

Apple II Technical Notes



Developer Technical Support

GS/OS

#12: All About Notify Procs

Written by: Matt Deatherage
September 1990

This Technical Note discusses the GS/OS notification procedure new to System Software 5.0 and enhances the discussion of these procedures in the Addison-Wesley *GS/OS Reference*.

Why Do I Want To Be Notified?

GS/OS notification procedures (or “notify procs”) are handy ways to let the operating system tell you when interesting things are happening. As documented in *GS/OS Reference*, they can tell you when you’re switching to ProDOS 8 (and back), when disks are inserted or ejected, when GS/OS is shut down, and even when a change occurs to a volume.

However, getting these notifications is not as simple as installing a procedure. Some behaviors are due to the way device drivers are designed and some are due to the design of GS/OS or device hardware. This Note discusses a few slightly unusual situations you can encounter when dealing with notification procedures.

I Get “Parameter out of range,” and There’s Only One Parameter

It seems incongruous to get error \$0053 (“Parameter out of range”) when there’s only one parameter, a pointer to the notification procedure. However, GS/OS checks the procedure header to ensure consistency. In particular, the `flags` field must not have any of the reserved bits set. Having any bits other than one through six set results in error \$53; it ensures you do not get strange behavior or are not passed values you cannot comprehend.

I'm Not Getting Notified

You've written your notification procedure correctly and tested it, but when you run your application you can eject and insert disks until your arm falls off and your code is never called.

This is a side effect of the design of most Apple II peripherals—no hardware interrupt is generated when you eject a disk. Without an interrupt to grab the CPU's attention, the drive just sits there until someone actually asks the drive if a disk is present.

Well-designed GS/OS drivers look to see if a disk has been switched every time they get control and call the System Service routine `SET_DISKSW`, which in turn causes the notification procedures to be told the disk has been switched. However, the driver cannot set this chain in motion until it gets control.

The easiest way to do this is to loop through all on-line devices, issuing a device call to each in turn. When the driver gets control, it starts the ball rolling. Note that you must make a device call that actually causes driver code to be executed. This includes all the application level device calls with less than two parameters, except `DRename` and `DInfo` (the third parameter is a block count, which causes a `Driver_Status` call to the driver). These calls are handled entirely by the Device Manager without actually transferring control to any driver code. `DStatus` with a `transferCount = 2` is a good choice.

I Get Notified About Insertion at Weird Times

When coming back to GS/OS from ProDOS 8, you get “insertion” notification even though no disks have actually been inserted. This is done for you by most drivers, which pretend that any media in the device has just come online at driver startup time—which is true as far as any application is concerned.

General Truths

Be careful when installing notification procedures from an application. Applications either go away or are made purgeable when they quit, and that means your notification procedure can get disposed. GS/OS tries to call the address anyway, and this is generally a bad idea. Make sure you remove all notification procedures before their code goes away.

Even though you have to poll to ensure you get disk insertion and ejection events, it’s still useful to install notification procedures. The notification queue allows **everyone** who’s interested in GS/OS events to be notified about them. Check the “disk has been switched” bit of the status word is not suitable, because this bit is only set once. If a desk accessory makes a status call to a switched device, it sees the “disk has been switched” bit and your application does not, so use the notification queue.

Operating system calls (i.e., `Write`) can generate volume changed events during execution; therefore, GS/OS could be busy when it calls your notification procedure. Volume changed events are not necessarily generated immediately. The AppleShare FST checks for volume changes approximately every 10 seconds, but it only generates these events for a given volume if it contains an open folder.

GS/OS can call your notification procedure from inside an interrupt, so make it short and sweet. One approach is setting a flag which you can check periodically from your main code; when the flag is set, you can process the event and clear the flag.

Further Reference

- *GS/OS Reference*