
Subject: Optimizing a lookup table

Posted by [ianpaul.freeley](#) on Sat, 03 Mar 2007 01:15:14 GMT

[View Forum Message](#) <> [Reply to Message](#)

I would like to write a function that uses a look-up table. Now the easiest (coding-wise), is to just save the look-up table and have the function restore it. However, I plan to be calling this function in a loop, so this would result in numerous unnecessary disk-reads. Anyway, what is the most efficient way to get a lookup table into an IDL function?

My thoughts so far:

1) Just print the look-up table to the screen, copy/paste into the function and add some brackets and commas, presto, variable is in the function and will stay loaded as long as the function is compiled. This will work for my present problem, but would be unwieldy for really large look-up tables and I worry about double-precision getting truncated on the print.

2) have a procedure that reads in the table and puts it in a common block, then just start my function with an if-statement to see if the common block exists and if not, call the reading procedure. My question here is, how can I check to see if a common block has already been created? I know I could call the common block maker outside the loop, but that seems lame and makes it more complicated if I want to share the code.

Unless someone comes up with something really witty, I'll just use option 1. Just seems like this would be a common problem that someone's solved before.

cheers,
IP Freeley
