Нарезка: подробно

English

Раздел режет камерные файлы **без перекодирования**. Потоки копируются как есть, поколение не теряется, метаданные и таймкод переезжают в результат. Это не конкурент монтажной программе, а точный отсекатель лишнего с гарантией, что материал с камеры доедет до монтажа таким, каким снят.

Звуковые файлы режутся тем же разделом и теми же тремя кнопками. Ключевых кадров у звука нет: WAV, FLAC и ALAC режутся **по сэмплу**, сжатый звук по умолчанию — по границам своих пакетов, без перекодирования, а в режиме «Точно» — по сэмплу ценой ещё одного кодирования. Об этом — раздел «Звуковые файлы» ниже.

Всё, что здесь написано, проверено прогоном на живых файлах; числа в тексте — измеренные, а не ожидаемые. Свод измерений — в `MEASUREMENTS_RU.md`, таблицы совместимости — в `TRIM_MATRIX_RU.md`, клавиши плеера — в `MPV_HOTKEYS_RU.md`.

Рабочий цикл

1. Положить дубли или звуковые файлы в Источник (`input`).
2. Выбрать файл бейджем — стрелки слева и справа шагают по папке, кнопка с таблицей показывает, что `ffprobe` знает о файле, ещё до запуска.
3. `Открыть в плеере` — дубль открывается в mpv отдельным окном.
4. Найти точки: `,` и `.` шагают ровно в кадр, `Shift+←` и `Shift+→` — точно на секунду, `l` ставит A, второе нажатие — B. Кусок между точками играет по кругу, и рез слышно до того, как он сделан.
5. `Взять A и B` — обе точки переезжают в поля панели. Или `Отметить IN` и `Отметить OUT`, если удобнее держать руки на панели: они берут позицию, где стоит плеер, и переносят туда же точку петли.
6. Выбрать, что сделать с куском между точками, и нажать `Запустить`.

Точки можно и просто набрать с клавиатуры: `00:01:28,400` или просто секунды. Способ, которому не нужно ничего, кроме цифры, — и он работает, даже если плеер не установлен.

Несколько кусков: кнопка `+`

Справа от последней строки стоит `+`. Он серый, пока в строке не взято всё нужное: для `Оставить` хватит одной точки, для `Вырезать` нужны обе, для `Разделить` — точка. Нажатие добавляет следующую строку и сбрасывает точки A и B в плеере, чтобы следующий кусок искать с чистого листа. Плеер всегда пишет в последнюю строку. Пустая строка просто не учитывается, поэтому удалять её не нужно. Предел — 20 строк и 30 точек разреза.

Справа от OUT в каждой строке `Оставить` и `Вырезать` — длительность куска: сколько останется или сколько будет вырезано. Один IN в `Оставить` считается до конца файла.

Спиннеры в полях точек

В каждом поле точки справа стоят `▲▼`. Поле остаётся одной строкой текста, так что таймкод по-прежнему копируется и вставляется целиком. Стрелка меняет ту секцию, где стоит курсор: часы, минуты, секунды или последнюю — она шагает кадром файла и ложится на сетку кадров (у звукового файла — миллисекундой). Перенос между секундами и минутами — сам собой. Без курсора в поле шагают секунды. Удержание ускоряет: каждый повтор на единицу больше предыдущего — 1, 2, 3… Проверено: при 25p `00:02:01,013` шагом вверх по кадру стало `00:02:01,040`, два шага вниз — `00:02:00,960`; секунда удержания в секундах дала 1, 3, 6, 10, 15, 21.

Плеер и выбранный файл

Стрелки бейджа меняют файл в разделе, а не в плеере. Поэтому кнопки плеера сначала спрашивают у mpv, какой файл он показывает, и переносят точки, только если это файл, выбранный в разделе. Иначе — предупреждение с именем файла в плеере и подсказка нажать `Открыть в плеере`.

`Открыть в плеере` с другим файлом сбрасывает точки A и B: в mpv они — настройка плеера, а не свойство файла, и переживают смену файла. Замерено: после обычной смены файла петля 1–3 с оставалась на новом.

Что сделать с куском между точками

Три кнопки. Больше не нужно: какой край уходит, уже сказано тем, какие поля заполнены.

`Оставить`

Остаётся то, что между точками, остальное уходит.

Заполнено	Что происходит
только IN	уходит начало: хлопушка, подход, фальстарт
только OUT	уходит конец: отход и остановка камеры
обе точки	уходят оба края, остаётся середина
ничего	раздел отказывается: «нечего резать, точка не задана»

`Вырезать`

Наоборот: кусок между точками исчезает, а начало и конец склеиваются в один файл. Оговорка, телефонный звонок, прошедший через кадр ассистент.

Работает в три прохода FFmpeg, ни один из них не перекодирует: голова до IN, хвост от ключевого кадра после OUT, склейка копированием пакетов. Промежуточные куски живут в `workspace` и удаляются, чем бы дело ни кончилось.

Строки — это варианты, а не одна монтажная линейка

В `Оставить` и `Вырезать` каждая строка, добавленная плюсом, — отдельный прогон по тому же дублю и отдельный файл: `имя_keep01`, `имя_keep02` или `имя_cut01`, `имя_cut02`. Номер — номер строки на экране. Строки ничего не знают друг о друге, поэтому могут пересекаться как угодно: один и тот же момент войдёт в два клипа, а из целого дубля в каждом файле вырезан свой кусок. С одной заполненной строкой всё как раньше, без суффикса.

Проверено на 30-секундном дубле (ключевой кадр каждую секунду): строки 5–12,5 и 8–20 в `Оставить` дали два файла по 188 и 300 кадров, в `Вырезать` — 550 и 450 кадров: хвост начинается с ключевого кадра после OUT.

`Разделить`

Ничего не выбрасывается. N точек режут файл на N + 1 частей: `имя_part1`, `имя_part2` и дальше; с десяти частей номер пишется с нулём — `имя_part01`, чтобы в проводнике части шли по порядку. У каждой точки одно поле, `+` добавляет следующее.

Точки берутся только вперёд. `Взять A` и `Отметить здесь` записывают позицию плеера в последнее поле, только если она дальше предыдущих точек; иначе панель говорит, где стоит

плеер и какую точку он не прошёл. Ползунок, случайно отведённый назад, ничего не испортит. Набранные руками точки движок сам ставит по порядку. Если две точки лягут на один ключевой кадр, прогон остановится и назовёт обе — пустую часть он не пишет.

Проверено: четыре точки 4, 11, 17 и 23 с на том же дубле дали пять частей, вместе ровно 750 кадров из 750; две точки на 30-секундном WAV — три части, вместе 1 440 000 сэмплов из 1 440 000.

Каждая точка ложится на **ближайший** ключевой кадр, и этот кадр — общий стык двух соседних частей: одна кончается перед ним, следующая с него начинается. Поэтому части, поставленные подряд, дают исходник кадр в кадр, а звук — сэмпл в сэмпл.

Два независимых реза, «убрать конец» и «убрать начало», встречаются, только если точка сама стоит на ключевом кадре. Замер на 30-секундном дубле (750 кадров, ключевой кадр каждую секунду): точка 10,2 с кладёт так **5 кадров в обе части**, точка 10,7 с оставляет **7 кадров ни в одной**. Разделение режет каждый стык один раз, и во всех пяти проверенных случаях — точка на ключевом кадре, после него, перед ним, две точки, две точки в обратном порядке — хэши кадров частей подряд совпали со списком исходника, а сырые сэмплы частей подряд — с 1 440 000 сэмплами исходника побайтово.

Если обе точки ложатся на один ключевой кадр, средняя часть была бы пустой — раздел отказывается и просит развести точки.

Почему начало ложится на ключевой кадр

Копируемый видеопоток нельзя начать с произвольного кадра: кадры между ключевыми описаны через соседей, и без опорного кадра их нечем декодировать. Поэтому начало куска ложится на ближайший ключевой кадр, а журнал говорит, куда оно попало на самом деле:

```
cut in  00:00:09.000 -> keyframe 00:00:08.400 (-0.6 s)
```

Конец считается иначе — кадрами по точной дроби частоты, поэтому попадает ровно в запрошенный кадр:

```
cut out 00:00:21.000 -> frame 300 of the piece (00:00:12.000)
result  300 frames, 00:00:12.000 (+0.000 s from the plan), timecode 01:00:09:00
```

У выреза середины это правило работает наоборот. Хвост начинается с **первого ключевого кадра после OUT**, а не с ближайшего: начнись он раньше — и часть того, что просили убрать, осталась бы в результате. Насколько больше запрошенного ушло, сказано словами:

```
tail at 00:00:09.000 (first keyframe past the piece, +0.7 s)
[WARNING] 0.7 s more than asked for was removed. Starting earlier would leave
          part of the removed piece in the result.
```

Если ключевой кадр стоит ровно в точке — не уходит ничего лишнего.

Частота берётся у файла, а не додумывается

`ffprobe` отдаёт точную дробь: `24000/1001` для 23,976 и `24/1` для честных 24,000 — камера снимает и так, и так. Дробь идёт в расчёт как дробь, поэтому на часовом файле нет накопленной ошибки.

Если файл снят с **переменной частотой** (телефон пишет 30 в одном поле и что-то среднее в другом), кадровый счёт соврал бы: раздел переходит на счёт по времени и говорит об этом. Конец тогда попадает внутрь пакета, а не в кадр.

Звук

Звук копируется вместе с картинкой. Два случая, где копия недостаточно точна, и раздел переписывает PCM тем же форматом и той же разрядностью — побитово тот же звук, но нарезанный по сэмплу:

- **MXF**: чанки большие, копия притащила бы около 8 мс звука с той стороны реза;
- **склейка** (`Вырезать`): излишек не просто повисает в конце, он приходится ровно на стык. Измерено на дубле 25 к/с: без этого стык уезжал на 72 мс, потом на 16 мс; сейчас — ровно 1 200 000 сэмплов на 25,000 с.

`Аудиоканал` умеет взять первый, второй или свести оба в моно — камера часто пишет пушку и петличку двумя моно-каналами. Выбор канала **пересобирает** звук: для PCM это бесплатно, для сжатого стоит поколения, и раздел об этом предупреждает.

Что ещё едет вместе с материалом

Камера пишет не только картинку и звук, и отброшенное молча человек обнаружил бы уже в монтаже. Поэтому здесь два флажка, оба включены по умолчанию.

`Телеметрия` — поток данных. GoPro пишет туда свою телеметрию. Такой поток держат **MOV и MP4**; **MKV не держит вовсе** («Only audio, video, and subtitles are supported for Matroska»), MXF тоже нет — и раздел говорит об этом до запуска, а не падает на середине.

`Субтитры` — у камерных файлов редкость, но DJI пишет субтитрами данные полёта. Их принимают MOV, MP4 и MKV.

Субтитры не копируются через рез, а собираются для него заново. Копирование пакетов проверяли, и на каждой границе оно ошибалось: при резе по числу кадров FFmpeg выбрасывал реплику, идущую через OUT; реплика, начавшаяся до IN, сохраняла полную длину; а склейка `Вырезать` превращала реплику 0,5–1,5 с в одну микросекунду и сдвигала следующую на секунду раньше.

Поэтому раздел вынимает текст каждой реплики в SRT (на часах самого файла, `-copyts`), обрезает реплики по сохраняемым кускам, сдвигает их на новую шкалу времени и вмешивает готовую дорожку вторым входом в тот же вызов FFmpeg, что режет картинку. Картинка по-прежнему копируется. Что получается:

- реплика, идущая через IN или OUT, остаётся своей частью внутри куска;
- кусок начинается с ключевого кадра перед IN — реплики отсчитываются от него же, вместе с картинкой;
- в `Вырезать` реплики хвоста переезжают вплотную к голове; реплика, разрезанная только вырезанным куском, снова целая;
- в `Разделить` реплика через стык достаётся обеим частям своими половинами.

Замер на 8-секундном дубле 25p (реплики 0,5–1,5, 2,5–3,5 и 5,0–7,5 с): `Оставить` 0–7 дал 0,5–1,5, 2,5–3,5 и 5,0–7,0; `Вырезать` 2–4 — 0,5–1,5 и 3,0–5,5; IN 1,2 (ключевой кадр 1,0) — 0,0–0,5, 1,5–2,5 и 4,0–6,0.

Каждый контейнер пишет текст в своём виде (замер на FFmpeg 8.0.1): MP4 и MOV — `mov_text`, MKV — SRT; другой формат туда копией не входит. Поэтому `mov_text` для MKV и SRT, ASS, WebVTT для MP4 переписываются — слова и тайминги едут, оформление ASS нет, и журнал это говорит. Субтитры картинкой (PGS, DVD) остаются: картинку текста нельзя обрезать по куску, не перерисовав её. Язык дорожки переезжает.

Поток данных (телеметрия) через склейку `Вырезать` не проходит, и журнал говорит об этом до запуска; `Оставить` и `Разделить` его переносят.

Дорожка таймкода — тоже поток данных, и она намеренно остаётся: FFmpeg не может назвать её кодек, MP4 на такой останавливается («Could not find tag for codec none»), а сам таймкод и так переезжает значением. В журнале это сказано словами, с именем дорожки:

```
[WARNING] The data stream (tmcd) has no codec FFmpeg can name, so it cannot be
          copied. A timecode track reads exactly like this and is written as a
          value instead.
```

Что переехало — пишется тоже:

```
carried over: telemetry, subtitles
```

Правило общее для панели и командной строки (`AUDION_TRIM_DATA`, `AUDION_TRIM_SUBTITLES`), и файлы у них по-прежнему совпадают побайтово.

Итог прогона и остановка

Статус внизу окна и отчёт говорят правду о прогоне. `Готово` — только если ни один кусок не упал и ни одна строка не отвергнута. Ошибка FFmpeg и отвергнутые точки (например, IN позже OUT) дают `Ошибка`, остановка кнопкой — `Остановлено`. Результат, который не перезаписан, потому что выключена `Перезапись`, — это пропуск по просьбе, а не ошибка.

Остановка не трогает то, что ещё не начато: файл следующего куска под тем же именем — от прежнего прогона, и при выключенной `Перезаписи` он чужой. Убирается только недописанный файл того куска, который писался в момент остановки.

Таймкод

Монтажёры синхронизируют камеры между собой и с рекордером по таймкоду, поэтому он переезжает в результат и сдвигается на длину отрезанного начала:

```
timecode 01:00:00:00 -> 01:00:09:00
```

`Ноль` вместо `Сдвиг` обнуляет таймкод результата. Если контейнер не умеет держать `tmcd`, раздел скажет об этом до запуска, а не после.

Звуковые файлы

WAV, FLAC, MP3, M4A, Opus, OGG и остальные звуковые форматы стоят в том же бейдже — со значком нот вместо плёнки — и режутся теми же тремя кнопками. Для звукового файла раздел прячет то, что к нему не относится (контейнер, таймкод видео, телеметрию, субтитры), и показывает своё.

Точная резка

Поле «Резка» = «Точно». По умолчанию стоит «Без потерь» (ниже): «Точно» кодирует сжатый звук ещё раз. Здесь звук декодируется, режется **ровно по сэмплу** и записывается в формат, выбранный кнопками ниже. Промежуточного файла нет: декодирование, рез и запись идут одним проходом FFmpeg.

Точка становится ближайшим сэмплом: 10,2345 с на 48 кГц — сэмпл 491 256. Журнал печатает оба числа:

```
Split take.wav into 3 parts on the sample:
    at 00:00:10.235 (sample 491256)
    at 00:00:20.000 (sample 960005)
[1/3] take.wav [part1] | PCM_S24LE 48000 Hz 2 ch | 00:00:30.000 | -> WAV
    piece   00:00:00.000 (sample 0) .. 00:00:10.235 (sample 491256)
    keeping 491256 samples (00:00:10.235)
    writes  PCM_S24LE as it was, sample for sample
    result  491256 samples, 00:00:10.235 (+0 from the plan)
```

`result` — не число из заголовка файла, а столько сэмплов, сколько отдал декодер: заголовки сжатых форматов расходятся со звуком (у AAC на 29 сэмплов меньше, у Opus на 312 больше).

Что получается — по замерам на 30-секундной стереозаписи, 1 440 000 сэмплов:

Источник → результат	Что совпало
WAV → WAV, FLAC → FLAC	кусок **бит в бит** равен тому же отрезку исходника; три части подряд — весь исходник

Источник → результат	Что совпало
MP3, AAC, Vorbis → WAV	кусок бит в бит равен тому же отрезку **декодированного** исходника
что угодно → MP3, Opus	длина точна до сэмпла
что угодно → M4A (AAC)	кодер дописывает хвост: +243 сэмпла к куску в 468 749
что угодно → FLAC, ALAC	бит в бит, пока исходник не глубже 24 бит

Формат куска

Кнопки — тот же набор, что в разделе АУДИО:

Кнопка	Что пишется
`Как есть`	WAV, FLAC, ALAC, WavPack, AIFF — собой, сэмпл в сэмпл. MP3, AAC, Opus, Vorbis — тем же кодеком заново, с битрейтом исходника: **одно поколение сжатия**, и журнал это говорит
`WAV`	декодированный звук с разрядностью исходника; сжатый исходник хранится во float — ровно так, как декодируется
`FLAC`, `ALAC`	без потерь, до 24 бит; float или 32-битный исходник хранится в 24, с предупреждением
`M4A / AAC`	AAC с битрейтом ниже
`MP3`	LAME: VBR V0, Insane 320 или по битрейту; частота выше 48 кГц приводится к 48 через soxr
`Opus`	не больше 256 kbps; Opus работает на 48 кГц, другие частоты приводятся через soxr

Сжатый исходник без потерь — это `WAV` или `FLAC`: всё, что слышно после декодирования, сохраняется, и дальше ничего не теряется.

`Как есть` отказывает там, где FFmpeg не умеет записать формат обратно: для APE, TAK и DSD раздел предлагает FLAC или WAV.

Что едет вместе со звуком

- **Broadcast WAV.** Рекордер пишет в чанк `bext` описание, источник, дату, время и `time_reference` — число сэмплов от полуночи, на котором начался дубль. Это звуковой таймкод, и он сдвигается так же, как таймкод видео: три части разделения в точках 491 256 и

960 005 получили 172 800 000, 173 291 256 и 173 760 005. Без явной просьбы FFmpeg теряет чанк `bext` целиком, а четыре его поля отдаёт под именами, которых сам же при записи не читает, — раздел передаёт их под нужными. Проверено чтением байтов чанка.

- **iXML** — имена дорожек, сцена, дубль — FFmpeg не читает и не пишет. Раздел предупреждает: `bext` и время переезжают, iXML остаётся.
- **Обложка** переезжает в MP3, M4A, FLAC и ALAC; WAV и Opus её не принимают, и журнал об этом говорит.
- Во FLAC, MP3 и M4A `time_reference` переезжает тегом — тоже сдвинутым.
- Из нескольких звуковых дорожек режется первая, и это сказано.

Без потерь — по умолчанию

Ничего не декодируется и не кодируется: пакеты сжатого звука копируются нетронутыми, поколение не тратится. Цена — точка уходит к **ближайшей границе пакета** (MP3 — 1 152 сэмпла, AAC — 1 024, Vorbis — до 1 024, Opus — 960), а стык звучит иначе, чем в исходнике. Насколько — измерено для каждого кодека декодированием частей и сравнением с декодированным исходником:

Кодек	Контейнер	Во что обходится стык
MP3	`mp3`	первые 34 мс после стыка (1 632 сэмпла): фрейм берёт биты у предыдущего; часть до конца файла, начатая не с начала, несёт хвостовое заполнение кодера — меньше фрейма
AAC	`m4a`, `m4b`	первые 21 мс после стыка (1 024 сэмпла): фреймы перекрываются
Vorbis	`ogg`, `oga`	часть копируется на пакет раньше — декодер выбрасывает первый пакет — и дальше бит в бит; стык склейки — 21 мс
Opus	`opus`, `ogg`, `oga`	около полусекунды, пока декодер приходит в себя; часть, начатая не с начала, теряет pre-skip — 312 сэмплов

Без поправок было хуже, и это тоже измерено: MP3 при разделении на три части терял 4 625 сэмплов и сдвигал звук после первого стыка — FFmpeg писал в каждую часть заголовок с задержкой кодера исходника, и декодер срезал её снова; склейка AAC выходила на 1 024 сэмпла длиннее и сдвинутой целиком, пока часы склейки не отвели назад на длину праймера.

WAV, FLAC, ALAC и AIFF режутся по сэмплу и «Без потерь»: терять им нечего, и журнал так и пишет. Кодеки и контейнеры не из таблицы в этом режиме не предлагаются — раздел просит выбрать «Точно».

```
Copying song.m4a packet by packet (Exact cut is off):
    point 00:00:10.235 -> packet boundary 00:00:10.240 (+5.5 ms, sample 491520)
    point 00:00:20.000 -> packet boundary 00:00:20.011 (+10.6 ms, sample 960512)
[1/1] song.m4a | AAC 48000 Hz 2 ch | 00:00:30.000 | -> M4A
    remove  00:00:10.240 (sample 491520) .. 00:00:20.011 (sample 960512)
    keeping 971008 samples (00:00:20.229)
    writes  AAC packets, copied untouched
```

Когда раздел откажет — до запуска

Случай	Что говорит
Камерный RAW в MXF (Sony X-OCN, ARRIRAW)	FFmpeg не читает такой поток вовсе — резать нечего; названа программа производителя
MP4 + ProRes, DNxHR, FFV1, HuffYUV, UTVideo, v210, CineForm	контейнер не примет такое видео; предложены MOV или MKV
MOV + AV1 или VP9	то же самое
MXF + HEVC, AV1, VP9	MXF таких не держит
MXF + сжатый звук	MXF принимает только PCM
MOV + FLAC, Opus, Vorbis	контейнер не примет такой звук
Запрошен канал, которого нет	сказано, сколько каналов в файле на самом деле
Обе точки пусты	«нечего резать: точка не задана»
Разделение: обе точки на одном ключевом кадре или сэмпле	средняя часть была бы пустой — развести точки
Звуковой файл `Как есть`: APE, TAK, DSD и другие, которые FFmpeg не записывает	предложены FLAC или WAV
MP3 из файла с тремя и больше каналами	MP3 держит один или два канала
Копирование пакетов для кодека или контейнера, которые не мерили	просит включить точную резку

Читаемые расширения видео: `avi`, `m2ts`, `m4v`, `mkv`, `mov`, `mp4`, `mts`, `mxf`, `ts`. Звука: `aac`, `ac3`, `aif`, `aiff`, `amr`, `ape`, `au`, `awb`, `caf`, `dff`, `dsf`, `dts`, `eac3`, `flac`, `m4a`, `m4b`, `mka`, `mp3`, `mpc`, `mpc8`, `oga`, `ogg`, `opus`, `snd`, `spx`, `tak`, `tta`, `w64`, `wav`, `wv`.

Плеер

mpv запускается **отдельной программой** и слушает команды по каналу — так же, как здесь используется `ffmpeg.exe`. Ставится один раз: `install\Install-Portable-mpv.cmd`.

Что это даёт:

- точная перемотка: попадание в запрошенный кадр, а не в ближайший ключевой;
- шаг ровно в кадр в обе стороны;
- воспроизведение **назад** (`b`);
- петля A/B — она и есть рез, слышимый до реза;
- те же камерные форматы, что читает сама программа: внутри mpv тот же FFmpeg.

Окно можно держать на втором мониторе; оно переживает смену файла — переоткрытие это команда, а не перезапуск. Если закрыть его руками, панель это увидит и кнопка `Открыть в плеере` поднимет его заново. Вместе с программой плеер закрывается сам.

Установщик кладёт рядом с плеером текст лицензии (`LICENSE.GPL`, `COPYRIGHT` — в архиве сборки их нет) и исходники, из которых он собран: сам mpv по коммиту сборки, встроенный в него FFmpeg — это **другой** FFmpeg, не тот, которым работает программа, — и сборочные скрипты, где перечислены версии всех остальных библиотек. Около 24 МБ; что откуда взято, записано в `Tools\mpv\SOURCES.txt`.

Настройки лежат в `config\mpv`. Там общая конфигурация плеера программ Audion, а ниже — короткий блок поправок для реза: интерполяция выключена (она рисует кадры, которых в файле нет, и точку по ним ставить нельзя), перемотка всегда точная, файл открывается туда, куда его открыли.

Клавиши — в `MPV_HOTKEYS_RU.md`.

Из командной строки

```
Scripts\ff-trim-start.cmd     убрать начало
Scripts\ff-trim-end.cmd       убрать конец
Scripts\ff-trim-both.cmd      оставить середину
Scripts\ff-trim-middle.cmd    вырезать середину и склеить
```

Точки задаются переменными окружения: `AUDION_TRIM_FILE`, `AUDION_TRIM_START`, `AUDION_TRIM_END`; контейнер, звук, канал и таймкод — `AUDION_TRIM_CONTAINER`, `AUDION_TRIM_AUDIO`, `AUDION_TRIM_AUDIO_CHANNEL`, `AUDION_TRIM_TIMECODE`.

Обёртки режут видео; звуковые файлы режутся из панели.

Арифметика общая с панелью, и это проверено, а не заявлено: один и тот же вырез, сделанный обёрткой и панелью, дал **побайтово одинаковые файлы**.

Соседний раздел: разворот без потерь

Разворот картинки живёт в разделе УПАКОВКА, потому что это не рез, а смена того, как файл показывать. Камеру повернули набок ради вертикального кадра, а она записала обычный горизонтальный файл и ничего об этом не сказала — здесь разворот записывается в контейнер, и монтажка открывает клип правильно.

Пиксели не трогаются: файл 6 085 307 байт стал 6 085 304, хэши кадров совпали, декодер отдаёт 540×960 вместо 960×540. Хранят разворот MP4, MOV и MKV; MXF нет, и раздел отказывает вместо того, чтобы выдать неповёрнутый файл. Предлагается для камерных H.264 и HEVC.