Subject: Re: Floating illegal operand
Posted by Karl Schultz on Fri, 04 Oct 2002 15:36:40 GMT
View Forum Message <> Reply to Message

"Ed Wright" <ed.wright@jpl.nasa.gov> wrote in message
news:B9C1C7C2.15A8%ed.wright@jpl.nasa.gov...
>
> Time to discuss my favorite IDL subject, Floating illegal operand...
>
> Yesterday, I received an inquiry from a user of a DLM I wrote questioning
me
> about the following:
>
> IDL> cspice_pxform, 'GNS_GIM_CEM1', 'J2000', 74276223.927918d0, rotmat
> % Program caused arithmetic error: Floating illegal operand
>
> The procedure calculates a series of rotation matrices between two
arbitrary
> frames (in this case the Genesis vehicle and the J2000 inertial frame) at
a
> stated time.
>
> The output matrix, rotmat, was correct. Nothing wrong there. I wrote a
small
> C program, linked against the same library as the DLM, to reproduce the
> error. The program ran, produced the same numerical results to round off,
> but signaled no FPE error. Similar behavior for a FORTRAN version.
>
> Question: the signal originates in the C library against which I link the
> DLM. How does the IDL environment capture this type of signal?

IDL checks the machine's floating point status register according to the
setting of !EXCEPT. I do not think that IDL installs a signal handler for
SIGFPE. There is a bit more discussion in the docs for !EXCEPT, "Math
Errors", and CHECK_MATH.

> Should the C
> program show no signal?

I think that the answer depends on the environment you are in and what the
library is doing.

If I run a simple C console (not Win32 or MFC) program on Windows that does
a divide by zero, it just quietly continues on, inserting a NaN for the
result. Same goes for a couple of Unix platforms. There's a good chance
that the OS or C runtime is trapping the signal/interrupt and making this
happen.

The library you are calling may also be handling these conditions itself, but I think that is unlikely.

One thing you might try in your C program is to read the fp status register before and after you call the library and look for any changes. It is pretty easy to do this on all the IDL platforms. _statusfp is the function to look up on Windows.

If you convince yourself that the problem is in the library, and you don't want to or can't find it and fix it, I think that you could change the value of !EXCEPT before and after the call so that it is 0 during the call. This would prevent the message from appearing.

If the library checks out OK, then is it possible that your IDL code surrounding the call to the library is causing the error? If you set !EXCEPT to 2 , you can at least narrow it down to the IDL statement.

Actually, I'd probably start by setting EXCEPT=2 in the IDL code.

Hope this helps,
Karl
