G2 Camera Control (v4l2-ctl, ioctl)

29.06.2017

V4L2 devices are controlled by iotcl calls. You can use either the command line tool v4l2-ctl to get/set the camera parameters or use iotcl calls directly from source code.

v4l2-ctl

Basic commands:

List all formats the device is capable of

```
v4l2-ctl -d /dev/video0 --list-formats-ext
```

List all controls a device offers

```
v4l2-ctl -d /dev/video0 -l
```

Get the value of a specific control

```
v4l2-ctl -d /dev/video0 -C control_name     ( e.g. v4l2-ctl -d
/dev/video0 -C gain )
```

Set the value of a specific control

```
v4l2-ctl -d /dev/video0 -c control_name=value     ( e.g. v4l2-ctl -d
/dev/video0 -c gain=104 )
```

The control parameters

The control parameter are currently directly linked to camera registers and have therefore non human readable values, this might change in further releases

control name	description	notes
coarse_time	the exposure time	exposure time is given in line time, see formula below for conversion in second for resolution 4320x2160: coars_time = exposure_time_in_seconds / (4320.0 / 317400000.0)
gain	analogue gain	gain is given in multiples of 52
group_hold	unused	
bypass_mode	bypass camera pipelin	setting this control will freeze the stream. Unfreezing is NOT possible at the moment
hdr_enable	enables On-Chip HDR synthesis	
focus_absolute	set the focus of the g2 module	only usable when a g2 module is used

binning_x binning_y	set the binning factor in horizontal/vertical direction	debug parameter. Do not set this control manualy. Binning is implicit used when setting the camera format
summed_binning	use binning in summed mode	pixels are added instead of averaged when this mode is enabled
roi_start_x roi_start_y	move the sensors ROI	when a resolution is set that does not cover the whole sensor, the ROI can be moved within the given boundaries
delayed_start	enable this control to pause the sensor after starting	unlock the stream by executing the following command (as root): echo "0x100 0x1" > /sys/kernel/debug/g2_a/d replace g2_a with g2_b, g2_c, ... to unlock the corresponding sensor

Using ioctl form source code

All commands use the v4l2_control structure (https://linuxtv.org/downloads/v4l-dvb-apis/uapi/v4l/vidioc-g-ctrl.html) and the ioctl() (http://man7.org/linux/man-pages/man2/ioctl.2.html) function.
In addition a file-descriptor for the device to control is needed and the ID of the control to manipulate.

See following example to how getting and setting a parameter works:

get a v4l2 control value

```
// Define IDs of exposure and gain control
unsigned long IOTCL_GAIN = (0x9A0000 | 0x2000) + 10;
unsigned long IOTCL_EXP = (0x9A0000 | 0x2000) + 1;

// Open the file descriptor with read/write access
int iFD = open("/dev/video0", I_RDWR)

// Create a v4l2_control struct
v4l2_control xCtrl;

// Set the id we want to read
xCtrl.id = IOTCL_GAIN;

if (ioctl(iFD, VIDIOC_G_CTRL, &xCtrl) == -1)
{
  // Handle error here
}

// Do something with the value received
int iValue = xCtrl.value;

// Close the file descriptor
close(fID);
```

set a v4l2 control value

```
// Define IDs of exposure and gain control
unsigned long IOTCL_GAIN = (0x9A0000 | 0x2000) + 10;
unsigned long IOTCL_EXP = (0x9A0000 | 0x2000) + 1;

// Open the file descriptor with read/write access
int iFD = open("/dev/video0", I_RDWR)

// Create a v4l2_control struct
v4l2_control xCtrl;

// Set the id we want to read
xCtrl.id = IOTCL_GAIN;
// Set the new value of the control here
xCtrl.value = 104;

if (ioctl(iFD, VIDIOC_S_CTRL, &xCtrl) == -1)
{
  // Handle error here
}

// Close the file descriptor
close(fID);
```