

## Re: \Storage Card doesn't always reload when ATADISK reloaded

---

*Source:*

<http://www.tech-archive.net/Archive/WindowsCE/microsoft.public.windowsce.platbuilder/2005-06/msg01405.html>

---

- *From:* Andrew Worsley <[andrew.worsley@xxxxxxxxxxx](mailto:andrew.worsley@xxxxxxxxxxx)>
  - *Date:* Thu, 30 Jun 2005 09:03:40 +1200
- 

Thanks Steve,

Yes, I have the 2004 yearly QFE and 2005 1st quarter QFE's applied.  
Can you give me any details of your workaround please?

Having trolled through the usenet archives, I think my problem may be related to the PnPUnloadDelay registry setting, which is currently set to the default value of 0x1000.

As far as I can tell, this setting is used to delay the dismount of a filesystem after the underlying block driver goes away.

With reference to the following thread (which you participated in...):

[http://groups-](http://groups-beta.google.com/group/microsoft.public.windowsce.talisker.techpreview/browse_thread/thread/da42dae49880a1a5/8fd41b032de2ca4f?q=pnpunloaddelay&rnum=24&hl=en)

[beta.google.com/group/microsoft.public.windowsce.talisker.techpreview/browse\\_thread/thread/da42dae49880a1a5/8fd41b032de2ca4f?q=pnpunloaddelay&rnum=24&hl=en](http://groups-beta.google.com/group/microsoft.public.windowsce.talisker.techpreview/browse_thread/thread/da42dae49880a1a5/8fd41b032de2ca4f?q=pnpunloaddelay&rnum=24&hl=en)

It seems that this functionality was introduced to allow for a block driver to be shut down at suspend time, then restarted on resume, with the associated filesystem "riding through" the change (the caveat being don't access the files until the block driver has reloaded).

In our case, we don't support suspend/resume (our power down really does power down, so we could boot the next time around), so that isn't a problem for us.

However, our sequence needs to be a little different in that we need to remove the filesystem before the block driver goes away, in order to ensure that any filesystem caches are written to the card before someone removes it from the slot.

I think the problem is that when I call DismountStore(), it doesn't actually remove its store object until after the PnPUnloadDelay timeout expires.

In the mean time, if I unload and reload ATADISK (as described in my original post), the MountStore() function (which gets called when

Re: \Storage Card doesn't always reload when ATADISK reloaded

ATADISK reloads) still finds the "old" store, and just reattaches it, rather than adding a new one. I believe the filesystem doesn't reappear because this code is assuming it never went away (in the suspend/resume cycle, I guess there is no explicit call to DismountStore(), just a pair of PnP notifications that ATADISK goes away, then comes back again, right?).

Anyway, I'm going to try setting the PnPUnloadDelay timeout to zero, which I believe will solve the problem. Of course, this is only a potential solution for us since we don't need to suspend/resume, which obviously isn't the typical case.

Andrew

In article <e0LEdDLfFHA.3316@xxxxxxxxxxxxxxxxxxxxxxxx>, nospam1@EntelechyConsulting.com says...

> Qfe's applied? There was a known issue with this that we worked with PSS to  
> come up with a workaround for in V4.2. The use of xxx\_Preclose and  
> xxx\_PreDeinit in V5.0 resolves the problem.  
>  
>  
.

---

• **Follow-Ups:**

- ◆ **Re: \Storage Card doesn't always reload when ATADISK reloaded**  
◇ From: Steve Maillet (eMVP)
- ◆ **Re: \Storage Card doesn't always reload when ATADISK reloaded**  
◇ From: Andrew Worsley

• **References:**

- ◆ **\Storage Card doesn't always reload when ATADISK reloaded**  
◇ From: Andrew Worsley
- ◆ **Re: \Storage Card doesn't always reload when ATADISK reloaded**  
◇ From: Steve Maillet (eMVP)

- Prev by Date: **Re: Closing a thread that is waiting for Irq**
- Next by Date: **Re: \Storage Card doesn't always reload when ATADISK reloaded**
- Previous by thread: **Re: \Storage Card doesn't always reload when ATADISK reloaded**
- Next by thread: **Re: \Storage Card doesn't always reload when ATADISK reloaded**
- Index(es):
  - ◆ **Date**
  - ◆ **Thread**