

```
#include <linux/kd.h>
#include <stdint.h>
#include "vcio.h"
#include <time.h>

// 'global' variables to store s
int fbfd = 0;
char *fbp = 0;
struct fb_var_screeninfo vinfo;
struct fb_fix_screeninfo finfo;
int i = 0;
int xres = 0;
int yres = 0;
int xres_virtual = 0;
```

Pseudo-random ramblings about programming and other geeky stuff

Low-level Graphics on Raspberry Pi [part X]

- Low-level Graphics on RPi

- Raspberry Pi
- Python

```

    x = x + dx;
    y = y + dy;

    // check for display sides
    if ((x < 0) || (x > (vinfo.xres - w))) {
        dx = -dx; // reverse direction
        x = x + 2 * dx; // counteract the move already done above
    }
    // same for vertical dir
    if ((y < 0) || (y > (vinfo.yres - h))) {
        dy = -dy;
        y = y + 2 * dy;
    }

    usleep(1000000 / fps);
    // to be exact, would need to time the above and subtract...
}

}
...

```

Save as `fbtestX.c` (complete code in [GitHub](#)) - build with `make fbtestX`. This should give us a moving white rectangle that bounces off the screen sides... Unfortunately the updates are not smooth (at least on most displays) - there is a quite prominent **tearing effect**.

Linux framebuffer **interface** does define some methods to overcome this - we could make the framebuffer virtual size double the height of the (smaller) physical one using the `FBIOPUT_VSCREENINFO` call:

```

// Set variable info
vinfo.xres = 640; // try a smaller resolution
vinfo.yres = 480;
vinfo.xres_virtual = 640;
vinfo.yres_virtual = 960; // double the physical
vinfo.bits_per_pixel = 8;
if (ioctl(fbfd, FBIOPUT_VSCREENINFO, &vinfo)) {
    printf("Error setting variable information.\n");
}

//long int screensize = vinfo.xres * vinfo.yres;
// have to use the virtual size for the mmap...
long int screensize = vinfo.xres_virtual * vinfo.yres_virtual;

```

And change our drawing loop to use the two halves of the virtual buffer using a call to `FBIOPAN_DISPLAY` for **page-flipping** tied to a **vertical sync** using `FB_ACTIVATE_VBL`:

```

int vs = 0;
// initially show upper half (0) - draw to lower half (1)
int cur_half = 1;
for (i = 0; i < 1000; i++) {

    fill_rect(x, y, 40, 40, 4);

    x = x + dx;
    y = y + dy;

    if ((x < 0) || (x > (vinfo.xres - 40))) {
        dx = -dx;
        x = x + 2 * dx;
    }
    if ((y < 0) || (y > (vinfo.yres - 40))) {
        dy = -dy;
        y = y + 2 * dy;
    }

    // switch page
    vinfo.yoffset = cur_page * vinfo.yres;
    vinfo.activate = FB_ACTIVATE_VBL;
    if (ioctl(fbfd, FBIOPAN_DISPLAY, &vinfo)) {
        printf("Error panning display.\n");
    }
}

```

```
}
```

Unfortunately these calls have not been implemented in the [RPI framebuffer driver](#) (does not seem to be in later versions either yet, see also <http://www.raspberrypi.org/phpBB3/viewtopic.php?f=67&t=19073>). So running the above code ([full code](#)) results in repeated output of `Error panning display.` and the rectangle flashing even worse (as we miss every second screen update by drawing outside of the visible area).

[Continued in [next part](#)]

[UPDATE on changes in the fb driver [here](#)]

Posted by [Unknown](#) at [12:19](#)



Labels: [C](#), [graphics](#), [Linux](#), [Raspberry Pi](#)

1 comment:



Raspberry Compote Wednesday, 19 March 2014 at 18:06:00 GMT

Switching the `ioctl(fbfd, FBIOPAN_DISPLAY, &vinfo)` to `ioctl(fbfd, FBIOPUT_VSCREENINFO, &vinfo)` seems to work slightly better - it does flip the page to the yoffset, but unfortunately it also redraws the console buffer, so one may end up with flashing directory listing or whatever was on display when starting the tester app...

[Reply](#)



Comment as:

Lhunden (Google+)

[Sign out](#)

[Publish](#)

[Preview](#)

☐ [Notify me](#)

Note: only a member of this blog may post a comment.

[Newer Post](#)

[Home](#)

[Older Post](#)

Subscribe to: [Post Comments \(Atom\)](#)