

New Technical Notes

Macintosh



Developer Support

A5 Within Trap Patches - Don't Depend on It

Overview

Revised by:

March 1988

Written by: Bryan Stearns

June 1986

Future software may allow desk accessories to have their own globals by changing register A5 when the accessory is entered and exited. This can cause problems for applications that patch traps without following certain rules.

If your application patches any traps, it's important that the patches not depend on register A5. This is because you may have intercepted a trap used by a desk accessory.

If you need access to your globals within your patch, you can save A5 (on the stack, perhaps), load A5 from the low-memory global `CurrentA5` (this is guaranteed to be correct for your application), do whatever you have to do within your patch, then restore A5 on the way out. Note that if you make any traps within your patch (or call the "real" version of the routine you patched), you should restore the caller's A5 before doing so.

There are several ways of depending on A5 within a patch that you should watch out for:

- Are you making any references to your global variables, or those of any units that you're using, such as `thePort` from QuickDraw? These are accessed with A5-relative references with negative offsets.
- Are you making any inter-segment subroutine calls? These are accessed with A5-relative references with positive offsets.
- Are you using any system calls (either traps or "glue" routines) which will depend on A5 during their execution? In this case, you need to be sure that you restore the caller's A5 before executing the call.

To be safest, patched traps should follow the same rules as interrupt handlers.

Note: In general, applications should not have to patch any traps, and risk compatibility problems if they do! If you'd like help in removing your dependence on patching, please contact Macintosh Developer Technical Support.

Further Reference:

- The Operating System Utilities