To be published in the
July 1993 issue of Silicon Graphics' *DevNotes*.

A Three Color Cursor for X

Mark J. Kilgard
Silicon Graphics Inc.
Revision : 1.4

May 21, 1993

1 Introduction

Graphics hardware built by Silicon Graphics supports three color cursors (excepting some old models of the Personal IRIS). Most X programmers are aware that the X Window System supports two color cursors. And IRIS GL programmers may be aware that the GL supports three color cursors.

From a hardware point of view, a three color cursor makes sense. The cursor generation hardware requires an image (usually 32 by 32 pixels). The image needs two colors to support X properly and some way to represent a transparent value. To hold these three values, two bits are required per pixel. But two bits allows four values to be represented so it is natural to use the left over value as a third cursor color.

To an application programmer, a third cursor color can allow more identifiable cursor icons to be generated. For example, a pencil icon could be drawn in yellow with a gray lead and a pink eraser for added realism.

This article describes how a three color cursor can be generated using X on Silicon Graphics workstations. A previously undocumented `XSGIMiscSetThirdCursorColor` routine is documented which allows the third color of an X cursor to be specified. The routine uses SGI's proprietary `SGI-SUNDRY-NONSTANDARD` extension. This interface is the same mechanism used by the IRIS GL's internal cursor management routines to implement three color cursors in the GL.

OpenGL developers are already aware that X is how non-3D rendering tasks are performed. This means OpenGL developers should anticipate using X to generate three color cursors.

Note that this mechanism is not part of the X standard and is only supported on Silicon Graphics X servers. But it is easy to query if a given X server supports it or not so programs can be written to use three color cursors if available. And if not, programs can fall back to use two color cursors for less capable X servers.

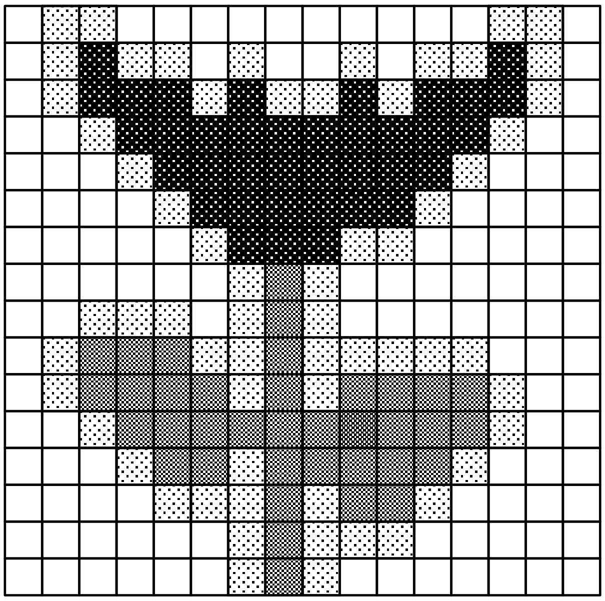

Figure 1: A 16x16 three color cursor tulip.

2 Normal X Cursors

Normally in X, a cursor can be specified on a per window basis. When the cursor is moved into a window, the window's cursor is displayed. If a window does not specify a cursor, the cursor is inherited from its parent.

Creating a cursor involves specifying a cursor image, a foreground and background color, and a hotspot. The hotspot defines the tracking position of the cursor. For example, the hotspot might be the tip of an arrow or center of a crosshair. The image can be supplied either as two bitmaps or two glyphs from a font. The cursor colors are specified as red-green-blue triples.

The two bitmaps (or glyphs) are referred to as the *source* and *mask* images for the cursor. The color value for a pixel in the cursor is determined by these images. The mask image defines the shape of the cursor. If a mask pixel is set, the foreground or background pixel color is displayed depending on if the source image is set or not.

The standard Xlib API supplies the following routines to implement cursor functionality:

`XCreateFontCursor` creates a cursor from the standard X cursor font.

`XCreateGlyphCursor` creates a cursor from an arbitrary font.

`XCreatePixmapCursor` creates a cursor from two supplied bitmaps.

`XDefineCursor` allows the cursor for a window to be set.

`XUndefineCursor` tells a window to resume inheriting its cursor from its parent.

`XRecolorCursor` allows the foreground and background colors for a cursor to be changed.

`XQueryBestCursorSize` returns the best size for a cursor on a given screen.

Note that only two color cursors are supported by the standard Xlib interface.

3 A Third Color

On Silicon Graphics workstations, a proprietary X extension named `SGI-SUNDRY-NONSTANDARD` but often referred to as the SGI Misc extension provides facilities not part of X but needed for IRIS GL functionality and/or to take advantage of proprietary capabilities.

One of the supported requests allows a third color to be specified for a cursor. The **`XSGIMiscSetThirdCursorColor`** routine requests the X server to supply a third color for a normal two color X cursor. The routine is part of the X extension library. Programs link with the library by specifying `-lXext` on their link line. The routine has the following interface:

```
#include <X11/Xlib.h>
#include <X11/extensions/SGIMisc.h>

Status
XSGIMiscSetThirdCursorColor(
   Display *display,
   Cursor cursor,
   XColor *color)
```

The routine returns 1 if successful or 0 if the SGI Misc extension is not available. `display` is a normal X display pointer. `color` specifies a red-green-blue triple. `cursor` specifies a cursor XID usually generated by one of the Xlib cursor creation routines.

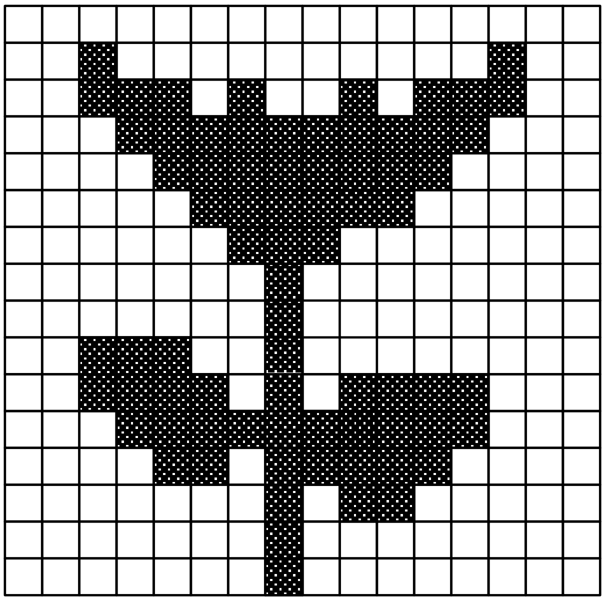

Figure 2: Foreground (source) image for a 16x16 three color cursor tulip.

So when does the third color get displayed? It is displayed if a pixel in the source image is set but the corresponding pixel in the mask is *not* set. Normally, a transparent cursor pixel is generated where ever a mask pixel is not set.

Since many developers are interested in writing X applications portable to other X servers, a routine is available to query if the SGI Misc extension is available (you should only expect to find it supported on SGI X servers). The routine is **`XSGIMiscQueryExtension`** and has the following interface:

```
Bool
XSGIMiscQueryExtension(
   Display *dpy,
   int *event_basep,
   int *error_basep)
```

It returns `True` if the extension is supported, `False` otherwise. The `event_basep` and `error_basep` parameters are used to return the extension event and error bases. These return values are unimportant for using the three color cursor functionality.

4 A Tulip Cursor Example

As an example of how to use a three color cursor, Appendix A supplies sample code for a program which changes the cursor for the root window into a small tulip with red petals, green leaves, and a white outline. Figure 1 shows the 16 by 16 tulip cursor shaded to indicate the various colors.

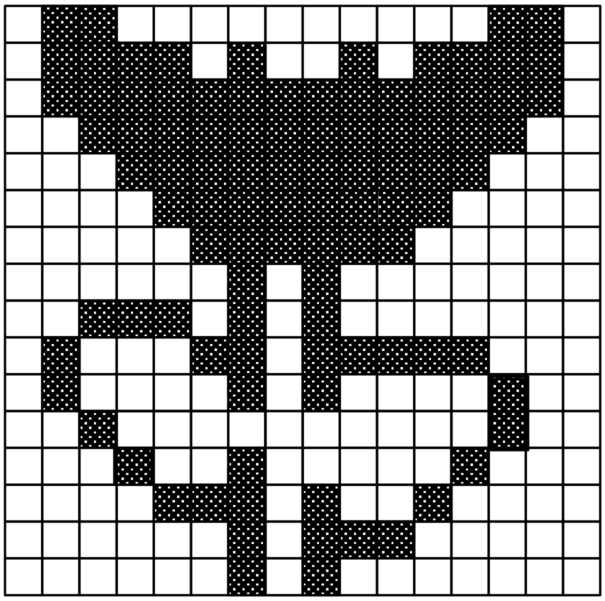

Figure 3: Background (mask) image for a 16x16 three color cursor tulip.

Figures 2 and 3 show the source and mask images for the cursor. Notice that the stem and leaves of the tulip are drawn in the source image but not in the mask image. This will allow the stem and leaves to be shown in green, the third cursor color.

To create the cursor image, the `bitmap` utility supplied with IRIX can be used to graphically generate the image data. Note you will need to generate two bitmaps.

Hopefully SGI developers will find three color cursors a useful way to add extra visual sizzle to X and OpenGL applications.

A tulip.c

```
/*
 * tulip - Turns root cursor into three-color tulip.
 *         COMPILE: cc -o tulip tulip.c -lXext -lX11_s
 */

#include <X11/Xlib.h>
#include <X11/extensions/SGIMisc.h>

/*
 * Basis for constructing a three color cursor:
 *
 * Cursor plane 0 is date ("foreground")
 * Cursor plane 1 is mask ("background")
 *
 * X server color display truth table:
 *   mask(bit1) source(bit0)  color
 *       0        0          transparent
 *       0        1          third color
 *       1        0          background
 *       1        1          foreground
 *
 * WARNING: Not all SGI hardware supports three-color cursors (ie,
 *          early Personal IRIS models).
 */

/*
 * Tulip 16x16 bitmaps (foreground and background) generated via bitmap.
 */

#define tulip_fg_width 16
#define tulip_fg_height 16
static char     tulip_fg_bits[] = {
    0x00, 0x00, 0x04, 0x20, 0x5c, 0x3a, 0xf8, 0x1f, 0xf0, 0x0f, 0xe0, 0x07,
    0xc0, 0x01, 0x80, 0x00, 0x80, 0x00, 0x9c, 0x1e, 0xbc, 0x1f, 0xf8, 0x0f,
    0xb0, 0x06, 0x80, 0x00, 0x80, 0x00, 0x80, 0x00};

#define tulip_bg_width 16
#define tulip_bg_height 16
static char     tulip_bg_bits[] = {
    0x06, 0x60, 0x5e, 0x7a, 0xfe, 0x7f, 0xfc, 0x3f, 0xf8, 0x1f, 0xf0, 0x0f,
    0xe0, 0x07, 0x40, 0x01, 0x5c, 0x1f, 0x62, 0x21, 0x42, 0x20, 0x04, 0x10,
    0x48, 0x09, 0x70, 0x07, 0x40, 0x01, 0x40, 0x01};

main()
{
    Display        *dpy;
    Window          root;
    Pixmap          fg_pixmap, bg_pixmap;
    Cursor          cursor;
    XColor          red, white, green;
    Status          status;
    Colormap        cmap;
    int             dummy;

    dpy = XOpenDisplay(NULL);
    if (dpy == NULL) {
        fprintf(stderr, "tulip: could not open display\n");
        exit(1);
```

```
    }
    if (!XSGIMiscQueryExtension(dpy, &dummy, &dummy)) {
        fprintf(stderr, "tulip: SGI-SUNDRY-NONSTANDARD extension required\n");
        exit(1);
    }
    root = DefaultRootWindow(dpy);
    cmap = DefaultColormap(dpy, DefaultScreen(dpy));
    status = XParseColor(dpy, cmap, "orangered", &red);
    status = status && XParseColor(dpy, cmap, "white", &white);
    status = status && XParseColor(dpy, cmap, "darkgreen", &green);
    if (status == 0) {
        fprintf(stderr, "tulip: unable to parse colors\n");
        exit(1);
    }
    fg_pixmap = XCreateBitmapFromData(dpy, root, tulip_fg_bits,
                                      tulip_fg_width, tulip_bg_height);
    bg_pixmap = XCreateBitmapFromData(dpy, root, tulip_bg_bits,
                                      tulip_bg_width, tulip_bg_height);
    cursor = XCreatePixmapCursor(dpy, fg_pixmap, bg_pixmap, &red, &white,
                                 tulip_fg_width / 2, tulip_bg_height / 2);
    XSGIMiscSetThirdCursorColor(dpy, cursor, &green);
    XDefineCursor(dpy, root, cursor);
    XCloseDisplay(dpy);
    printf("to restore: xsetroot -cursor_name X_cursor -fg red -bg white\n");
    exit(0);
}
```