ET61X[12]51 PC Camera Controllers
Driver for Linux

- Documentation -

Index
=====

1. Copyright

2. Disclaimer
Etoms is a trademark of Etoms Electronics Corp.
This software is not developed or sponsored by Etoms Electronics.

3. License

4. Overview and features
This driver supports the video interface of the devices mounting the ET61X151
or ET61X251 PC Camera Controllers.

It's worth to note that Etoms Electronics has never collaborated with the author during the development of this project; despite several requests, Etoms Electronics also refused to release enough detailed specifications of the video compression engine.

The driver relies on the Video4Linux2 and USB core modules. It has been designed to run properly on SMP systems as well.

The latest version of the ET61X[12]51 driver can be found at the following URL: http://www.linux-projects.org/

Some of the features of the driver are:

- full compliance with the Video4Linux2 API (see also "Notes for V4L2 application developers" paragraph);
- available mmap or read/poll methods for video streaming through isochronous data transfers;
- automatic detection of image sensor;
- support for any window resolutions and optional panning within the maximum pixel area of image sensor;
- image downscaling with arbitrary scaling factors from 1 and 2 in both directions (see "Notes for V4L2 application developers" paragraph);
- two different video formats for uncompressed or compressed data in low or high compression quality (see also "Notes for V4L2 application developers" paragraph);
- full support for the capabilities of every possible image sensors that can be connected to the ET61X[12]51 bridges, including, for instance, red, green, blue and global gain adjustments and exposure control (see "Supported devices" paragraph for details);
- use of default color settings for sunlight conditions;
- dynamic I/O interface for both ET61X[12]51 and image sensor control (see "Optional device control through 'sysfs'" paragraph);
- dynamic driver control thanks to various module parameters (see "Module parameters" paragraph);
- up to 64 cameras can be handled at the same time; they can be connected and disconnected from the host many times without turning off the computer, if the system supports hotplugging;
- no known bugs.

5. Module dependencies

For it to work properly, the driver needs kernel support for Video4Linux and USB.

The following options of the kernel configuration file must be enabled and corresponding modules must be compiled:

```
# Multimedia devices
#
CONFIG_VIDEO_DEV=m
```

To enable advanced debugging functionality on the device through /sysfs:

```
# Multimedia devices
#
```

```
CONFIG_VIDEO_ADV_DEBUG=y

# USB support
#
CONFIG_USB=m
```

In addition, depending on the hardware being used, the modules below are necessary:

```
# USB Host Controller Drivers
#
CONFIG_USB_EHCI_HCD=m
CONFIG_USB_UHCI_HCD=m
CONFIG_USB_OHCI_HCD=m
```

And finally:

```
# USB Multimedia devices
#
CONFIG_USB_ET61X251=m
```

6. Module loading

To use the driver, it is necessary to load the "et61x251" module into memory after every other module required: "videodev", "v4l2_common", "compat_ioctl32", "usbcore" and, depending on the USB host controller you have, "ehci-hcd", "uhci-hcd" or "ohci-hcd".

Loading can be done as shown below:

```
[root@localhost home]# modprobe et61x251
```

At this point the devices should be recognized. You can invoke "dmesg" to analyze kernel messages and verify that the loading process has gone well:

```
[user@localhost home]$ dmesg
```

7. Module parameters

Module parameters are listed below:

Name:	video_nr
Type:	short array (min = 0, max = 64)
Syntax:	<-1|n[,...]>
Description:	Specify V4L2 minor mode number: -1 = use next available n = use minor number n You can specify up to 64 cameras this way. For example: video_nr=-1,2,-1 would assign minor number 2 to the second registered camera and use auto for the first one and for every other camera.
Default:	-1

```
Name:           force_munmap
Type:           bool array (min = 0, max = 64)
Syntax:         <0|1[,...]>
Description:    Force the application to unmap previously mapped buffer memory
                before calling any VIDIOC_S_CROP or VIDIOC_S_FMT ioctl's. Not
                all the applications support this feature. This parameter is
                specific for each detected camera.
                0 = do not force memory unmapping
                1 = force memory unmapping (save memory)
Default:        0
-------------------------------------------------------------------------------
Name:           frame_timeout
Type:           uint array (min = 0, max = 64)
Syntax:         <n[,...]>
Description:    Timeout for a video frame in seconds. This parameter is
                specific for each detected camera. This parameter can be
                changed at runtime thanks to the /sys filesystem interface.
Default:        2
-------------------------------------------------------------------------------
Name:           debug
Type:           ushort
Syntax:         <n>
Description:    Debugging information level, from 0 to 3:
                0 = none (use carefully)
                1 = critical errors
                2 = significant informations
                3 = more verbose messages
                Level 3 is useful for testing only, when only one device
                is used at the same time. It also shows some more informations
                about the hardware being detected. This module parameter can be
                changed at runtime thanks to the /sys filesystem interface.
Default:        2
-------------------------------------------------------------------------------
```

8. Optional device control through "sysfs"

If the kernel has been compiled with the CONFIG_VIDEO_ADV_DEBUG option enabled, it is possible to read and write both the ET61X[12]51 and the image sensor registers by using the "sysfs" filesystem interface.

There are four files in the /sys/class/video4linux/videoX directory for each registered camera: "reg", "val", "i2c_reg" and "i2c_val". The first two files control the ET61X[12]51 bridge, while the other two control the sensor chip. "reg" and "i2c_reg" hold the values of the current register index where the following reading/writing operations are addressed at through "val" and "i2c_val". Their use is not intended for end-users, unless you know what you are doing. Remember that you must be logged in as root before writing to them.

As an example, suppose we were to want to read the value contained in the register number 1 of the sensor register table - which is usually the product identifier - of the camera registered as "/dev/video0":

```
        [root@localhost #] cd /sys/class/video4linux/video0
        [root@localhost #] echo 1 > i2c_reg
        [root@localhost #] cat i2c_val
```

Note that if the sensor registers cannot be read, "cat" will fail.
To avoid race conditions, all the I/O accesses to the files are serialized.

9. Supported devices

None of the names of the companies as well as their products will be mentioned here. They have never collaborated with the author, so no advertising.

From the point of view of a driver, what unambiguously identify a device are its vendor and product USB identifiers. Below is a list of known identifiers of devices mounting the ET61X[12]51 PC camera controllers:

Vendor ID	Product ID
0x102c	0x6151
0x102c	0x6251
0x102c	0x6253
0x102c	0x6254
0x102c	0x6255
0x102c	0x6256
0x102c	0x6257
0x102c	0x6258
0x102c	0x6259
0x102c	0x625a
0x102c	0x625b
0x102c	0x625c
0x102c	0x625d
0x102c	0x625e
0x102c	0x625f
0x102c	0x6260
0x102c	0x6261
0x102c	0x6262
0x102c	0x6263
0x102c	0x6264
0x102c	0x6265
0x102c	0x6266
0x102c	0x6267
0x102c	0x6268
0x102c	0x6269

The following image sensors are supported:

Model	Manufacturer
TAS5130D1B	Taiwan Advanced Sensor Corporation

All the available control settings of each image sensor are supported through the V4L2 interface.

10. Notes for V4L2 application developers

This driver follows the V4L2 API specifications. In particular, it enforces two rules:

- exactly one I/O method, either "mmap" or "read", is associated with each file descriptor. Once it is selected, the application must close and reopen the device to switch to the other I/O method;

- although it is not mandatory, previously mapped buffer memory should always be unmapped before calling any "VIDIOC_S_CROP" or "VIDIOC_S_FMT" ioctl's. The same number of buffers as before will be allocated again to match the size of the new video frames, so you have to map the buffers again before any I/O attempts on them.

Consistently with the hardware limits, this driver also supports image downscaling with arbitrary scaling factors from 1 and 2 in both directions. However, the V4L2 API specifications don't correctly define how the scaling factor can be chosen arbitrarily by the "negotiation" of the "source" and "target" rectangles. To work around this flaw, we have added the convention that, during the negotiation, whenever the "VIDIOC_S_CROP" ioctl is issued, the scaling factor is restored to 1.

This driver supports two different video formats: the first one is the "8-bit Sequential Bayer" format and can be used to obtain uncompressed video data from the device through the current I/O method, while the second one provides "raw" compressed video data (without frame headers not related to the compressed data). The current compression quality may vary from 0 to 1 and can be selected or queried thanks to the VIDIOC_S_JPEGCOMP and VIDIOC_G_JPEGCOMP V4L2 ioctl's.

11. Contact information

The author may be contacted by e-mail at <luca.risolia@studio.unibo.it>.

GPG/PGP encrypted e-mail's are accepted. The GPG key ID of the author is 'FCE635A4'; the public 1024-bit key should be available at any keyserver; the fingerprint is: '88E8 F32F 7244 68BA 3958 5D40 99DA 5D2A FCE6 35A4'.