig

--

 -- --------------
Craig B. Markwardt, Ph.D.   EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
 -- --------------