

Re: dumping call stack in c++

Source:

<http://www.tech-archive.net/Archive/VisualStudio/microsoft.public.vstudio.development/2005-05/msg00002.html>

- *From:* "Steve Muccione" <home*DOT*muccione@xxxxxxxxxxx>
 - *Date:* Mon, 02 May 2005 12:11:15 GMT
-

Hey Paul,

If you haven't done so you might also want to check out SoftICE from Compuware (I don't know if you can but it seperately any more or if you have to buy the whole device driver development kit). It supports remote debugging, but it also (and here's the best part) allows you to fully debug into the windows kernel. It let's you trap events before windows even sees them (any and all faults are trappable). It's quite a handy tool for debugging at that level (although since you have bounds checker you're probably familiar with it, but I don't know if you've given it a try).

best,

steve

ps: You the Paul Davis from rgvac? Are you working on a new multi-board? One of Clays or something new? (I won't tell)

"Paul Davis" <paul.davis@xxxxxxxxxxxxxxxxxxxxxxxx> wrote in message [news:d4nsjm\\$si8\\$1@xxxxxxxxxxxxxxxxxxxxxxxx](mailto:news:d4nsjm$si8$1@xxxxxxxxxxxxxxxxxxxxxxxx)

> Well I think I have discovered where my problem might lay. One of the
> major design faults in the hardware we are developing for (I wish I had a
> say in the design it would never have been this bad) is that we have an
> ati radeon 9000 onboard which shares conventional memory. It does not
> have its own :(I think somewhere there is a problem with memory being
> flooded by the gfx hardware and windows is not protecting this.

>
> This goes back to my point about windows memory manager abilities of
> windows :)

>
> I noted your points Phil, but I have seen a lot of the base code for areas
> of windows and I stand by my comments about the memory manager etc.

>
>
>
> "Philip Taylor (MS)" <~ptaylor@xxxxxxxxxxxx> wrote in message
> news:eXcKIrsSFHA.2128@xxxxxxxxxxxxxxxx forphx.gbl...

>>

Re: dumping call stack in c++

>>>
>>> If windows memory manager etc had been written correctly and the os was
>>> a true multitasking system then this sort of thing could be avoided but
>>> as windows is a very unstable platform crashes like this that result in
>>> total lockup do happen :(it then becomes a major problem in finding
>>> such bugs.
>>>
>>
>> its an aside, but usually this has nothing to do with how the memory
>> manager was written or how multitasking works. NT based OS's do have true
>> multitasking, which XP Embedded is.
>>
>> I infer from your comments re game development and coin mechs that you
>> are doing something DX related or graphics related on this embedded
>> system?
>>
>> crashes like this usually are root-caused from the decision to move the
>> graphics layer into the kernel for performance, so if an app that uses DX
>> or the graphics driver does something bad, it does indeed have a chance
>> to hose the system. the original NT design had that subsystem in
>> user-land which freed the OS from hard crashes due to the graphics
>> system, but the round-trips to the kernel by the OS cost so much in
>> performance that the cost of the change was deemed worth the benefit.
>>
>> when you are doing DX graphics, you are directing the API and the DDI
>> under the covers to write directly into registers on the hw. that does
>> give a lot of rope. graphics IHVs are starting to add fault-detection and
>> restart capability in their hw, and this is a spec-ed feature for
>> Longhorn. that should drastically reduce the hard locks from errant
>> graphics apps.
>>
>> can you use a different graphics card/driver combo, from a different IHV?
>> that may reduce the bad effect of this crash such that you can debug.
>> another thing to try is does XPe have a DX debug runtime? and a debug
>> output logger, if it has a "log to file" option?. then run using the
>> debug runtime, turn the debug level to high, capture to a file, and at
>> least you have an offline trace that might help.
>>
>> another thing is to try to recreate the same hw environment on a desktop
>> pc and see if you can recreate the graphics crash. if so, then you could
>> get more traction in debugging.
>>
>> not much help, but maybe some.
>>
>>
>>
>
>

- Prev by Date: [*newbie: trying to make an ole or ocs control*](#)
- Next by Date: [*Functions exported in C++ and global variables;*](#)
- Previous by thread: [*newbie: trying to make an ole or ocs control*](#)
- Next by thread: [*Functions exported in C++ and global variables;*](#)
- Index(es):
 - ◆ [*Date*](#)
 - ◆ [*Thread*](#)