

file: media/v4l/v4l2grab.c

```
/* V4L2 video picture grabber
Copyright (C) 2009 Mauro Carvalho Chehab <mchehab@kernel.org>

This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation version 2 of the License.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <fcntl.h>
#include <errno.h>
#include <sys/ioctl.h>
#include <sys/types.h>
#include <sys/time.h>
#include <sys/mman.h>
#include <linux/videodev2.h>
#include "../libv4l/include/libv4l2.h"

#define CLEAR(x) memset(&(x), 0, sizeof(x))

struct buffer {
    void *start;
    size_t length;
};

static void xioctl(int fh, int request, void *arg)
{
    int r;

    do {
        r = v4l2_ioctl(fh, request, arg);
    } while (r == -1 && ((errno == EINTR) || (errno == EAGAIN)));

    if (r == -1) {
        fprintf(stderr, "error %d, %s\n", errno, strerror(errno));
        exit(EXIT_FAILURE);
    }
}

int main(int argc, char **argv)
{
    struct v4l2_format          fmt;
    struct v4l2_buffer          buf;
    struct v4l2_requestbuffers  req;
    enum v4l2_buf_type          type;
    fd_set                      fds;
    struct timeval              tv;
    int                          r, fd = -1;
    unsigned int                i, n_buffers;
    char                         *dev_name = "/dev/video0";
    char                         out_name[256];
    FILE                         *fout;
    struct buffer                *buffers;

    fd = v4l2_open(dev_name, O_RDWR | O_NONBLOCK, 0);
    if (fd < 0) {
        perror("Cannot open device");
        exit(EXIT_FAILURE);
    }

    CLEAR(fmt);
    fmt.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
    fmt.fmt.pix.width      = 640;
    fmt.fmt.pix.height     = 480;
    fmt.fmt.pix.pixelformat = V4L2_PIX_FMT_RGB24;
    fmt.fmt.pix.field      = V4L2_FIELD_INTERLACED;
    xioctl(fd, VIDIOC_S_FMT, &fmt);
    if (fmt.fmt.pix.pixelformat != V4L2_PIX_FMT_RGB24) {
        printf("Libv4l didn't accept RGB24 format. Can't proceed.\n");
    }
}
```

```

        exit(EXIT_FAILURE);
    }
    if ((fmt.fmt.pix.width != 640) || (fmt.fmt.pix.height != 480))
        printf("Warning: driver is sending image at %dx%d\n",
               fmt.fmt.pix.width, fmt.fmt.pix.height);

    CLEAR(req);
    req.count = 2;
    req.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
    req.memory = V4L2_MEMORY_MMAP;
    xioctl(fd, VIDIOC_REQBUFS, &req);

    buffers = calloc(req.count, sizeof(*buffers));
    for (n_buffers = 0; n_buffers < req.count; ++n_buffers) {
        CLEAR(buf);

        buf.type      = V4L2_BUF_TYPE_VIDEO_CAPTURE;
        buf.memory     = V4L2_MEMORY_MMAP;
        buf.index      = n_buffers;

        xioctl(fd, VIDIOC_QUERYBUF, &buf);

        buffers[n_buffers].length = buf.length;
        buffers[n_buffers].start = v4l2_mmap(NULL, buf.length,
                                             PROT_READ | PROT_WRITE, MAP_SHARED,
                                             fd, buf.m.offset);

        if (MAP_FAILED == buffers[n_buffers].start) {
            perror("mmap");
            exit(EXIT_FAILURE);
        }
    }

    for (i = 0; i < n_buffers; ++i) {
        CLEAR(buf);
        buf.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
        buf.memory = V4L2_MEMORY_MMAP;
        buf.index = i;
        xioctl(fd, VIDIOC_QBUF, &buf);
    }
    type = V4L2_BUF_TYPE_VIDEO_CAPTURE;

    xioctl(fd, VIDIOC_STREAMON, &type);
    for (i = 0; i < 20; i++) {
        do {
            FD_ZERO(&fds);
            FD_SET(fd, &fds);

            /* Timeout. */
            tv.tv_sec = 2;
            tv.tv_usec = 0;

            r = select(fd + 1, &fds, NULL, NULL, &tv);
        } while ((r == -1 && (errno == EINTR)));
        if (r == -1) {
            perror("select");
            return errno;
        }

        CLEAR(buf);
        buf.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
        buf.memory = V4L2_MEMORY_MMAP;
        xioctl(fd, VIDIOC_DQBUF, &buf);

        sprintf(out_name, "out%03d.ppm", i);
        fout = fopen(out_name, "w");
        if (!fout) {
            perror("Cannot open image");
            exit(EXIT_FAILURE);
        }
        fprintf(fout, "P6\n%d %d 255\n",
               fmt.fmt.pix.width, fmt.fmt.pix.height);
        fwrite(buffers[buf.index].start, buf.bytesused, 1, fout);
        fclose(fout);

        xioctl(fd, VIDIOC_QBUF, &buf);
    }

    type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
    xioctl(fd, VIDIOC_STREAMOFF, &type);
    for (i = 0; i < n_buffers; ++i)

```

```
        v4l2_munmap (buffers[i].start, buffers[i].length);  
        v4l2_close (fd);  
  
        return 0;  
    }
```