Subject: Re: IDL_IDLBridge GetVar vs. Shared Memory
Posted by JD Smith on Tue, 05 Dec 2006 23:16:43 GMT
View Forum Message <> Reply to Message

On Tue, 05 Dec 2006 21:27:26 +0000, Dick Jackson wrote:

> Hi all,
>
> Using IDL_IDLBridge has been very helpful in some of our work lately, helping us
> to get more processing done while one IDL process is waiting on a DLL, and other
> fun. I thought I'd share some findings that show how results can come back 4-10
> times faster using shared memory (SHMMAP, etc.) rather than GetVar(). Any
> comments on this are very welcome, and I'd be interested to see how this plays
> out on other hardware.

Thanks Dick, looks very interesting.

I tried it, and shared memory definitely seems to be much faster for me,
up to 10x. I also added back your 1E8 case, and got this unhelpful error:

```
IDL> shmvsgetvar
% Loaded DLM: IDL_IDLBRIDGE.
Testing with byte array of size: 1E8
% IDL_IDLBRIDGE Error: Undefined variable: 'a'
% Execution halted at: SHMVSGETVAR       39
  /home/jdsmith/idl/test/test_shm_iib.pro
%                    $MAIN$
```

Any reason why you can't create and retrieve a 100MB array? Then I
recalled advice from earlier this week:

```
 echo 1073741824 > /proc/sys/kernel/shmmax
```

After increasing this limit, it ran fine. The default is 32MB in
Linux/Intel.

What's quite interesting, however, is that I noticed the SHMMap version
didn't run into this limit. Without any modification to shmmax, it would
still happily send 1GB across shared memory. It wasn't abundantly clear
why IDL_IDLBridge's GetVar hits this limit, while SHMMap does not.

I thought perhaps the Bridge uses SYSV shared memory segments (whereas
the latter defaults to the more modern POSIX flavor), I modified the
code to read:

```
   SHMMap, shmName,/Byte,Dimension=sizeNum,/SYSV
```

and sure enough, hit the 32MB limit quickly. Also, somewhat

strangely, with SYSV shared memory, only 0b was returned from the shared memory segment. Then I realized you need an "OS handle" for SYSV, but not for POSIX, since POSIX shared memory just uses the supplied name with a slash. After passing OS_HANDLE back to the IDL Bridge, it worked fine. The speed was almost identical (actually SYSV shared memory seems about 5% faster).

So, it appears that:

1) POSIX shmem "does the right" thing, without any arbitrary limits on the amount of shared memory.

2) SYSV shmem runs into a maximum memory limit defined by your kernel.

3) IDL_IDLBridge uses SYSV shared memory under UNIX to implement GetVar, thus inheriting this limit.

It's not at all clear why they use SYSV shmem for the Bridge, since they go on about how important POSIX shared memory is and how all decent Unixes support it, etc.

The bottom line is, not only is it much faster to use SHMMap (without /SYSV), but it's probably more portable and obviously less likely to run into arbitrary memory boundaries. So if you plan on passing any data larger than a few MB through the IDL_IDLBridge, do look into it.

Thanks,

JD
