# API видео для Linux

**COLLABORATORS**

| | *TITLE* :<br><br>API видео для Linux | |
|---|---|---|

| *ACTION* | *NAME* | *DATE* | *SIGNATURE* |
|---|---|---|---|
| WRITTEN BY | Сергей Ларионов | 1.0, 17 мая 2017 | |

**REVISION HISTORY**

| NUMBER | DATE | DESCRIPTION | NAME |
|---|---|---|---|
| 1.0.0 | 1.0, 17 мая 2017 | | СЛ |

# Содержание

# Список иллюстраций



# Список таблиц

Этот документ является частичным переводом джокумента ”Video for Linux Two API Specification”, взятого с https://linuxtv.org/downloads/v4l-dvb-apis/uapi/v4l/common.html .

# Общие элементы API

Программирование v4l2 устройства состоит из следующих этапов:

- Открытие устройства
- Изменение свойств устройства, выбор видео-и звукового ввода, стандарт видео, яркость изображения и т.д.
- Согласование формата данных
- Согласование метода ввода-вывода
- Фактический цикл ввода-вывода
- Закрытие устройства

На практике большинство шагов являются необязательными и могут быть выполнены вне порядка. Это зависит от типа устройства v4l2, Вы можете прочитать сведения о деталях в интерфейсах. В этой главе обсуждаются основные концепции, применимые ко всем устройствам.

## Открытие и закрытие устройств

### Именование устройств

Драйверы v4l2 реализуются в качестве модулей ядра, загружаются вручную системным администратором или автоматически, при обнаружении устройства. Модули драйверов подключаются к модулю ядра ”videodev”. Они предоставляют вспомогательные функции и общий интерфейс приложения, описаный в этом документе.

Каждый из этих драйверов регистрирует один или несколько узлов устройств с major номером 81 и minor номером в диапазоне от 0 до 255. Minor номера выделяются динамически, если ядро не компилируется с параметром ядра CONFIG_VIDEO_FIXED_MINOR_RANGES. В этом случае minor номера распределяются в диапазоне, зависящем от типа узла устройства (видео, Радио и т. д.).

Многие драйверы поддерживают параметры модуля ”video_nr”, ”radio_nr” или ”vbi_nr” для выбора конкретных номеров узлов видео/радио/vbi. Это позволяет пользователю запросить устройство по имени узла , например /dev/video5, вместо того, чтобы предоставлять дело случаю. Если драйвер поддерживает несколько устройств одного типа, можно назначить несколько номеров узлов, разделенных запятыми:

```
> modprobe mydriver video_nr=0,1 radio_nr=0,1
```

В /etc/modules.conf это может быть написано следующим образом:

```
options mydriver video_nr=0,1 radio_nr=0,1
```

Если в качестве параметра модуля не задан номер узла устройства, драйвер предоставляет значение по умолчанию.

Обычно udev автоматически создаст узлы устройств в /dev. Если udev не установлен, то необходимо включить параметр ядра CONFIG_VIDEO_FIXED_MINOR_RANGES, чтобы иметь возможность правильно связать minor с major устройства. Т. е. необходимо убедиться, что minor 5 соответствует имени узла устройства video5. При этом параметре ядра различные типы устройств имеют разные диапазоны minor номеров. Эти диапазоны перечислены в интерфейсах.

Создание специальных символьных файлов (с помощью mknod) является привилегированной операцией, а устройства не могут открываться по номерам major.minor. Это означает, что приложения не могут надежно проверять загруженные или установленные драйверы. Пользователь должен задать имя устройства, или приложение может попытаться использовать традиционные имена устройств.

### Связанные устройства

Устройства могут поддерживать несколько функций. Пример, видеозахват, VBI захват и поддержка радиовещания.

API V4L2 создает различные узлы для каждой из этих функций.

Интерфейс API V4L2 был разработан с идеей о том, что один узел устройства может поддерживать все функции. Однако на практике это никогда не работало: Эта ”функция” никогда не использовалась приложениями, и многие драйверы не поддерживали ее, и если они сделали это, то, безусловно, это никогда не проверялось. Кроме того, переключение узла устройства между различными функциями работает только при использовании API потокового ввода-вывода, а не с помощью API read()/write ().

Сегодня каждый узел устройства поддерживает только одну функцию.

Помимо ввода и вывода видео данных, оборудование может также поддерживать оцифровку или воспроизведение звука. Если это так, то эти функции реализованы в качестве устройств с помощью ALSA PCM с дополнительными устройствами звукомикшера ALSA.

Одна из проблем со всеми этими устройствами заключается в том, что API V4L2 не даёт никаких подсказок для поиска связи этих устройств. Некоторые очень сложные устройства используют контроллер мультимедиа (см. часть IV — API-интерфейс контроллера мультимедиа), который можно использовать для этой цели. Но большинство драйверов не используют его, и хотя существует некоторый код, использующий sysfs для обнаружения связанных устройств (см. libmedia_dev в V4L-служебном репозитории), нет библиотеки, которая может предоставить единый интерфейс API для устройств и устройств на базе контроллера мультимедиа, которые не используют контроллер мультимедиа. Если вы хотите работать над этим, пожалуйста, напишите в: Список рассылки для Linux-Media:https://linuxtv.org/lists.php.

### Множественные открытия

V4L2 устройства могут открываться более одного раза. [1] При поддержке драйвера пользователи могут, например, запустить ”панельное” приложение , чтобы изменить такие элементы управления, как яркость или громкость звука, в то время как другое приложение захватывает видео и аудио. Другими словами, прикладные приложения сопоставимы с приложением для микширования звука в ALSA. Однако, открытие V4L2 устройства не должно изменять состояние этого устройства. [2]

После того как приложение выделило буферы памяти, необходимые для потоковой передачи данных (с помощью вызова *ioctl* VIDIOC_REQBUFS или *ioctl* VIDIOC_CREATE_BUFS, или неявно путем вызова функций *read()* или *write())*, приложение (файловый указатель) становится владельцем устройства. После этого не разрешается вносить изменения, влияющие на размеры буфера (например, путем вызова *ioctl* с VIDIOC_S_FMT), а другим приложениям больше не разрешается выделять буферы, запускать или останавливать потоковую передачу. Вместо этого будет возвращен код ошибки EBUSY.

[1] По-прежнему существуют старые и закрытые драйверы, которые не были обновлены, чтобы разрешить множественные открытия. Это означает, что для таких драйверов open() может возвращать код ошибки EBUSY, когда устройство уже используется.

[2] К сожалению, во многих драйверах, открытие радиоустройства часто переключает состояние устройства в режим радиовещания. Это поведение должно быть исправлено в конце концов, поскольку оно нарушает спецификацию V4L2

Простое открытие V4L2 устройства не предоставляет монопольного доступа.[3] инициализация обмена данными, однако, присваивает этому дескриптору файла право на чтение или запись запрошенного типа данных, а также на изменение связанных свойств. Приложения могут запросить дополнительные привилегии доступа с помощью механизма приоритета, описанного в ”Приоритет приложения”.

#### Общие потоки данных

V4L2 драйверы не должны поддерживать чтения или записи одного потока данных несколькими приложениями на одном устройстве путем копирования буферов, мультиплексирования по времени или аналогичных средств. Это лучше осуществляется прокси-приложением в пространстве пользователя.

#### Функции

Чтобы открыть и закрыть V4L2 устройство, приложения должны использовать функции *open()* и *close()*, соответственно. Устройства программируются с помощью функции *ioctl()*, как описано в следующих разделах.

## Возможности запросов

Поскольку V4L2 охватывает широкий спектр устройств, не все аспекты API в равной степени применимы ко всем типам устройств. Кроме того, устройства одного и того же типа имеют разные возможности, и эта спецификация позволяет пропустить несколько сложных и менее важных частей API.

*ioctl* команда VIDIOC_QUERYCAP позволяет проверить уровень совместимости устройства ядра с данной спецификацией и для запроса функций и методов ввода-вывода, поддерживаемых устройством.

Начиная с ядра версии 3.1, *ioctl* команда VIDIOC_QUERYCAP возвращает версию API V4L2, используемую драйвером, при этом, она обычно соответствует версии ядра. Нет необходимости использовать *ioctl* команду VIDIOC_QUERYCAP, чтобы проверить, поддерживается ли конкретный *ioctl*, ядро V4L2 теперь возвращает ENOTTY, если драйвер не обеспечивает поддержку конкретной *ioctl* коанды.

Другие возможности могут быть запрошены путем вызова соответствующего *ioctl*, например *ioctl* команда VIDIOC_ENUMINPUT, позволяет узнать о количестве, типах и именах видеоразъемов на устройстве. Хотя абстракция является одной из основных задач этого API, *ioctl* команда VIDIOC_QUERYCAP используется для надежной идентификации драйвера, приложениями, предназначенными для конкретного драйвера.

Все драйверы V4L2 должны поддерживать *ioctl* команду VIDIOC_QUERYCAP. Приложения всегда должны вызывать этот запрос *ioctl* после открытия устройства.

## Приоритет приложения

Если несколько приложений совместно используют устройство, может быть желательным назначить им различные приоритеты. Вопреки традиционной точке зрения, приложение видеозаписи может, например, блокировать изменение элементов управления видео или переключение текущего ТВ-канала другими приложениями. Другая цель заключается в том, чтобы разрешить приложениям с низким приоритетом работать в фоновом режиме, что может быть прервано приложениями, управляемыми пользователями, и автоматически восстановить контроль над устройством позднее.

[3] Драйверы могут распознать флаг открытия O_EXCL. В настоящее время это не требуется, так как приложения не могут знать, действительно ли это работает

Поскольку эти функции не могут быть реализованы полностью в пространстве пользователя, V4L2 определяет *ioctl* команды VIDIOC_G_PRIORITY и VIDIOC_S_PRIORITY, чтобы запросить у них связь с дескриптором файла. При открытии устройства назначается средний приоритет, совместимый с более ранними версиями V4L2, а драйверы не поддерживают эти *ioctl*. Приложения, требующие другого приоритета, обычно будут вызывать VIDIOC_S_PRIORITY после проверки устройства с помощью *ioctl* команды VIDIOC_QUERYCAP.

*ioctl* изменяющие свойства драйвера, например VIDIOC_S_INPUT, возвращают код ошибки EBUSY после того, как другое приложение получило более высокий приоритет.

## Видео вход и выход

Видеовходы и выходы являются физическими соединителями устройства. Это может быть, например, RF-соединители (антенна/кабель), CVBS, известные как композитное видео, s-video и RGB соединители. Датчики камеры также считаются входным видео. Устройства захвата видео и VBI имеют входные данные. Устройства вывода видео и VBI имеют выход, по крайней мере, по одному. Радиоустройства не имеют видео-входов и выходов.

Для получения сведений о количестве и атрибутах доступных входов и выходов, приложения могут перечислять их с помощью вызова *ioctl* VIDIOC_ENUMINPUT и VIDIOC_ENUMOUTPUT, соответственно. Структура V4L2_input, возвращенная запросом *ioctl* VIDIOC_ENUMINPUT, также содержит сигнал: сведения о состоянии, применимые при запросе текущего видеовхода.

Вызовы *ioctl* VIDIOC_G_INPUT и VIDIOC_G_OUTPUT возвращают индекс текущего видео-ввода или вывода. Чтобы выбрать другие ввод или вывод, приложения вызываеет *ioctl* VIDIOC_S_OUTPUT. Драйверы должны реализовывать все входные запросы *ioctl*, если устройство имеет один или несколько входов, все выходные *ioctl*, если устройство имеет один или несколько выходных данных.

#### Пример: Сведения о текущем вводе видео

```
struct v4l2_input input;
int index;

if (-1 == _ioctl_(fd, VIDIOC_G_INPUT, &index)) {
        perror(”VIDIOC_G_INPUT”);
        exit(EXIT_FAILURE);
}

memset(&input, 0, sizeof(input));
input.index = index;

if (-1 == _ioctl_(fd, VIDIOC_ENUMINPUT, &input)) {
        perror(”VIDIOC_ENUMINPUT”);
        exit(EXIT_FAILURE);
}

printf(”Current input: %s\\n”, input.name);
```

#### Пример: Переключение на первый видеовход

```
int index;

index = 0;

if (-1 == _ioctl_(fd, VIDIOC_S_INPUT, &index)) {
```

```
        perror("VIDIOC_S_INPUT");
        exit(EXIT_FAILURE);
}
```

## Аудио входы и выходы

Аудио-и входы и выходы физически соединители на устройстве. При выборе источника видео также выбирается источник звука. Это становится очевидно, когда источник видео и аудио - один тюнер. Дополнительные звуковые разъемы могут сочетаться с более чем одним видеовходом или выходом. Если предположить, что существуют два композитных видео входа и два звуковых входа, то возможно, до четырех допустимых комбинаций. Связь видео и аудио коннекторами определяется в поле audioset сопровождения соответствующей структуры V4L2_input или структуры V4L2_output, где каждый бит представляет номер индекса, начиная с нуля, одного звукового ввода или вывода.

Аудиовходы и видеовходы и выводы связаны. Выбор источника видеосигнала также выбирает аудио источник. Это является наиболее очевидным, когда источник видеосигнала и источник звука - тюнер. Более того, один аудиоразъем может объединиться более чем с одним видеовходом или выходом. При наличии двух входов составного видеосигнала и двух аудиовходов, может быть до четырех допустимых комбинаций. Отношение видео и аудио соединителей определены в поле audioset соответствующей структуры v4l2_input или v4l2_output, где каждый бит представляет индекс, начиная с нуля, одного аудиовхода или выхода.

Для получения сведений о количестве и атрибутах доступных входов и выходов, приложения могут перечислять их с помощью вызова *ioctl* VIDIOC_ENUMINPUT и VIDIOC_ENUMOUTPUT, соответственно. Структура V4L2_input, возвращенная запросом *ioctl* VIDIOC_ENUMINPUT, также содержит сигнал: сведения о состоянии, применимые при запросе текущего видеовхода.

VIDIOC_G_AUDIO и VIDIOC_G_AUDOUT сообщают о текущем аудио вводе и выходе, соответственно.

---

**Замечание**
Обратите внимание, что в отличие от VIDIOC_G_INPUT и VIDIOC_G_OUTPUT эти *ioctl* возвращают структуру как и вызов *ioctl* VIDIOC_ENUMAUDIO и VIDIOC_ENUMAUDOUT, а не только индекс.

---

Для выбора звукового входа и изменения его свойств, приложения должны вызвать *ioctl* VIDIOC_S_AUD
Чтобы выбрать звуковой выход (которые, в настоящее время, не имеют изменяемых свойств), вызовите *ioctl* VIDIOC_S_AUDOUT.

Драйверы должны реализовывать все входные запросы *ioctl*, если устройство имеет один или несколько входов, все выходные *ioctl*, если устройство имеет один или несколько выходных данных. Когда устройство имеет какие-либо аудио входы или выходы, драйвер должен установить флаг 4L2_CAP_AUDIO в структуре v4l2_capability, возвращаемой *ioctl* VIDIOC_QUERYCAP.

### Пример: Сведения о текущем аудио входе

```
struct v4l2_audio audio;

memset(&audio, 0, sizeof(audio));

if (-1 == _ioctl_(fd, VIDIOC_G_AUDIO, &audio)) {
        perror("VIDIOC_G_AUDIO");
        exit(EXIT_FAILURE);
}

printf("Current input: %s\\n", audio.name);
```

**Пример: Переключение на первый звуковой вход**

```
struct v4l2_audio audio;

memset(&audio, 0, sizeof(audio)); /* clear audio.mode, audio.reserved */

audio.index = 0;

if (-1 == _ioctl_(fd, VIDIOC_S_AUDIO, &audio)) {
        perror(”VIDIOC_S_AUDIO”);
        exit(EXIT_FAILURE);
}
```

## Тюнеры и модуляторы

### Тюнер

Устройства видеоввода могут иметь один или несколько тюнеров, для демодуляции RF сигнала. Каждый тюнер связан с одним или несколькими видеовходами в зависимости от количества RF разъёмов на тюнере. Поле ”type” соответствующей структуры v4l2_input, возвращенной *ioctl* VIDIOC_ENUMINPUT, установлено значением ”V4L2_INPUT_TYPE_TUNER”, а в поле ”tuner” содержится номер индекса тюнера.

Устройства ввода радиоприемника имеют ровно один тюнер с нулевым индексом, без видеовходов.

Чтобы запросить и изменить свойства тюнера, используйте *ioctl* VIDIOC_G_TUNER и VIDIOC_S_TUNER соответственно. Структура v4l2_tuner, возвращенная вызовом VIDIOC_G_TUNER, также содержит сведения о состоянии сигнала, которые имеют смысл, когда текущий видео или радио вход запрашивает данные.

---

**Замечание**
VIDIOC_S_TUNER не переключается на текущий тюнер, если в нем есть несколько. Тюнер определяется только текущим видео входом. Драйверы должны поддерживать оба IOCTL и устанавливать флаг V4L2_CAP_TUNER в структуре v4l2_capability, возвращаемой *ioctl* VIDIOC_QUERYCAP, когда устройство имеет один или несколько тюнеров.

---

### Модуляторы

Устройства видеовывода могут иметь один или несколько модуляторов, которые модулируют видеосигнал для передачи или при соединении с антенным входом, подключенным к телевизору или видеомагнитофону. Каждый модулятор связан с одним или несколькими видеовыходами, в зависимости от количества разъемов RF на модуляторе. Поле ”type” соответствующей структуры v4l2_output *ioctl* , возвращенной вызовом *ioctl* VIDIOC_ENUMOUTPUT, имеет значение ”V4L2_OUTPUT_TYPE_MODULATOR”, а поле ”modulator” содержит указатель количества модуляторов.

Устройства вывода радиоприемников имеют ровно один модулятор с нулевым индексом, без видеовыходов.

Видео или радиоустройство не могут поддерживать одновременнно как тюнер, так и модулятор. Для такого оборудования необходимо использовать два отдельных узла устройств, один из которых поддерживает функции тюнера и один из них поддерживает функции модулятора. Причина заключается в ограничении VIDIOC_S_FREQUENCY, где нельзя указать, задана ли эта частота для тюнера или модулятора.

Для запроса и изменения свойств модулятора приложения используют *ioctl* как VIDIOC_G_MODULATO
так и VIDIOC_S_MODULATOR. Обратите внимание, что VIDIOC_S_MODULATOR не переключает-
ся на текущий модулятор если в нем есть их несколько. Модулятор определяется только теку-
щим видеовыходом. Драйверы должны поддерживать оба IOCTL и устанавливать флаг V4L2_CAP_MOD
в структуре v4l2_capability, возвращаемой *ioctl* VIDIOC_QUERYCAP, когда устройство имеет один
или несколько модуляторов. 1.6.3. Радиочастота

Чтобы получить и установить для тюнера или модулятора радиочастоту, приложения должны использовать *ioctl* VIDIOC_G_FREQUENCY и VIDIOC_S_FREQUENCY, которые принимают как параметр указатель на структуру v4l2_frequency. Эти *ioctl* используются как для ТВ, так и для радиоустройств. Драйверы должны поддерживать оба *ioctl*, когда поддерживаются запросы для тьюнера или модулятора, или если устройство является радиоустройством.

## Стандарты видео

Видеоустройства, как правило, поддерживают один или несколько различных стандартов видео или разновидностей стандартов. Каждый видеовход и выход может поддерживать различный набор стандартов. Этот набор сообщается полем ”std” в структуре v4l2_input и v4l2_output, возвращаемым, соответственно, запросами *ioctl* VIDIOC_ENUMINPUT и VIDIOC_ENUMOUTPUT.

V4L2 определяет один бит для каждого аналогового видео стандарта, используемого в настоящее время во всем мире, и отменяет биты для стандартов, определяемых драйверами, e. g. Гибридные стандарты для просмотра видеозаписей NTSC на телевизорах PAL и наоборот. Приложения могут использовать стандартные биты для выбора определенного стандарта, но при этом пользователю рекомендуется выбрать меню поддерживаемых стандартов. Для нумерации и запроса атрибутов поддерживаемых стандартов, приложение используется запрос IOCTL VIDIOC_ENUM

Многие из установленных стандартов фактически являются лишь вариациями нескольких основных стандартов. Оборудование может фактически не различать их, или делать это внутренним образом и автоматически переключаться. Поэтому перечисленные стандарты также содержат наборы из одного или нескольких стандартных битов.

Предположим, что гипотетический тьюнер имеет возможность демодуляции сигналов B/PAL, G/PAL и I/PAL. Первым перечислением стандартов является набор B и G/PAL, который автоматически переключается в зависимости от выбранной радиочастоты в диапазоне УВЧ или ОВЧ. Перечисление задаётся выбором ”PAL-B/G” или ”PAL-I”. Аналогичный комплексный ввод может свернуть стандарты, перечисление как ”PAL-B/G/H/I”, ”NTSC-M” и ”CEKAM-D/K”. [4]

Чтобы запросить и выбрать стандарт, используемый текущими приложениями видеоввода или вывода, вызовите *ioctl* VIDIOC_G_STD и VIDIOC_S_STD соответственно. Стандарт приёма можно определить с помощью *ioctl* VIDIOC_QUERYSTD,

**Внимание**
Параметр всех этих *ioctl* является указателем на тип v4l2_std_id ( набор стандартов), а не на индекс в перечислении стандартнов. Драйверы должны реализовывать все стандартные *ioctl*, если устройство имеет один или несколько видеовходов или выходов данных.

Специальные правила применяются к таким устройствам, как USB-камеры, в которых понятие видеостандартов имеет мало смысла. Как правило, для любого устройства захвата или вывода, которое:

- не способно захвата полей или кадров при номинальном значении потока для видеостандарта,
- или вообще не поддерживает стандартные форматы видео.

[4] Некоторые пользователи уже путаются с техническими терминами PAL, NTSC и SEKAM. Нет смысла просить их различать B, G, D или K, когда программное или аппаратное обеспечение может сделать это автоматически.

В этом случае драйвер должен устанавливать поле ”std” в структурах v4l2_input и struct v4l2_output равным нулю, а вызовы *ioctl* VIDIOC_G_STD, VIDIOC_S_STD, VIDIOC_QUERYSTD и VIDIOC_ENUMSTD, возвращают код ошибки ENOTTY или код ошибки EINVAL.

Приложения могут использовать флаги возможностей ввода и вывода, чтобы определить, могут ли быть использованы для работы с заданным вводом или выводом.

**Пример: Сведения о текущем стандарте видео**

```
v4l2_std_id std_id;
struct v4l2_standard standard;

if (-1 == _ioctl_(fd, VIDIOC_G_STD, &std_id)) {
        /* Note when VIDIOC_ENUMSTD always returns ENOTTY this
        is no video device or it falls under the USB exception,
        and VIDIOC_G_STD returning ENOTTY is no error. */

        perror(”VIDIOC_G_STD”);
        exit(EXIT_FAILURE);
}

memset(&standard, 0, sizeof(standard));
standard.index = 0;

while (0 == _ioctl_(fd, VIDIOC_ENUMSTD, &standard)) {
        if (standard.id & std_id) {
                printf(”Current video standard: %s\\n”, standard.name);
                exit(EXIT_SUCCESS);
        }

        standard.index++;
}

/* EINVAL indicates the end of the enumeration, which cannot be
empty unless this device falls under the USB exception. */

if (errno == EINVAL || standard.index == 0) {
        perror(”VIDIOC_ENUMSTD”);
        exit(EXIT_FAILURE);
}
```

**Пример: Перечисление стандартов видео, поддерживаемых текущим вводом**

```
struct v4l2_input input;
struct v4l2_standard standard;

memset(&input, 0, sizeof(input));

if (-1 == _ioctl_(fd, VIDIOC_G_INPUT, &input.index)) {
        perror(”VIDIOC_G_INPUT”);
        exit(EXIT_FAILURE);
}

if (-1 == _ioctl_(fd, VIDIOC_ENUMINPUT, &input)) {
        perror(”VIDIOC_ENUM_INPUT”);
        exit(EXIT_FAILURE);
}
```

```
printf(”Current input %s supports:\\n”, input.name);

memset(&standard, 0, sizeof(standard));
standard.index = 0;

while (0 == _ioctl_(fd, VIDIOC_ENUMSTD, &standard)) {
        if (standard.id & input.std)
        printf(”%s\\n”, standard.name);

        standard.index++;
}

/* EINVAL indicates the end of the enumeration, which cannot be
empty unless this device falls under the USB exception. */

if (errno != EINVAL || standard.index == 0) {
        perror(”VIDIOC_ENUMSTD”);
        exit(EXIT_FAILURE);
}
```

**Пример: Выбор нового стандарта видео**

```
struct v4l2_input input;
v4l2_std_id std_id;

memset(&input, 0, sizeof(input));

if (-1 == _ioctl_(fd, VIDIOC_G_INPUT, &input.index)) {
        perror(”VIDIOC_G_INPUT”);
        exit(EXIT_FAILURE);
}

if (-1 == _ioctl_(fd, VIDIOC_ENUMINPUT, &input)) {
        perror(”VIDIOC_ENUM_INPUT”);
        exit(EXIT_FAILURE);
}

if (0 == (input.std & V4L2_STD_PAL_BG)) {
        fprintf(stderr, ”Oops. B/G PAL is not supported.\\n”);
        exit(EXIT_FAILURE);
}

/* Note this is also supposed to work when only B
   or G/PAL is supported. */

std_id = V4L2_STD_PAL_BG;

if (-1 == _ioctl_(fd, VIDIOC_S_STD, &std_id)) {
        perror(”VIDIOC_S_STD”);
        exit(EXIT_FAILURE);
}
```

## Время в цифровом видео (DV)

Обсуждаемые сейчас видеостандарты тесно связаны с аналоговым телевидением и соответствующего таймирования видео . Сегодня существует множество других аппаратных интерфейсов, таких как ТВ-интерфейсы высокого разрешения (HDMI), VGA и DVI разъемы, и т. д., которые

выдают видеосигналы, и необходимо расширить интерфейс API для выбора времени видео для этих интерфейсов. Поскольку невозможно расширить v4l2_std_id из-за ограниченного количества бит, новый набор *ioctl* был добавлен для установки/получения времени на входе и выходе видео.

Эти *ioctl* определяют детальные временные характеристики цифрового видео, определяющие каждый видеоформат. Сюда входят такие параметры, как активная ширина и высота видео, полярность сигналов, входы, выходы, ширина синхронизации и т. д. Файл заголовка linux/v4l2-dv-timings. h может использоваться для получения времени для форматов в стандартах CEA-861-E и VESA DTM.

Чтобы перечислить и запросить атрибуты параметров времени DV устройств, поддерживаемых приложениями, используйте *ioctl* VIDIOC_ENUM_DV_TIMINGS, VIDIOC_SUBDEV_ENUM_DV_TIMINGS и *ioctl* VIDIOC_DV_TIMINGS_CAP, VIDIOC_SUBDEV_DV_TIMINGS_CAP. Чтобы установить время на DV устройстве, приложение должно использовать *ioctl* VIDIOC_S_DV_TIMINGS, а для получения текущих параметров времени они должны использовать *ioctl* команду VIDIOC_G_DV_TIMINGS. Чтобы обнаружить время видеозаписи, как это видно из приложений-приемников, используйте запрос *ioctl* VIDIOC_QUERY_DV_TIMINGS.

Приложения могут использовать флаги возможностей ввода и вывода, чтобы определить, можно ли использовать функции IOCTL цифрового видео с заданными входными или выходными данными.

## Пользовательские элементы управления

Обычно устройства имеют ряд пользовательских элементов управления, таких как яркость, насыщенность и т. д., которые будут представлены пользователю в графическом пользовательском интерфейсе. Однако в разных устройствах будут доступны различные элементы управления, а также диапазон возможных значений и значение по умолчанию будет отличаться от устройства к устройству. Вызовы *ioctl* предоставляют сведения об элементах управления и механизм для создания хорошего пользовательского интерфейса для этих элементов управления, которые будут работать корректно с любым устройством.

Доступ к всем элементам управления осуществляется с помощью значения их идентификатора (ID). V4L2 определяет несколько идентификаторов (ID) для конкретных целей. Драйверы также могут реализовывать собственные пользовательские элементы управления с помощью V4L2_CID_PRIVATE_BASE [5] и старших значений. Предварительно определенные идентификаторы элементов управления имеют префикс V4L2_CID_ и перечислены в Control IDs. ID используется при запросе значений атрибутов элемента управления или установке текущего значения.

Как правило, приложения должны представлять пользователю элементы управления без предположения об их назначении. Каждый элемент управления должен предоставляться со строкой имени, которую пользователь может понять. Если цель не является интуитивно понятной, разработчики драйверов должны предоставить пользователю руководство пользователя, подключаемый модуль пользовательского интерфейса или специальное приложение панели для драйвера. Предопределенные идентификаторы были введены для изменения нескольких элементов управления программным путём, например для отключения устройства во время переключения канала.

Драйверы могут перенумеровать различные элементы управления после переключения текущего видеовхода или вывода, тюнера или модулятора, а также аудио-ввода или вывода. Отличающиеся, в смысле других границ, других значений по умолчанию и текущему значению, размеру

[5] Использование V4L2_CID_PRIVATE_BASE является проблематичным, поскольку различные драйверы могут использовать один и тот же V4L2_CID_PRIVATE_BASE ID для различных элементов управления. Это затрудняет возможность программирования набора таких элементов управления, поскольку назначение таких элементов управления с одним и тем же идентификатором зависит от драйвера. Чтобы разрешить эту проблему, драйверы должны использовать уникальные ID, а V4L2_CID_PRIVATE_BASE ID мапируются на уникальные идентификаторы ядра. Рассмотрим эти V4L2_CID_PRIVATE_BASE идентификаторы, как псевдонимы для реальных идентификаторов. Многие приложения сегодня по-прежнему используют ID V4L2_CID_PRIVATE_BASE вместо использования *ioctl* VIDIOC_QUERYCTRL, VIDIOC_QUERY_EXT_CTRL and VIDIOC_QUERYMENU с флагом V4L2_CTRL_FLAG_NEXT_CTRL для перечисления всех ID поэтому поддержка V4L2_CID_PRIVATE_BASE все еще продолжается.

шага или другим пунктам меню. Элемент управления с определенным ID также может изменить имя и тип.

Если элемент управления более не применим к текущей конфигурации устройства (например, он не применим к текущему видеовходу), драйверы устанавливают флаг V4L2_CTRL_FLAG_INACTIVE.

Значения элементов управления хранятся в глобальном контексте, поэтому, при переключении они не меняются, за исключением того, что они должны остаться в пределах известных границ. Они также не меняют и тогда, когда устройство открывается или закрывается, при изменении частоты радиотюнера , как правило, никогда без запроса приложения.

V4L2 задаёт механизм событий для уведомления приложений, когда элементы управления изменяют значение (см. пункт *ioctl* VIDIOC_SUBSCRIBE_EVENT, VIDIOC_UNSUBSCRIBE_EVENT, событие V4L2_EVENT_CTRL), панельным приложениям может понадобиться использовать это, чтобы всегда отражать правильное значение элемента управления.

Все элементы управления используют машино независимый порядок байт.

### Идентификаторы элементов управления

**V4L2_CID_BASE** Первый предварительно определенный идентификатор, равный V4L2_CID_BRIGHT

**V4L2_CID_USER_BASE** Синоним V4L2_CID_BASE.

**V4L2_CID_BRIGHTNESS (integer)** Яркость изображения, или точнее, уровень черного.

**V4L2_CID_CONTRAST (integer)** Контрастность изображения или коэффициент яркости.

**V4L2_CID_SATURATION (integer)** Насыщенность цвета изображения или цветовой коэффициент.

**V4L2_CID_HUE (integer)** Цветовой тон или баланс цветов.

**V4L2_CID_AUDIO_VOLUME (integer)** Общая громкость звука. Примечание: Некоторые драйверы также обеспечивают интерфейс микшера OSS или ALSA.

**V4L2_CID_AUDIO_BALANCE (integer)** Аудио стерео-баланс. Минимальное значение соответствует тому, что вся мощность идёт влево, максимальное значение - вправо.

**V4L2_CID_AUDIO_BASS (integer)** Настройка басов.

**V4L2_CID_AUDIO_TREBLE (integer)** Настройка дисканта.

**V4L2_CID_AUDIO_MUTE (boolean)** Отключение звука, т. е. установка громкости в ноль, но без влияния на V4L2_CID_AUDIO_VOLUME. Как и драйверы ALSA, драйверы V4L2 должны выключить звук во время загрузки, чтобы избежать чрезмерного шума. На самом деле, все устройство должно быть сброшено в состояние низкого энергопотребления.

**V4L2_CID_AUDIO_LOUDNESS (boolean)** Режим громкости (усиление басов).

**V4L2_CID_BLACK_LEVEL (integer)** Другое имя для яркости (не синоним V4L2_CID_BRIGHTNESS). Этот элемент управления является устаревшим и не должен использоваться в новых драйверах и приложениях.

**V4L2_CID_AUTO_WHITE_BALANCE (boolean)** Автоматический баланс белого (камеры).

**V4L2_CID_DO_WHITE_BALANCE (button)** Это элемент управления действиями. При установке (значение игнорируется), устройство будет выполнять баланс белого, а затем удерживать текущую настройку. Сравните это с логическим значением V4L2_CID_AUTO_WHITE_BALANC которое при активации продолжает корректировать баланс белого.

**V4L2_CID_RED_BALANCE (integer)** Насыщенность красного.

**V4L2_CID_BLUE_BALANCE (integer)** Насыщенность синего.

**V4L2_CID_GAMMA (integer)** Гамма-регулировка.

**V4L2_CID_WHITENESS (integer)** Белизна для устройств серого цвета. Это синоним для V4L2_CID_G
Этот элемент управления является устаревшим и не должен использоваться в новых драйверах и приложениях.

**V4L2_CID_EXPOSURE (integer)** Экспозиция (камеры). [Единицы?]

**V4L2_CID_AUTOGAIN (boolean)** Автоматическое управление усилением/экспозицией.

**V4L2_CID_GAIN (integer)** Управление усилением.

**V4L2_CID_HFLIP (boolean)** Отразить изображение по горизонтали.

**V4L2_CID_VFLIP (boolean)** Зеркальное отражение рисунка по вертикали.

**V4L2_CID_POWER_LINE_FREQUENCY (enum)** Включает фильтр частоты линий электропитания, чтобы избежать мерцания. Возможные значения для перечисления v4l2_power_line_freque
V4L2_CID_POWER_LINE_FREQUENCY_DISABLED (0), V4L2_CID_POWER_LINE_FREQUENCY_50HZ
(1), V4L2_CID_POWER_LINE_FREQUENCY_60HZ (2) and V4L2_CID_POWER_LINE_FREQUENCY_AU
(3).

**V4L2_CID_HUE_AUTO (boolean)** Включение автоматического управления оттенком на устройстве. Влияние установки V4L2_CID_HUE в то время, как функция автоматического управления оттенком включена, не определена, драйверы должны игнорировать такой запрос.

**V4L2_CID_WHITE_BALANCE_TEMPERATURE (integer)** Этот элемент управления определяет параметры баланса белого в виде цветовой температуры по Кельвину. Драйвер должен иметь минимум 2800 (лампа накаливания) до 6500 (дневной свет). Дополнительные сведения о цветовой температуре см Wikipedia.

**V4L2_CID_SHARPNESS (integer)** Настройка фильтров резкости в камере. Минимальное значение отключает фильтры, более высокие значения дают более четкое изображение.

**V4L2_CID_BACKLIGHT_COMPENSATION (integer)** Настройка компенсации заднего света в камере. Минимальное значение отключает компенсацию подсветки.

**V4L2_CID_CHROMA_AGC (boolean)** Автоматическое управление насыщенностью цвета.

**V4L2_CID_CHROMA_GAIN (integer)** Настройка управления насыщенностью цвета (для использования при отключении автоматической регулировки AGC).

**V4L2_CID_COLOR_KILLER (boolean)** Разрешает отключение цветовой (т.е. принудительное включение черного и белого изображения в случае слабого видеосигнала).

**V4L2_CID_COLORFX (enum)** Выбор цветового эффекта. Определены следующие значения:

- V4L2_COLORFX_NONE Цветовой эффект отключен.
- V4L2_COLORFX_ANTIQUE Эффект устаревания (Старое фото).
- V4L2_COLORFX_ART_FREEZE Цветовой эффект холодного цвета.
- V4L2_COLORFX_AQUA Цвет воды, холодный тон.
- V4L2_COLORFX_BW Черный и белый.
- V4L2_COLORFX_EMBOSS Рельеф, блики и тени заменяют светлые/темные границы, а области с низкой контрастностью задаются серым фоном.
- V4L2_COLORFX_GRASS_GREEN Травяная зелень.
- V4L2_COLORFX_NEGATIVE Негатив.
- V4L2_COLORFX_SEPIA Тональность сепия.
- V4L2_COLORFX_SKETCH Эскиз.

- V4L2_COLORFX_SKIN_WHITEN Отбеливание кожи.
- V4L2_COLORFX_SKY_BLUE Небо голубое.
- V4L2_COLORFX_SOLARIZATION При использовании соляризации, изображение частично инвертировано в тональности, но инвертируются только цветовые значения выше или ниже определенного порога.
- V4L2_COLORFX_SILHOUETTE Силуэт (контур).
- V4L2_COLORFX_VIVID Яркие цвета.
- V4L2_COLORFX_SET_CBCR Компоненты цвета Cb и Cr заменяются фиксированными коэффициентами, определяемыми в элементе управления V4L2_CID_COLORFX_CBCR.

**V4L2_CID_COLORFX_CBCR (integer)** Определяет коэффициенты CB и CR для цветового эффекта V4L2_COLORFX_SET_CBCR. Биты [7:0] предоставленного 32-битового значения интерпретируются как компонент Cr, биты [15:8] как компонент Cb, а биты [31:16] должны быть равны нулю.

**V4L2_CID_AUTOBRIGHTNESS (boolean)** Включение автоматической яркости.

**V4L2_CID_ROTATE (integer)** Поворот изображения на заданный угол. Общие углы — 90, 270 и 180. При повороте изображения на 90 и 270 будет переставлены высота и ширина окна отображения. Необходимо задать новую высоту и ширину изображения, используя VIDIOC_S_FMT в соответствии с выбранным углом поворота.

**V4L2_CID_BG_COLOR (integer)** Задает цвет фона для текущего устройства вывода. Цвет фона должен быть указан в формате RGB24. Предоставленное 32-битовое значение интерпретируется как биты 0-7 красный цвет, биты 8-15 зелёный цвет, биты 16-23 синий цвет и биты 24-31 должны быть равны нулю.

**V4L2_CID_ILLUMINATORS_1 V4L2_CID_ILLUMINATORS_2 (boolean)** Включите или выключите подсветку 1 или 2 устройства (обычно на микроскопе).

**V4L2_CID_MIN_BUFFERS_FOR_CAPTURE (integer)** Это элемент управления, предназначенный только для чтения, который может быть прочитан приложением и использован в качестве подсказки для определения количества буферов захвата для передачи в REQBUFS. Это минимальное количество буферов захвата, необходимых для работы оборудования.

**V4L2_CID_MIN_BUFFERS_FOR_OUTPUT (integer)** Это элемент управления, предназначенный только для чтения, который может быть прочитан приложением и использован в качестве подсказки для определения количества ВЫХОДНЫХ буферов для передачи в REQBUFS. Значение является минимальным количеством ВЫХОДНЫХ буферов, необходимых для работы оборудования.

**V4L2_CID_ALPHA_COMPONENT (integer)** Задает альфа-компонент цвета. Когда устройство захвата (или очередь захвата устройства в памяти) создает формат кадра, включающий альфа-компонент (например, упакованные форматы изображений RGB), а значение альфа-канала не определяется устройством или входными данными очереди в памяти, этот элемент управления позволяет выбрать значение альфа-компонента для всех пикселов. Если устройство вывода (или очередь выходного устройства в памяти) использует формат кадра, не включающий альфа-канал, и устройство поддерживает обработку альфа-канала, этот элемент управления позволяет установить значение альфа-компонента для всех пикселов для дальнейшей обработки в устройстве.

**V4L2_CID_LASTP1** Конец предопределенных идентификаторов элементов управления (в настоящее время V4L2_CID_ALPHA_COMPONENT + 1).

**V4L2_CID_PRIVATE_BASE** Идентификатор первого пользовательского элемента управления (конкретного драйвера). Приложения, зависящие от конкретных пользовательских элементов управления, должны проверять имя и версию драйвера, см.

Приложения могут перечислять доступные элементы управления с помощью *ioctl* VIDIOC_QUERYCTRL VIDIOC_QUERY_EXT_CTRL и VIDIOC_QUERYMENU, получить и задать значение элемента управления с помощью *ioctl* VIDIOC_G_CTRL и VIDIOC_S_CTRL. Драйверы должны реализовывать VIDIOC_QU VIDIOC_G_CTRL и VIDIOC_S_CTRL, когда устройство имеет один или несколько элементов управления, VIDIOC_QUERYMENU при наличии одного или нескольких элементов управления типа меню.

**Пример: Перечисление всех элементов управления**

```
struct v4l2_queryctrl queryctrl;
struct v4l2_querymenu querymenu;

static void enumerate_menu(__u32 id)
{
        printf(”  Menu items:\\n”);

        memset(&querymenu, 0, sizeof(querymenu));
        querymenu.id = id;

        for (querymenu.index = queryctrl.minimum;
                querymenu.index <= queryctrl.maximum;
                querymenu.index++) {
                        if (0 == _ioctl_(fd, VIDIOC_QUERYMENU, &querymenu)) {
                        printf(”  %s\\n”, querymenu.name);
                }
        }
}

memset(&queryctrl, 0, sizeof(queryctrl));

queryctrl.id = V4L2_CTRL_FLAG_NEXT_CTRL;
while (0 == _ioctl_(fd, VIDIOC_QUERYCTRL, &queryctrl)) {
        if (!(queryctrl.flags & V4L2_CTRL_FLAG_DISABLED)) {
                printf(”Control %s\\n”, queryctrl.name);

                if (queryctrl.type == V4L2_CTRL_TYPE_MENU)
                        enumerate_menu(queryctrl.id);
        }

        queryctrl.id |= V4L2_CTRL_FLAG_NEXT_CTRL;
}

if (errno != EINVAL) {
        perror(”VIDIOC_QUERYCTRL”);
        exit(EXIT_FAILURE);
}
```

**Пример: Перечисление всех элементов управления, включая составные элементы управления**

```
struct v4l2_query_ext_ctrl query_ext_ctrl;

memset(&query_ext_ctrl, 0, sizeof(query_ext_ctrl));

query_ext_ctrl.id = V4L2_CTRL_FLAG_NEXT_CTRL | V4L2_CTRL_FLAG_NEXT_COMPOUND;
while (0 == _ioctl_(fd, VIDIOC_QUERY_EXT_CTRL, &query_ext_ctrl)) {
        if (!(query_ext_ctrl.flags & V4L2_CTRL_FLAG_DISABLED)) {
                printf(”Control %s\\n”, query_ext_ctrl.name);
```

```
                if (query_ext_ctrl.type == V4L2_CTRL_TYPE_MENU)
                        enumerate_menu(query_ext_ctrl.id);
        }

        query_ext_ctrl.id |= V4L2_CTRL_FLAG_NEXT_CTRL | V4L2_CTRL_FLAG_NEXT_COMPOUND;
}

if (errno != EINVAL) {
        perror("VIDIOC_QUERY_EXT_CTRL");
        exit(EXIT_FAILURE);
}
```

**Пример: Перечисление всех пользовательских элементов управления (старый стиль)**

```
memset(&queryctrl, 0, sizeof(queryctrl));

for (queryctrl.id = V4L2_CID_BASE;
        queryctrl.id < V4L2_CID_LASTP1;
        queryctrl.id++) {

        if (0 == _ioctl_(fd, VIDIOC_QUERYCTRL, &queryctrl)) {
                if (queryctrl.flags & V4L2_CTRL_FLAG_DISABLED)
                continue;

                printf("Control %s\\n", queryctrl.name);

                if (queryctrl.type == V4L2_CTRL_TYPE_MENU)
                        enumerate_menu(queryctrl.id);
        } else {
                if (errno == EINVAL)
                continue;

                perror("VIDIOC_QUERYCTRL");
                exit(EXIT_FAILURE);
        }
}

for (queryctrl.id = V4L2_CID_PRIVATE_BASE;;
                queryctrl.id++) {
        if (0 == _ioctl_(fd, VIDIOC_QUERYCTRL, &queryctrl)) {
                if (queryctrl.flags & V4L2_CTRL_FLAG_DISABLED)
                        continue;

                printf("Control %s\\n", queryctrl.name);

                if (queryctrl.type == V4L2_CTRL_TYPE_MENU)
                        enumerate_menu(queryctrl.id);
        } else {
                if (errno == EINVAL)
                        break;
                        perror("VIDIOC_QUERYCTRL");
                exit(EXIT_FAILURE);
        }
}
```

**Пример: Изменение элементов управления**

```
struct v4l2_queryctrl queryctrl;
struct v4l2_control control;

memset(&queryctrl, 0, sizeof(queryctrl));
queryctrl.id = V4L2_CID_BRIGHTNESS;

if (-1 == _ioctl_(fd, VIDIOC_QUERYCTRL, &queryctrl)) {
        if (errno != EINVAL) {
                perror("VIDIOC_QUERYCTRL");
                exit(EXIT_FAILURE);
        } else {
                printf("V4L2_CID_BRIGHTNESS is not supportedn");
        }
} else if (queryctrl.flags & V4L2_CTRL_FLAG_DISABLED) {
        printf("V4L2_CID_BRIGHTNESS is not supportedn");
} else {
        memset(&control, 0, sizeof (control));
        control.id = V4L2_CID_BRIGHTNESS;
        control.value = queryctrl.default_value;

        if (-1 == _ioctl_(fd, VIDIOC_S_CTRL, &control)) {
                perror("VIDIOC_S_CTRL");
                exit(EXIT_FAILURE);
        }
}

memset(&control, 0, sizeof(control));
control.id = V4L2_CID_CONTRAST;

if (0 == _ioctl_(fd, VIDIOC_G_CTRL, &control)) {
        control.value += 1;

        /* The driver may clamp the value or return ERANGE, ignored here */

        if (-1 == _ioctl_(fd, VIDIOC_S_CTRL, &control)
                        && errno != ERANGE) {
                perror("VIDIOC_S_CTRL");
                exit(EXIT_FAILURE);
        }
        /* Ignore if V4L2_CID_CONTRAST is unsupported */
} else if (errno != EINVAL) {
        perror("VIDIOC_G_CTRL");
        exit(EXIT_FAILURE);
}

control.id = V4L2_CID_AUDIO_MUTE;
control.value = 1; /* silence */

/* Errors ignored */
_ioctl_(fd, VIDIOC_S_CTRL, &control);
```

## Расширенные элементы управления

### Введение

Изначально спроектированный механизм управления, был предназначен для использования в настройках пользователя (яркость, насыщенность и т. д.). Однако это оказалось очень полезной

моделью для реализации более сложных API-интерфейсов драйверов, в которых каждый драйвер реализует только подмножество более крупного API.

API для кодирования MPEG был движущей силой при проектировании и реализации этого расширенного механизма управления: стандарт MPEG очень большой и, в настоящее время поддерживается аппаратными кодировщиками MPEG частично - каждый реализует только подмножество этого стандарта. Кроме того, многие параметры, относящиеся к тому, как видео кодируется в поток MPEG, специфичны для чипов кодировки MPEG, поскольку стандарт MPEG определяет формат результирующего потока MPEG, а не то, как видео фактически было закодировано в этот формат.

К сожалению, у исходного API-интерфейса отсутствовали некоторые возможности, необходимые для этих новых видов использования, и поэтому он был расширен в расширенном (без ужасных первоначальных иимён) API-интерфейсе управления.

Хотя API-интерфейс MPEG был первой попыткой использования расширенного API-интерфейс, в настоящее время уже существуют и другие классы расширенных элементов управления, например, элементы управления фотокамерой и элементы управления FM-передатчиком. Интерфейс API расширенных элементов управления, а также все расширенные классы элементов управления описаны в нижеследующем тексте.

### Расширенный интерфейс API

Доступны три новых *ioctl*: VIDIOC_G_EXT_CTRLS, VIDIOC_S_EXT_CTRLS и VIDIOC_TRY_EXT_CTRLS. Эти *ioctl* действуют на массивах элементов управления (в отличие от VIDIOC_G_CTRL и VIDIOC_S_CTRL *ioctl*, которые действуют на единственный элемент управления). Это необходимо, поскольку часто требуется атомарное изменение нескольких элементов управления одновременно.

Каждый из новых *ioctl* ожидает указатель на структуру v4l2_ext_controls. Эта структура содержит указатель на массив элементов управления, количество элементов управления в этом массиве и класс элемента управления. Классы элементов управления используются для группирования сходных элементов управления в один класс. Пример, класс управления V4L2_CTRL_CLASS_USER содержит все пользовательские элементы управления (i. e. все элементы управления, которые также могут быть заданы с помощью старого *ioctl* VIDIOC_S_CTRL). Класс управляющих элементов V4L2_CTRL_CLASS_MPEG содержит все элементы управления, относящиеся к кодировке MPEG и т. д.

Все элементы управления в массиве элементов управления должны принадлежать к указанному классу элемента управления. Если это не так, возвращается ошибка.

Также можно использовать пустой массив элементов управления (count == 0), чтобы проверить, поддерживается ли указанный класс элементов управления.

Массив элементов управления является массивом структур v4l2_ext_control. Структура v4l2_ext_contro очень похожа на структуру v4l2_control, за исключением того, что она также позволяет передавать 64-разрядные значения и указатели.

Поскольку структура v4l2_ext_control поддерживает указатели, теперь можно также иметь элементы управления с составными типами, такими как n-мерные массивы и/или структуры. Необходимо задать флаг V4L2_CTRL_FLAG_NEXT_COMPOUND при перечислении элементов управления, которые фактически могут иметь возможность видеть такие составные элементы управления. Другими словами, эти элементы управления с составными типами должны использоваться только программными средствами.

Поскольку такие составные элементы управления должны предоставлять больше информации о себе, чем это возможно при вызовах *ioctl* VIDIOC_QUERYCTRL, VIDIOC_QUERY_EXT_CTRL и VIDIOC_QUERYMENU, то был добавлен вызов *ioctl* VIDIOC_QUERY_EXT_CTRL. В частности, этот запрос IOCTL предоставляет размеры n-мерного массива, если этот элемент управления состоит из нескольких элементов.

**Замечание**
Важно понимать, что благодаря гибкости элементов управления необходимо проверять, поддерживается ли в драйвере элемент управления, который требуется установить, и каков допустимый диапазон значений. Чтобы проверить это, используйте *ioctl* VIDIOC_QUERYCTRL, VIDIOC_QUERY_EXT_CTRL и VIDIOC_QUERYMENU (или VIDIOC_QUERY_EXT_CTRL) и VIDIOC_QUERYMENU. Возможно, что некоторые индексы меню в элементе управления типа V4L2_CTRL_TYPE_MENU могут не поддерживаться (VIDIOC_QUERYMENU возвратит ошибку). Хорошим примером является список поддерживаемых скоростей передачи звука в MPEG. Некоторые драйверы поддерживают только одну или две скорости, другие поддерживают более широкий диапазон.

Все элементы управления используют машино независимый порядок байт.

### Перечисление расширенных элементов управления

Рекомендуемым способом перечисления дополнительных элементов управления является использование *ioctl* VIDIOC_QUERYCTRL, VIDIOC_QUERY_EXT_CTRL и VIDIOC_QUERYMENU в сочетании с флагом V4L2_CTRL_FLAG_NEXT_CTRL:

```
struct v4l2_queryctrl qctrl;

qctrl.id = V4L2_CTRL_FLAG_NEXT_CTRL;
while (0 == _ioctl_(fd, VIDIOC_QUERYCTRL, &qctrl)) {
        /* ... */
        qctrl.id |= V4L2_CTRL_FLAG_NEXT_CTRL;
}
```

Начальный идентификатор элемента управления устанавливается равным 0 ORed с флагом V4L2_CTRL
Вызов *ioctl* VIDIOC_QUERYCTRL вернет первый элемент управления с более высоким ID, чем заданный. Если такие элементы управления не найдены, возвращается ошибка.

Если требуется получить все элементы управления в определенном классе элементов управления, можно задать начальное значение qctrl.id для этого класса элемента управления и добавить дополнительную проверку, чтобы выйти из цикла при обнаружении элемента управления другого класса элемента управления:

```
qctrl.id = V4L2_CTRL_CLASS_MPEG | V4L2_CTRL_FLAG_NEXT_CTRL;
while (0 == _ioctl_(fd, VIDIOC_QUERYCTRL, &qctrl)) {
        if (V4L2_CTRL_ID2CLASS(qctrl.id) != V4L2_CTRL_CLASS_MPEG)
        break;
        /* ... */
        qctrl.id |= V4L2_CTRL_FLAG_NEXT_CTRL;
}
```

32-разрядное значение qctrl.id делится на три битовых диапазона: верхние 4 бита зарезервированы для флагов (e. g. V4L2_CTRL_FLAG_NEXT_CTRL) и фактически не является частью идентификатора. Оставшиеся 28 бит образуют идентификатор элемента управления, из которых старшие 12 бит определяют класс элемента управления а младшие 16 бит идентифицируют элемент управления в классе элемента управления. Гарантируется, что эти последние 16 бит всегда не равны нулю для элементов управления. Диапазон 0x1000 и выше зарезервирован для элементов управления, предназначенных для конкретного драйвера. Макрос V4L2_CTRL_ID2CLASS(id) возвращает идентификатор класса элемента управления на основе идентификатора элемента управления.

Если драйвер не поддерживает расширенные элементы управления, то VIDIOC_QUERYCTRL завершится ошибкой при использовании в комбинации с V4L2_CTRL_FLAG_NEXT_CTRL. В этом случае следует использовать старый метод перечисления элементов управления (см. Пример:

Перечисление всех элементов управления). Но если расширенный набор элементов поддерживается, то он гарантирует перечисление всех элементов управления, в том числе и приватных элементов управления драйвера.

### Создание панелей управления

Можно создавать панели управления для графического пользовательского интерфейса, в котором пользователь может выбрать различные элементы управления. В основном вам придется перебирать все элементы управления с помощью описанного выше метода. Каждый класс элемента управления начинается с элемента управления типа V4L2_CTRL_TYPE_CTRL_CLASS. Вызов *ioctl* VIDIOC_QUERYCTRL вернет имя этого класса элемента управления, который может использоваться в качестве заголовка вкладки в панели управления.

Поле флагов структуры v4l2_queryctrl также содержит подсказки о поведении элемента управления. Дополнительные сведения содержатся в документации по вызовам *ioctl* VIDIOC_QUERYCTRL, VIDIOC_QUERY_EXT_CTRL и VIDIOC_QUERYMENU.

### Справочник по элементам управления кодеков

Ниже описаны все элементы управления в классе Codec . Сначала перечислены общие элементы управления, а затем элементы управления, специфичные для определенного оборудования.

---

**Замечание**
Эти элементы управления применимы ко всем кодекам, а не только к MPEG. Эти определения имеют префикс V4L2_CID_MPEG/V4L2_MPEG, поскольку элементы управления изначально были сделаны для кодеков MPEG, а затем расширены для охвата всех форматов кодирования.

---

### Универсальные элементы управления кодеками

====== Идентификаторы элементов управления кодеками

**V4L2_CID_MPEG_CLASS (class)** Дескриптор класса кодека. Вызове *ioctl* комнад VIDIOC_QUERYCTR VIDIOC_QUERY_EXT_CTRL и VIDIOC_QUERYMENU для этого элемента управления будет возвращать описание элемента управления этого класса. Это описание может использоваться, например, как заголовок страницы вкладки в графическом интерфейсе пользователя.

V4L2_CID_MPEG_STREAM_TYPE (enum) enum v4l2_mpeg_stream_type - Тип выходного потока MPEG-1,-2 или-4. Здесь нельзя ничего предполагать. В каждом аппаратном кодировщике MPEG, как правило, поддерживаются различные подмножества доступных типов потоков MPEG. Этот элемент управления специфичен для мультиплексных потоков MPEG. В настоящее время определены типы потоков:

| | |
|---|---|
| V4L2_MPEG_STREAM_TYPE_MPEG2_PS | Поток программы MPEG-2 |
| V4L2_MPEG_STREAM_TYPE_MPEG2_TS | Транспортный поток MPEG-2 |
| V4L2_MPEG_STREAM_TYPE_MPEG1_SS | Системный поток MPEG-1 |
| V4L2_MPEG_STREAM_TYPE_MPEG2_DVD | Поток MPEG-2, совместимый с DVD |
| V4L2_MPEG_STREAM_TYPE_MPEG1_VCD | Поток MPEG-1 VCD-совместимый |
| V4L2_MPEG_STREAM_TYPE_MPEG2_SVCD | SVCD-совместимый поток MPEG-2 |

Таблица 1: Типы потоков

**V4L2_CID_MPEG_STREAM_PID_PMT (integer)** ID пакета таблицы программного мапирования для транспортного потока MPEG (по умолчанию 16)

**V4L2_CID_MPEG_STREAM_PID_AUDIO (integer)** ID аудио пакета для транспортного потока MPEG (по умолчанию 256)

**V4L2_CID_MPEG_STREAM_PID_VIDEO (integer)** ID видео-пакета для транспортного потока MPEG (по умолчанию 260)

**V4L2_CID_MPEG_STREAM_PID_PCR (integer)** ID пакета для транспортного потока MPEG, переуjczotuj поля PCR (по умолчанию 259)

**V4L2_CID_MPEG_STREAM_PES_ID_AUDIO (integer)** ID звука для MPEG PE

**V4L2_CID_MPEG_STREAM_PES_ID_VIDEO (integer)** ID видео для MPEG PE

**V4L2_CID_MPEG_STREAM_VBI_FMT (enum)** enum v4l2_mpeg_stream_vbi_fmt - Некоторые карты могут внедрить VBI данные (e. g. Closed Caption, Teletext) в MPEG поток. Этот элемент управления выбирает, следует ли внедрять VBI данные, и если да, то какой метод внедрения следует использовать. Список возможных форматов VBI зависит от драйвера. В настоящее время определены типы форматов VBI:

| | |
|---|---|
| V4L2_MPEG_STREAM_VBI_FMT_NONE | Нет VBI в потоке MPEG |
| V4L2_MPEG_STREAM_VBI_FMT_IVTV | VBI в приватных пакетах, формата IVTV (задокументировано в исходниках ядра в файле Documentation/video4linux/cx2341x/README.vbi) |

Таблица 2: Типы форматов VBI

**V4L2_CID_MPEG_AUDIO_SAMPLING_FREQ (enum)** enum v4l2_mpeg_audio_sampling_freq - Частота выборки звука MPEG. Возможные значения:

| | |
|---|---|
| V4L2_MPEG_AUDIO_SAMPLING_FREQ_44100 | 44.1 kHz |
| V4L2_MPEG_AUDIO_SAMPLING_FREQ_48000 | 48 kHz |
| V4L2_MPEG_AUDIO_SAMPLING_FREQ_32000 | 32 kHz |

Таблица 3: Частота выборки звука MPEG

**V4L2_CID_MPEG_AUDIO_ENCODING (enum)** enum v4l2_mpeg_audio_encoding - Кодировка звука MPEG. Этот элемент управления специфичен для мультиплексных потоков MPEG. Возможные значения:

| | |
|---|---|
| V4L2_MPEG_AUDIO_ENCODING_LAYER_1 | Кодировка MPEG-1/2 уровня I. |
| V4L2_MPEG_AUDIO_ENCODING_LAYER_2 | Кодировка MPEG-1/2 уровня II |
| V4L2_MPEG_AUDIO_ENCODING_LAYER_3 | Кодировка MPEG-1/2 уровня III |
| V4L2_MPEG_AUDIO_ENCODING_AAC | MPEG-2/4 AAC (усовершенствованное аудио кодирование) |
| V4L2_MPEG_AUDIO_ENCODING_AC3 | AC-3, известный как кодировка ATSC A/52 |

Таблица 4: Кодировка звука в MPEG

**V4L2_CID_MPEG_AUDIO_L1_BITRATE (enum)** enum v4l2_mpeg_audio_l1_bitrate - Скорость потока MPEG-1/2 на уровне I. Возможные значения:

| | |
|---|---|
| V4L2_MPEG_AUDIO_L1_BITRATE_32K | 32 kbit/s |
| V4L2_MPEG_AUDIO_L1_BITRATE_64K | 64 kbit/s |
| V4L2_MPEG_AUDIO_L1_BITRATE_96K | 96 kbit/s |
| V4L2_MPEG_AUDIO_L1_BITRATE_128K | 128 kbit/s |
| V4L2_MPEG_AUDIO_L1_BITRATE_160K | 160 kbit/s |
| V4L2_MPEG_AUDIO_L1_BITRATE_192K | 192 kbit/s |
| V4L2_MPEG_AUDIO_L1_BITRATE_224K | 224 kbit/s |
| V4L2_MPEG_AUDIO_L1_BITRATE_256K | 256 kbit/s |
| V4L2_MPEG_AUDIO_L1_BITRATE_288K | 288 kbit/s |
| V4L2_MPEG_AUDIO_L1_BITRATE_320K | 320 kbit/s |
| V4L2_MPEG_AUDIO_L1_BITRATE_352K | 352 kbit/s |
| V4L2_MPEG_AUDIO_L1_BITRATE_384K | 384 kbit/s |
| V4L2_MPEG_AUDIO_L1_BITRATE_416K | 416 kbit/s |
| V4L2_MPEG_AUDIO_L1_BITRATE_448K | 448 kbit/s |

Таблица 5: Скорость потока MPEG-1/2 первого уровня

**V4L2_CID_MPEG_AUDIO_L2_BITRATE (enum)** enum v4l2_mpeg_audio_l2_bitrate - Скорость потока в MPEG-1/2 уровня II. Возможные значения:

| | |
|---|---|
| V4L2_MPEG_AUDIO_L2_BITRATE_32K | 32 kbit/s |
| V4L2_MPEG_AUDIO_L2_BITRATE_48K | 48 kbit/s |
| V4L2_MPEG_AUDIO_L2_BITRATE_56K | 56 kbit/s |
| V4L2_MPEG_AUDIO_L2_BITRATE_64K | 64 kbit/s |
| V4L2_MPEG_AUDIO_L2_BITRATE_80K | 80 kbit/s |
| V4L2_MPEG_AUDIO_L2_BITRATE_96K | 96 kbit/s |
| V4L2_MPEG_AUDIO_L2_BITRATE_112K | 112 kbit/s |
| V4L2_MPEG_AUDIO_L2_BITRATE_128K | 128 kbit/s |
| V4L2_MPEG_AUDIO_L2_BITRATE_160K | 160 kbit/s |
| V4L2_MPEG_AUDIO_L2_BITRATE_192K | 192 kbit/s |
| V4L2_MPEG_AUDIO_L2_BITRATE_224K | 224 kbit/s |
| V4L2_MPEG_AUDIO_L2_BITRATE_256K | 256 kbit/s |
| V4L2_MPEG_AUDIO_L2_BITRATE_320K | 320 kbit/s |
| V4L2_MPEG_AUDIO_L2_BITRATE_384K | 384 kbit/s |

Таблица 6: Скорость потка MPEG-1/2 второго уровня

**V4L2_CID_MPEG_AUDIO_L3_BITRATE (enum)** enum v4l2_mpeg_audio_l3_bitrate - Скорость потока MPEG-1/2 уровня III. Возможные значения:

**V4L2_CID_MPEG_AUDIO_AAC_BITRATE (integer)** Скорость AAC в битах в секунду.

**V4L2_CID_MPEG_AUDIO_AC3_BITRATE (enum)** enum v4l2_mpeg_audio_ac3_bitrate - Битрейт AC-3. Возможные значения:

**V4L2_CID_MPEG_AUDIO_MODE (enum)** enum v4l2_mpeg_audio_mode -Режим звука MPEG. Возможные значения:

**V4L2_CID_MPEG_AUDIO_MODE_EXTENSION (enum)** enum v4l2_mpeg_audio_mode_extension - Расширение режима joint-стерео аудио. На уровне I и II они указывают, какие подгруппы находятся в интенсивном стерео. Все остальные подгруппы кодируются по стереосистеме. Уровень III (пока) не поддерживается. Возможные значения:

| | |
|---|---|
| V4L2_MPEG_AUDIO_L3_BITRATE_32K | 32 kbit/s |
| V4L2_MPEG_AUDIO_L3_BITRATE_40K | 40 kbit/s |
| V4L2_MPEG_AUDIO_L3_BITRATE_48K | 48 kbit/s |
| V4L2_MPEG_AUDIO_L3_BITRATE_56K | 56 kbit/s |
| V4L2_MPEG_AUDIO_L3_BITRATE_64K | 64 kbit/s |
| V4L2_MPEG_AUDIO_L3_BITRATE_80K | 80 kbit/s |
| V4L2_MPEG_AUDIO_L3_BITRATE_96K | 96 kbit/s |
| V4L2_MPEG_AUDIO_L3_BITRATE_112K | 112 kbit/s |
| V4L2_MPEG_AUDIO_L3_BITRATE_128K | 128 kbit/s |
| V4L2_MPEG_AUDIO_L3_BITRATE_160K | 160 kbit/s |
| V4L2_MPEG_AUDIO_L3_BITRATE_192K | 192 kbit/s |
| V4L2_MPEG_AUDIO_L3_BITRATE_224K | 224 kbit/s |
| V4L2_MPEG_AUDIO_L3_BITRATE_256K | 256 kbit/s |
| V4L2_MPEG_AUDIO_L3_BITRATE_320K | 320 kbit/s |

Таблица 7: Скорость потока MPEG-1/2 третьего уровня

| | |
|---|---|
| V4L2_MPEG_AUDIO_AC3_BITRATE_32K | 32 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_40K | 40 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_48K | 48 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_56K | 56 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_64K | 64 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_80K | 80 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_96K | 96 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_112K | 112 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_128K | 128 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_160K | 160 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_192K | 192 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_224K | 224 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_256K | 256 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_320K | 320 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_384K | 384 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_448K | 448 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_512K | 512 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_576K | 576 kbit/s |
| V4L2_MPEG_AUDIO_AC3_BITRATE_640K | 640 kbit/s |

Таблица 8: Битрейт AC-3

| | |
|---|---|
| V4L2_MPEG_AUDIO_MODE_STEREO | Стерео |
| V4L2_MPEG_AUDIO_MODE_JOINT_STEREO | Joint стерео |
| V4L2_MPEG_AUDIO_MODE_DUAL | Двуязычный |
| V4L2_MPEG_AUDIO_MODE_MONO | Моно |

Таблица 9: Режим звука MPEG

| | |
|---|---|
| V4L2_MPEG_AUDIO_MODE_EXTENSION_BOUND_4 | Подгруппы 4-31 в интенсивном стерео |
| V4L2_MPEG_AUDIO_MODE_EXTENSION_BOUND_8 | Подгруппы 8-31 в интенсивном стерео |
| V4L2_MPEG_AUDIO_MODE_EXTENSION_BOUND_12 | Подгруппы 12-31 в интенсивном стерео |
| V4L2_MPEG_AUDIO_MODE_EXTENSION_BOUND_16 | Подгруппы 16-31 в интенсивном стерео |

Таблица 10: Расширения режима joint-стерео аудио

**V4L2_CID_MPEG_AUDIO_EMPHASIS (enum)** enum v4l2_mpeg_audio_emphasis - Акцент на аудио. Возможные значения:

| | |
|---|---|
| V4L2_MPEG_AUDIO_EMPHASIS_NONE | Нет |
| V4L2_MPEG_AUDIO_EMPHASIS_50_DIV_15_uS | Акцент 50/15 микросекунд |
| V4L2_MPEG_AUDIO_EMPHASIS_CCITT_J17 | CCITT J.17 |

Таблица 11: Акцент на аудио

**V4L2_CID_MPEG_AUDIO_CRC (enum)** enum v4l2_mpeg_audio_crc - Метод CRC. Возможные значения:

| | |
|---|---|
| V4L2_MPEG_AUDIO_CRC_NONE | Нет |
| V4L2_MPEG_AUDIO_CRC_CRC16 | 16-битная проверка четности |

Таблица 12: Метод CRC

**V4L2_CID_MPEG_AUDIO_MUTE (boolean)** Отключение звука при захвате. Это не делается путём выключения аудио-оборудования, которое по-прежнему может привести к незначительному шипению, но в самом кодировщике, гарантируя фиксированный и воспроизводимый звуковой поток. 0 = Звук, 1 = Выкл.

**V4L2_CID_MPEG_AUDIO_DEC_PLAYBACK (enum)** enum v4l2_mpeg_audio_dec_playback - Определяет способ воспроизведения одноязычного звука. Возможные значения:

| | |
|---|---|
| V4L2_MPEG_AUDIO_DEC_PLAYBACK_AUTO | Автоматическое определение наилучшего режима воспроизведения. |
| V4L2_MPEG_AUDIO_DEC_PLAYBACK_STEREO | Воспроизведение стерео. |
| V4L2_MPEG_AUDIO_DEC_PLAYBACK_LEFT | Воспроизведение левого канала. |
| V4L2_MPEG_AUDIO_DEC_PLAYBACK_RIGHT | Воспроизведение правого канала. |
| V4L2_MPEG_AUDIO_DEC_PLAYBACK_MONO | Моно воспроизведение. |
| V4L2_MPEG_AUDIO_DEC_PLAYBACK_SWAPPED_STEREO | Воспроизведение стерео с перестановкой левого и правого каналов. |

**V4L2_CID_MPEG_AUDIO_DEC_MULTILINGUAL_PLAYBACK (enum)** enum v4l2_mpeg_audio_dec_pl - Определяет, как следует воспроизвести многоязычный звук.

**V4L2_CID_MPEG_VIDEO_ENCODING (enum)** enum v4l2_mpeg_video_encoding - Метод кодирования MPEG-видео. Этот элемент управления специфичен для мультиплексных потоков MPEG. Возможные значения:

| | |
|---|---|
| V4L2_MPEG_VIDEO_ENCODING_MPEG_1 | Кодирование видео в формате MPEG-1 |
| V4L2_MPEG_VIDEO_ENCODING_MPEG_2 | Кодирование видео в формате MPEG-2 |
| V4L2_MPEG_VIDEO_ENCODING_MPEG_4_AVC | Кодирование видео MPEG-4 AVC (H.264) |

**V4L2_CID_MPEG_VIDEO_ASPECT (enum)** enum v4l2_mpeg_video_aspect - Видео соотношение сторон. Возможные значения:

| |
|---|
| V4L2_MPEG_VIDEO_ASPECT_1x1 |

| |
|---|
| V4L2_MPEG_VIDEO_ASPECT_4x3 |
| V4L2_MPEG_VIDEO_ASPECT_16x9 |
| V4L2_MPEG_VIDEO_ASPECT_221x100 |

**V4L2_CID_MPEG_VIDEO_B_FRAMES (integer)** Число B-кадров (по умолчанию 2)

**V4L2_CID_MPEG_VIDEO_GOP_SIZE (integer)** GOP - количество кадров в группе (по умолчанию 12)

**V4L2_CID_MPEG_VIDEO_GOP_CLOSURE (boolean)** Замыкатели в группе (по умолчанию 1)

**V4L2_CID_MPEG_VIDEO_PULLDOWN (boolean)** Включить 3:2 (по умолчанию 0)

**V4L2_CID_MPEG_VIDEO_BITRATE_MODE (enum)** enum v4l2_mpeg_video_bitrate_mode - Режим скорости видео. Возможные значения:

| | |
|---|---|
| V4L2_MPEG_VIDEO_BITRATE_MODE_VBR | Переменная скорость |
| V4L2_MPEG_VIDEO_BITRATE_MODE_CBR | Постоянная скорость |

**V4L2_CID_MPEG_VIDEO_BITRATE (integer)** Скорость видео в битах в секунду.

**V4L2_CID_MPEG_VIDEO_BITRATE_PEAK (integer)** Пиковая скорость видео в битах в секунду. Должна быть больше или равна средней скорости видеозаписи. Игнорируется, если для режима скорости видео задана постоянная скорость.

**V4L2_CID_MPEG_VIDEO_TEMPORAL_DECIMATION (integer)** Для каждого захваченного кадра пропустить указанное количество последующих фреймов (по умолчанию 0).

**V4L2_CID_MPEG_VIDEO_MUTE (boolean)** «Отключить» видео до захвата фиксированного цвета. Это полезно для тестирования, для создания фиксированного видео битстреам. 0 = без отключения, 1 = отключить.

**V4L2_CID_MPEG_VIDEO_MUTE_YUV (integer)** Устанавливает цвет для отключения видео. Предоставленное 32-разрядное целое число интерпретируется следующим образом (бит 0 = наименьший значащий бит):

| | |
|---|---|
| Bit 0:7 | V цветовая информация |
| Bit 8:15 | U цветовая информация |
| Bit 16:23 | Y информация о яркости |
| Bit 24:31 | Должно быть ноль. |

Таблица 13: Структура описания цвета для отключения видео

**V4L2_CID_MPEG_VIDEO_DEC_PTS (integer64)** Этот элемент управления, доступный только для чтения, возвращает 33-битную метку времени показа видео, как определено в ITU T-REC-H.222.0 и ISO/IEC 13818-1 в отображаемом в данный момент фрейме. Этот же PTS используется в *ioctl* VIDIOC_DECODER_CMD, VIDIOC_TRY_DECODER_CMD.

**V4L2_CID_MPEG_VIDEO_DEC_FRAME (integer64)** Этот элемент управления, доступный только для чтения, возвращает счетчик фрейма кадра, отображаемого в данный момент (декодированный). При запуске декодера это значение сбрасывается до 0.

**V4L2_CID_MPEG_VIDEO_DECODER_SLICE_INTERFACE (boolean)** Если этот параметр включен, декодер ожидает получения одного фрагмента для каждого буфера, в противном случае декодер ожидает один кадр в буфере. Применимо к декодеру, все кодеки.

**V4L2_CID_MPEG_VIDEO_H264_VUI_SAR_ENABLE (boolean)** Разрешает запись соотношения сторон фрейма в информацию VUI (Информация для удобства использования видео). Применимо к кодировщику H264.

**V4L2_CID_MPEG_VIDEO_H264_VUI_SAR_IDC (enum)** enum v4l2_mpeg_video_h264_vui_sar_idc - VUI индикатор соотношения сторон для кодирования H.264. Это значение определяется в таблице E-1 в стандарте. Применимо к кодировщику H264.

| | |
|---|---|
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_UNSPECIFIED | Не задано |
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_1x1 | 1x1 |
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_12x11 | 12x11 |
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_10x11 | 10x11 |
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_16x11 | 16x11 |
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_40x33 | 40x33 |
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_24x11 | 24x11 |
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_20x11 | 20x11 |
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_32x11 | 32x11 |
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_80x33 | 80x33 |
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_18x11 | 18x11 |
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_15x11 | 15x11 |
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_64x33 | 64x33 |
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_160x99 | 160x99 |
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_4x3 | 4x3 |
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_3x2 | 3x2 |
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_2x1 | 2x1 |
| V4L2_MPEG_VIDEO_H264_VUI_SAR_IDC_EXTENDED | Расширенный SAR |

Таблица 14: Индикатор соотношения сторон

**V4L2_CID_MPEG_VIDEO_H264_VUI_EXT_SAR_WIDTH (integer)** Расширенная SAR (пропорции) для кодирования VUI в h. 264. Применимо к кодировщику H264.

**V4L2_CID_MPEG_VIDEO_H264_VUI_EXT_SAR_HEIGHT (integer)** Высота в расширенной пропорции (SAR) для кодирования VUI H.264. Применимо к кодировщику H264.

**V4L2_CID_MPEG_VIDEO_H264_LEVEL (enum)** enum v4l2_mpeg_video_h264_level - Сведения о уровне для элементарного потока видео H.264. Применимо к кодировщику H264. Возможные значения:

**V4L2_CID_MPEG_VIDEO_MPEG4_LEVEL (enum)** enum v4l2_mpeg_video_mpeg4_level - Сведения о уровне для элементарного потока видео MPEG4. Применимо к кодировщику MPEG4. Возможные значения:

**V4L2_CID_MPEG_VIDEO_H264_PROFILE (enum)** enum v4l2_mpeg_video_h264_profile - Сведения о профиле для H.264. Применимо к кодировщику H264. Возможные значения:

**V4L2_CID_MPEG_VIDEO_MPEG4_PROFILE (enum)** enum v4l2_mpeg_video_mpeg4_profile - Сведения о профиле для MPEG4. Применимо к кодировщику MPEG4. Возможные значения:

**V4L2_CID_MPEG_VIDEO_MAX_REF_PIC (integer)** Максимальное число опорных изображений, используемых для кодирования. Применимо к кодировщику.

**V4L2_CID_MPEG_VIDEO_MULTI_SLICE_MODE (enum)** enum v4l2_mpeg_video_multi_slice_mode - Определяет, как кодировщик должен обрабатывать разделение кадра на фрагменты. Применимо к кодировщику. Возможные значения:

| | |
|---|---|
| V4L2_MPEG_VIDEO_H264_LEVEL_1_0 | Уровень 1.0 |
| V4L2_MPEG_VIDEO_H264_LEVEL_1B | Уровень 1B |
| V4L2_MPEG_VIDEO_H264_LEVEL_1_1 | Уровень 1.1 |
| V4L2_MPEG_VIDEO_H264_LEVEL_1_2 | Уровень 1.2 |
| V4L2_MPEG_VIDEO_H264_LEVEL_1_3 | Уровень 1.3 |
| V4L2_MPEG_VIDEO_H264_LEVEL_2_0 | Уровень 2.0 |
| V4L2_MPEG_VIDEO_H264_LEVEL_2_1 | Уровень 2.1 |
| V4L2_MPEG_VIDEO_H264_LEVEL_2_2 | Уровень 2.2 |
| V4L2_MPEG_VIDEO_H264_LEVEL_3_0 | Уровень 3.0 |
| V4L2_MPEG_VIDEO_H264_LEVEL_3_1 | Уровень 3.1 |
| V4L2_MPEG_VIDEO_H264_LEVEL_3_2 | Уровень 3.2 |
| V4L2_MPEG_VIDEO_H264_LEVEL_4_0 | Уровень 4.0 |
| V4L2_MPEG_VIDEO_H264_LEVEL_4_1 | Уровень 4.1 |
| V4L2_MPEG_VIDEO_H264_LEVEL_4_2 | Уровень 4.2 |
| V4L2_MPEG_VIDEO_H264_LEVEL_5_0 | Уровень 5.0 |
| V4L2_MPEG_VIDEO_H264_LEVEL_5_1 | Уровень 5.1 |

Таблица 15: Уровень элементарного потока H.264

| | |
|---|---|
| V4L2_MPEG_VIDEO_LEVEL_0 | Уровень 0 |
| V4L2_MPEG_VIDEO_LEVEL_0B | Уровень 0b |
| V4L2_MPEG_VIDEO_LEVEL_1 | Уровень 1 |
| | V4L2_MPEG_VIDEO_LEVEL_2 |
| Уровень 2 | V4L2_MPEG_VIDEO_LEVEL_3 |
| Уровень 3 | V4L2_MPEG_VIDEO_LEVEL_3B |
| Уровень 3b | V4L2_MPEG_VIDEO_LEVEL_4 |
| Уровень 4 | V4L2_MPEG_VIDEO_LEVEL_5 |

Таблица 16: Уровень элементарного потока MPEG4

| | |
|---|---|
| V4L2_MPEG_VIDEO_H264_PROFILE_BASELINE | Базовый профиль |
| V4L2_MPEG_VIDEO_H264_PROFILE_CONSTRAINED_BASELINE | Ограниченный базовый профиль |
| V4L2_MPEG_VIDEO_H264_PROFILE_MAIN | Основной профиль |
| V4L2_MPEG_VIDEO_H264_PROFILE_EXTENDED | Расширенный профиль |
| V4L2_MPEG_VIDEO_H264_PROFILE_HIGH | Высокий профиль |
| V4L2_MPEG_VIDEO_H264_PROFILE_HIGH_10 | Высокий профиль 10 |
| V4L2_MPEG_VIDEO_H264_PROFILE_HIGH_422 | Высокий профиль 422 |
| V4L2_MPEG_VIDEO_H264_PROFILE_HIGH_444_PREDICTIVE | Высокий прогнозируемый профиль 444 |
| V4L2_MPEG_VIDEO_H264_PROFILE_HIGH_10_INTRA | Высокий внутренний профиль 10 |
| V4L2_MPEG_VIDEO_H264_PROFILE_HIGH_422_INTRA | Высокий внутренний профиль 422 |
| V4L2_MPEG_VIDEO_H264_PROFILE_HIGH_444_INTRA | Высокий внутренний профиль 444 |
| V4L2_MPEG_VIDEO_H264_PROFILE_CAVLC_444_INTRA | Внутренний профиль CAVL 444 |
| V4L2_MPEG_VIDEO_H264_PROFILE_SCALABLE_BASELINE | Масштабируемый базовый профиль |
| V4L2_MPEG_VIDEO_H264_PROFILE_SCALABLE_HIGH | Масштабируемый высокий профиль |
| V4L2_MPEG_VIDEO_H264_PROFILE_SCALABLE_HIGH_INTRA | Масштабируемый высокий внутренний профиль |
| V4L2_MPEG_VIDEO_H264_PROFILE_STEREO_HIGH | Высокий профиль стерео |
| V4L2_MPEG_VIDEO_H264_PROFILE_MULTIVIEW_HIGH | Многовидовой высокий профиль |

Таблица 17: Сведения о профиле H.264

| | |
|---|---|
| V4L2_MPEG_VIDEO_PROFILE_SIMPLE | Простой профиль |
| V4L2_MPEG_VIDEO_PROFILE_ADVANCED_SIMPLE | Расширенный простой профиль |
| V4L2_MPEG_VIDEO_PROFILE_CORE | Основной профиль |
| V4L2_MPEG_VIDEO_PROFILE_SIMPLE_SCALABLE | Простой масштабируемый профиль |
| V4L2_MPEG_VIDEO_PROFILE_ADVANCED_CODING_EFFICIENCY | |

Таблица 18: Сведения о профиле MPEG4

| | |
|---|---|
| V4L2_MPEG_VIDEO_MULTI_SLICE_MODE_SINGLE | Один фрагмент на кадр. |
| V4L2_MPEG_VIDEO_MULTI_SLICE_MODE_MAX_MB | Несколько фрагментов с установленным максимальным числом макроблоков на фрагмент. |
| V4L2_MPEG_VIDEO_MULTI_SLICE_MODE_MAX_BYTES | Несколько фрагментов с максимальным размером в байтах на фрагмент. |

Таблица 19: Разделение кадра на фрагметы

**V4L2_CID_MPEG_VIDEO_MULTI_SLICE_MAX_MB (integer)** Максимальное число макроблоков в фрагменте. Когда используется с V4L2_CID_MPEG_VIDEO_MULTI_SLICE_MODE iдолжно быть установлено в V4L2_MPEG_VIDEO_MULTI_SLICE_MODE_MAX_MB. Применимо к кодировщику.

**V4L2_CID_MPEG_VIDEO_MULTI_SLICE_MAX_BYTES (integer)** Максимальный размер фрагмента в байтах. Когда используется V4L2_CID_MPEG_VIDEO_MULTI_SLICE_MODE должно быть установлено в V4L2_MPEG_VIDEO_MULTI_SLICE_MODE_MAX_BYTES. Применимо к кодировщику.

**V4L2_CID_MPEG_VIDEO_H264_LOOP_FILTER_MODE (enum)** enum v4l2_mpeg_video_h264_loop_fi
- Режим циклической фильтрации для кодировщика H.264. Возможные значения:

| | |
|---|---|
| V4L2_MPEG_VIDEO_H264_LOOP_FILTER_MODE_ENABLED | Циклический фильтр включен. |
| V4L2_MPEG_VIDEO_H264_LOOP_FILTER_MODE_DISABLED | Циклический фильтр выключен. |
| V4L2_MPEG_VIDEO_H264_LOOP_FILTER_MODE_DISABLED_AT_SLICE_BOUNDARY | Циклический фильтр включен по контуру фрагмента. |

Таблица 20: Режим циклической фильтрации для кодировщика H.264

**V4L2_CID_MPEG_VIDEO_H264_LOOP_FILTER_ALPHA (integer)** Циклический фильтр по альфа-коэффициенту, определенный в стандарте H.264. Применимо к кодировщику H264.

**V4L2_CID_MPEG_VIDEO_H264_LOOP_FILTER_BETA (integer)** Циклический фильтр по бета-коэффициенту, определенный в стандарте H.264. Применимо к кодировщику H264.

**V4L2_CID_MPEG_VIDEO_H264_ENTROPY_MODE (enum)** enum v4l2_mpeg_video_h264_entropy_mo
- Режим энтропийного кодирования для CABAC/CAVALK. Применимо к кодировщику H264. Возможные значения:

**V4L2_CID_MPEG_VIDEO_H264_8X8_TRANSFORM (boolean)** Включить преобразование 8x8 для H.264. Применимо к кодировщику H264.

**V4L2_CID_MPEG_VIDEO_CYCLIC_INTRA_REFRESH_MB (integer)** Циклическое обновление внутренних макроблоков. Число последовательных макроблоков, обновляемых каждый кадр. Каждый кадр предыдущий набор макроблоков обновляется до тех пор, пока цикл не завершится и не начнется с верхней части кадра. Применимо к H264 и MPEG4 кодировщику.

| | |
|---|---|
| V4L2_MPEG_VIDEO_H264_ENTROPY_MODE_CAVLC | Используется CAVALC энтропийное кодирование. |
| V4L2_MPEG_VIDEO_H264_ENTROPY_MODE_CABAC | Использовать CABAC энтропийное кодирование. |

Таблица 21: Режим энтропийного кодирования

**V4L2_CID_MPEG_VIDEO_FRAME_RC_ENABLE (boolean)** Включить управление коэффициентом скорости кадров. Если этот элемент управления отключен, параметр дискретности для каждого типа кадра является константой и устанавливается по соответствующими элементами управления (например, V4L2_CID_MPEG_VIDEO_H263_I_FRAME_QP). Если включено управление частотой кадров, то параметр дискретности корректируется в соответствии с выбранной скоростью. Минимальное и максимальное значение параметра дискретности можно задать с помощью соответствующих элементов управления (например, V4L2_CID_MPEG_VI Применимо к кодировщикам.

**V4L2_CID_MPEG_VIDEO_MB_RC_ENABLE (boolean)** Включить управление уровнем скорости макроблоков. Применимо к кодировщикам MPEG4 и H.264.

**V4L2_CID_MPEG_VIDEO_MPEG4_QPEL (boolean)** Оценка движения четверти пикселей для MPEG4. Применимо к кодировщику MPEG4.

**V4L2_CID_MPEG_VIDEO_H263_I_FRAME_QP (integer)** Параметр дискретности для I-фрейма для H263. Допустимый диапазон: от 1 до 31.

**V4L2_CID_MPEG_VIDEO_H263_MIN_QP (integer)** Минимум параметра дискретности для H263. Допустимый диапазон: от 1 до 31.

**V4L2_CID_MPEG_VIDEO_H263_MAX_QP (integer)** Максимум параметра дискретности для H263. Допустимый диапазон: от 1 до 31.

**V4L2_CID_MPEG_VIDEO_H263_P_FRAME_QP (integer)** Параметр дискретности для P-фрейма для H263. Допустимый диапазон: от 1 до 31.

**V4L2_CID_MPEG_VIDEO_H263_B_FRAME_QP (integer)** Параметр дискретности для B-фрейма для H263. Допустимый диапазон: от 1 до 31.

**V4L2_CID_MPEG_VIDEO_H264_I_FRAME_QP (integer)** Параметр дискретности для I-фрейма для H264. Допустимый диапазон: от 0 до 51.

**V4L2_CID_MPEG_VIDEO_H264_MIN_QP (integer)** Минимум параметра дискретности для H264. Допустимый диапазон: от 0 до 51.

**V4L2_CID_MPEG_VIDEO_H264_MAX_QP (integer)** Максимум параметра дискретности для H264. Допустимый диапазон: от 0 до 51.

**V4L2_CID_MPEG_VIDEO_H264_P_FRAME_QP (integer)** Параметр дискретности для P-фрейма для H264. Допустимый диапазон: от 0 до 51.

**V4L2_CID_MPEG_VIDEO_H264_B_FRAME_QP (integer)** Параметр дискретности для B-фрейма для H264. Допустимый диапазон: от 0 до 51.

**V4L2_CID_MPEG_VIDEO_MPEG4_I_FRAME_QP (integer)** Параметр дискретности для I-фрейма для H264. Допустимый диапазон: от 1 до 31.

**V4L2_CID_MPEG_VIDEO_MPEG4_MIN_QP (integer)** Минимум параметра дискретности для мпег4. Допустимый диапазон: от 1 до 31.

**V4L2_CID_MPEG_VIDEO_MPEG4_MAX_QP (integer)** Максимум параметра дискретности для MPEG4. Допустимый диапазон: от 1 до 31.

**V4L2_CID_MPEG_VIDEO_MPEG4_P_FRAME_QP (integer)** Параметр дискретности для P-фрейма для MPEG4. Допустимый диапазон: от 1 до 31.

**V4L2_CID_MPEG_VIDEO_MPEG4_B_FRAME_QP (integer)** Параметр дискретности для B-фрейма для MPEG4. Допустимый диапазон: от 1 до 31.

**V4L2_CID_MPEG_VIDEO_VBV_SIZE (integer)** Размер проверочного видео буфера (VBV) в килобайтах, используется в качестве ограничения для пропуска кадра. VBV определяется в стандарте для того, чтобы убедиться, что произведенный поток будет успешно декодирован. Стандарт описывает его как ”Часть гипотетического декодера, который концептуально соединён с выходом кодировщика. Его цель заключается в том, чтобы обеспечить ограничение изменения скорости обработки данных, которую может производить кодировщик или процесс редактирования ”. Применимо к MPEG1, MPEG2, MPEG4 кодировщикам.

**V4L2_CID_MPEG_VIDEO_VBV_DELAY (integer)** Задает начальную задержку в миллисекундах для управления буфером VBV.

**V4L2_CID_MPEG_VIDEO_MV_H_SEARCH_RANGE (integer)** Горизонтальный диапазон поиска определяет максимальную горизонтальную область поиска в пикселах для поиска совпадения с текущим макроблоком (MB) в ссылочном изображении. Этот управляющий макрос V4L2 используется для задания горизонтального диапазона поиска для модуля оценки движения в кодировщике видео.

**V4L2_CID_MPEG_VIDEO_MV_V_SEARCH_RANGE (integer)** Вертикальный диапазон поиска определяет максимальную область поиска по вертикали в пикселах для поиска совпадения для текущего макроблока (MB) в ссылочном изображении. Этот управляющий макрос V4L2 используется для задания вертикального диапазона поиска для модуля оценки движения в кодировщике видео.

**V4L2_CID_MPEG_VIDEO_FORCE_KEY_FRAME (button)** Принудительно переключить ключевой кадр на следующий буфер в очереди. Применимо к кодировщикам. Это общий, независимый от кодеков, элемент управления ключевыми фреймами.

**V4L2_CID_MPEG_VIDEO_H264_CPB_SIZE (integer)** Размер закодированного буфера изображения (CPB) в килобайтах используется в качестве ограничения для пропуска кадра. CPB определен в стандарте H264 для того, чтобы убедиться, что произведенный поток будет успешно декодирован. Применимо к кодировщику H264.

**V4L2_CID_MPEG_VIDEO_H264_I_PERIOD (integer)** Период между I-фреймами в открытом GOP для H264. В случае открытого GOP это период между двумя I-фреймами. Период между IDR (мгновенное обновление декодирования) фреймами будет взят из элемента управления GOP_SIZE. IDR фрейм, который поддерживает мгновенное обновление декодирования, — это I-фрейм, после которого нет ссылок на предыдущие кадры. Это означает, что поток может быть перезапущен с IDR фрейма без необходимости хранения или расшифровки предыдущих кадров. Применимо к кодировщику H264.

**V4L2_CID_MPEG_VIDEO_HEADER_MODE (enum)** enum v4l2_mpeg_video_header_mode - Определяет, возвращается ли заголовок в качестве первого буфера, или он возвращается вместе с первым кадром. Применимо к кодировщикам. Возможные значения:

| | |
|---|---|
| V4L2_MPEG_VIDEO_HEADER_MODE_SEPARATE | Заголовок потока возвращается отдельно в первом буфере. |
| V4L2_MPEG_VIDEO_HEADER_MODE_JOINED_WITH_1ST_FRAME | Заголовок потока возвращается вместе с первым закодированным фреймом. |

Таблица 22: Режим возврата заголовка

**V4L2_CID_MPEG_VIDEO_REPEAT_SEQ_HEADER (boolean)** Повторить заголовки видео последовательности. Повторение этих заголовков облегчает произвольный доступ к потоку видео. Применимо к кодировщику MPEG1, 2 и 4.

**V4L2_CID_MPEG_VIDEO_DECODER_MPEG4_DEBLOCK_FILTER (boolean)** Включена блокировка фильтра POST-обработки для декодера MPEG4. Применимо к кодировщику MPEG4.

**V4L2_CID_MPEG_VIDEO_MPEG4_VOP_TIME_RES (integer)** vop_time_increment_resolution value for MPEG4. Применимо к кодировщику MPEG4.

**V4L2_CID_MPEG_VIDEO_MPEG4_VOP_TIME_INC (integer)** Значение vop_time_increment для MPEG4. Применимо к кодировщику MPEG4.

**V4L2_CID_MPEG_VIDEO_H264_SEI_FRAME_PACKING (boolean)** Включить генерирование дополнительной информации об улучшении упаковки кадров в закодированном битстреам. В сообщении SEI, упаковки фрейма, содержится расположение плоскостей L и R для 3D просмотра. Применимо к кодировщику H264.

**V4L2_CID_MPEG_VIDEO_H264_SEI_FP_CURRENT_FRAME_0 (boolean)** Установить текущий фрейм как фрейм-0 в кадре упаковки SEI. Применимо к кодировщику H264.

**V4L2_CID_MPEG_VIDEO_H264_SEI_FP_ARRANGEMENT_TYPE (enum)** enum v4l2_mpeg_video_h2
- Тип упорядочения упаковки для H264 SEI. Применимо к кодировщику H264. Возможные значения:

| | |
|---|---|
| V4L2_MPEG_VIDEO_H264_SEI_FP_ARRANGEMENT_TYPE_CHECKERBOARD | Пиксели чередуются по столбцам. |
| V4L2_MPEG_VIDEO_H264_SEI_FP_ARRANGEMENT_TYPE_COLUMN | Пиксели чередуются по строкам. |
| V4L2_MPEG_VIDEO_H264_SEI_FP_ARRANGEMENT_TYPE_ROW | Слева, с боков. |
| V4L2_MPEG_VIDEO_H264_SEI_FP_ARRANGEMENT_TYPE_SIDE_BY_SIDE | Сверху, снизу. |
| V4L2_MPEG_VIDEO_H264_SEI_FP_ARRANGEMENT_TYPE_TOP_BOTTOM | Один за другим. |
| V4L2_MPEG_VIDEO_H264_SEI_FP_ARRANGEMENT_TYPE_TEMPORAL | Один кадр на кадр. |

Таблица 23: Тип упорядочения упаковки для H264 SEI

**V4L2_CID_MPEG_VIDEO_H264_FMO (boolean)** Включает гибкое упорядочивание макроблоков в закодированном потоке. Это техника, используемая для реструктуризации порядка макроблоков в изображении. Применимо к кодировщику H264.

**V4L2_CID_MPEG_VIDEO_H264_FMO_MAP_TYPE (enum)** enum v4l2_mpeg_video_h264_fmo_map_ty
- При использовании FMO, тип карты делит изображение на различные паттерны сканирования макроблоков. Применимо к кодировщику H264. Возможные значения:

**V4L2_CID_MPEG_VIDEO_H264_FMO_SLICE_GROUP (integer)** Количество групп фрагментов в FMO. Применимо к кодировщику H264.

**V4L2_CID_MPEG_VIDEO_H264_FMO_CHANGE_DIRECTION (enum)** enum v4l2_mpeg_video_h264_
- Задает направление изменения группы фрагментов для растровых и стирающих карт. Применимо к кодировщику H264. Возможные значения:

**V4L2_CID_MPEG_VIDEO_H264_FMO_CHANGE_RATE (integer)** Задает размер первой группы фрагментов для растровой и стирающей карты. Применимо к кодировщику H264.

**V4L2_CID_MPEG_VIDEO_H264_FMO_RUN_LENGTH (integer)** Задает число последовательных макроблоков для перемежающейся карты. Применимо к кодировщику H264.

**V4L2_CID_MPEG_VIDEO_H264_ASO (boolean)** Разрешает произвольное упорядочивание фрагментов в закодированном потоке. Применимо к кодировщику H264.

| | |
|---|---|
| V4L2_MPEG_VIDEO_H264_FMO_MAP_TYPE_INTERLEAVED_SLICES | Фрагменты перемежаются один за другим, с макроблоками в порядке длительности выполнения. |
| V4L2_MPEG_VIDEO_H264_FMO_MAP_TYPE_SCATTERED_SLICES | Макроблоки распределяются на основе математической функции, известной как кодировщику, так и декодеру. |
| V4L2_MPEG_VIDEO_H264_FMO_MAP_TYPE_FOREGROUND_WITH_LEFT_OVER | Макроблоки организуются в прямоугольных областях или регионах, представляющих интерес. |
| V4L2_MPEG_VIDEO_H264_FMO_MAP_TYPE_BOX_OUT | Группы фрагментов растут циклическими способом от центра к краям. |
| V4L2_MPEG_VIDEO_H264_FMO_MAP_TYPE_RASTER_SCAN | Группы фрагментов растут в образце растрового сканирования слева направо. |
| V4L2_MPEG_VIDEO_H264_FMO_MAP_TYPE_WIPE_SCAN | Группы фрагментов растут в образце стирающего сканирования сверху вниз. |
| V4L2_MPEG_VIDEO_H264_FMO_MAP_TYPE_EXPLICIT | Тип карты, определяемый пользователем. |

Таблица 24: Тип карты H.264

| | |
|---|---|
| V4L2_MPEG_VIDEO_H264_FMO_CHANGE_DIR_RIGHT | Растровое сканирование или стирание вправо. |
| V4L2_MPEG_VIDEO_H264_FMO_CHANGE_DIR_LEFT | Обратное растровое сканирование или стирание влево. |

Таблица 25: Направление изменения группы фрагментов

| | |
|---|---|
| Bit 0:15 | ID фрагмента |
| Bit 16:32 | Положение фрагмента или порядок |

Таблица 26: Порядок фрагментов в ASO

**V4L2_CID_MPEG_VIDEO_H264_ASO_SLICE_ORDER (integer)** Задает порядок фрагментов в ASO. Применимо к кодировщику H264. Предоставленное 32-разрядное целое число интерпретируется следующим образом (бит 0 = наименьший значащий бит):

**V4L2_CID_MPEG_VIDEO_H264_HIERARCHICAL_CODING (boolean)** Включает иерархическое кодирование H.264. Применимо к кодировщику H264.

**V4L2_CID_MPEG_VIDEO_H264_HIERARCHICAL_CODING_TYPE (enum)** enum v4l2_mpeg_video_h - Задает тип иерархического кодирования. Применимо к кодировщику H264. Возможные значения:

| | |
|---|---|
| V4L2_MPEG_VIDEO_H264_HIERARCHICAL_CODING_B | Иерархическая кодирование типа B. |
| V4L2_MPEG_VIDEO_H264_HIERARCHICAL_CODING_P | Иерархическое кодирование типа P. |

Таблица 27: Тип иерархического кодирования

**V4L2_CID_MPEG_VIDEO_H264_HIERARCHICAL_CODING_LAYER (integer)** Указывает количество слоев иерархического кодирования. Применимо к кодировщику H264.

**V4L2_CID_MPEG_VIDEO_H264_HIERARCHICAL_CODING_LAYER_QP (integer)** Задаёт определённое пользователем QP для каждого слоя. Применимо к кодировщику H264. Предоставленное 32-разрядное целое число интерпретируется следующим образом (бит 0 = наименьший значащий бит):

| | |
|---|---|
| Bit 0:15 | Значение QP |
| Bit 16:32 | Номер слоя |

Таблица 28: Описатель QP для слоя

### Элементы управления MPEG в MFC 5.1

Следующие MPEG классы элементов управления занимаются настройкой MPEG декодирования и кодирования, которые специфичны для Multi Format Codec 5.1 устройств, представленных в семействе S5P из SoCs от Samsung.

====== Идентификаторы элементов управления MFC 5.1

**V4L2_CID_MPEG_MFC51_VIDEO_DECODER_H264_DISPLAY_DELAY_ENABLE (boolean)** Если задержка отображения включена, декодер должен возвращать буфер ЗАХВАТА (декодированный кадр) после обработки определенного количества ВЫХОДНЫХ буферов. Задержка может быть установлена с помощью V4L2_CID_MPEG_MFC51_VIDEO_DECODER_H264_DISPLAY_D Эта функция может использоваться, например, для создания миниатюр видеороликов. Применимо к H.264 декодеру.

**V4L2_CID_MPEG_MFC51_VIDEO_DECODER_H264_DISPLAY_DELAY (integer)** Значение задержки отображения для декодера H.264. Декодер должен возвращать декодированный фрейм после заданного «display delay» количества фреймов. Если это число мало, то это может привести к тому, что кадры, возвращенные в порядке отображения, будут использоваться аппаратурой в качестве опорного изображения для последующих фреймов.

**V4L2_CID_MPEG_MFC51_VIDEO_H264_NUM_REF_PIC_FOR_P (integer)** Число ссылочных изображений, используемых для кодирования P фрейма. Применимо к кодировщику H264.

**V4L2_CID_MPEG_MFC51_VIDEO_PADDING (boolean)** Включить заполнение в кодировщике - использовать цвет вместо повторяющихся пикселов границы. Применимо к кодировщикам.

**V4L2_CID_MPEG_MFC51_VIDEO_PADDING_YUV (integer)** Цвет заполнения в кодировщике. Применимо к кодировщикам. Предоставленное 32-разрядное целое число интерпретируется следующим образом (бит 0 = наименьший значащий бит):

| | |
|---|---|
| Bit 0:7 | V цветовая информация |
| Bit 8:15 | U цветовая информация |
| Bit 16:23 | Y информация о яркости |
| Bit 24:31 | Должно быть ноль. |

Таблица 29: Цвет заполенеия

**V4L2_CID_MPEG_MFC51_VIDEO_RC_REACTION_COEFF (integer)** Коэффициент реакции для управления скоростью потока в MFC. Применимо к кодировщикам.

---

**Замечание**
Действительно только при включенном RC уровне кадра. Для напряжённого фиксированного битрейта это поле должно быть небольшим (напр. 2 10). Для переменного битрейта это поле должно быть большим (напр. 100 1000). Не рекомендуется использовать число, большее чем FRAME_RATE * ($10^9$ / BIT_RATE).

---

**V4L2_CID_MPEG_MFC51_VIDEO_H264_ADAPTIVE_RC_DARK (boolean)** Адаптивный контроль битрэйта для темной области. Допустимо только для H.264 при включенном параметре уровень макроблоков RC (V4L2_CID_MPEG_VIDEO_MB_RC_ENABLE). Применимо к кодировщику H264.

**V4L2_CID_MPEG_MFC51_VIDEO_H264_ADAPTIVE_RC_SMOOTH (boolean)** Адаптивный контроль битрэйта для гладкого региона. Допустимо только для H.264 при включенном параметре уровень макроблоков RC (V4L2_CID_MPEG_VIDEO_MB_RC_ENABLE). Применимо к кодировщику H264.

**V4L2_CID_MPEG_MFC51_VIDEO_H264_ADAPTIVE_RC_STATIC (boolean)** Адаптивный контроль битрэйта для статичного региона. Допустимо только для H.264 при включенном параметре уровень макроблоков RC (V4L2_CID_MPEG_VIDEO_MB_RC_ENABLE). Применимо к кодировщику H264.

**V4L2_CID_MPEG_MFC51_VIDEO_H264_ADAPTIVE_RC_ACTIVITY (boolean)** Адаптивный контроль битрэйта для активных областей. Допустимо только для H.264 при включенном параметре уровень макроблоков RC (V4L2_CID_MPEG_VIDEO_MB_RC_ENABLE). Применимо к кодировщику H264.

**V4L2_CID_MPEG_MFC51_VIDEO_FRAME_SKIP_MODE (enum)** enum v4l2_mpeg_mfc51_video_fram - Указывает, в каких условиях кодировщик должен пропустить кадры. Если кодирование фрейма приведет к тому, что закодированный поток будет больше, чем выбранное ограничение данных, фрейм будет пропущен. Возможные значения:

**V4L2_CID_MPEG_MFC51_VIDEO_RC_FIXED_TARGET_BIT (integer)** Включить управление скоростью с фиксированным целевым битом. Если этот параметр включен, то логика управления скоростью кодировщика вычисляет среднюю скорость для GOP и сохраняет её ниже или равной установленной целевой скорости. В противном случае логика управления скоростью вычисляет общую среднюю скорость потока и удерживает ее ниже или равной установленной скорости. В первом случае средняя скорость для всего потока будет меньше, чем установленная скорость. Это вызвано тем, что среднее значение вычисляется для меньшего числа кадров, с другой стороны, включение этого параметра гарантирует, что поток будет отвечать жестким ограничениям пропускной способности. Применимо к кодировщикам.

| | |
|---|---|
| V4L2_MPEG_MFC51_FRAME_SKIP_MODE_DISABLED | Режим пропуска кадра отключен. |
| V4L2_MPEG_MFC51_FRAME_SKIP_MODE_LEVEL_LIMIT | Режим пропуска кадра включен, а предел буфера устанавливается выбранным уровнем и определяется стандартом. |
| V4L2_MPEG_MFC51_FRAME_SKIP_MODE_BUF_LIMIT | Режим пропуска кадра включен и предельный размер буфера задается с помощью элемента управления размером буфера VBV (MPEG1/2/4) или CPB (H.264). |

Таблица 30: Условие пропуска кадров

**V4L2_CID_MPEG_MFC51_VIDEO_FORCE_FRAME_TYPE (enum)** enum v4l2_mpeg_mfc51_video_forc
- Принудительно переключить тип фрейма для следующего буфера в очереди. Применимо к кодировщикам. Возможные значения:

| | |
|---|---|
| V4L2_MPEG_MFC51_FORCE_FRAME_TYPE_DISABLED | Принудительное переключение типа фрейма запрещено. |
| V4L2_MPEG_MFC51_FORCE_FRAME_TYPE_I_FRAME | Переключить тип на I-фрейм.. |
| V4L2_MPEG_MFC51_FORCE_FRAME_TYPE_NOT_CODED | Переключить тип на незакодированный фрейм. |

Таблица 31: Принудительное переключение типа фрейма

### Элементы управления MPEG для CX2341x

Следующие элементы управления класса MPEG относятся к параметрам кодирования MPEG, специфичным для чипов Conexant CX23415 и CX23416.

====== ID элементов управления CX2341x

**V4L2_CID_MPEG_CX2341X_VIDEO_SPATIAL_FILTER_MODE (enum)** enum v4l2_mpeg_cx2341x_vid
- Задает режим пространственного фильтра (По умолчанию - ручной). Возможные значения:

| | |
|---|---|
| V4L2_MPEG_CX2341X_VIDEO_SPATIAL_FILTER_MODE_MANUAL | Выбор фильтра вручную |
| V4L2_MPEG_CX2341X_VIDEO_SPATIAL_FILTER_MODE_AUTO | Автоматический выбор фильтра |

Таблица 32: Режим пространственного фильтра

**V4L2_CID_MPEG_CX2341X_VIDEO_SPATIAL_FILTER (integer (0-15))** Параметр пространственного фильтра. 0 = Откл, 15 = максимум. (по умолчанию равен 0.)

**V4L2_CID_MPEG_CX2341X_VIDEO_LUMA_SPATIAL_FILTER_TYPE (enum)** enum v4l2_mpeg_cx234
- Выбор алгоритма, который будет использоваться для пространственного фильтра Luma (по умолчанию 1D_HOR). Возможные значения:

**V4L2_CID_MPEG_CX2341X_VIDEO_CHROMA_SPATIAL_FILTER_TYPE (enum)** enum v4l2_mpeg_cx
- Выбор алгоритма пространственного фильтра насыщенности цвета (по умолчанию 1D_HOR). Возможные значения:

**V4L2_CID_MPEG_CX2341X_VIDEO_TEMPORAL_FILTER_MODE (enum)** enum v4l2_mpeg_cx2341x_
- Устанавливает режим временного фильтра (Ручной по умолчанию). Возможные значения:

| | |
|---|---|
| V4L2_MPEG_CX2341X_VIDEO_LUMA_SPATIAL_FILTER_TYPE_OFF | Нет фильтра |
| V4L2_MPEG_CX2341X_VIDEO_LUMA_SPATIAL_FILTER_TYPE_1D_HOR | Одномерный горизонтальный |
| V4L2_MPEG_CX2341X_VIDEO_LUMA_SPATIAL_FILTER_TYPE_1D_VERT | Одномерный вертикальный |
| V4L2_MPEG_CX2341X_VIDEO_LUMA_SPATIAL_FILTER_TYPE_2D_HV_SEPARABLE | Двумерный разделяемый |
| V4L2_MPEG_CX2341X_VIDEO_LUMA_SPATIAL_FILTER_TYPE_2D_SYM_NON_SEPARABLE | Двумерный симметричный неразделяемый |

Таблица 33: Алгоритм пространственного фильтра Luma

| | |
|---|---|
| V4L2_MPEG_CX2341X_VIDEO_CHROMA_SPATIAL_FILTER_TYPE_OFF | Нет фильтра |
| V4L2_MPEG_CX2341X_VIDEO_CHROMA_SPATIAL_FILTER_TYPE_1D_HOR | Одномерный горизонтальный |

Таблица 34: Алгоритм фильтра насыщенности

**V4L2_CID_MPEG_CX2341X_VIDEO_TEMPORAL_FILTER (integer (0-31))** Настройка временного фильтра. 0 = Откл, 31 = максимум. (значение по умолчанию — 8 для полного захвата и 0 для масштабированного захвата.)

**V4L2_CID_MPEG_CX2341X_VIDEO_MEDIAN_FILTER_TYPE (enum)** enum v4l2_mpeg_cx2341x_vide - Тип медианного фильтра (по умолчанию Откл). Возможные значения:

**V4L2_CID_MPEG_CX2341X_VIDEO_LUMA_MEDIAN_FILTER_BOTTOM (integer (0-255))** Пороговое значение, при превышении которого, включается медианный фильтр по яркости (по умолчанию 0)

**V4L2_CID_MPEG_CX2341X_VIDEO_LUMA_MEDIAN_FILTER_TOP (integer (0-255))** Пороговое значение, при снижении ниже которого, включается медианный фильтр по яркости (по умолчанию 255)

**V4L2_CID_MPEG_CX2341X_VIDEO_CHROMA_MEDIAN_FILTER_BOTTOM (integer (0-255))** Пороговое значение, при превышении которого, включается медианный фильтр по насыщенности (по умолчанию 0)

**V4L2_CID_MPEG_CX2341X_VIDEO_CHROMA_MEDIAN_FILTER_TOP (integer (0-255))** Пороговое значение, при снижении ниже которого, включается медианный фильтр по насыщенности (по умолчанию 255)

**V4L2_CID_MPEG_CX2341X_STREAM_INSERT_NAV_PACKETS (boolean)** Кодировщик CX2341X MPEG может вставлять один пустой пакет PES MPEG-2 в поток между каждой четвёркой видеокадров. Размер пакета составляет 2048 байт, включая поля packet_start_code_prefix и stream_id. stream_id - 0xB (Приватный поток 2). Полезная нагрузка содержит байты 0x00, записанных приложением. 0 = не вставлять, 1 = вставлять пакеты.

### Справочник по элементам управления VPX

Элементы управления VPX включают элементы управления для кодирования параметров VPx видео кодека.

====== ID элементов управления VPX

| | |
|---|---|
| V4L2_MPEG_CX2341X_VIDEO_TEMPORAL_FILTER_MODE_MANUAL | Выбор фильтра вручную |
| V4L2_MPEG_CX2341X_VIDEO_TEMPORAL_FILTER_MODE_AUTO | Автоматический выбор фильтра |

Таблица 35: Режим фильтра по времени

| | |
|---|---|
| V4L2_MPEG_CX2341X_VIDEO_MEDIAN_FILTER_TYPE_OFF | Нет фильтра |
| V4L2_MPEG_CX2341X_VIDEO_MEDIAN_FILTER_TYPE_HOR | Горизонтальный фильтр |
| V4L2_MPEG_CX2341X_VIDEO_MEDIAN_FILTER_TYPE_VERT | Вертикальный фильтр |
| V4L2_MPEG_CX2341X_VIDEO_MEDIAN_FILTER_TYPE_HOR_VERT | Горизонтальный и вертикальный фильтр |
| V4L2_MPEG_CX2341X_VIDEO_MEDIAN_FILTER_TYPE_DIAG | Диагональный фильтр |

Таблица 36: Тип медианного фильтра

| | |
|---|---|
| V4L2_CID_MPEG_VIDEO_VPX_1_PARTITION | 1 Коэффициент раздела |
| V4L2_CID_MPEG_VIDEO_VPX_2_PARTITIONS | 2 коэффициента разделов |
| V4L2_CID_MPEG_VIDEO_VPX_4_PARTITIONS | 4 коэффициента разделов |
| V4L2_CID_MPEG_VIDEO_VPX_8_PARTITIONS | 8 коэффициента разделов |

Таблица 37: Число разделов

**V4L2_CID_MPEG_VIDEO_VPX_NUM_PARTITIONS (enum)** enum v4l2_vp8_num_partitions - Число маркеров зазделов, используемых в кодировщике VP8. Возможные значения:

**V4L2_CID_MPEG_VIDEO_VPX_IMD_DISABLE_4X4 (boolean)** Установка этого параметра предотвращает использование внутреннего режима 4x4, при выборе внутреннего режима.

**V4L2_CID_MPEG_VIDEO_VPX_NUM_REF_FRAMES (enum)** enum v4l2_vp8_num_ref_frames - Число ссылочных фреймов, используемых для кодирования P фрейма. Возможные значения:

| | |
|---|---|
| V4L2_CID_MPEG_VIDEO_VPX_1_REF_FRAME | Поиск в последнем закодированном фрейме |
| V4L2_CID_MPEG_VIDEO_VPX_2_REF_FRAME | Поиск будет веститись в двух фреймах, между последним закодированным фреймом, золотым фреймом и фреймом по альтернативной ссылке (altref). Реализация кодировщика решит, какие два фрейма будут выбраны. |
| V4L2_CID_MPEG_VIDEO_VPX_3_REF_FRAME | Поиск будет вестись в последнем закодированном фрейме, золотом фрейме и altref фрейме. |

Таблица 38: Число ссылочных фреймов

**V4L2_CID_MPEG_VIDEO_VPX_FILTER_LEVEL (integer)** Указывает уровень фильтра цикла. Корректировка уровня фильтра цикла выполняется с помощью дельта-значения по отношению к базовому значению фильтра цикла.

**V4L2_CID_MPEG_VIDEO_VPX_FILTER_SHARPNESS (integer)** Этот параметр влияет на фильтр цикла. Все, что выше нуля, ослабляет эффект деблокировки в фильтре цикла.

**V4L2_CID_MPEG_VIDEO_VPX_GOLDEN_FRAME_REF_PERIOD (integer)** Устанавливает период обновления для золотого фрейма. Период определяется количеством кадров. Для значения ’ n ’ каждый n-й кадр, начинающийся с первого ключевого кадра, будет взят в качестве золотого фрейма. Пример, для кодирования последовательности 0, 1, 2, 3, 4, 5, 6, 7, в которой период золотого кадра равен 4, кадры 0, 4, 8 и т. д. будут взяты в качестве золотых фреймов , а фрейм 0 является золотым фреймом, так как он всегда ключевой кадр.

**V4L2_CID_MPEG_VIDEO_VPX_GOLDEN_FRAME_SEL (enum)** enum v4l2_vp8_golden_frame_sel - Выбор золотого фрейма для кодирования. Возможные значения:

| | |
|---|---|
| V4L2_CID_MPEG_VIDEO_VPX_GOLDEN_FRAME_USE_PREV | Использовать (n-2)-й фрейм в качестве золотого фрейма, номер текущего фрейма - ”n”. |
| V4L2_CID_MPEG_VIDEO_VPX_GOLDEN_FRAME_USE_REF_PERIOD | Использовать предыдущий конкретный кадр, обозначенный V4L2_CID_MPEG_VIDEO_VPX_GOLDEN_FRAME_REF_ в качестве золотого фрейма. |

Таблица 39: Выбор золотого фрейма

**V4L2_CID_MPEG_VIDEO_VPX_MIN_QP (integer)** Минимальный параметр дискретности для VP8.

**V4L2_CID_MPEG_VIDEO_VPX_MAX_QP (integer)** Максимальный параметр дискретности для VP8.

**V4L2_CID_MPEG_VIDEO_VPX_I_FRAME_QP (integer)** Параметр дискретности для I-фрейма для VP8.

**V4L2_CID_MPEG_VIDEO_VPX_P_FRAME_QP (integer)** Параметр дискретности для P-фрейма для VP8.

**V4L2_CID_MPEG_VIDEO_VPX_PROFILE (integer)** Выберите требуемый профиль для кодировщика VPx. Допустимыми значениями являются 0, 1, 2 и 3, соответствующие профилю кодировщика 0, 1, 2 и 3.

### Справочник по управлению камерой

Класс камеры включает элементы управления для механических (или эквивалентных цифровых) функций устройства, таких как управляемые линзы или датчики.

### ID управления камерой

**V4L2_CID_CAMERA_CLASS (class)** Дескриптор класса камеры. Вызове *ioctl* комнад VIDIOC_QUERY VIDIOC_QUERY_EXT_CTRL и VIDIOC_QUERYMENU для этого элемента управления будет возвращать описание элемента управления этого класса.

**V4L2_CID_EXPOSURE_AUTO (enum)** enum v4l2_exposure_auto_type - Включает автоматическую коррекцию времени экспозиции и/или ирисовой диафрагмы. Влияние ручных изменений времени экспозиции или диафрагмы, при включении этих функций не определено, драйверы должны игнорировать такие запросы. Возможные значения:

| | |
|---|---|
| V4L2_EXPOSURE_AUTO | Автоматическое время экспозиции, автоматическая диафрагма. |
| V4L2_EXPOSURE_MANUAL | Ручное время экспозиции, ручная диафрагма. |
| V4L2_EXPOSURE_SHUTTER_PRIORITY | Ручное время экспозиции, автоматическая диафрагма. |
| V4L2_EXPOSURE_APERTURE_PRIORITY | Автоматическое время экспозиции, ручная регулировка диафрагмы. |

Таблица 40: Управление автоматикой экспозиции

**V4L2_CID_EXPOSURE_ABSOLUTE (integer)** Определяет время экспозиции датчика камеры. Время экспозиции ограничено скоростью смены кадров. Драйверы должны интерпретировать значения как единицы по 100 µS, где значение 1 соответствует 1/10000 секунды, 10000 - 1 секунде и 100000 соответствует 10 секундам.

**V4L2_CID_EXPOSURE_AUTO_PRIORITY (boolean)** Если контрол V4L2_CID_EXPOSURE_AUTO установлен в значение AUTO или APERTURE_PRIORITY,toj этот элемент управления определяет, может ли устройство динамически изменять частоту кадров. По умолчанию эта функция отключена (0), а частота кадров должна оставаться постоянной.

**V4L2_CID_EXPOSURE_BIAS (integer menu)** Определяет автоматическую компенсацию экспозиции, которая действует только в том случае, если контрол V4L2_CID_EXPOSURE_AUTO установлен в значение AUTO, SHUTTER_PRIORITY или APERTURE_PRIORITY. Его значение задаётся в единицах EV, драйверы должны интерпретировать значения как 0,001 EV, где значение 1000 соответствует + 1 EV. Увеличение значения коррекции экспозиции эквивалентно уменьшению экспозиционного числа (EV) и будет увеличивать количество света на матрице. Фотокамера выполняет компенсацию экспозиции, корректируя абсолютное время экспозиции и/или диафрагму.

**V4L2_CID_EXPOSURE_METERING (enum)** enum v4l2_exposure_metering - Определяет, как фотокамера измеряет количество света, доступного для экспозиции кадра. Возможные значения:

| | |
|---|---|
| V4L2_EXPOSURE_METERING_AVERAGE | Использовать информацию о свете, поступающий от всего фрейма в среднем, не задавая никакого веса какой-либо определенной части области измерения. |
| V4L2_EXPOSURE_METERING_CENTER_WEIGHTED | Усреднять информацию о источнике света, поступающую из всего фрейма, отдает приоритет центральной области замера. |
| V4L2_EXPOSURE_METERING_SPOT | Измерить только очень маленькую область в центре кадра. |
| V4L2_EXPOSURE_METERING_MATRIX | Многозонный замер. Интенсивность света измеряется в нескольких точках кадра, а результаты комбинируются. Алгоритм выбора зон и их вкладе при вычислении окончательного значения зависит от устройства. |

Таблица 41: Режим замера освещённости

**V4L2_CID_PAN_RELATIVE (integer)** Этот элемент управления вращает камеру по горизонтали на заданную величину. Единица измерения не определена. Положительное значение перемещает камеру вправо (по часовой стрелке при просмотре сверху), отрицательное значение влево. Нулевое значение не вызывает движения. Это элемент управления, доступный только для записи.

**V4L2_CID_TILT_RELATIVE (integer)** Этот элемент управления поворачивает камеру по вертикали на заданную величину. Единица измерения не определена. Положительное значение перемещает камеру вверх, отрицательное значение вниз. Нулевое значение не вызывает движения. Это элемент управления, доступный только для записи.

**V4L2_CID_PAN_RESET (button)** Когда этот элемент управления установлен, камера переходит по горизонтали на позицию по умолчанию.

**V4L2_CID_TILT_RESET (button)** Когда этот элемент управления установлен, камера перемещается по вертикали в положение по умолчанию.

**V4L2_CID_PAN_ABSOLUTE (integer)** Этот элемент управления перемещает камеру горизонтально в указанное положение. Положительные значения перемещают камеру вправо (по часовой стрелке при просмотре сверху), отрицательные значения - влево. Драйверы должны интерпретировать эти значения как угловые секунды, с допустимыми значениями между-180 * 3600 и + 180 * 3600 включительно.

**V4L2_CID_TILT_ABSOLUTE (integer)** Этот элемент управления вращает камеру по вертикали в указанное положение. Положительные значения перемещайте камеру вверх, отрицательные значения вниз. Драйверы должны интерпретировать эти значения как угловые секунды, с допустимыми значениями между-180 * 3600 и + 180 * 3600 включительно.

**V4L2_CID_FOCUS_ABSOLUTE (integer)** Этот элемент управления устанавливает точку фокуса камеры в заданную позицию. Единица измерения не определена. Положительные значения задается приближение к камере, а отрицательные значения — к бесконечности.

**V4L2_CID_FOCUS_RELATIVE (integer)** Этот элемент управления перемещает точку фокуса камеры на заданную величину. Единица измерения не определена. Положительные значения перемещают фокус ближе к фотокамере, отрицательные значения к бесконечности. Это элемент управления, доступный только для записи.

**V4L2_CID_FOCUS_AUTO (boolean)** Разрешить непрерывную автоматическую коррекцию фокуса. Влияние корректировок фокуса вручную во время включения этой функции не определено, драйверы должны игнорировать такие запросы.

**V4L2_CID_AUTO_FOCUS_START (button)** Запускает однократный процесс автоматической фокусировки. Влияние установки этого элемента управления, когда контрол V4L2_CID_FOCUS_AUTO установлн в TRUE (1), не определено, драйверы должны игнорировать такие запросы.

**V4L2_CID_AUTO_FOCUS_STOP (button)** Прерывать автоматическую фокусировку, начатую контролом V4L2_CID_AUTO_FOCUS_START. Этот контрол эффективен только в том случае, если непрерывная автофокусировка отключена, т. е. если контрол V4L2_CID_FOCUS_AUTO имеет значение FALSE (0).

**V4L2_CID_AUTO_FOCUS_STATUS (bitmask)** Состояние автоматического фокуса. Это элемент управления, доступный только для чтения. Установка бита блокировки V4L2_LOCK_FOCUS элемента управления V4L2_CID_3A_LOCK может остановить обновление значения контрола V4L2_CID_AUTO_FOCUS_STATUS.

| | |
|---|---|
| V4L2_AUTO_FOCUS_STATUS_IDLE | Автоматический фокус не активен. |
| V4L2_AUTO_FOCUS_STATUS_BUSY | Выполняется автоматическая фокусировка. |
| V4L2_AUTO_FOCUS_STATUS_REACHED | Фокус достигнут. |
| V4L2_AUTO_FOCUS_STATUS_FAILED | Автоматическая фокусировка не удалась, драйвер не выйдет из этого состояния до тех пор, пока приложение не выполнит некое другое действие. |

Таблица 42: Состояние автофокуса

**V4L2_CID_AUTO_FOCUS_RANGE (enum)** enum v4l2_auto_focus_range - Определяет диапазон расстояний автоматической фокусировки, для которого может быть скорректирован объектив.

**V4L2_CID_ZOOM_ABSOLUTE (integer)** Задаёт фокусное расстояние объектива в абсолютном значении. Единица масштабирования является специфичной для драйвера, и ее значение должно быть положительным целым числом.

| | |
|---|---|
| V4L2_AUTO_FOCUS_RANGE_AUTO | Фотокамера автоматически выбирает диапазон фокусировки. |
| V4L2_AUTO_FOCUS_RANGE_NORMAL | Нормальный диапазон расстояний, ограничивающий наилучшую автоматическую фокусировку. |
| V4L2_AUTO_FOCUS_RANGE_MACRO | Автоматический фокус (Макросъемка). Фотокамера будет использовать минимальное возможное расстояние для автофокусировки. |
| V4L2_AUTO_FOCUS_RANGE_INFINITY | Объектив устанавливает фокус на бесконечность. |

Таблица 43: Диапазон фокусировки

**V4L2_CID_ZOOM_RELATIVE (integer)** Задаёт фокусное расстояние объектива относительно текущего значения. Положительное значения переместит группу линз зум-объективов к телескопическому положению, отрицательное значение - в широкоугольном направлении. Единица zoom является специфичной для драйвера. Это элемент управления, доступный только для записи.

**V4L2_CID_ZOOM_CONTINUOUS (integer)** Перемещать группу линз объектива с заданной скорость до тех пор, пока она не достигнет физических ограничений или до явного запроса на остановку движения. Положительное значение перемещает группу линз зум-объектива в направлении телефото. Нулевое значение останавливает движение группы линз зум-объектива. Отрицательное значение перемещает группу линз зум-объектива в широкоугольном направлении. Единица скорости зуммирования является специфичной для драйвера.

**V4L2_CID_IRIS_ABSOLUTE (integer)** Этот элемент управления устанавливает диафрагму фотокамеры на заданное значение. Единица измерения не определена. Большие значения открывают диафрагму шире, меньшие значения закрывают.

**V4L2_CID_IRIS_RELATIVE (integer)** Этот элемент управления изменяет диафрагму фотокамеры на заданную величину. Единица измерения не определена. Положительные значения открывайте диафрагму на еще один шаг вперед, отрицательные значения закрывают на один шаг. Это элемент управления, доступный только для записи.

**V4L2_CID_PRIVACY (boolean)** Прекращает получение видео камерой. Если этот элемент управления имеет значение true (1), изображение не может быть захвачено камерой. Общим средством обеспечения конфиденциальности являются механические закрытие сенсора и остановка обработки изображений встроенным по, но устройство может не ограничивается этими методами. Устройства, реализующие элемент управления конфиденциальностью, должны поддерживать доступ на чтение и могут поддерживать доступ на запись.

**V4L2_CID_BAND_STOP_FILTER (integer)** Включить или выключите режим band-stop фильтра камеры или задать его силу. Такие band-stop фильтры можно использовать, например, для фильтрации мерцающего компонента люминесцентного освещения.

**V4L2_CID_AUTO_N_PRESET_WHITE_BALANCE (enum)** enum v4l2_auto_n_preset_white_balance - Установить баланса белого на автоматический, ручной или стандартный. Предварительные настройки определяют цветовую температуру света в качестве подсказки к фотокамере для коррекции баланса белого, что приводит к наиболее точной цветопередаче. Следующие предустановки баланса белого перечислены в порядке возрастания цветовой температуры.

**V4L2_CID_WIDE_DYNAMIC_RANGE (boolean)** Включение или отключение функции широкого динамического диапазона фотокамеры. Эта функция позволяет получать четкие снимки в ситуациях, когда интенсивность освещенности значительно варьируется по всей сцене,

| | |
|---|---|
| V4L2_WHITE_BALANCE_MANUAL | Ручной баланс белого. |
| V4L2_WHITE_BALANCE_AUTO | Автоматическая корректировка баланса белого. |
| V4L2_WHITE_BALANCE_INCANDESCENT | Настройка баланса белого для освещения лампой накаливания (вольфрам). Обычно это понижает температуру до диапазона 2500.... 3500 K цветовой температуре. |
| V4L2_WHITE_BALANCE_FLUORESCENT | Предустановка баланса белого для люминесцентного освещения. Она соответствует приблизительно 4000... 5000 k цветовой температуры. |
| V4L2_WHITE_BALANCE_FLUORESCENT_H | С помощью этой настройки фотокамера компенсирует флуоресцентное освещение типа H. |
| V4L2_WHITE_BALANCE_HORIZON | Настройка баланса белого для дневного света. Соответствует цветовой температуре около 5000 K. |
| V4L2_WHITE_BALANCE_DAYLIGHT | Набор параметров баланса белого для дневного света (с ясным небом). Соответствует приблизительно 5000... 6500 K цветовой температуры. |
| V4L2_WHITE_BALANCE_FLASH | С помощью этого параметра фотокамера компенсирует свет вспышки. Цвет слегка теплее и соответствует приблизительно 5000... 5500 K цветовой температуры. |
| V4L2_WHITE_BALANCE_CLOUDY | Набор параметров баланса белого для неба с умеренной облачностью. Этот параметр соответствует диапазону приблизительно 6500... 8000 K цветовой температуры. |
| V4L2_WHITE_BALANCE_SHADE | Набор параметров баланса белого для тени или сильно облачного неба. Соответствует приблизительно 9000...10000 K |

Таблица 44: Режим баланса белого

т. е. одновременно находятся очень темные и очень яркие участки. Чаще всего она реализуется в камерах, сочетая два последовательных кадра с разной экспозицией. [6]

**V4L2_CID_IMAGE_STABILIZATION (boolean)** Включает или отключает стабилизацию изображения.

**V4L2_CID_ISO_SENSITIVITY (integer menu)** Определяет ISO эквивалент матрицы, указывающий чувствительность датчика к свету. Числа выражаются в арифметической шкале, в соответствии со стандартом ISO 12232:2006, в котором удвоение чувствительности датчика представлено удвоением численного значения ISO. Приложения должны интерпретировать значения как стандартные значения ISO, умноженные на 1000, например, значение элемента управления 800 соответствует стандарту ISO 0.8. Драйверы обычно поддерживают только подмножество стандартных значений ISO. Эффект установки этого элемента управления, когда элемент управления V4L2_CID_ISO_SENSITIVITY_AUTO имеет значение, отличное от V4L2_CID_ISO_SENSITIVITY_MANUAL, не определен, драйверы должны игнорировать такие запросы.

**V4L2_CID_ISO_SENSITIVITY_AUTO (enum)** enum v4l2_iso_sensitivity_type - Включение или отключение автоматической коррекции чувствительности ISO.

| | |
|---|---|
| V4L2_CID_ISO_SENSITIVITY_MANUAL | Чувствительность ISO устанавливается вручную. |
| V4L2_CID_ISO_SENSITIVITY_AUTO | Автоматическая корректировка чувствительности ISO. |

Таблица 45: Чувствительность ISO

**V4L2_CID_SCENE_MODE (enum)** enum v4l2_scene_mode - Этот элемент управления позволяет выбирать сюжетные программы в качестве автоматических режимов фотокамеры, оптимизированных для обычных сцен съемки. В этих режимах фотокамера определяет наилучшую экспозицию, диафрагму, фокусировка, легкий замер, баланс белого и эквивалентную чувствительность. На элементы управления этими параметрами влияет элемент управления сюжетным режимом. Точное поведение в каждом режиме зависит от спецификации камеры. Если функция сюжетного режима не используется, этот элемент управления должен быть установлен на V4L2_SCENE_MODE_NONE, чтобы убедиться, что другие, возможно, связанные элементы управления доступны. Определены следующие программы сцен:

**V4L2_CID_3A_LOCK (bitmask)** Этот элемент управления блокирует или разблокирует автоматический фокус, экспозицию и баланс белого. Автоматическая настройка может быть приостановлена независимо по каждому параметру, если установить соответствующий бит блокировки в 1. После этого камера сохраняет настройки до тех пор, пока бит блокировки не будет очищен. Определены следующие биты блокировки: Если заданный алгоритм не включен, драйверы должны игнорировать запросы блокировки и не возвращать ошибку. Примером может быть приложение, устанавливающее бит V4L2_LOCK_WHITE_BALANCE, когда элемент управления V4L2_CID_AUTO_WHITE_BALANCE имеет значение false. Значение этого элемента управления может быть изменено контролами экспозиции, баланса белого или управления фокусом.

**V4L2_CID_PAN_SPEED (integer)** Этот элемент управления вращает камеру горизонтально с конкретной скоростью. Единица измерения не определена. Положительное значение перемещает камеру вправо (по часовой стрелке при просмотре сверху), отрицательное значение влево. Нулевое значение останавливает движение, если оно выполняется, и не имеет никакого эффекта.

[6] Этот элемент управления может быть изменен на элемент управления меню в будущем, если потребуются дополнительные параметры.

| | |
|---|---|
| V4L2_SCENE_MODE_NONE | Функция сюжетного режима отключена. |
| V4L2_SCENE_MODE_BACKLIGHT | Контровый свет. Компенсирует темные тени, когда свет выходит из-за объекта, а также автоматически включает вспышку. |
| V4L2_SCENE_MODE_BEACH_SNOW | Пляж и снег. Этот режим компенсирует все белые или яркие сцены, которые, как правило, выглядят серыми с низким контрастом, так как автоматическая экспозиция фотокамеры основана на средней яркости сцены. Чтобы это компенсировать, режим автоматически слегка переэкспонирует кадр. Баланс белого может также корректироваться, чтобы компенсировать тот факт, что снег выглядит синеватым, а не белым. |
| V4L2_SCENE_MODE_CANDLELIGHT | Свеча. Камера обычно повышает чувствительность ISO и снижает выдержку. Этот режим компенсирует относительно близкий объект в сцене. Вспышка отключена, чтобы сохранить атмосферу света. |
| V4L2_SCENE_MODE_DAWN_DUSK | Рассвет и сумерки. Сохраняет цвета, видимые при низком естественном освещении до заката и после наступления сумерек. Фотокамера может выключить вспышку и автоматически фокусироваться на бесконечности. Это обычно повышает насыщенность и снижает выдержку. |
| V4L2_SCENE_MODE_FALL_COLORS | Осенние цвета. Увеличивает насыщенность и корректирует баланс белого для улучшения цвета. Картинки осенних листьев получают насыщенные красные и желтые цвета. |
| V4L2_SCENE_MODE_FIREWORKS | Салют Длительное время экспозиции используется для того, чтобы запечатлеть растущий всплеск света от фейерверка. Камера может включить стабилизацию изображения. |
| V4L2_SCENE_MODE_LANDSCAPE | Пейзаж. Фотокамера может выбрать маленькую диафрагму для обеспечения большой глубины резкости и продолжительную экспозицию, что поможет запечатлеть детали в условиях недостаточной освещённости. Фокус установлен на бесконечность. Подходит для дальних и широких пейзажей. |
| V4L2_SCENE_MODE_NIGHT | Ночь, также известная как ночной пейзаж. Спроектированная для недостаточного освещения, она сохраняет детали в темных областях, не затемняя яркие объекты. Фотокамера обычно устанавливает чувствительность ISO от среднего до высокого уровня с относительно длительным временем экспозиции и выключает вспышку. Поэтому увеличивается шум изображения и возможность размытия изображения. |
| V4L2_SCENE_MODE_PARTY_INDOOR | Вечеринка в помещении. Предназначен для съемки сюжетов, освещенных фоновым освещением в помещении, а также вспышкой. Фотокамера обычно увеличивает чувствительность ISO и корректирует экспозицию для низкого освещения. |
| V4L2_SCENE_MODE_PORTRAIT | Портрет. Фотокамера корректирует |

| | |
|---|---|
| V4L2_LOCK_EXPOSURE | Автоматическая регулировка экспозиции заблокирована. |
| V4L2_LOCK_WHITE_BALANCE | Автоматическая настройка баланса белого заблокирована. |
| V4L2_LOCK_FOCUS | Автоматическая фокусировка заблокирована . |

Таблица 47: Блокировка автоматических регулировок

**V4L2_CID_TILT_SPEED (integer)** Этот элемент управления поворачивает камеру вертикально с заданной скоростью. Единица измерения не определена. Положительное значение перемещает камеру вверх, отрицательное значение вниз. Нулевое значение останавливает движение, если оно выполняется, и не имеет никакого эффекта.

### Справочник по управлению FM-передатчиком

Класс FM-передатчика (FM_TX) включает элементы управления для распространенных функций устройств, поддерживающих FM-передачу. В настоящее время этот класс включает параметры сжатия звука, генерации пилотных частот, ограничения девиации звука, RDS передачи и настройки особенностей питания.

### Идентификаторы элементов управления FM_TX

**V4L2_CID_FM_TX_CLASS (class)** Дескриптор класса FM_TX. Вызове *ioctl* комнад VIDIOC_QUERYCT VIDIOC_QUERY_EXT_CTRL и VIDIOC_QUERYMENU для этого элемента управления будет возвращать описание элемента управления этого класса.

**V4L2_CID_RDS_TX_DEVIATION (integer)** Настройка уровня девиации частоты сигналов RDS в Гц. Диапазон и шаг являются специфичными для драйвера.

**V4L2_CID_RDS_TX_PI (integer)** Задает поле идентификации программы RDS для передачи.

**V4L2_CID_RDS_TX_PTY (integer)** Задает поле типа программы RDS для передачи. Здесь кодируется до 31 предварительно определенных типов программ.

**V4L2_CID_RDS_TX_PS_NAME (string)** Задает имя службы программ сервиса (PS_NAME) для передачи. Оно предназначено для статического отображения на приемнике. Предназначено для помощи слушателям в идентификации и отборе службы программ. В приложении E к IEC 62106, описывающем RDS, содержится полное описание правильного кодирования символов для строк имени службы программ. Также из спецификации RDS следует, что, как правило, PS представляет собой текст от одного до восьми символов. Однако можно также найти приемников, которые могут прокручивать строки размером 8 x N символов. Таким образом, этот элемент управления должен быть настроен на шаг в 8 символов. В результате он всегда должен содержать строку с размером, кратным 8.

**V4L2_CID_RDS_TX_RADIO_TEXT (string)** Устанавливает текстовую информацию для передачи. Это текстовое описание того, что транслируется. RDS Radio Text может применяться, когда необходимо передавать широковещательно более длинные имена PS, информацию о программах или любой другой текст. В этих случаях RadioText следует использовать в дополнение к V4L2_CID_RDS_TX_PS_NAME. Кодирование текстовых строк для Radio Text также полностью описано в приложении E к IEC 62106. Длина строк Radio Text зависит от того, какой блок RDS используется для их передачи, либо 32 (блок 2A), либо 64 (блок 2B). Однако возможно найти приемники, которые могут прокручивать строки размером 32 x N или 64 x N. Таким образом, этот элемент управления должен настраиваться с шагом в 32 или 64 символа. В результате он всегда должен содержать строку с размером, кратным 32 или 64.

**V4L2_CID_RDS_TX_MONO_STEREO (boolean)** Задает бит моно-стерео идентификационного кода декодера. Если установить, то звук был записан как стерео.

**V4L2_CID_RDS_TX_ARTIFICIAL_HEAD (boolean)** Задает бит искусственного заголовка идентификационного кода декодера. Если установить, то звук был записан с использованием искусственного заголовка.

**V4L2_CID_RDS_TX_COMPRESSED (boolean)** Задает бит сжатия в идентификационном коде декодера. Если установлен, то звук сжимается.

**V4L2_CID_RDS_TX_DYNAMIC_PTY (boolean)** Задает бит динамического PTY в идентификационном коде декодера. Если задан, то код PTY может переключатся динамически.

**V4L2_CID_RDS_TX_TRAFFIC_ANNOUNCEMENT (boolean)** Если задано, то выполняется динамическое отображение трафика.

**V4L2_CID_RDS_TX_TRAFFIC_PROGRAM (boolean)** Если установлен, то отображается настроенная программой частота несущей трафика.

**V4L2_CID_RDS_TX_MUSIC_SPEECH (boolean)** Если установлен, то этот канал передает музыку. Если этот флажок снят, он передает речь. Если передатчик не различает эти режимы, то этот бит должен быть установлен.

**V4L2_CID_RDS_TX_ALT_FREQS_ENABLE (boolean)** Если задано, передаются альтернативные частоты.

**V4L2_CID_RDS_TX_ALT_FREQS (__u32 array)** Альтернативные частоты в единицах кГц. Стандарт RDS позволяет определять до 25 частот. Драйверы могут поддерживать меньше частот, поэтому должны проверять размер массива.

**V4L2_CID_AUDIO_LIMITER_ENABLED (boolean)** Включает или отключает функцию ограничения девиации звука. Это ограничение полезно при попытке максимизировать громкость звука, свести к минимуму искажения в приемнике и предотвратить перемодуляцию.

**V4L2_CID_AUDIO_LIMITER_RELEASE_TIME (integer)** Задает время отключения функции ограничителя девиации звука. В единицах useconds. Шаг и диапазон являются специфичными для драйвера.

**V4L2_CID_AUDIO_LIMITER_DEVIATION (integer)** Настраивает уровень отклонения частоты звука в Гц. Диапазон и шаг являются специфичными для драйвера.

**V4L2_CID_AUDIO_COMPRESSION_ENABLED (boolean)** Включает или отключает функцию сжатия звука. Эта функция усиливает сигналы ниже порогового значения с помощью фиксированного коэффициента и сжимает звуковые сигналы, превышающие порог, по соотношению порог/(коэффициент + порог).

**V4L2_CID_AUDIO_COMPRESSION_GAIN (integer)** Установить коэффициент сжатия звука. Это значение в дБ. Диапазон и шаг являются специфичными для драйвера.

**V4L2_CID_AUDIO_COMPRESSION_THRESHOLD (integer)** Устанавливает пороговый уровень для функции сжатия звука. Это значение в дБ. Диапазон и шаг являются специфичными для драйвера.

**V4L2_CID_AUDIO_COMPRESSION_ATTACK_TIME (integer)** Установка времени атаки для функции сжатия звука. Это значение в usecond. Диапазон и шаг являются специфичными для драйвера.

**V4L2_CID_AUDIO_COMPRESSION_RELEASE_TIME (integer)** Устанавливает время отключения для сжатия звука. Это значение в usecond. Диапазон и шаг являются специфичными для драйвера.

**V4L2_CID_PILOT_TONE_ENABLED (boolean)** Включает или отключает функцию генерации пилотных частот.

**V4L2_CID_PILOT_TONE_DEVIATION (integer)** Настраивает уровень отклонения частоты пилотного сигнала. Единица измерения — Гц. Диапазон и шаг являются специфичными для драйвера.

**V4L2_CID_PILOT_TONE_FREQUENCY (integer)** Настраивает значение пилотной частоты. Единица измерения — Гц. Диапазон и шаг являются специфичными для драйвера.

**V4L2_CID_TUNE_PREEMPHASIS (enum)** enum v4l2_preemphasis - Настраивает значения предварительно выделенные для вещания. Для подъёма верхних частот в широковещательной передаче применяется предварительно выделенный фильтр. В зависимости от региона используется константа времени, равной 50 или 75 useconds. В перечислении v4l2_preemphasis определены все возможные значения для предварительного выделения. Вот они:

| | |
|---|---|
| V4L2_PREEMPHASIS_DISABLED | Предварительное выделение не применяется. |
| V4L2_PREEMPHASIS_50_uS | Используется предварительное выделение 50 uS. |
| V4L2_PREEMPHASIS_75_uS | Используется предварительное выделение 75 uS. |

Таблица 48: Предварительное выделение

**V4L2_CID_TUNE_POWER_LEVEL (integer)** Задает уровень выходной мощности для передачи сигнала. Единица измерения — dBuV. Диапазон и шаг являются специфичными для драйвера.

**V4L2_CID_TUNE_ANTENNA_CAPACITOR (integer)** Устанавливает значение конденсатора настройки антенны вручную или автоматически если значение ”ноль”. Единицы измерения, диапазон и шаг являются специфичными для драйвера.

Дополнительные сведения о спецификации RDS см. в документе IEC 62106, от CENELEC.

### Справочник по управлению вспышкой

Элементы управления V4L2 Flash предназначены для обеспечения универсального доступа к контроллеру вспышки. Устройства-контроллеры вспышки обычно используются в цифровых фотокамерах.

Интерфейс может поддерживать как LED вспышки, так и ксеноновые. На момент написания, не существует драйвера ксеноновой вспышки, использующего этот интерфейса.

### Поддерживаемые варианты использования

====== Несинхронизированная LED вспышка (программный строб)

Несинхронизированный светодиодная вспышка управляется непосредственно хостом, как и сенсор. Вспышка должна быть активирована хостом до начала экспозиции и отключения после её завершения. Хост несет полную ответственность за время вспышки.

Пример такого устройства: Nokia N900.

====== Синхронная LED вспышка (аппаратный строб)

Синхронизированный светодиодный индикатор предварительно программируется хостом (мощность и тайм-аут), но контролируется сенсором посредством строба от сенсора к вспышке.

Сенсор определяет длительность и время вспышки. Эта информация, как правило, должна предоставляться сенсору.

====== LED вспышка как фонарь

LED вспышка может использоваться в качестве фонаря в сочетании с другим вариантом использования с камерой или индивидуально.

======= ID управления вспышкой

**V4L2_CID_FLASH_CLASS (class)** Дескриптор класса FLASH.

**V4L2_CID_FLASH_LED_MODE (menu)** Определяет режим LED вспышки, мощного белого светодиода, соединённого с контроллером вспышки. Установка этого элемента управления может оказаться невозможной, при наличии некоторых ошибок. Смотри V4L2_CID_FLASH_FAULT.

| | |
|---|---|
| V4L2_FLASH_LED_MODE_NONE | Выключено |
| V4L2_FLASH_LED_MODE_FLASH | Режим вспышки. |
| V4L2_FLASH_LED_MODE_TORCH | Режим фонарика. Смотри V4L2_CID_FLASH_TORCH_INTENSITY. |

Таблица 49: Режим вспышки

**V4L2_CID_FLASH_STROBE_SOURCE (menu)** Определяет источник строба вспышки.

| | |
|---|---|
| V4L2_FLASH_STROBE_SOURCE_SOFTWARE | Строб вспышки срабатывает с помощью элемента управления V4L2_CID_FLASH_STROBE. |
| V4L2_FLASH_STROBE_SOURCE_EXTERNAL | Строб вспышки приходит от внешнего источника. Обычно это сенсор, что позволяет синхронизировать строб вспышки с началом экспозиции. |

Таблица 50: Запуск строба вспышки

**V4L2_CID_FLASH_STROBE (button)** Строб вспышки. Допустимо, когда V4L2_CID_FLASH_LED_MO установлен в значение V4L2_FLASH_LED_MODE_FLASH и V4L2_CID_FLASH_STROBE_SOURCE установлен в значение V4L2_FLASH_STROBE_SOURCE_SOFTWARE. Установка этого элемента управления может оказаться невозможной, при наличии некоторых ошибок. Смотри V4L2_CID_FLASH_FAULT.

**V4L2_CID_FLASH_STROBE_STOP (button)** Немедленно прервать строб вспышки.

**V4L2_CID_FLASH_STROBE_STATUS (boolean)** Состояние строба: в данный момент вспышка под стробом, или нет. Это элемент управления, доступный только для чтения.

**V4L2_CID_FLASH_TIMEOUT (integer)** Аппаратный тайм-аут для вспышки. Строб вспышки закнчивается после того, как этот период времени прошел с начала строба.

**V4L2_CID_FLASH_INTENSITY (integer)** Интенсивность строба вспышки, когда LED находится в режиме вспышки (V4L2_FLASH_LED_MODE_FLASH). Должна задаваться в единицах миллиампер (MA), если это возможно.

**V4L2_CID_FLASH_TORCH_INTENSITY (integer)** Интенсивность индикатора вспышки в режиме фонаря (V4L2_FLASH_LED_MODE_TORCH). Должна задаваться в единицах миллиампер (MA), если это возможно. Установка этого элемента управления может оказаться невозможной, при наличии некоторых ошибок. Смотри V4L2_CID_FLASH_FAULT.

**V4L2_CID_FLASH_INDICATOR_INTENSITY (integer)** Интенсивность индикаторного LED. Индикаторный LED может быть полностью независим от LED вспышки. Должна задаваться, по возможности, микроамперами (uA).

**V4L2_CID_FLASH_FAULT (bitmask)** Ошибки, связанные со вспышкой. Ошибки сообщают о конкретных проблемах в самой микросхеме вспышки или LED, соединённых с ней. Ошибки могут препятствовать дальнейшему использованию некоторых элементов управления вспышкой. В частности, V4L2_CID_FLASH_LED_MODE устанавливается в значения V4L2_FLASH_L если ошибка связана с LED. В точности, какие ошибки имеют такой эффект, зависит от чипа. Считывание ошибок сбрасывает элемент управления и возвращает чип в пригодное для использования состояние, если это возможно.

| | |
|---|---|
| V4L2_FLASH_FAULT_OVER_VOLTAGE | Напряжение контроллера вспышки на LED вспышки превысило предельное значение, определенное для контроллера вспышки. |
| V4L2_FLASH_FAULT_TIMEOUT | Строб вспышки все еще включен, когда истек тайм-аут, установленный пользователем — V4L2_CID_FLASH_TIMEOUT. Не все контроллеры вспышек могут устанавливать это во всех таких условиях. |
| V4L2_FLASH_FAULT_OVER_TEMPERATURE | Контроллер вспышки перегрелся. |
| V4L2_FLASH_FAULT_SHORT_CIRCUIT | Сработала защита от короткого замыкания контроллера вспышки. |
| V4L2_FLASH_FAULT_OVER_CURRENT | Ток в блоке питания LED превысил предельное значение, определенное для контроллера вспышки. |
| V4L2_FLASH_FAULT_INDICATOR | Контроллер вспышки обнаружил короткое замыкание или обрыв цепи LED индикатора. |
| V4L2_FLASH_FAULT_UNDER_VOLTAGE | Напряжение контроллера вспышки на LED вспышки ниже минимального предела, определенного для контроллера вспышки. |
| V4L2_FLASH_FAULT_INPUT_VOLTAGE | Входное напряжение контроллера вспышки находится ниже предельного значения, при котором вспышка в полном объеме не может быть выполнена. Условие сохраняется до тех пор, пока этот флаг больше не будет установлен. |
| V4L2_FLASH_FAULT_LED_OVER_TEMPERATURE | Температура LED превысила допустимый верхний предел. |

Таблица 51: Коды ошибок в вспышке

**V4L2_CID_FLASH_CHARGE (boolean)** Разрешает или запрещает зарядку конденсатора для ксеноновой вспышки.

**V4L2_CID_FLASH_READY (boolean)** Вспышка готова к срабатыванию? Ксеноновая вспышка требует, чтобы конденсаторы были заряжены до прихода строба. LED вспышки часто требуется период для охлаждения после срабатывания, в течение которого еще один строб невозможен. Это элемент управления, доступный только для чтения.

### Справочник по управлению JPEG

Класс JPEG содержит элементы управления для общих функций кодировщиков и декодеров JPEG. В настоящее время он включает функции для кодеков, реализующих прогрессивный процесс сжатия DCT с энтропийным кодированием по Хафману.

### ID элементов управления JPEG

**V4L2_CID_JPEG_CLASS (class)** Дескриптор класса JPEG. Вызове *ioctl* комнад VIDIOC_QUERYCTRL, VIDIOC_QUERY_EXT_CTRL и VIDIOC_QUERYMENU для этого элемента управления будет возвращать описание элемента управления этого класса.

**V4L2_CID_JPEG_CHROMA_SUBSAMPLING (menu)** Коэффициенты насыщенности цвета описывают, как сэмплируется каждый компонент входного изображения, в том что касается максимальной скорости сэмплирования в каждом пространственном измерении. Смотри ITU-T.81, раздел A.1.1. для больших подробностей. Элемент управления V4L2_CID_JPEG_CHROMA определяет, каким образом компоненты Cb и Cr будут передискретизированы после преобразования входного изображения из RGB в цветовое пространство Y'CbCr.

| | |
|---|---|
| V4L2_JPEG_CHROMA_SUBSAMPLING_444 | Отсутствует цветовая передискретизация, каждая точка имеет значения Y, Cr и Cb. |
| V4L2_JPEG_CHROMA_SUBSAMPLING_422 | Горизонтальная передискретизация CR, Cr компонент в 2 раза. |
| V4L2_JPEG_CHROMA_SUBSAMPLING_420 | Передискретизаци Cr и Cb, компонент по горизонтали и вертикали в 2 раза. |
| V4L2_JPEG_CHROMA_SUBSAMPLING_411 | Горизонтальная передискретизация CR, Cr компонент в 4 раза. |
| V4L2_JPEG_CHROMA_SUBSAMPLING_410 | Передискретизаця Cr и Cb, компонент по горизонтали в 4 раза и вертикали в 2 раза. |
| V4L2_JPEG_CHROMA_SUBSAMPLING_GRAY | Использовать только компонент яркости. |

Таблица 52: Режим передикретизации

**V4L2_CID_JPEG_RESTART_INTERVAL (integer)** Интервал перезапуска определяет интервал вставки маркеров RSTm (m = 0.. 7). Целью этих маркеров является дополнительно повторная инициализация процессов кодирования для независимой обработки блоков изображения. Для процессов сжатия с потерей данных единицей интервала перезапуска является MCU (минимальная кодируемая единица), а ее значение содержится в маркере DRI (Определение интервала перезапуска). Если для элемент управления V4L2_CID_JPEG_RESTART_INTERV установлен в значение 0, маркеры DRI и RSTm не будут вставляться.

**V4L2_CID_JPEG_COMPRESSION_QUALITY (integer)** Элемент управления V4L2_CID_JPEG_COMPR определяет компромисс между качеством изображения и его размером. Он обеспечивает более простой способ управления качеством изображения, не требуя прямой перенастройки дискретных таблиц яркости и цветности. В тех случаях, когда драйвер использует таблицы дискретности, настроенные непосредственно приложением, с помощью интерфейсов, определенных в других местах, управляющий элемент V4L2_CID_JPEG_COMPRESSION_QUALITY должен быть установлен драйвером в 0. Диапазон значений этого элемента управления является специфическим для драйвера. Значимыми являются только положительные, ненулевые значения. Рекомендуемый диапазон — 1-100, где большие значения соответствуют лучшему качеству изображения.

**V4L2_CID_JPEG_ACTIVE_MARKER (bitmask)** Указывает, какие маркеры JPEG включаются в сжатый поток. Этот элемент управления действителен только для кодировщиков.

Для получения дополнительных сведений о спецификации JPEG обратитесь к ITU-T.81, JFIF, W3C JPEG JFIF.

### Справочник по управлению источниками изображений

Класс элемента управления Image Source предназначен для низкоуровневого управления устройствами - источниками изображений, такими как датчики изображения. Устройства обладают

| | |
|---|---|
| V4L2_JPEG_ACTIVE_MARKER_APP0 | Сегмент данных приложения APP0. |
| V4L2_JPEG_ACTIVE_MARKER_APP1 | Сегмент данных приложения APP1. |
| V4L2_JPEG_ACTIVE_MARKER_COM | Сегмент комментария. |
| V4L2_JPEG_ACTIVE_MARKER_DQT | Сегмент таблиц дискретизации. |
| V4L2_JPEG_ACTIVE_MARKER_DHT | Сегмент таблиц Хаффмана. |

Таблица 53: Сегменты данных, включаемые в поток

аналого-цифровым преобразователем и передатчиком на шину данных для передачи данных изображения из устройства.

#### ID элементов управления источниками изображений

**V4L2_CID_IMAGE_SOURCE_CLASS (class)** Дескриптор класса IMAGE_SOURCE.

**V4L2_CID_VBLANK (integer)** Пробел по вертикали Период ожидания после каждого кадра, в течение которого данные изображения не передаются. Единица измерения по вертикали — это строка. Каждая строка имеет длину, равную ширине изображения плюс горизонтальный пробел в пикселях, определяемый элементом управления V4L2_CID_PIXEL_RATE в том же подустройстве.

**V4L2_CID_HBLANK (integer)** Горизонтальный пробел. Период бездействия после каждой строки данных изображения, в течение которых не передаются данные изображения. Единица горизонтального пробела — пикселы.

**V4L2_CID_ANALOGUE_GAIN (integer)** Аналоговый коэффициент усиления — это усиление, влияющее на все компоненты цвета в пиксельной матрице. Операция усиления выполняется в аналоговой части до преобразования A/D.

**V4L2_CID_TEST_PATTERN_RED (integer)** Тестовый образец красного компонента цвета.

**V4L2_CID_TEST_PATTERN_GREENR (integer)** Тестовый образец зелёного (следующего за красным) компонента цвета.

**V4L2_CID_TEST_PATTERN_BLUE (integer)** Тестовый образец синего компонента цвета.

**V4L2_CID_TEST_PATTERN_GREENB (integer)** Тестовый образец зелёного (следующего за синим) компонента цвета.

#### Справочник по управлению обработкой изображений

Класс Image Process предназначен для низкоуровневого управления функциями обработки изображения. В отличие от V4L2_CID_IMAGE_SOURCE_CLASS, элементы управления данного класса влияют на обработку изображения и не управляют его захватом.

#### ID управления обработкой изображения

**V4L2_CID_IMAGE_PROC_CLASS (class)** Дескриптор класса IMAGE_PROC.

**V4L2_CID_LINK_FREQ (integer menu)** Частота шины данных. Вместе с пиксельным кодом шины мультимедиа, типом шины (тактов на один сэмпл) частота шины данных определяет пиксельную скорость (V4L2_CID_PIXEL_RATE) в матрице пикселей (или, возможно, в другом месте, если устройство не является матрицей изображения). Частота кадров может быть рассчитана на основе пиксельной частоты, ширины и высоты изображения, а также

горизонтального и вертикального пробелов. Хотя управление пиксельной скоростью может быть осуществлено в другом месте, а не в подустройстве, содержащем пиксельную матрицу, частота кадров не может быть определена из этой информации. Это объясняется тем, что только в пиксельном массиве точно известно, чему равны пробелы по вертикали и горизонтали: никакие другие пустые значения не допускаются в массиве пикселей. Выбор частоты кадров выполняется путем выбора требуемого горизонтального и вертикального размера пробелов. Единицей этого элемента управления является Гц.

**V4L2_CID_PIXEL_RATE (64-bit integer)** Пиксельная скорость на площадках источников подустройств. Этот элемент управления только для чтения, а его едницы — Пиксели/секунды.

**V4L2_CID_TEST_PATTERN (menu)** Некоторые устройства захвата/отображения/сенсоры имеют возможность создавать изображения тестовых образов. Эти специфичные для оборудования тестовые образы можно использовать для проверки правильности работы устройства.

**V4L2_CID_DEINTERLACING_MODE (menu)** Режим преобразования черезстрочной развёртки в обычную (например, Bob, Weave,...). Пункты меню являются специфичными для драйвера и задокументированы в Video4Linux (V4L) документации по драйверу.

### Справочник по управляющим элементам цифрового видео

Класс управляющего элемента Digital Video предназначен для управления приемниками и передатчиками для VGA, DVI (цифрового визуального интерфейса), HDMI и DisplayPort (DP). Как правило, эти элементы управления должны быть приватными в подустройствах приемников или передатчиков, реализующих их, так что они были доступны только посредством /dev/Дев/v4l-subdev* дерева устройств.

---

**Замечание**
Обратите внимание, что эти устройства могут иметь несколько разъёмов ввода или вывода, например HDMI. Даже если подустройство будет получать или передавать видео только с одной из этих Pad, другие Pad все еще могут быть активными, когда речь идет о EDID (расширенные идентификационные данные дисплея, EDID) и HDCP (система защиты цифрового контента с высокой пропускной способностью, HDCP), что позволяет устройству заблаговременно обрабатывать довольно медленную обработку EDID/HDCP. Это позволяет быстро переключаться между соединителями.

---

Эти разъёмы отображаются в нескольких элементах управления в этом разделе как битовая маска, один бит для каждого разъёма. Бит 0 соответствует разъёму 0, бит 1 для разъёма 1 и т. д. Максимальное значение элемента управления — это набор допустимых разъёмов.

### ID управляющих элементов цифрового видео

**V4L2_CID_DV_CLASS (class)** Дескриптор класса цифрового видео.

**V4L2_CID_DV_TX_HOTPLUG (bitmask)** Многие соединители имеют контакт, напряжение на котором становится высоким, если информация EDID доступна из источника сигнала. Этот элемент управления показывает состояние контакта подключения устройства, как задано на передатчике. Каждый бит соответствует выходному разъёму на передатчике. Если разъём вывода не имеет связанного с ним контакта доступности, то бит для этой разъёма будет равен 0. Этот элемент управления только для чтения, применим к разъемам DVI-D, HDMI и DisplayPort.

**V4L2_CID_DV_TX_RXSENSE (bitmask)** Детектор RX обнаружил скачок на линиях синхронизации TMDS. Обычно это означает, что приемник имеет вышел/вошёл в режим ожидания (т.

е. передатчик может обнаружить, что приемник готов к приему видео). Каждый бит соответствует выходному разъёму на передатчике. Если у разъёма вывода отсутствует связанный с ней Rx детектор, то бит для этого разъёма будет равен 0. Этот элемент управления только для чтения применим к устройствам DVI-D и HDMI.

**V4L2_CID_DV_TX_EDID_PRESENT (bitmask)** Когда передатчик увидит сигнал горячего подключения от приемника, он попытается прочитать EDID. Если установлен, то передатчик должен прочитать, по крайней, мере первый блок (= 128 байт). Каждый бит соответствует выходному разъёму на передатчике. Если разъём вывода не поддерживает EDID, то бит для этого разъёма будет равен 0. Этот элемент управления только для чтения, применим к разъемам VGA, DVI-A/D, HDMI и DisplayPort.

**V4L2_CID_DV_TX_MODE (enum)** enum v4l2_dv_tx_mode - Передатчики HDMI могут передавать в режиме DVI-D (только видео) или в режиме HDMI (видео + аудио + вспомогательные данные). Этот элемент управления выбирает используемый режим: V4L2_DV_TX_MODE_DVI_D или V4L2_DV_TX_MODE_HDMI. Этот элемент управления применим к разъемам HDMI.

**V4L2_CID_DV_TX_RGB_RANGE (enum)** enum v4l2_dv_rgb_range - Установить диапазон дискретности для вывода RGB. V4L2_DV_RANGE_AUTO следует за диапазоном дискретизации RGB, заданным в стандарте для видеоинтерфейса (например HDMI). V4L2_DV_RANGE_LIMITED и V4L2_DV_RANGE_FULL переопределяют стандарт для совместимости с раковинами, забыли реализовать стандарт правильно (к сожалению, это довольно распространено для HDMI и DVI-D). Полный диапазон позволяет использовать все возможные значения, в то время как ограниченный диапазон задает диапазон (16 < < (N-8))-(235 < (N-8)), где N — число бит на компонент. Этот элемент управления применим к разъемам VGA, DVI-A/D, HDMI и DisplayPort.

**V4L2_CID_DV_TX_IT_CONTENT_TYPE (enum)** enum v4l2_dv_it_content_type - Настраивает тип контента в переданном видео. Эта информация отправляется по разъемам HDMI и DisplayPort как часть AVI InfoFrame. Термин «IT-контент» используется для содержимого, которое исходит от компьютера в противоположность содержимому из телевизионной передачи или аналогового источника. Перечисление v4l2_dv_it_content_type определяет возможные типы контента:

| | |
|---|---|
| V4L2_DV_IT_CONTENT_TYPE_GRAPHICS | Графическое содержимое. Пиксельные данные должны передаваться без фильтрации и без восстановления аналогового сигнала. |
| V4L2_DV_IT_CONTENT_TYPE_PHOTO | Фотография. Содержимое извлекается из цифровых фотографий. Содержимое должно быть передано с минимальным масштабированием и улучшениями изображения. |
| V4L2_DV_IT_CONTENT_TYPE_CINEMA | Кино |
| V4L2_DV_IT_CONTENT_TYPE_GAME | Игровое содержимое Задержка звука и видео должна быть минимизирована. |
| V4L2_DV_IT_CONTENT_TYPE_NO_ITC | Информация об IT содержимом не доступна, и бит ITC в AVI InfoFrame имеет значение 0. |

Таблица 54: Типы конетента

**V4L2_CID_DV_RX_POWER_PRESENT (bitmask)** Определяет, получает ли приемник питание от источника (например, HDMI передает 5 в на один из контактов). Это часто используется для питания EEPROM, в которой есть информация EDID, так что источник может прочитать EDID даже в том случае, если приемник находится в ждущем режиме или выключен. Каждый бит соответствует разъёму ввода в передатчике. Если разхъём ввода не может определить, присутствует ли питание, то бит для этого разъёма будет равен 0. Этот элемент управления только для чтения, применим к разъемам DVI-D, HDMI и DisplayPort.

**V4L2_CID_DV_RX_RGB_RANGE (enum)** enum v4l2_dv_rgb_range - Установить диапазон дискретности для ввода RGB. V4L2_DV_RANGE_AUTO следует за диапазоном дискретизации RGB, заданным в стандарте для видеоинтерфейса (например HDMI). V4L2_DV_RANGE_LIMITED и V4L2_DV_RANGE_FULL переопределяют стандарт для совместимости с источниками, забыли реализовать стандарт правильно (к сожалению, это довольно распространено для HDMI и DVI-D). Полный диапазон позволяет использовать все возможные значения, в то время как ограниченный диапазон задает диапазон (16 < < (N-8))-(235 < (N-8)), где N — число бит на компонент. Этот элемент управления применим к разъемам VGA, DVI-A/D, HDMI и DisplayPort.

**V4L2_CID_DV_RX_IT_CONTENT_TYPE (enum)** enum v4l2_dv_it_content_type - Считывает тип контента в принимаемом видео. Эта информация отправляется по разъемам HDMI и DisplayPort как часть AVI InfoFrame. Термин «IT-контент» используется для содержимого, которое исходит от компьютера в противоположность содержимому из телевизионной передачи или аналогового источника. См. V4L2_CID_DV_TX_IT_CONTENT_TYPE для доступных типов контента.

### Справочник по управлению FM-приемником

Класс FM-приемника (FM_RX) включает в себя элементы управления для распространенных функций устройств, поддерживающих ЧМ-прием.

### ID элементов управления FM_RX

**V4L2_CID_FM_RX_CLASS (class)** Дескриптор класса FM_RX. Вызове *ioctl* комнад VIDIOC_QUERYCT VIDIOC_QUERY_EXT_CTRL и VIDIOC_QUERYMENU для этого элемента управления будет возвращать описание элемента управления этого класса.

**V4L2_CID_RDS_RECEPTION (boolean)** Включение/отключение RDS приема по радио-тюнеру

**V4L2_CID_RDS_RX_PTY (integer)** Возвращает поле типа программы RDS. Здесь кодируется до 31 предварительно определенных типов программ.

**V4L2_CID_RDS_RX_PS_NAME (string)** Получает имя програмы службы (PS_NAME). Оно предназначено для статического отображения на приемнике. Предназначено для помощи слушателям в идентификации и отборе службы программ. В приложении E к IEC 62106, описывающем RDS, содержится полное описание правильного кодирования символов для строк имени службы программ. Также из спецификации RDS следует, что, как правило, PS представляет собой текст от одного до восьми символов. Однако можно также найти приемников, которые могут прокручивать строки размером 8 x N символов. Таким образом, этот элемент управления должен быть настроен на шаг в 8 символов. В результате он всегда должен содержать строку с размером, кратным 8.

**V4L2_CID_RDS_RX_RADIO_TEXT (string)** Получает текстовую информацию по радио. Это текстовое описание того, что транслируется. RDS Radio Text может применяться, когда необходимо передавать широковещательно более длинные имена PS, информацию о программах или любой другой текст. В этих случаях RadioText следует использовать в дополнение к V4L2_CID_RDS_RX_PS_NAME. Кодирование текстовых строк для Radio Text также полностью описано в приложении E к IEC 62106. Длина строк Radio Text зависит от того, какой блок RDS используется для их передачи, либо 32 (блок 2A), либо 64 (блок 2B). Однако возможно найти приемники, которые могут прокручивать строки размером 32 x N или 64 x N. Таким образом, этот элемент управления должен настраиваться с шагом в 32 или 64 символа. В результате он всегда должен содержать строку с размером, кратным 32 или 64.

**V4L2_CID_RDS_RX_TRAFFIC_ANNOUNCEMENT (boolean)** Если задано, то выполняется динамическое отображение трафика.

**V4L2_CID_RDS_RX_TRAFFIC_PROGRAM (boolean)** Если установлен, то отображается настроенная программой частота несущей трафика.

**V4L2_CID_RDS_RX_MUSIC_SPEECH (boolean)** Если установлен, то этот канал передает музыку. Если этот флажок снят, он передает речь. Если передатчик не различает эти режимы, то этот бит должен быть установлен.

**V4L2_CID_TUNE_DEEMPHASIS (enum)** enum v4l2_deemphasis - Настраивает значение параметра уменьшения высокочастотных составляющих для приема. Для подъёма верхних звуковых частот в широковещательной передаче применяется фильтр устранение высокочастотных составляющих. В зависимости от региона используется константа времени, равной 50 или 75 useconds. Перечисление v4l2_deemphasis определяет возможные значения для фильтра высокочастотных составляющих . Вот они:

| | |
|---|---|
| V4L2_DEEMPHASIS_DISABLED | Не применять фильтр высокочастотных составляющих. |
| V4L2_DEEMPHASIS_50_uS | Фильтр использует значение 50 uS. |
| V4L2_DEEMPHASIS_75_uS | Фильтр используется значение 75. |

Таблица 55: Фильтр высоких частот

### Справочник по управлению детектором

Класс ”Detect” включает элементы управления для общих функций различных устройств, поддерживающих обнаружение движения или объектов.

### ID элементов управления детектором

**V4L2_CID_DETECT_CLASS (class)** Дескриптор класса детектора. Вызове *ioctl* комнад VIDIOC_QUE VIDIOC_QUERY_EXT_CTRL и VIDIOC_QUERYMENU для этого элемента управления будет возвращать описание элемента управления этого класса.

**V4L2_CID_DETECT_MD_MODE (menu)** Задает режим обнаружения движения.

**V4L2_CID_DETECT_MD_GLOBAL_THRESHOLD (integer)** Задает глобальный порог обнаружения движения, используемый с режимом обнаружения движения V4L2_DETECT_MD_MODE_G

**V4L2_CID_DETECT_MD_THRESHOLD_GRID (__u16 matrix)** Устанавливает пороги обнаружения движения для каждой ячейки в сетке. Используется с режимом обнаружения движения V4L2_DETECT_MD_MODE_THRESHOLD_GRID. Элемент матрицы (0, 0) представляет ячейку в верхнем левом углу сетки.

**V4L2_CID_DETECT_MD_REGION_GRID (__u8 matrix)** Задает области определения движения для каждой ячейки в сетке. Используется с режимом обнаружения движения V4L2_DETECT_M Элемент матрицы (0, 0) представляет ячейку в верхнем левом углу сетки.

### Справочник по управлению RF тюнером

Класс RF-тюнера (RF_TUNER) включает элементы управления для общих особенностей устройств, имеющих радиотюнер.

В этом контексте RF-тюнер является радиоприемником между антенной и демодулятором. Он получает радиочастоту (RF) с антенны и преобразует полученный сигнал в нижнюю промежуточную частоту (IF) или базовую частоту (BB). Тюнеры, которые могут выдавать базовую частоту

| | |
|---|---|
| V4L2_DETECT_MD_MODE_DISABLED | Отключает обнаружение движения. |
| V4L2_DETECT_MD_MODE_GLOBAL | Использовать один порог обнаружения движения. |
| V4L2_DETECT_MD_MODE_THRESHOLD_GRID | Изображение делится на сетку, каждая ячейка с собственным порогом обнаружения движения. Эти пороговые значения устанавливаются с помощью матричного элемента управления V4L2_CID_DETECT_MD_THRESHOLD_GRID. |
| V4L2_DETECT_MD_MODE_REGION_GRID | Изображение делится на сетку, каждая ячейка с собственным региональным значением, определяющим, какие пороговые значения для обнаружения движения в регионе должны использоваться. У каждого региона есть свои пороговые значения. Настройка пороговых значений для каждого региона является специфичной для драйвера. Регионы для сетки задаются с помощью матричного элемента управления V4L2_CID_DETECT_MD_REGION_GRID. |

Таблица 56: Режим обнаружения движения

(BB), часто называются Zero-IF тюнерами. Старые тюнеры обычно были простыми ПЛЛ тюнерами внутри металлического корпуса, а новые — высокоинтегрированные чипы без металлического корпуса - «полупроводниковые тюнеры». Эти элементы управления в основном применимы для новых полнофункциональных полупроводниковых тюнеров, просто потому, что устаревшие тюнеры не имеют большинства регулируемых функций.

Для получения дополнительной информации о RF тюнерах РФ см. Tuner (radio) и RF front end из Wikipedia

### ID элементов управления RF_TUNER

**V4L2_CID_RF_TUNER_CLASS (class)** Дескриптор класса RF_TUNER. Вызове *ioctl* комнад VIDIOC_Q VIDIOC_QUERY_EXT_CTRL и VIDIOC_QUERYMENU для этого элемента управления будет возвращать описание элемента управления этого класса.

**V4L2_CID_RF_TUNER_BANDWIDTH_AUTO (boolean)** Включает/отключает настройку полосы пропускания канала тюнера. В автоматическом режиме, настройка полосы пропускания выполняется драйвером.

**V4L2_CID_RF_TUNER_BANDWIDTH (integer)** Фильтр (ы) при прохождении сигнала в тюнере, используется для фильтрации сигнала в соответствии с потребностями принимающей стороны. Драйвер настраивает фильтры для достижения требований к пропускной способности. Используется, когда V4L2_CID_RF_TUNER_BANDWIDTH_AUTO не задан. Единица измерения — Гц. Диапазон и шаг являются специфичными для драйвера.

**V4L2_CID_RF_TUNER_LNA_GAIN_AUTO (boolean)** Включение/отключение автоматического управления усилением (AGC) в малошумящем усилителе (LNA).

**V4L2_CID_RF_TUNER_MIXER_GAIN_AUTO (boolean)** Включение/отключение автоматического управления усилением (AGC) в смесителе

**V4L2_CID_RF_TUNER_IF_GAIN_AUTO (boolean)** Включение/отключение автоматического управления усилением (AGC) по промежуточной частоте (IF)

**V4L2_CID_RF_TUNER_RF_GAIN (integer)** Усилитель RF-это самый первый усилитель на пути сигнала по приемнику, сразу после антенны. Разница между усилением в LNA и усилением RF в этом документе, заключается в том, что LNA интегрирован в тюнер, а усилитель RF - это отдельный чип. В одном и том же устройстве могут быть как RF, так и LNA управление усилением. Диапазон и шаг являются специфичными для драйвера.

**V4L2_CID_RF_TUNER_LNA_GAIN (integer)** LNA (малошумящий усилитель) — это первый стадия усиления сигнала при прохождении по RF-тюнеру. Он расположен очень близко к антенному входу тюнера. Используется, когда V4L2_CID_RF_TUNER_LNA_GAIN_AUTO не задан. См. V4L2_CID_RF_TUNER_RF_GAIN для того, чтобы понять, чем RF-усиление и LNA-усиление отличаются друг от друга. Диапазон и шаг являются специфичными для драйвера.

**V4L2_CID_RF_TUNER_MIXER_GAIN (integer)** Усиление в смесителе — это второй этап усиления на пути сигнала в RF-тюнере. Он расположен внутри блока смесителя, где сигнал RF преобразуется в промежуточную частоту смесителем. Используется, когда V4L2_CID_RF_TUNER_ не задан. Диапазон и шаг являются специфичными для драйвера.

**V4L2_CID_RF_TUNER_IF_GAIN (integer)** Усиление по ПЧ (IF) является последним этапом на пути сигнала в RF-тюнере. Он расположен на выходе RF-тюнера. Он контролирует уровень сигнала промежуточной частоты или ширину полосы данных. Используется, когда V4L2_CID_RF_TUNER_IF_GAIN_AUTO не задан. Диапазон и шаг являются специфичными для драйвера.

**V4L2_CID_RF_TUNER_PLL_LOCK (boolean)** Синтезатор ФАПЧ(PLL) заблокирован? При установке этого элемента управления радиотюнер принимает заданную частоту. Это элемент управления, доступный только для чтения.

## Форматы данных

### Согласование формата данных

Различные устройства обмениваются разными типами данных с приложениями, например видеоизображениями, необработанными или фрагментированными VBI данными, RDS-датаграммами. Даже в пределах одного вида возможны различные форматы, в частности существует изобилие форматов изображений. Хотя драйверы должны обеспечивать настройки по умолчанию, а выбранные настройки должны сохраняется при закрытии и повторном открытии устройства, приложения всегда должны согласовывать формат данных перед началом обмена данными. Согласование означает, что приложение запрашивает определенный формат, а драйвер выбирает и сообщает о наилучшем оборудовании для удовлетворения запроса. Приложения, разумеется, могут также просто запросить текущий выбор.

Существует единый механизм для согласования всех форматов данных с использованием сложной структуры v4l2_format и *ioctl* команд VIDIOC_G_FMT и VIDIOC_S_FMT. Кроме того, команду *ioctl* VIDIOC_TRY_FMT можно использовать для проверки того, что может сделать оборудование, без фактического выбора нового формата данных. Форматы данных, поддерживаемые V4L2 API, рассматриваются в соответствующем разделе устройства в интерфейсах. Для более тщательный знакомства с форматами изображений см. раздел Image Formats.

Вызов *ioctl* VIDIOC_S_FMT является важной поворотной точкой в последовательности инициализации. До этой точки множество графических приложений могут одновременно обращаться к одному устройству, чтобы выбрать текущий ввод, изменить элементы управления или изменить другие свойства. Первый вызов VIDIOC_S_FMT назначает логический поток (видео-данные, VBI данные и т. д.) монопольно одному дескриптору файла.

Монопольно - означает, что нет других приложений, точнее говоря, другого дескриптора файла, который может захватить этот поток или изменить свойства устройства, несовместимые с согласованными параметрами. Пример, изменение стандарта видео, когда новый стандарт использу-

ет другое количество строк развёртки, может сделать выбранный формат изображения недействительным. Поэтому только дескриптор файла, владеющий потоком, может сделать недействительными изменения. Поэтому несколько дескрипторов файлов, которые захватили различные логические потоки, не позволяют друг другу вмешиваться в их параметры. Когда, например, наложение видео начинается или уже выполняется, одновременный захват видео может быть ограничен одним и тем же кадрированием и размером изображения.

Если в приложениях опущен вызов *ioctl* VIDIOC_S_FMT, то блокировка побочных эффектов возникает на следующем шаге, при выборе метода ввода-вывода с помощью вызова *ioctl* команды VIDIOC_REQBUFS или неявно, при вызове первого read() или write ().

Обычно дескриптору файла может быть назначен только один логический поток, исключение — это драйверы, позволяющие одновременное захват и перекрытие видео с использованием одного дескриптора файла для совместимости с V4L или более ранними версиями V4L2. Переключение логического потока или возврат в режим графического приложения возможен путем закрытия и повторного открытия устройства. Драйверы могут поддерживать переключение при использовании VIDIOC_S_FMT.

Все драйверы, которые обмениваются данными с приложениями, должны поддерживать *ioctl* команды VIDIOC_S_FMT и VIDIOC_S_FMT. Реализация VIDIOC_TRY_FMT настоятельно рекомендуется, но не обязательно.

#### Перечисление форматов изображения

Помимо функций согласования универсального формата, для перечисления всех форматов изображений, поддерживаемых при видеозахвате, наложении или выводе, устройствами, имеется специальная команда *ioctl* . [7]

Команда IOCTL VIDIOC_ENUM_FMT должна поддерживаться всеми драйверами, которые обмениваются данными изображений с приложениями.

**Важно**
Драйверы не должны преобразовывать форматы изображений в пространстве ядра. Они должны перечислять только форматы, непосредственно поддерживаемые оборудованием. Если есть необходимость преобразования, разработчики драйверов должны публиковать пример процедуры преобразования или библиотеки для применения в приложениях.

## Одно- и многоплоскостные API

Некоторые устройства требуют, чтобы данные для каждого входного или выходного видеокадра были размещены в несмежных буферах памяти. В таких случаях один видеокадр должен адресоваться с использованием более одного адреса памяти, т. е. одного указателя на ”срез”. Срез - вложенный буфер текущего кадра. Примеры подобных форматов см. ”Форматы изображений”.

Первоначально в API V4L2 не поддерживались многосрезные буферы, и для их обработки был введен набор расширений. Эти расширения представляют собой то, что называется ”многосрезным интерфейсом API”.

Некоторые вызовы и структуры API V4L2 интерпретируются по-разному, в зависимости от того, используется ли API для одного или несколько срезов. Приложение может выбрать, следует ли использовать один или другой вариант, передавая соответствующий тип буфера в *ioctl* вызовы. Многоплоскостные версии типов буферов обявляются с суффиксами _MPLANE. Список доступных типов много-срезных буферов см. в перечислении v4l2_buf_type.

[7] Приложение, перечисляющее форматы, не имеет априори знания (в противном случае оно может явно запросить их и не нужно перечислять) кажется бесполезным, но есть приложения, которые служат прокси-сервером между драйверами и фактическими видеоприложениями, для которых это полезно.

#### Многоплоскостные форматы

Многоплоскостный API интерфейс вводит новые многоплоскостные форматы. В этих форматах используется отдельный набор кодов FourCC. Важно различать многоплоскостные API интерфейсы и многоплоскостный формат. Многоплоскостные вызовы API могут обрабатывать все одноплоскостные форматы (до тех пор, пока они передаются в многоплоскостных структурах API), в то время как одноплоскостный интерфейс API не может обрабатывать многоплоскостные форматы.

#### Вызовы, отличающиеся для одноплоскостные и многоплоскостных API интерфейсов

**VIDIOC_QUERYCAP** Добавлены две дополнительные многоплоскостные возможности. Они могут быть установлены вместе с не-многоплоскостными для устройств, которые обрабатывают как одно, так и много-плоскостные форматы.

**VIDIOC_G_FMT, VIDIOC_S_FMT, VIDIOC_TRY_FMT** Добавлены новые структуры для описания многоплоскостных форматов: v4l2_pix_format_mplane и v4l2_plane_pix_format. Драйверы могут определять новые многоплоскостные форматы, имеющие различные FourCC коды из существующих одноплоскостных форматов.

**VIDIOC_QBUF, VIDIOC_DQBUF, VIDIOC_QUERYBUF** Добавлена новая структура v4l2_plane для описания плоскостей. Массив таких структур передаётся в новом поле m.planes структуры v4l2_buffer.

**VIDIOC_REQBUFS** Будет выделять vногоплоскостные буферы по запросу.

### Кадрирование изображений, вставка и масштабирование

Некоторые устройства видеозахвата могут сэмплировать подраздел кадра и уменьшать или увеличивать его до изображения произвольного размера. Эти способности называются обрезкой и масштабированием. Некоторые видео устройства вывода могут масштабировать изображение вверх или вниз и вставлять его в произвольной строке кадра с заданным горизонтальном смещением.

Приложения могут использовать следующий API для выбора области видеосигнала, запрашивать область по умолчанию и аппаратные ограничения.

---

**Внимание**
Несмотря на их названия, *ioctl* команды VIDIOC_CROPCAP, VIDIOC_G_CROP и VIDIOC_S_CROP применяются как к устройствам ввода, так и к устройствам вывода.

---

Для масштабирования требуется источник и цель. На устройстве захвата видео или наложения источником является видеосигналом, а IOCTL для кадрирования определяют область, в которой фактически снимается изображение. Цель — это изображения, прочитанные приложением или наложенные на графический экран. Их размер (и положение для наложения) согласовываются с пощью ioct команд VIDIOC_G_FMT и VIDIOC_S_FMT.

На устройстве видеовывода источник — это изображения, передаваемые приложением, и их размер вновь согласовывается с помощью ioct команд VIDIOC_G_FMT и VIDIOC_S_FMT или может быть закодирован в сжатом видеопотоке. Цель — это видеосигнал, а IOCTL для кадрирования определяют область, в которую вставляются изображения.

Исходные и целевые прямоугольники определяются даже в том случае, если устройство не поддерживает масштабирование или *ioctl* комады VIDIOC_G_CROP и VIDIOC_S_CROP. Их размер (и положение там, где это применимо) будет зафиксировано в данном случае.

**Внимание**
Все устройства захвата и вывода должны поддерживать *ioctl* команду VIDIOC_CROPCAP, чтобы приложения могли определить, происходит ли масштабирование.

### Структуры обрезки

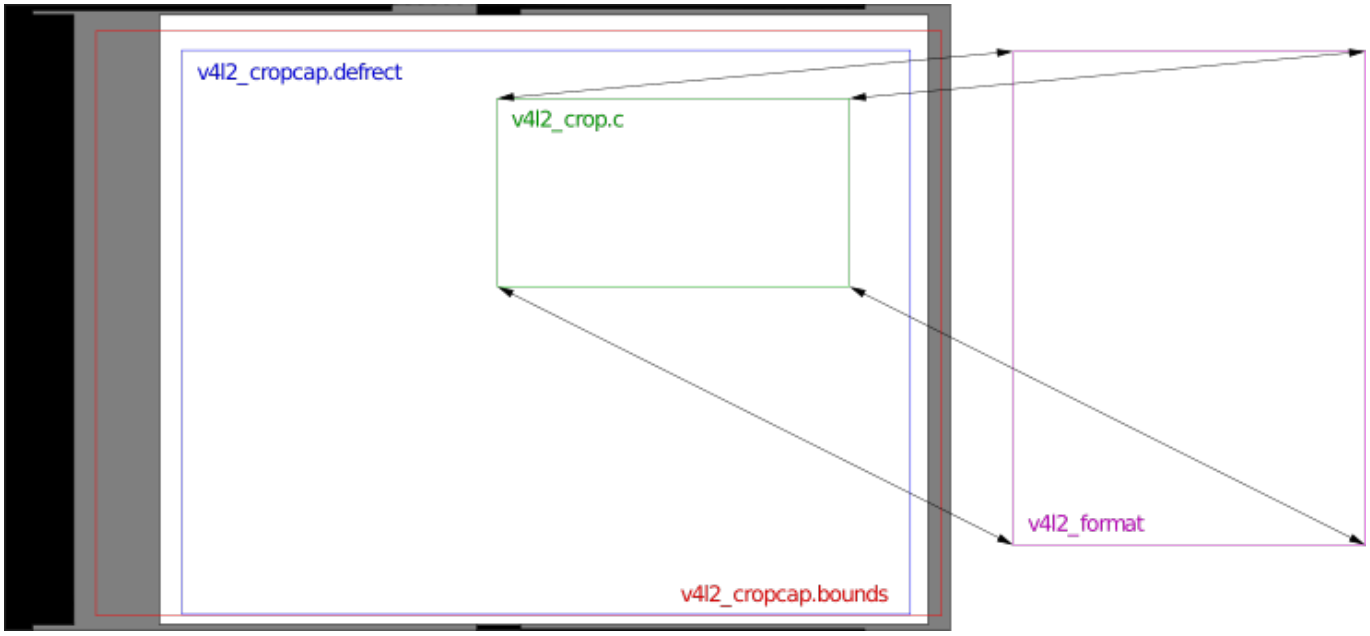


Рис. 1: Кадрирование, вставка и масштабировани

Процесс кадрирования, вставки и масштабирования

Для устройств захвата координаты левого верхнего угла, ширины и высоты области, которые могут быть отобраны для отбора, задается подструктурой границ структуры v4l2_cropcap, возвращаемой *ioctl* командой VIDIOC_CROPCAP. Для поддержки широкого спектра оборудования данная спецификация не определяет начало координат или единицы измерения. Однако, для согласованности, драйверы должны отсчитывать немасштабированные сэмплы относительно 0 (передний фронт импульса горизонтальной синхронизации, см. рис. 4.1). Синхронизация строк). Вертикальные номера строк ITU-R первого поля (см. ITU R-525 нумерация строк для 525 и 625 строк), умноженные на два, если драйвер может захватить оба типа поля.

Верхний левый угол, ширина и высота исходного прямоугольника, то есть область, в которой выполняется сэмплирование задается структурой v4l2_crop, использующей ту же систему координат, что и структура v4l2_cropcap. Приложения могут использовать *ioctl* гкоманды VIDIOC_G_CROP и VIDIOC_S_CROP для получения и установки этого прямоугольника. Он должен полностью лежать в пределах границ захвата, и драйвер может дополнительно скорректировать требуемый размер и/или положение в соответствии с аппаратными ограничениями.

Каждое устройство захвата имеет прямоугольник по умолчанию, предоставленный подструктурой defrect структуры v4l2_cropcap. Центр этого прямоугольника выравнивается по центру активной области изображения видеосигнала и покрывает обасть, котрую разарботчик драйвера считает полной картиной. Драйверы должны сбросить исходный прямоугольник в значения по умолчанию при первой загрузке драйвера, но не позднее не делать этого.

Для устройств вывода эти структуры и *ioctl* используются соответствующим образом, определяя конечный прямоугольник, в котором изображения будут вставляться в видеосигнал.

### Корректировка масштабирования

Видеооборудование может иметь различные ограничения кадрирования, вставки и масштабирования. Оно может масштабировать только вверх или вниз, поддерживать только дискретные коэффициенты масштабирования или иметь различные возможности масштабирования в горизонтальном и вертикальном направлении. Кроме того, оно может не поддерживать масштабирование совсем. В то же время прямоугольник структуры v4l2_crop может быть выровнен , а исходный и конечный прямоугольники могут иметь произвольные верхние и нижние пределы размера. В частности, максимальная ширина и высота в структуре v4l2_crop может быть меньше, чем область границ структуры v4l2_cropccap.bounds . Поэтому, как обычно, драйверы должны корректировать запрошенные параметры настройки и возвращать фактические установленные значения.

Приложения могут сначала изменить исходный или конечный прямоугольник, поскольку они могут предпочесть определенный размер изображения или определенную область в видеосигнале. Если драйвер должен скорректировать настройки с учетом аппаратных ограничений, последний запрашиваемый прямоугольник должен иметь приоритет, и драйвер должен быть предпочесть его регулировки а не первый. Однако *ioctl* команда VIDIOC_TRY_FMT не изменяет состояние драйвера и таким образом корректирует только запрошенный прямоугольник.

Предположим, что масштабирование устройства захвата видео ограничено коэффициентом 1:1 или 2:1 в любом направлении, а размер целевого изображения должен быть кратным 16 × 16 пикселов. Для исходного прямоугольника кадрирования установлены значения по умолчанию, которые также являются верхним пределами в данном примере 640 × 400 пикселей со смещением 0, 0. Приложение запрашивает размер изображения размером 300 × 225 пикселей, предполагая, что видео будет масштабировано с ”полной картины”. Драйвер задает размер изображения максимально близкими значениями 304 × 224, а затем выбирает прямоугольник обрезки, ближайший к запрошенному размеру, т. е. 608 × 224 (224 × 2:1 превысит предел 400). Смещение 0, 0 остается действительным, поэтому не меняется. С учетом стандартного прямоугольника кадрирования, полученного от VIDIOC_CROPCAP, приложение может легко предложить другое смещение для центрирования прямоугольника кадрирования.

Теперь приложение может настаивать на том, чтобы покрыть область изображения с отношением сторон изображения ближе к исходному запросу, поэтому оно запрашивает прямоугольник 608 × 456 пикселей. Текущие коэффициенты масштабирования ограничивают кадрирование 640 × 384, поэтому драйвер возвращает размер кадрирования 608 × 384 и корректирует размер изображения до ближайшего возможного 304 × 192.

### Примеры

Исходные и целевые прямоугольные области должны оставаться неизменными при закрытии и повторном открытии устройства, так как потоковые данные в устройство или из него будут работать без специальной подготовки. Более продвинутые приложения должны перед запуском ввода-вывода убедиться в соответствии параметров.

---

**Замечание**
В следующих двух примерах предполагается устройство захвата видео; Измените V4L2_BUF_TYPE_VIDEO_CAPTURE для других типов устройств.

---

### Пример: Сброс параметров обрезки

```
struct v4l2_cropcap cropcap;
struct v4l2_crop crop;

memset (&cropcap, 0, sizeof (cropcap));
```

```
cropcap.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;

if (-1 == _ioctl_(fd, VIDIOC_CROPCAP, &cropcap)) {
        perror (”VIDIOC_CROPCAP”);
        exit (EXIT_FAILURE);
}

memset (&crop, 0, sizeof (crop));
crop.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
crop.c = cropcap.defrect;

/* игнорировать, если кадрирование не поддерживается (EINVAL). */

if (-1 == _ioctl_ (fd, VIDIOC_S_CROP, &crop) && errno != EINVAL) {
        perror (”VIDIOC_S_CROP”);
        exit (EXIT_FAILURE);
}
```

**Пример: Простое масштабирование вниз**

```
struct v4l2_cropcap cropcap;
struct v4l2_format format;

reset_cropping_parameters ();

/* Масштабирование до 1/4 размера полной картины. */

memset (&format, 0, sizeof (format)); /* defaults */

format.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;

format.fmt.pix.width = cropcap.defrect.width >> 1;
format.fmt.pix.height = cropcap.defrect.height >> 1;
format.fmt.pix.pixelformat = V4L2_PIX_FMT_YUYV;

if (-1 == _ioctl_ (fd, VIDIOC_S_FMT, &format)) {
        perror (”VIDIOC_S_FORMAT”);
        exit (EXIT_FAILURE);
}

/* Мы можем проверить фактический размер изображения, фактический коэффициент масштабирования
или если драйвер может масштабироваться. */
```

**Пример: Выбор области вывода**

**Замечание**
В этом примере предполагается использование устройства вывода.

```
struct v4l2_cropcap cropcap;
struct v4l2_crop crop;

memset (&cropcap, 0, sizeof (cropcap));
cropcap.type = V4L2_BUF_TYPE_VIDEO_OUTPUT;

if (-1 == _ioctl_ (fd, VIDIOC_CROPCAP;, &cropcap)) {
```

```
        perror (”VIDIOC_CROPCAP”);
        exit (EXIT_FAILURE);
}

memset (&crop, 0, sizeof (crop));

crop.type = V4L2_BUF_TYPE_VIDEO_OUTPUT;
crop.c = cropcap.defrect;

/* масштабирование ширины и высоты до 50% от их исходного размера
и центрирование выходных данных. */

crop.c.width /= 2;
crop.c.height /= 2;
crop.c.left += crop.c.width / 2;
crop.c.top += crop.c.height / 2;

/* игнорировать, если кадрирование не поддерживается (EINVAL). */

if (-1 == _ioctl_ (fd, VIDIOC_S_CROP, &crop) && errno != EINVAL) {
        perror (”VIDIOC_S_CROP”);
        exit (EXIT_FAILURE);
}
```

**Пример: Текущий коэффициент масштабирования и пиксельный коэффициент**

**Внимание**
В этом примере предполагается устройство захвата видео.

```
struct v4l2_cropcap cropcap;
struct v4l2_crop crop;
struct v4l2_format format;
double hscale, vscale;
double aspect;
int dwidth, dheight;

memset (&cropcap, 0, sizeof (cropcap));
cropcap.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;

if (-1 == _ioctl_ (fd, VIDIOC_CROPCAP, &cropcap)) {
        perror (”VIDIOC_CROPCAP”);
        exit (EXIT_FAILURE);
}

memset (&crop, 0, sizeof (crop));
crop.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;

if (-1 == _ioctl_ (fd, VIDIOC_G_CROP, &crop)) {
        if (errno != EINVAL) {
                perror (”VIDIOC_G_CROP”);
                exit (EXIT_FAILURE);
        }

        /* Кадрирование не поддерживается. */
        crop.c = cropcap.defrect;
}
```

```
memset (&format, 0, sizeof (format));
format.fmt.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;

if (-1 == _ioctl_ (fd, VIDIOC_G_FMT, &format)) {
        perror (”VIDIOC_G_FMT”);
        exit (EXIT_FAILURE);
}

/* масштабирование, применяемое драйвером. */

hscale = format.fmt.pix.width / (double) crop.c.width;
vscale = format.fmt.pix.height / (double) crop.c.height;

aspect = cropcap.pixelaspect.numerator /
        (double) cropcap.pixelaspect.denominator;
aspect = aspect * hscale / vscale;

/* Устройства после ITU-R BT.601 не захватывают
квадратные пикселы. Для воспроизведения на мониторе компьютера
мы должны масштабировать изображения до этого размера. */

dwidth = format.fmt.pix.width / aspect;
dheight = format.fmt.pix.height;
```

## API для кадрирования, составления и масштабирования

### Введение

Некоторые устройства видеозахвата могут сэмплировать подраздел кадра и уменьшать или увеличивать его до изображения произвольного размера. Далее, устройства могут вставлять изображение в большее. Некоторые устройства видеовывода могут обрезать часть входного изображения, масштабировать его вверх или вниз и вставлять его в произвольной строке развёртки и с горизонтальном смещении в выходной видеосигнал. Эти способности называются обрезкой, масштабированием и композицией.

На устройстве видеозахвата источник является видеосигналом, а целевой объект обрезки определяет область, в которой фактически выполняется сэмплирование. Приемник — это сохранение изображения в буфере памяти. В составной области указывается, какая часть буфера фактически записывается оборудованием.

На устройстве видеовывода источником является изображение в буфере памяти, а целью обрезки — часть изображения, которая должна отображаться на дисплее. Приемник — это экран или графический экран. Приложение может выбрать часть дисплея, в которой должно отображаться изображение. Размер и положение такого окна управляются компоновкой цели.

Прямоугольные области для обрезания и компоновки цели определяются даже в том случае, если устройство не поддерживает ни кадрирование, ни компоновку. Их размер в данном случае (и положение там, где это применимо) будет зафиксировано. Если устройство не поддерживает масштабирование, то прямоугольники кадрирования и компоновки имеют одинаковый размер.

### Выбор цели

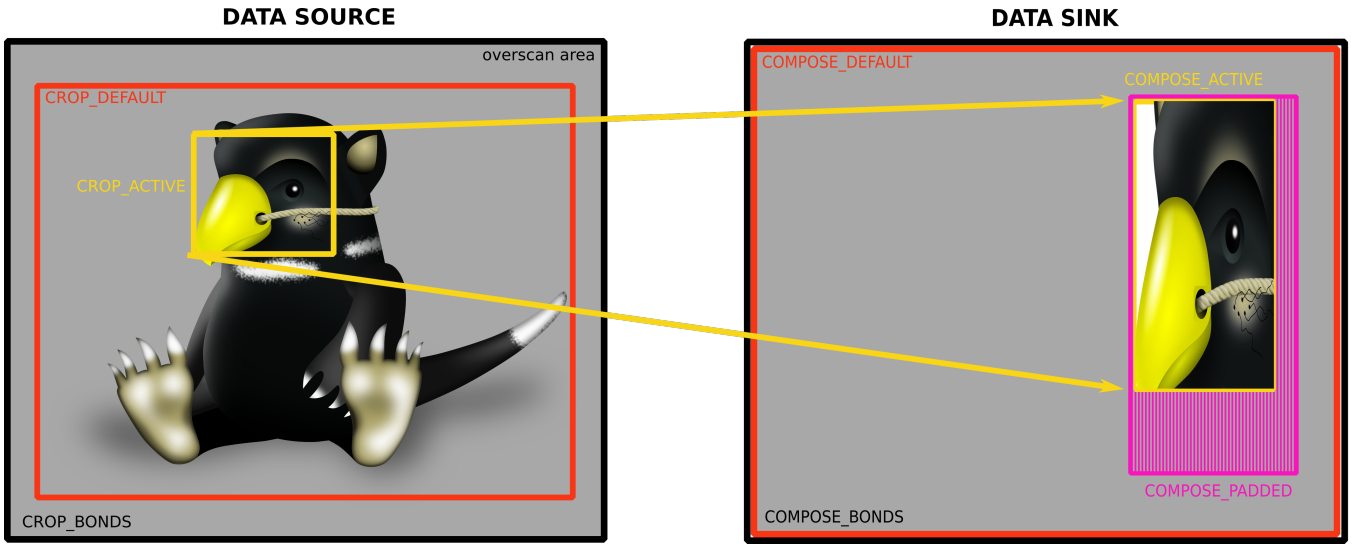


Рис. 2: Цели кадрирования и композици

Цели используются процессами кадрирования, компоновки и масштабирования

Дополнительные сведения см. ”Selection targets”

### Конфигурация

Приложения могут использовать API выбора для выбора области в видеосигнале или буфере, а также для запроса параметров по умолчанию и ограничений оборудования.

Видеооборудование может иметь различные ограничения кадрирования, вставки и масштабирования. Оно может масштабироваться только вверх или вниз, поддерживать только дискретные коэффициенты масштабирования или иметь различные возможности масштабирования в горизонтальном и вертикальном направлениях. Кроме того, оно может не поддерживать масштабирование совсем. В то же время может возникнуть необходимость выравнивания прямоугольника обрезки/композиции, а источник и приемник могут иметь произвольные верхние и нижние пределы размеров. Поэтому, как обычно, драйверы должны корректировать запрошенные параметры настройки и возвращать фактические установленные значения. Приложение может управлять поведением округления с помощью флагов ограничений.

### Конфигурация видеозахвата

См. рисунок ” Cropping and composing targets” для примера выбора целей, доступных для устройств захвата видео. Рекомендуется настроить цель объекты обрезки до цели компоновки.

Диапазон координат от верхнего левого угла, ширины и высоты областей, которые могут быть отобраны, в объекте цели V4L2_SEL_TGT_CROP_BOUNDS. Рекомендуется, чтобы разработчики драйверов поместили верхний/левый угол в позицию (0, 0). Координаты прямоугольника выражаются в пикселях.

Верхний левый угол, ширина и высота исходного прямоугольника, задают область фактической выборка, задаются целевым объектом V4L2_SEL_TGT_CROP. Используется та же система координат, что и в V4L2_SEL_TGT_CROP_BOUNDS. Активная область кадрирования должна полностью

находиться внутри границ захвата. Драйвер может дополнительно скорректировать требуемый размер и/или положение в соответствии с аппаратными ограничениями.

Каждое устройство захвата имеет прямоугольник захвата по умолчанию, предоставленный целью V4L2_SEL_TGT_CROP_DEFAULT. Этот прямоугольник должен располагаться поверх того, что автор драйвера считает полным изображением. Драйверы должны сбросить исходный прямоугольник в значения по умолчанию при первой загрузке драйвера, но позднее не делать этого.

Цели композиции ссылаются на буфер памяти. Пределы координат композиций получаются посредством вызова *ioctl* команды V4L2_SEL_TGT_COMPOSE_BOUNDS. Все координаты выражаются в пикселях. Верхний/левый угол прямоугольника должен располагаться на позиции (0, 0). Ширина и высота равны размеру изображения, установленному *ioctl* командой VIDIOC_S_FMT.

Часть буфера, в которую изображение вставляется оборудованием, управляется целью V4L2_SEL_TGT_
Координаты прямоугольника также выражаются в той же системе координат, что и границы прямоугольника. Прямоугольник композиции должен лежать полностью внутри границ прямоугольника. Драйвер должен корректировать прямоугольник композиции так, чтобы он укладывался в полные границы. Кроме того, драйвер может выполнять другие корректировки в зависимости от аппаратных ограничений. Приложение может управлять поведением округления с помощью флагов ограничений.

Для устройств захвата стандартный прямоугольник для композиции запрашивается с помощью *ioctl* команды V4L2_SEL_TGT_COMPOSE_DEFAULT. Обычно он равен ограничивающему прямоугольнику.

Часть буфера, изменяемая оборудованием, задаётся V4L2_SEL_TGT_COMPOSE_PADDED. Он содержит все пикселы, определенные с помощью V4L2_SEL_TGT_COMPOSE плюс все данные заполнения, измененные оборудованием во время процесса вставки. Все пикселы за пределами этого прямоугольника не должны изменяться оборудованием. Содержимое пикселов, находящихся внутри области вставки, но вне активной области, не определено. Приложение может использовать прямоугольники заполнения и активности, чтобы определить, где находятся битые пиксели, и при необходимости удалить их.

### Настройка видеовыхода

Для устройств вывода целевые объекты и *ioctl* используются аналогично, как для устройств видеозахвата. Прямоугольник композиции ссылается на вставку изображения в видеосигнал. Прямоугольники обрезки ссылаются на буфер памяти. Рекомендуется настроить копозитные цели до целей кадрирования.

Цели обрезки ссылаются на буфер памяти, содержащий изображение, вставляемое в видеосигнал или графический экран. Пределы координат кадрирования определяются с помощью V4L2_SEL_TGT_CROP_BOUNDS. Все координаты выражаются в пикселях. Верхний/левый угол всегда является точкой (0, 0). Ширина и высота равны размеру изображения, установленному *ioctl* командой VIDIOC_S_FMT.

Верхний левый угол, ширина и высота исходного прямоугольника, это область, из которой данные изображения обрабатывается оборудованием, заданы в V4L2_SEL_TGT_CROP. Его координаты выражаются в той же системе координат, что и границы прямоугольника. Активная область кадрирования должна полностью лежать внутри границ обрезки, а драйвер может дополнительно скорректировать требуемый размер и/или положение в зависимости от аппаратных ограничений.

Для устройств вывода прямоугольник кадрирования по умолчанию запрашивается с помощью *ioctl* команды V4L2_SEL_TGT_CROP_DEFAULT. Обычно он равен ограничивающему прямоугольнику.

Часть видеосигнала или графического дисплея, куда устройством вставляется изображение, контролируется целью V4L2_SEL_TGT_COMPOSE. Координаты прямоугольника выражаются в пикселях. Прямоугольник композиции должен лежать полностью внутри границ прямоугольника. Драйвер должен настроить область в соответствии с переделами границ. Кроме того, драйвер может выполнять другие корректировки в зависимости от аппаратных ограничений.

Устройство имеет стандартный прямоугольник композиции, заданный целевым объектом V4L2_SEL_T
Этот прямоугольник должен располагаться поверх того, что автор драйвера считает полным изображением. Рекомендуется, чтобы разработчики драйверов поместили верхний/левый угол в позицию (0, 0). При первой загрузке драйвера, он должен установить активный прямоугольник композиции по умолчанию.

Устройства могут вводить дополнительное содержимое для видеосигнала, кроме изображения из буферов памяти. Она включает границы вокруг изображения. Однако такой заполненный участок является зависимым от драйвера элементом, и не охватываемым данным документом. Разработчикам драйверов предлагается сохранять прямоугольник с заполнения, равным активному. Доступ к цели заполнения осуществляется с помощью идентификатора V4L2_SEL_TGT_COMPOSE_
Он должен содержать все пиксели из цели V4L2_SEL_TGT_COMPOSE.

### Управление масштабированием

Приложение может определить, выполняется ли масштабирование путем сравнения ширины и высоты прямоугольников, полученных с помощью целей V4L2_SEL_TGT_CROP и V4L2_SEL_TGT_COMP
Если они не равны, выполняется масштабирование. Приложение может вычислять коэффициенты масштабирования с использованием этих значений.

### Сравнение со старым API кадрирования

Выбор API был сделан для устранения недостатков предыдущего API, предназначенных для управления простыми устройствами захвата. Впоследствии API кадрирования был подтвержден видео драйверами. Запрос *ioctl* используются для выбора части экрана дисплея, куда будет вставлен видеосигнал. Он должен рассматриваться как обман API, поскольку описанная операция, фактически, является составной. В API выбора проводится четкое разграничение между операциями композиции и обрезки, устанавливая соответствующие целевые объекты. API V4L2 не имеет никакой поддержки для создания и обрезки изображения внутри буфера памяти. Приложение может настроить устройство захвата для заполнения только части изображения, обманывая V4L2 API. Кадрирование меньшего изображения из большего размера достигается путем задания поля bytesperline в структуре v4l2_pix_format. Введение смещения изображения может быть выполнено путем изменения поля m_userptr в структуре v4l2_buffer перед вызовом *ioctl* команд VIDIOC_QBUF, VIDIOC_DQBUF. Следует избегать этих операций , поскольку они не являются портируемыми (не зависящими от порядка байт) и не работают в форматах макроблока и Байера и в буферов mmap. Избарнный API в части конфигурации кадрирования/композиции буферов является четким, интуитивном и портируемым. Далее, с помощью API выбора, вводятся понятия «заполненные поля целевого объекта» и «флаг ограничения». Наконец, в структурах v4l2_crop и v4l2_cropcap нет зарезервированных полей. Поэтому не существует способа расширения функциональности. Новая структура v4l2_selection предоставляет много места для будущих расширений. Разработчикам драйверов рекомендуется реализовывать только API выбора. Предыдущий API кадрирования будет смоделирован с использованием нового.

### Примеры

(предполагается устройство захвата видео; изменение V4L2_BUF_TYPE_VIDEO_CAPTURE для других устройств; изменение целевого объекта на семейство V4L2_SEL_TGT_COMPOSE_* для настройки составной области)

### Пример: Сброс параметров обрезки

```
struct v4l2_selection sel = {
        .type = V4L2_BUF_TYPE_VIDEO_CAPTURE,
        .target = V4L2_SEL_TGT_CROP_DEFAULT,
```

```
};

ret = _ioctl_(fd, VIDIOC_G_SELECTION, &sel);
if (ret) exit(-1);
sel.target = V4L2_SEL_TGT_CROP;
ret = _ioctl_(fd, VIDIOC_S_SELECTION, &sel);
if (ret) exit(-1);
```

Задание области композиции на выходе размера, составляющего более половины предела, размещённой в центре дисплея.

#### Пример: Простое масштабирование вниз

```
struct v4l2_selection sel = {
        .type = V4L2_BUF_TYPE_VIDEO_OUTPUT,
        .target = V4L2_SEL_TGT_COMPOSE_BOUNDS,
};
struct v4l2_rect r;

ret = _ioctl_(fd, VIDIOC_G_SELECTION, &sel);
if (ret) exit(-1);

/* Задание меньшего прямоугольника компорзиции */
r.width = sel.r.width / 2;
r.height = sel.r.height / 2;
r.left = sel.r.width / 4;
r.top = sel.r.height / 4;

sel.r = r;
sel.target = V4L2_SEL_TGT_COMPOSE;
sel.flags = V4L2_SEL_FLAG_LE;
ret = _ioctl_(fd, VIDIOC_S_SELECTION, &sel);
if (ret) exit(-1);
```

Предполагается устройство видеовыхода; изменение V4L2_BUF_TYPE_VIDEO_OUTPUT для других устройств

#### Пример: Запрос коэффициентов масштабирования

```
struct v4l2_selection compose = {
        .type = V4L2_BUF_TYPE_VIDEO_OUTPUT,
        .target = V4L2_SEL_TGT_COMPOSE,
};
struct v4l2_selection crop = {
        .type = V4L2_BUF_TYPE_VIDEO_OUTPUT,
        .target = V4L2_SEL_TGT_CROP,
};
double hscale, vscale;

ret = _ioctl_(fd, VIDIOC_G_SELECTION, &compose);
if (ret) exit(-1);
ret = _ioctl_(fd, VIDIOC_G_SELECTION, &crop);
if (ret) exit(-1);

/* Вычисление коэффициентов масштабирования */
hscale = (double)compose.r.width / crop.r.width;
vscale = (double)compose.r.height / crop.r.height;
```

### Параметры потоковой передачи

Потоковые параметры предназначены для оптимизации процесса захвата видео, а также ввода-вывода. в настоящее время приложения могут запрашивать высококачественный режим съемки с помощью *ioctl* вызова VIDIOC_S_PARM.

Текущий стандарт видео определяет номинальное число кадров в секунду. Если количество кадров меньше, чем должно быть выведено, то приложения могут запросить пропуск или дублирование на стороне драйвера. Это особенно полезно при использовании функции read () или write (), которые не дополняется штампами времени или счетчиками последовательности, а также во избежание ненужного копирования данных.

Наконец, эти *ioctl* можно использовать для определения количества буферов, используемых драйвером в режиме чтения/записи. Сведения о реализации см. раздел, в котором обсуждается функция read().

Для получения и установки потоковых параметров, приложения вызывают *ioctl* команды VIDIOC_G_PA
и VIDIOC_S_PARM соответственно,. Они задают указатель на структуру v4l2_streamparm, которая содержит объединение, содержащее отдельные параметры для устройств ввода и вывода.

Это *ioctl* является необязательными, поэтому драйверы не обязаны реализовывать его. Если так, то они возвращают код ошибки EINVAL.

## Форматы изображений

API V4L2 был разработан в основном для устройств, обменивающихся данными изображения с приложениями. Структуры v4l2_pix_format и v4l2_pix_format_mplane определяют формат и расположение изображения в памяти. Первый используется с плоским API, в то время как последний используется с многослойной версией (см. одно- и много- слойные API). Форматы изображений согласовываются с помощью вызова *ioctl* команды VIDIOC_S_FMT. (пояснения в этом разделе сосредоточены на захвате и выводе видео, для форматов наложения буфера кадра см. также VIDIOC_G_FBUF.)