

Technical Note TN1020

Color Cursing: Two Major Causes

CONTENTS

[Using Color Cursors](#)[References](#)[Change History](#)[Downloadables](#)

The creation and use of color cursors is discussed in the chapter on cursor utilities in [Inside Macintosh: Imaging with QuickDraw](#) .

If you're building an application that uses color cursors, you may encounter some quirks present in Color QuickDraw. This Technote provides solutions to a few common problems.

[Dec 23 1997]

Using Color Cursors

To avoid system errors or crashes while using a color cursor, incorporate the following information in your application development.

Setting Bounds for your Cursor

If your cursor is, for example, 15 pixels tall and 9 pixels wide, you might be tempted to use these values for the `bounds.bottom` and `bounds.right`, respectively, in your cursor's pixel map. *Don't* . When the cursor's image needs to be expanded (that is, when you specify a two bit-per-pixel cursor and the mouse pointer is on an eight-bit screen) the `SetCCursor` trap rounds the width of the pixel map in such a way that you'll get only the space required for a 15 by 8 pixel map allocated for the expanded cursor data. When the cursor's image is expanded into this too-small expanded cursor data handle as a 15 by 9 pixel map, something in your heap will get munched.

To avoid this problem, *always* specify the `pixmapHandle^^.bounds` to be 16 by 16. This will cause `SetCCursor` to properly allocate the expanded data area. Since the amount of data *drawn* for a cursor is specified by the cursor's pixel values and 'clut' resource, trying to save a few bytes by making the bounds rectangle smaller than 16 by 16 won't be very helpful anyway.

Inopportune Purging of a `CLUT' resource

If you load a color cursor's color table from a 'clut' resource using `GetCTable`, make sure that the 'clut' is marked non-purgeable while the color cursor is in use. If you don't take this precaution, bombs will occur if your 'clut' gets purged at an inopportune time.

For more information, see [Inside Macintosh: Imaging with QuickDraw](#) , "Cursor Utilities."

Calling `SetCCursor` can change the `GDevice`

Starting with System 7.5.2, the cursor images are copied into offscreen GWorlds via `CopyBits`. To do this, `SetCCursor` saves and restores the port and device using a fairly standard set of code:

```

{
    GetGWorld (&savePort, &saveDevice);
    SetGWorld (offPort, NULL);
    ...copybits and other stuff goes here

    SetGWorld (savePort, saveDevice);
}

```

A documented side effect of `SetGWorld` is that the device parameter will be ignored when you pass in a GWorld port. So, if the current port is a GWorld port, the save-restore code in `SetCCursor` will always make the current device that of the GWorld.

One place where this is especially dangerous is when the current port is a disposed GWorld. `DisposeGWorld` sets the current `GDevice` to a safe device, but doesn't affect the current port. If you then call `SetCCursor` before a valid port has been set, then `SetCCursor` will cause the current `GDevice` to point at the disposed `GDevice`, and the system will crash shortly afterwards.

The easiest way to avoid this side-effect of `SetCCursor` is to:

- Ensure that your port and device always match. Never allow a situation where the port and device can be mismatched. This helps cure many problems unrelated to color cursors.
- Never dispose of the current port without setting a valid QuickDraw port first. Drawing to a disposed port or device will most likely result in a crash.

Don't Call `SetCCursor` at Interrupt Time

`SetCCursor` is documented as moving memory. In 7.5.2 and later, `SetCCursor` makes calls to `CopyBits` and other QuickDraw routines. Never, ever call `SetCCursor` at interrupt time! Calling `SetCursor` at interrupt time is still allowed, however.

References

[*Inside Macintosh: Imaging with QuickDraw*](#)

[Back to top](#)

Change History

01-June-1989	Originally written.
01-February-1996	The warning about purgeable 'clut' resources added.
23-December-1997	Warnings about <code>SetCCursor</code> changing the <code>GDevice</code> , and <code>SetCCursor</code> moving memory added.

[Back to top](#)

Downloadables



Acrobat version of this Note (K).

[Download](#)

[Back to top](#)

Technical Notes by [API](#) | [Date](#) | [Number](#) | [Technology](#) | [Title](#)
[Developer Documentation](#) | [Technical Q&As](#) | [Development Kits](#) | [Sample Code](#)