V4L2 Sensor Driver Programming Guide

DG_07725-001 | May 11, 2016

TABLE OF CONTENTS

Document Change History

Document number: DA_07725-001

Version	Date	Authors	Description of Change
v1.0	13 Jul 2015	jbang/msum	Initial release.
v2.0	05 Oct 2015	jbang/hlang	Change in document title, other content updates.
v3.0	20 Jan 2016	jbang/gmead	Adds device tree and other BSP-integration information.
V4.0	21 Apr 2016	gigon bae/hlang	Added GNU licenses

Note: Apparent hyperlinks in this document are a legacy of the HTML version and may not operate as expected in the PDF version.

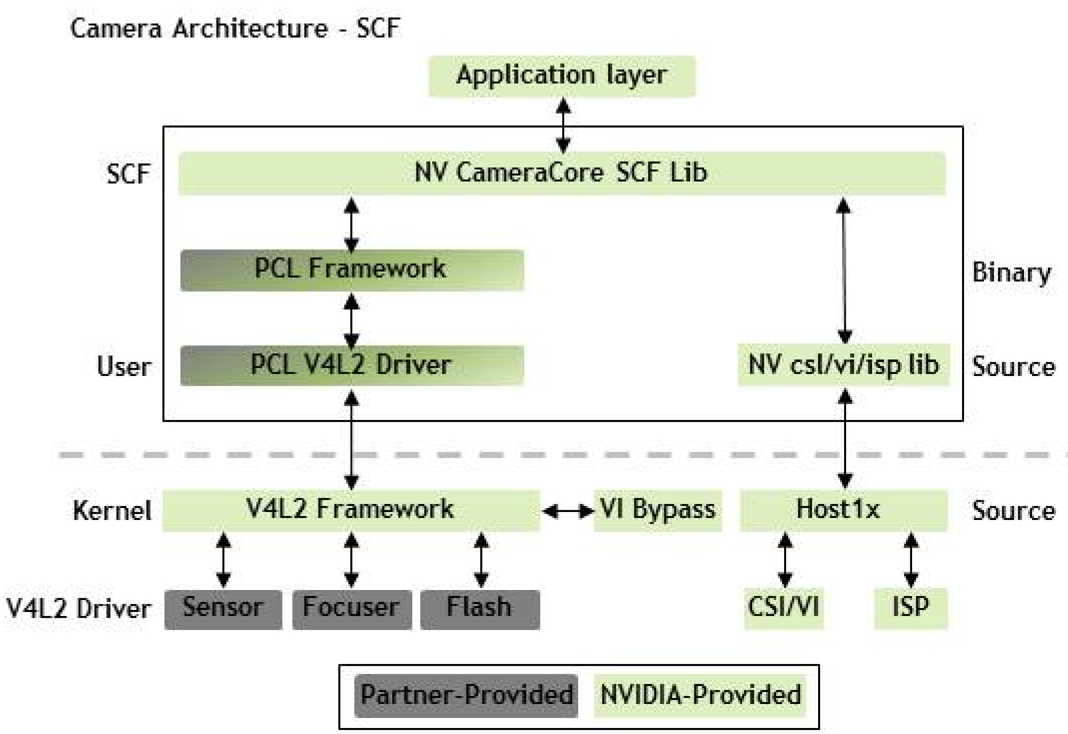
V4L2 Sensor Driver Programming Guide

This chapter provides advanced information for developing USB cameras and Bayer and YUV image sensor with NVIDIA® Tegra® Board Support Package (BSP) releases.

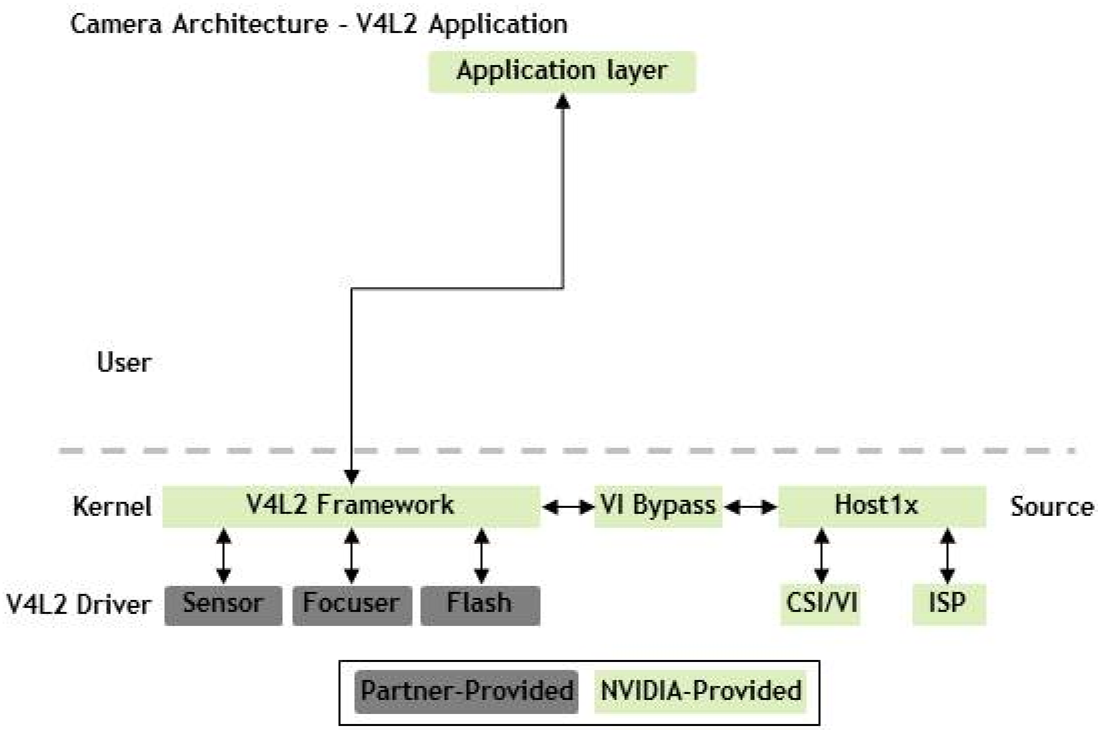
This document describes the sensor driver architecture required by Tegra® platform software, and provides guidance on the implementation of drivers suitable for use with this software release. Implementation of a camera sensor driver will enable acquisition of camera data over the CSI bus, in the native format provided by the sensor, and is intended for use in enabling sensors that contain an ISP (e.g. YUV output).

The programming interfaces for the Tegra® ISP and associated image quality tuning are not covered in this document. NVIDIA provides a software implementation for supported camera modules. (For more information, refer to product documentation.)

The following block diagram show the overall architecture of NVIDIA camera software.



The Scalable Camera Framework (SCF) is the main camera user-mode library.If the application supports direct V4L2 interface, it communicates to the NVIDIA V4L2 driver without SCF. Typically, this path is used when the sensor is a YUV sensor since this sensor has a built-in ISP and frame does not need extra processing. For Bayer sensor, applications use the SCF library to convert RGB format to YUV format and to do various post processing tasks.



This chapter describes how to bring up a Bayer sensor with Tegra BSP. Bringing up the sensor requires customers to develop two things:

- Device Tree in the Linux kernel
- V4L2 sensor driver

Read the following sections to learn how to develop these; our examples use Omnivision OV5693 sensor, and the source code for OV5693 sensor is available to customers.

> **Note**: the examples in this document show Tegra® X1-based implementations. To implement for other Tegra® devices, contact your customer engineer.

Camera Modules

A camera module installed on the target platform can consist of one or more devices. A typical rear camera module includes a complementary metal-oxide semiconductor (CMOS) sensor, a Visual Computing Module (VCM) focuser, or both. A typical front camera module might include a single CMOS sensor only.

To add one or more camera modules to a device tree, find or create a tegra-camera-platform device node in the kernel source tree. Usually, that node is in the following directory:

```
arch/arm[64]/boot/dts
```

In a tegra-camera-platform device node, you must create a module table (modules) with one or more modules. Each module must contain its basic information and the definition of the devices that are inside that module.

> **Note**: Except for the ones that refer to other device nodes, all value fields in camera-related device nodes must use the string data type.

A typical device-tree node for a camera module looks like this:

```
tegra-camera-platform {
compatible = "nvidia, tegra-camera-platform";
modules {
module0 {
badge = "e3326_front_P5V27C";
position = "rear";
orientation = "1";
drivernode0 {
pcl_id = "v4l2_sensor";
proc-device-tree = "/proc/device-tree/host1x/i2c@546c0000/ov5693_c@36";
};
drivernode1 {
pcl_id = "v4l2_focuser_stub";
};
};
};
};
```

Module Properties

The following table shows the information for `moduleX: module` (or `moduleX: modules`).

Property	Value
badge	A unique name that identifies this module. Guidelines for naming the three parts of the `badge_info` property: • The first part is the camera board ID (`camera_board_id`) of the module. • The second part contains the position of the module, for example, rear or front. • The third part contains the last six characters of a part number, which you can find in the data sheet on the module from the vender.
position	The camera-facing information, either rear or front.
orientation	The orientation related to the display panel. Valid values: 0, 90, 180, or 270.
drivernodeX	The information on the driver node; the X: 0-based index.

Driver Properties

The following table shows the information for `drivernodeX: device` (or `drivernodeX: devices`).

Property	Value
pcl-id	A unique name that identifies this device.
proc-device-tree*	The device-tree file path for the device-specific data node. For example, details on the device GPIO, regulator, clock, mode, and so forth.

Individual Imaging Device

An imaging device (a component inside the camera module) can be a sensor, focuser, or flash. Be sure to add all the required information to the device-tree node to support the device operation.

For each device-tree node for a device, assign a device node that contains the name of the device, its slave address, and a compatible string that identifies the node.

> **Note**: Except for the ones that refer to other device nodes, all value fields in camera-related device nodes must use the string data type.

An example device-tree node for the OV5693 V4L2 sensor driver:

```
ov5693_c@36 {
compatible = "nvidia,ov5693";
reg = <0x36>;
devnode = "video2";

physical_w = "3.674";
physical_h = "2.738";

avdd-reg = "vana";
iovdd-reg = "vif";

mode0 { // OV5693_MODE_2592X1944
mclk_khz = "24000";
num_lanes = "2";
tegra_sinterface = "serial_c";
discontinuous_clk = "no";
dpcm_enable = "false";
cil_settletime = "0";
```

```
active_w = "2592";
active_h = "1944";
pixel_t = "bayer_bggr";
readout_orientation = "90";
line_length = "2688";
inherent_gain = "1";
mclk_multiplier = "6.67";
pix_clk_hz = "160000000";

min_gain_val = "1.0";
max_gain_val = "16";
min_hdr_ratio = "1";
max_hdr_ratio = "64";
min_framerate = "1.816577";
max_framerate = "30";
min_exp_time = "34";
max_exp_time = "550385";
};
...
};
```

Device Properties

For the node of the V4L2 sensor-device, define the required hardware resource for the device, as shown in the following table.

Property	Value
compatible	Specifies the device identifier.
reg	Specifies the I2C slave address.
mclk	Specifies the input clock for the device.
XXX-gpio	Specifies the general-purpose input/output (GPIO) pins for the device.
XXX-supply	Specifies the regulator for the device, where XXX is the actual regulator name defined somewhere else in the device tree. The -supply suffix is mandatory.
XXX-reg	Specifies the name of the regulator, where XXX is the regulator name for the sensor driver. The value of this field is the regulator name with the suffix -reg.
physical_w	Specifies the physical width of the sensor.

physical_h	Specifies the physical height of the sensor.
devnode	Specifies a value used to derive the kernel device node. Android path: /dev/camera/<devnode> L4T path: /dev/<devnode>
formulaXXX	Specifies the predefined formula for calculating the parameters (pixel clock, frame rate, and so forth.) This formula is parsed and evaluated in the user space. Note these rules: • The variables in the formula can be numbers or any of the entries in the device tree. For example: `formulaPixelClk="(mclk*multiplier)/(pre_div*post_div)"` • The formula must contain only these operators: '*' '/' '+' '-' '(' ')' • At least one operator must be inside the parentheses. • No spaces are allowed in the formula.

Property-Value Pairs

The following table describes the property-value pairs that apply to the sensor mode for the V4L2 implementation.

Property	**Value**
modeX	Specifies the sensor-mode information, that is, the X: 0-based index.
Ports	Specifies the media controller graph binding info.
mclk_khz	Specifies the standard MIPI driving clock, which is typically 24MHz.
num_lanes	Specifies the number of lane channels the sensor is programmed to output.
tegra_sinterface	Specifies the base Tegra serial interface to which the lanes are connected.
discontinuous_clk	Specifies the indication that the sensor is programmed to use a discontinuous clock on MIPI lanes.
cil_settletime	Specifies the value of the settle time of the MIPI lane. A 0 value attempts to auto-calibrate according to the mclk_multiplier parameter.
active_w	Specifies the width of the pixel-active region.
active_h	Specifies the height of the pixel-active region.
pixel_t	Specifies the readout pixel pattern of the sensor.
readout_orientation	Specifies the readout orientation that is based on the orientation of the camera module. Change this property if you would like to program a different readout order for this mode.

line_length	Specifies the pixel line length (width) for sensor mode for calibrating the features in our camera stack.
mclk_multiplier	Specifies the multiplier to MCLK for timing the capture sequence of the hardware.
pix_clk_hz	Specifies the sensor pixel clock for calculating the exposure, frame rate, and so forth.
inherent_gain	Specifies the gain obtained inherently from the mode, that is, pixel binning.
min_gain_val	Specifies the minimum gain limit for the mode.
max_gain_val	Specifies the maximum gain limit for the mode.
min_exp_time	Specifies the minimum exposure time limit for the mode in microseconds.
max_exp_time	Specifies the maximum exposure time limit for the mode in microseconds.
min_hdr_ratio	Specifies the minimum high-dynamic-range (HDR) ratio limit for the mode (for HDR sensors).
max_hdr_ratio	Specifies the maximum HDR ratio limit for the mode (for HDR sensors).
min_framerat	Specifies the minimum frame-rate limit for the mode in frames per second (fps).
max_framerat	Specifies the maximum frame-rate limit for the mode in fps.

Example Focuser Driver Properties

The following table shows the required information for the example of the LC898212 focuser driver.

```
lc898212@72 {
compatible = "nvidia,lc898212";
reg = <0x72>;

devnode = "video6";
type = "default";
ports {
#address-cells = <1>;
#size-cells = <0>;
port@0 {
reg = <0>;
lc898212_out0: endpoint {
remote-endpoint = <&vi_in1>;
```

```
};
};
};
};
```

The following table describes the focuser driver properties.

Property	Value
compatible	Specifies the device identifier.
reg	Specifies the I2C slave address.
type	Specifies the focuser type: • default: VCM focuser. • steppermotor: Stepper motor focuser.
Ports	Media controller graph binding info.

V4L2 Kernel Driver

The content of this chapter is based on the Video for Linux 2 (V4L2) driver for the OmniVision OV5693 sensor (`ov5693.c`) at:

```
kernel/drivers/media/i2c/ov5693.c
```

NVIDIA suggests that you look at that source before reading this the rest of this chapter.

Macro Definitions

Following are the sensor-specific macro values:

- The minimum and maximum values for each control.
- The default value for each control.
- The macro values that is required for sensor timing or general functionality.

Sensor-Private Data

The following structure contains the private data that are specific to the sensor:

```
struct ov5693 {
struct camera_common_power_rail power;
int numctrls;
struct v4l2_ctrl_handler ctrl_handler;
struct camera_common_eeprom_data eeprom[OV5693_EEPROM_NUM_BLOCKS];
u8 eeprom_buf[OV5693_EEPROM_SIZE];
struct i2c_client *i2c_client;
```

```
struct v4l2_subdev *subdev;
struct media_pad pad;
s32 group_hold_prev;
bool group_hold_en;
struct regmap *regmap;
struct camera_common_data *s_data;
struct camera_common_pdata *pdata;
struct v4l2_ctrl *ctrls[];
};
```

The following table describes the sensor properties.

Property	Value
power	Holds generic power controls, include regulators,clks, and GPIOs.
numctrls	Holds the number of v4l2 controls for the sensor.
ctrl_handlerr	Holds the required v4l2 handler to controls, needed for v4l2 control init.
i2c_client	Holds a handle to i2c_client, used to access to sensor i2c client instance.
subdev	Holds a handle for v4l2 sub-device, needed to run subdev operations (ops).
eeprom	Holds common EEPROM device data.
eeprom_buf	Holds eeprom buffer storage.
pad	Holds media pad used for media controller initialization for a device to work as SINK or SOURCE.
group_hold_prev	Holds previous state use by group hold control handler to check for change of state.
group_hold_en	Holds group hold enable flag directly related to group hold control.
reg_map	Holds a register map setup for I2C read and write.
s_data	Holds a handle to common data, see documentation for camera_common_data.
pdata	Holds a handle to common platform data, populated by read Device Tree.
ctrls	Holds handles to initialized v4l2 controls, dynamic array, MUST BE LAST FIELD.

Configuring Regmap

Depending on the I2C interface of the sensor, you should update the values of `reg_bits` and `val_bits`.

```
regmap_config {
reg_bits = 16;
val_bits = 8;
};
```

The following table describes the `regmap_config` properties.

Property	Value
`reg_bits`	Specifies the number of bits needed to represent the I2C register offset.
`val_bits`	Specifies the number of bits in the buffer to store data to be transferred over I2C.

Check the vendor register programming table of your sensor to determine the size of `reg_bits` and `val_bits`.

Configuring Controls

To link the controls to their control handlers, set up the function pointers:

- Point `g_volatile_ctrl` to the internal `get` volatile control handler of the sensor.
- Point `s_ctrl` to the internal `set` control handler of the sensor.

```
static const struct v4l2_ctrl_ops ov5693_ctrl_ops = {
.g_volatile_ctrl = ov5693_g_volatile_ctrl,
.s_ctrl = ov5693_s_ctrl,
};
```

The following code-snippet lists the controls and their initialized values. This list is looped through during the `ctrls_init` call to initialize each of the controls. Each control is then accessible through the `ctrls` handler in the private data. The set of controls defined for OV5693 are the standard ones used by the user-mode PCL V4L2 driver.

> **Note**: Additional controls require a change in the PCL driver.

Three types of controls are defined for the OV5693 sensor.

```
static struct v4l2_ctrl_config ctrl_config_list[] = {
/* Integer Control: setting integer values such as gain, coarse
* time, *and frame length.
*/
{
```

```
.ops = &ov5693_ctrl_ops, //pointer to control ops
//function

.id = V4L2_CID_GAIN, //id, defined in
//camera_common.h

.name = "Gain", //string name of control
.type = V4L2_CTRL_TYPE_INTEGER, //type of control
.flags = V4L2_CTRL_FLAG_SLIDER, //control flags

// The following three are the value most likely
// needs changing
.min = OV5693_MIN_GAIN, //control value lower bound
.max = OV5693_MAX_GAIN, //control value upper bound
.def = OV5693_DEFAULT_GAIN, //default control value
.step = 1, //increment step size for
//value
},
...
/* String Data Control: converts data to string format then
* send to PCL
* drivers, used for EEPROM, OTP, and fuse id.
*/
{
.ops = &ov5693_ctrl_ops,
.id = V4L2_CID_EEPROM_DATA,
.name = "EEPROM Data",
.type = V4L2_CTRL_TYPE_STRING,
.flags = V4L2_CTRL_FLAG_VOLATILE,
.min = 0,
/* the following one is the value that likely needs
* changing, the string size is 2 times actual
* buffer size
*/
.max = OV5693_EEPROM_STR_SIZE,
.step = 2,
},
```

```
...

/* Menu Control: used as on/off switch for group hold and HDR.
* switch_ctrl_qmenu is used to define the states on/off there
* shouldn't be a need to change these controls, unless a
* completely new one is being added.
*/
{
.ops = &ov5693_ctrl_ops,
.id = V4L2_CID_GROUP_HOLD,
.name = "Group Hold",
.type = V4L2_CTRL_TYPE_INTEGER_MENU,
.min = 0,
.max = ARRAY_SIZE(switch_ctrl_qmenu) - 1,
.menu_skip_mask = 0,
.def = 0,
.qmenu_int = switch_ctrl_qmenu,
},
};
```

Setting Up Control Registers

Set up register writes for integer controls with the following functions. `addr` depends on the sensor being used; `val` is the source from the control handler. These functions are called by each control handler to set up register writes for each of the controls.

- ov5693_get_frame_length_regs
- ov5693_get_coarse_time_regs
- ov5693_get_coarse_time_short_regs
- ov5693_get_gain_reg
- ov5693_get_gain_short_reg

Read-Write Wrapper in the Register

The following functions are wrappers for the read-write interface of the I2C register. For OV5693, use the `regmap` interface. However, you can modify these functions for other interfaces.

- ov5693_read_reg
- ov5693_write_reg
- ov5693_write_table

Power Functions

The following table describes the functions for power-related controls.

Function	Description
ov5693_power_on	Contains the power-on sequence. You must modify this function according to the specification sheets. Note the usage of controls from common power rail, including regulators and GPIOs.
ov5693_power_off	Contains the power-off sequence. You must modify this function according to the specification sheets. Note the usage of controls from common power rail, including regulators and GPIOs.
ov5693_power_put	Calls `regulator_put` on all regulators.
ov5693_power_get	Calls `regulator_get` on all regulators.

Setting Up V4L2 Subdevice and Camera Common

The `ov5693_s_stream` function is mainly for writing mode tables by making calls to register the `write_table` function. You set up mode tables in the `ov5693_mode_tbls.h` file. For details, see Mode Tables in this chapter.

In addition to writing mode tables and enabling the stream through stream-enable register writes, `ov5693_s_stream` also writes the initial integer-control values to the register through direct calls to the integer-control handlers. For details, see the section Control Handlers.

```
control.id = V4L2_CID_GAIN;
err = v4l2_g_ctrl(&priv->ctrl_handler, &control);
err |= ov5693_set_gain(priv, control.value);
…
```

If a test pattern is supported by the sensor and you can create a register table for that pattern, you can add a `test_mode` flag to write the test-mode table here.

Camera common is a set of functions that are common to camera drivers of the NVIDIA kernel, to which this driver refers. Camera common sets up most of the V4L2 framework, requiring linkage from the driver in the form of the following:

```
struct camera_common_sensor_ops
struct camera_common_data
```

For details on modifying and adding new modes, see Mode Tables in this chapter.

The following code snippets set up the V4L2 subdevices and camera common for registering the sensor driver with the V4L2 framework. The `s_stream` pointer must point to the internal `s_stream` function of the sensor; you can leave the other pointers as is.

```
static struct v4l2_subdev_video_ops ov5693_subdev_video_ops = {
.s_stream = ov5693_s_stream,
...
};
```

You need not modify this structure:

```
static struct v4l2_subdev_core_ops ov5693_subdev_core_ops = {
...
};
```

Match the name for the pointer for the `core` and `video` operations function with the two from above, as follows:

```
static struct v4l2_subdev_ops ov5693_subdev_ops = {
.core = &ov5693_subdev_core_ops,
.video = &ov5693_subdev_video_ops,
};
```

During the parsing of the device tree, `of_device_id` matches the compatible field with the one in the Device Tree.

```
static struct of_device_id ov5693_of_match[] = {
{ .compatible = "nvidia,ov5693", },
{ },
};
```

This structure is required for camera common and you must set up the function pointers appropriately. Link the `power_on`, `power_off`, `write_reg`, and `read_reg` functions here:

```
static struct camera_common_sensor_ops ov5693_common_ops = {
.power_on = ov5693_power_on,
.power_off = ov5693_power_off,
.write_reg = ov5693_write_reg,
.read_reg = ov5693_read_reg,
};
```

Control Handlers

This section describes the control handlers.

Set Control

The two subsections describe the handlers for set control.

V4L2 Set-Control Operation

The V4L2 set-control function contains a `switch` statement to redirect set-control calls to their appropriate control handlers.

> **Note**: Read-only controls, such as `fuse_id` and One-Time Programmable Read-Only Memory (OTP ROM), do not have a case statement in the control ID `switch` statement.

```
ov5693_s_ctrl {
...
switch (ctrl->id) {
case V4L2_CID_GAIN:
...
case V4L2_CID_EEPROM_DATA:
...
case V4L2_CID_HDR_EN:
...
}
...
}
```

> **Note**: For EEPROM_DATA, the string control must have a preallocated string passed in. Hence, a null check is necessary before passing the string to the control handler.
>
> HDR_EN is a pure software control. No control handler writes to the hardware so simply break out of the switch statement.

Setter-Control Handlers (Writes)

Setter-control handlers are the control handlers called by `s_ctrl`. They perform additional control-handling operations, such as writes to registers. The majority of these controls make calls to the control register setup functions (see the section Setting Up Control Registers). The exception is `write_eeprom`, which acts as a separate I2C device with its own I2C write interface.

- ov5693_set_group_hold
- ov5693_set_gain
- ov5693_set_frame_length
- ov5693_set_coarse_time
- ov5693_set_coarse_time_short
- ov5693_write_eeprom

Note a special case for the gain-control handler, which contains a function call to:

```
ov5693_calculate_gain(val, OV5693_GAIN_SHIFT);
```

The function takes a binary-coded decimal from user space as input and computes the gain-register value according to the vendor-provided gain-calculation formula. Because that formula can vary from sensor to sensor, you must rewrite this function in the V4L2 sensor driver for each of the sensors.

The input of this function is the gain value passed in from user space. The value is a binary-coded decimal ranging from 1 to 16. The binary-coded decimal is divided into a six-byte integer representation and a two-byte decimal representation. The most significant six bytes of `val` is the integer representation. The least significant two bytes is the decimal representation, which is actually the numerator of a fraction over the maximum decimal representation of 0xFF.

The output of this function is the actual gain-register value programmed over I2C. That value depends on the gain-calculation formula provided by the sensor vendor (usually found in the data sheets on the sensor). The goal of this function is to convert the decimal `val` input into the gain-register value with as little truncation as possible.

For the OV5693 formula on which the `ov5693_calculate_gain` (or _`to_gain`) function is based, see the *OV5693 Software Reference Manual*.

Get-Volatile Control

Following are the get-volatile controls:

- **V4L2 get-volatile control operation**: The `ov5693_g_volatile_ctrl` function contains a switch statement for redirecting get-control calls to their appropriate control handlers.

  Note: For non-volatile controls, get-control simply returns the previously written value stored in the control handler.

- **Get-control handler (reads)**: The `ov5693_read_eeprom` control handler is an example of a get-volatile control handler. This handler reads the volatile value directly from the EEPROM register and updates the value that is read back every time get is called on this control.

Other Control-Related Functions

Following are two other control-related functions:

- **EPROM device-related controls**: These two functions are for setting up EEPROM as a separate I2C device with its own `regmap` interface:

  ```
  ov5693_eeprom_device_init
  ov5693_eeprom_device_release
  ```

- **Handlers called only on `ctrls_init`**: These control handlers are called only once from the `ctrls_init` function (see the section Boot-Time Initialization). That is because OTP and `fuse_id` controls are read-only and their values only need to be read once during boot time.

Boot-Time Initialization

This section describes the functions for initializing boot time.

Control Initialization

The `ov5693_ctrls_init` function iterates through `ctrl_config_list` (see Configuration Controls) and registers each control as a new custom control with the V4L2 framework. `s_ctrl` is also called for each control to set it to its default value defined in `ctrl_config_list`.

You need not modify the function except for the calls for initializing the value for the read-only controls. See the calls to `otp_setup` and `fuse_id_setup` in the section Control Handlers.

```
ov5693_ctrls_init {
...
err = ov5693_otp_setup(priv);
...
err = ov5693_fuse_id_setup(priv);
...
}
```

Device Tree Parser

The `ov5693_parse_dt` function, which parses device trees, takes the Device Tree node and looks for the parameters (according to the sensor-related private data) required by the sensor driver.

The values include but are not limited to the following:

```
mclk
pwdn-gpios
reset-gpios
af-gpios
avdd-reg
dvdd-reg
iovdd-reg
```

For details on how to set up device trees with the appropriate values, see the documentation. You must match the name list with the names in the Device Tree for their respective name-value pairs.

Port binding

```
vi {
ports {
port@0 {
reg = <0>;
vi_in0: endpoint {
bus-width = <2>;
remote-endpoint = <&ov5693_out0>;
};
};
};
};
```

```
ov5693_c@36 {
ports {
port@0 {
reg = <0>;
ov5693_out0: endpoint {
csi-port = <2>;
bus-width = <2>;
remote-endpoint = <&vi_in0>;
};
};
};
};
```

The following table describes the port binding properties.

Function	Description
port	Specifies the mediapad that the port is connected. All imager devices will have only one media pad for binding the connection with VI.
csi-port	This value defines the CSI port sensor is connected. For imager devices like focuser or flash this field is not required.
bus-width	Bus width defines the number of CSI lanes connected to sensor.
remote-endpoint	Remote end point is the label used for binding two ports. The binding expects one port is the sink and the other one is the source.

Media Controller Setup

A few additional components are needed to setup the media controller for V4L2 use. The open call is a placeholder operation for satisfying the V4L2 sub device internal operation requirements. The setup likely will not need to be change besides a name change to match the devices.

```
ov5693_open
```

Sub device function ops need to link to an the open operation:

```
static const struct v4l2_subdev_internal_ops ov5693_subdev_internal_ops = {
.open = ov5693_open,
};
```

Media entity function ops need to link to the V4L2 sub device link validation method:

```
static const struct media_entity_operations ov5693_media_ops = {
.link_validate = v4l2_subdev_link_validate,
};
```

Sensor-Driver Probing

The following subsections explain the probe function of the sensor driver.

Entry Point for Initialization During Boot

The `ov5693_probe` function is the entry point of the driver. The function starts off by allocating memory for common data and sensor-private data (see the section Sensor-Private Data).

```
common_data = devm_kzalloc(&client->dev,
sizeof(struct camera_common_data), GFP_KERNEL);

priv = devm_kzalloc(&client->dev,
sizeof(struct ov5693) + sizeof(struct v4l2_ctrl *) *
ARRAY_SIZE(ctrl_config_list),
GFP_KERNEL);
```

The function then initializes `regmap`:

```
priv->regmap = devm_regmap_init_i2c(client, &sensor_regmap_config);
```

Next, the function calls the other initialization functions, including the following:

```
// See the section Parser for Device Trees: stored in private
// data pdata field.
ov5693_parse_dt(client);

// See the section Power Functions.
ov5693_power_get(priv);



// Link the appropriate subdev ops (see the section
// Setting Up V4L2 Subdevice and Camera Common.
v4l2_i2c_subdev_init(&common_data->subdev, client, &subdev_ops);

// See the section Initialization of Controls.
ov5693_ctrls_init(priv);
```

```
// See the section Other Control-Related Functions.
ov5693_eeprom_device_init(priv);
```

Next, the function links the common data and sensor-private values:

```
// Link to ops. See Setting Up V4L2 Subdevice and Camera Common.
common_data->ops = &ov5693_common_ops;

// Control handler linking
common_data->ctrl_handler = &priv->ctrl_handler;

// I2C client passed in to probe
common_data->i2c_client = client;

// Default to frame format 0. See Mode Tables in this chapter.
common_data->frmfmt = &ov5693_frmfmt[0];

// Based on default data format definition, generally defaults
// to the same format. See the section Macro Definitions.
common_data->colorfmt = camera_common_find_datafmt(
OV5693_DEFAULT_DATAFMT);

// Power-handler linking
common_data->power = &priv->power;

// Sensor-private data linking
common_data->priv = (void *)priv;


// Number of format is frame format. See the section Macro Definitions.
common_data->numfmts = ARRAY_SIZE(ov5693_frmfmt);
```

```
// Set up of port information for device
camera_common_parse_ports(...)
// Port information is also used to create the name for debugfs node setup
sprintf(debugfs_name, "ov5693_%c", common_data->csi_port + 'a');
```

```
camera_common_create_debugfs(common_data, debugfs_name);
```

Setup of Default Values for Common Data

See the section Macro Definitions.

```
common_data->def_mode = OV5693_DEFAULT_MODE;
common_data->def_width = OV5693_DEFAULT_WIDTH;
common_data->def_height = OV5693_DEFAULT_HEIGHT;
common_data->def_clk_freq = OV5693_DEFAULT_CLK_FREQ;
debugfs_name

// I2C client passed in to probe
priv->i2c_client = client;

// Link to common data above
priv->s_data = common_data;

// Link to V4L2 subdevice handler
priv->subdev = &common_data->subdev;
// Link to subdevice dev to i2c_client dev (for media controller usage)
priv->subdev->dev = &client->dev;

// Initialize previous group hold state to 0
priv->group_hold_prev = 0;
```

Setup of for Media Controller

```
// Link subdevice internal and media entity operations
priv->subdev->internal_ops = &ov5693_subdev_internal_ops;
priv->subdev->entity.ops = &ov5693_media_ops;

// Setup subdevice flags and media entity type
priv->subdev->flags |= V4L2_SUBDEV_FL_HAS_DEVNODE | V4L2_SUBDEV_FL_HAS_EVENTS;
priv->subdev->entity.type = MEDIA_ENT_T_V4L2_SUBDEV_SENSOR;

// Setup media controller pad flags
priv->pad.flags = MEDIA_PAD_FL_SOURCE;
```

```
// Initialize and register subdevice and media entity with media controller
framework.
media_entity_init(&priv->subdev->entity, 1, &priv->pad, 0);
v4l2_async_register_subdev(priv->subdev);
```

All imager devices must register themselves as sub devices to media controller framework. VI acts as the master device which controls the binding, parsing the DT and establish the media links once all the sub devices are registered.

Each sub device can register to media controller framework by defining the entity and pads information.

Entity: Media entity is the unit device represented by media controller framework for establishing connections. Each media entity can have multiple pads. Framework provides a list of known entity types and the corresponding media operations.

Pad: Pad represents the device behaves as SINK or SOURCE port. Imager device must have a SOURCE port which gets binded to VI. If imager device has a SINK port then the SOURCE port which is getting binded the device SINK port must be represented in DT to complete the binding.

Removal of Sensor Drivers

The `ov5693_remove` function removes the sensor-device instance and calls on a device shutdown.

This function makes calls to various `free`, `put`, and `destroy` functions that match up with several calls in probe. It must remain largely the same.

```
// See the section Initialization of Controls for details on
// freeing the control handler.
v4l2_ctrl_handler free(&priv->ctrl_handler);

// See the section Power Functions.
ov5693_power_put(priv);
```

Debugging Tips

When the driver is complete, your primary goal is then to get the driver to probe and register the `/dev/camera/video#` node for the driver. That node is the file I/O interface with which the user-space driver accesses the driver.

Problems might occur in the probing process and require debugging. Typically, problems occur in the clock, GPIO, and regulator setup. See below for a few tips.

Verify that names match the Device Tree

To debug those problems, first verify that the names that are being read through the `parse_dt` function matches those in the Device Tree.

Verify that Device-Tree values match the hardware

Ensure that the values assigned to the Device Tree fields match the hardware they describe. Oftentimes, regulator names, GPIO numbers, and clock names are out-of-date in the Device Tree, causing probing to fail.

Verify that functions run to completion

Ensure that the `power_get` calls runs to completion. For details, see the sections Power Functions and Boot-Time Initialization.

Anther common problem that occurs during probe is in `control init`.

Verify that default values are correctly linked

Verify that default values are all linked correctly in the control configuration (see the section Sensor-Private Data) and that the default macros (see the section Macro Definitions) are updated appropriately with the values in the mode table of the sensor.

If new controls have been added to the control configuration list (on rare occasions), ensure that they contain the appropriate control handlers and default values.

After probing succeeds, problems could still occur with the control setting. A common problem is in the values of the register writes.

Verify that control-register values are correct

Ensure that the control-register setup (see the section Setting Up Control Registers) contains the correct register address and formats according to the mode table and data sheets.

For gain control, be sure to create a new `calculate_gain (to_gain)` function for calculating the gain value according to the gain formula in the data sheets.

Verify that mode-specific settings are correct

Double-check the mode-specific settings in the Device Tree. Update them if necessary. Problems might arise from the minimum and maximum values, potentially resulting in timing issues and sync-point timeouts.

Mode Tables

This topic explains mode tables and describes the procedure for adding them.

The modes tables of the register reside in a separate header file called `sensor_mode_tbls.h` and are included by the main driver. Those tables can be a list of `reg_8` or `reg_16` address-value pairs. Mode tables are separated by resolution. For the `ov5693`driver, a common mode table also exists, which contains the common register values among all the resolutions.

```
static ov5693_reg mode_table_common[] = {
...
};
```

The `start` and `stop` stream-register values must be in a separate mode table. When you separate them, delete the `start` stream-register values at the end of the resolution mode tables.

```
static ov5693_reg ov5693_start[] = {
```

```
{ 0x0100, 0x01 }, /* mode select streaming on */
{ OV5693_TABLE_END, 0x00 }
};

static ov5693_reg ov5693_stop[] = {
{ 0x0100, 0x00 }, /* mode select streaming off */
{ OV5693_TABLE_END, 0x00 }
};
```

Also, a table for the color bars of the test pattern is used if the `test_mode` flag is enabled, activating the test-pattern mode of the sensor. That table is required only if test pattern is supported by the sensor.

```
static ov5693_reg tp_colorbars[] = {
...
};
```

At the end of the header of the mode tables is an enumeration of all the mode tables, as well as a list that maps the enumeration to table pointers.

```
enum {
OV5693_MODE_4096X3072,
...
};
static ov5693_reg *mode_table[] = {
[OV5693_MODE_2593X1944] = mode_2593X1944,
...
};
```

The `camera_common_frmfmt` array list is a required table that sets up the format for the V4L2 framework. Each of the elements on that list contains the resolutions, the `is_hdr` flag, and the enumeration for the mode.

```
static const struct camera_common_frmfmt ov5693_frmfmt[] = {
{{2592, 1944}, 0, OV5693_MODE_2593X1944},
...
};
```

Adding a register mode table is a relatively simple and straightforward process.

To add a register mode table

1. Obtain the address-value pairs for the mode table you would like to add.

   Note: If separate stream-on and stream-off mode tables exist, you can omit them from the mode table.

2. Format the pairs according to the register structure and initialize a static array with the mode resolution as part of the name. For example:

```
static ov5693_reg mode_####x####[] = {
{ 0x0100, 0x01 }, /* mode select streaming on */
...
/* your addr and val pairs */
};
```

The mode table is now in place. Be sure to end the table array with the following:

```
{ OV5693_TABLE_END, 0x00 }
```

3. Create a new enumeration in the list of enumerations and add it to the array of mode tables:

```
enum {
...
OV5693_MODE_####X####,
}

static ov5693_reg *mode_table[] = {
...
[OV5693_MODE_####X####] = mode_####x####,
};
```

4. Add the new mode to the camera_common frmfmt array:

```
static const struct camera_common_frmfmt ov5693_frmfmt[] = {
...
{{####, ####}, 0, OV5693_MODE_####X####},
};
```

Open Source Software Licenses

This topic provides license information about the open source software libraries included in this NVIDIA product.

GNU General Public License 2.0

mtd-utils

mtd-utils is a collection of utilities licensed under GPLv2 and copyrighted by various individuals. Vibrante Android uses the following utilities from this collection.

mkfs

Copyright (C) 2008 Nokia Corporation.

ubinize

FreeType 6 version 2.4.12

ddccontrol version 0.2

FreeImage 3 version 3.15.1

FreeImage is currently maintained by HarvÃ© Drolon.

FreeImage is licensed under the GNU General Public License (GPL) and the FreeImage Public License (FIPL). You can choose the license that has the most advantages for you.

License Information

This product includes copyrighted third-party software licensed under the terms of the GNU General Public License. All third-party software packages are copyright by their respective authors. Third Party Terms are hereby incorporated into the Agreement by this reference

GNU General Public License

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it.By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it.(Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.)You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price.Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have.You must make sure that they, too, receive or can get the source code.And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software.If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents.We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary.To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License.The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language.(Hereinafter, translation is included without limitation in the term "modification".)Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope.The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License.(Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole.If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works.But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code.(This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it.For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable.However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License.Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights,

from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it.However, nothing else grants you permission to modify or distribute the Program or its derivative works.These actions are prohibited by law if you do not accept this License.Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions.You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License.If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all.For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices.Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded.In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time.Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number.If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation.If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission.For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this.Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program.It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.> Copyright (C) <year><name of author>

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; either version 2 of the License, or (at your option) any later version.

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.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'. This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License.Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary.Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program `Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989 Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs.If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library.If this is what you want to do, use the GNU Lesser General Public License instead of this License.

GNU Lesser General Public License 2.1

This product includes copyrighted third-party software licensed under the terms of the GNU Lesser General Public License. All third-party software packages are copyright by their respective authors.

GNU Lesser General Public License

Version 2.1, February 1999

[This is the first released version of the Lesser GPL.It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it.By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it.You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price.Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights.These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you.You must make sure that they, too, receive or can get the source code.If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it.And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library.Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program.We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder.Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License.This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License.We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library.The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom.The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License.It also provides other free software developers Less of an advantage over competing non-free programs.These disadvantages are the reason we use the ordinary General Public License for many libraries.However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard.To achieve this, non-free programs must be allowed to use the library.A more frequent case is that a free library does the same job as widely used non-free libraries.In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software.For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow.Pay close attention to the difference between a "work based on the library" and a "work that uses the library".The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms.A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language.(Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it.For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope.The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it).Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) The modified work must itself be a software library.

b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application.Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole.If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works.But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library.To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License.(If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.)Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library".Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library".The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library.The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work.(Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of

your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License.You must supply a copy of this License.If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License.Also, you must do one of these things:

a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library.(It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library.A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it.However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system.Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities.This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License.Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License.However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it.However, nothing else grants you permission to modify or distribute the Library or its derivative works.These actions are prohibited by law if you do not accept this License.Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions.You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License.If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all.For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices.Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded.In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number.If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation.If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission.For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this.Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE

LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU.SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change.You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library.It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.> Copyright (C) <year><name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library 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 Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA02110-1301USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary.Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library `Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990 Ty Coon, President of Vice

That's all there is to it!

Legal Information

Notice

Trademarks

Copyright