

Обрезка: матрица форматов, методы проверки и грабли FFmpeg

Про интерфейс в этом документе. Точки реза здесь находят в плеере trpv, который открывается кнопкой из раздела, и переносят в поля кнопками — либо вводят таймкодом с клавиатуры. Форма волны и прослушивание кадра, о которых говорится ниже, относятся к панели, отложенной до отдельного этапа: движок и измерения от неё не зависят и остаются в силе.

Справочник по разделу «Обрезка». Всё, что здесь написано, **измерено**, а не взято из документации: каждая ячейка таблиц — реальная попытка записи с `-c copy` и чтением результата обратно, каждое число синхрона — замер по кадрам и сэмплам. Числа сняты на **FFmpeg 9.0.1** и переснятые на 8.0.1 совпали до ячейки. Какая сборка попадёт в поставку — решает установщик по версии драйвера NVIDIA (на большинстве машин это ветка 8.x), поэтому обе проверены.

Замысел раздела и решения — в [PLAN_TRIM_RU.md](#) и [DECISIONS.md](#). Здесь — фактура по обрезке; полный свод измерений, включая пересчёт частоты и пределы самого FFmpeg, — в [MEASUREMENTS_RU.md](#).

1. Что раздел делает

Обрезка копирует потоки (`-c:v copy`) и не перекодирует картинку. Отсюда три свойства, которые определяют всё остальное:

- частота берётся у файла как есть: `ffprobe` отдаёт точную дробь (`24000/1001` для 23,976 и `24/1` для честных 24,000), и ничего не домысливается — камера снимает и то, и другое;
- начало куска ложится на **ключевой кадр** (иначе поток нечем начать);
- конец режется **числом кадров** по точной дроби частоты;

- звук копируется как есть, кроме двух случаев (выбор канала и PCM в MXF), где он переписывается сэмпл-в-сэмпл без потери поколения.

Режим	Что остаётся	Точек
Убрать начало	всё после IN	одна (IN)
Убрать конец	всё до OUT	одна (OUT)
Оставить середину	кусок между IN и OUT	две
Разделить надвое	оба куска, ничего не выбрасывается	одна (CUT)

1.1. Панель: чем ищут точку

Раздел ищут не глазами по цифрам, а слухом и картинкой, поэтому панель — часть инструмента, а не оформление.

Полоса формы волны одна — на весь файл; десятисекундное окно убрано, потому что не поспевало за ней и расходилось с ней. Под полосой фейдер, под ним транспорт.

Звук под курсором. Фейдер звучит: на остановке руки проигрывается кусок ровно **в один кадр**, выровненный по границе кадра, — 41,7 мс на 23,976, 40 мс на 25. Так делает монтажная программа: слышен кадр, на котором стоишь, а не секунда после него. Кнопкой «Кусок прослушивания» берётся 3 кадра (125 мс), если одного мало, чтобы узнать слог.

Откуда так быстро. Звуковая дорожка декодируется в память целиком (моно 24 кГц): полчаса — 82 МБ и 2 секунды, после чего любой кусок стоит **0,84 мс** и не требует ни процесса, ни обращения к диску. Выше трёх часов дорожка целиком не держится, и работают блоки по две минуты; ими же отвечают первые секунды, пока полная дорожка грузится в фоне.

Непрерывное прослушивание. ► играет от точки до конца файла через `ffplay` (без окна, только звук), II останавливает и **запоминает место** — следующее ► продолжает оттуда. Во время игры едет белая линия по волне, а фейдеры стоят: точка принадлежит оператору, а не проигрывателю. Сдвинули фейдер — запомненное место забывается, потому что звук из старого места врал бы о положении головы.

Как нарисована волна. Три правила, каждое проверено сравнением:

- **шкала — кубический корень.** На одних и тех же десяти секундах речи `lin` даёт ниточку со всплесками, `sqrt` — тонкую форму, `log` — сплошную заливку без структуры, `cbirt` — тело, в котором читается фраза;
- **нормализация по пику всей дорожки**, а не окна. Пик окна перерисовывал бы масштаб на каждом движении: тихое место раздувалось бы до полной высоты, и два места одной записи стало бы нельзя сравнить. Пик, а не RMS: RMS заглаживает атаки, а рез целятся именно в них;
- **воздух у краёв:** полоса рисуется короче своей высоты и дополняется прозрачным, чтобы пик 0 dB останавливался за 3–4 px до границы.

Скорость. Кадр превью декодируется аппаратно там, где это умеют QSV, NVDEC и AMF (h264, hevc, av1, vp9, vp8, vvc, vc1, mpeg1/2/4, mjpeg) — 4K H.264 занимает 0,48 с против 0,85 с на процессоре. Монтажным кодекам ускорение не предлагается: аппаратного пути для ProRes нет ни у одного производителя, а попытка его просить ломает цветовой обход, которым превью ProRes 4444 только и держится.

2. Контейнер × видеокодек

Пары «видео + звук, который контейнер точно принимает», запись `-с copy`, затем чтение обратно и сверка кодека.

Видео	MP4	MOV	MKV	MXF	TS	M2TS
H.264	да	да	да	да	да	да
HEVC	да	да	да	нет	да	да
ProRes	нет	да	да	да	нет	нет
DNxHD/DNxHR	нет	да	да	да	нет	нет
MPEG-2	да	да	да	да	да	да
AV1	да	нет	да	нет	нет	нет
VP9	да	нет	да	нет	нет	нет

Что из этого следует для практики:

- **ProRes и DNxHR живут в MOV, MKV и MXF.** MP4 их не примет — попытка даёт «Could not find tag for codec» и файл нулевой длины, поэтому такая пара отклоняется до запуска.
- **HEVC не идёт в MXF.** Камерный HEVC (Canon, Sony) в MXF не переложить без перекодирования — это отдельная операция, не обрезка.
- **AV1 и VP9 не идут в MOV.** Раньше код этого не знал; теперь знает.

3. Контейнер × аудиокодек

Звук	MP4	MOV	MKV	MXF	TS	M2TS
PCM 16 LE	да	да	да	да	нет	нет
PCM 16 BE (twos, Canon)	да	да	да	нет	нет	нет
PCM 24 LE	да	да	да	да	нет	нет
PCM 32 float	да	да	да	нет	нет	нет
AAC	да	да	да	нет	да	да
AC-3	да	да	да	нет	да	да
E-AC-3	да	да	да	нет	да	да
MP3	да	да	да	нет	да	да
FLAC	да	нет	да	нет	нет	нет
ALAC	да	да	да	нет	нет	нет
Opus	да	нет	да	нет	да	да

Главное:

- **MXF принимает только PCM, и только little-endian.** Это важно для Canon: камера пишет `pcm_s16be`, и при выводе в MXF звук переписывается в `pcm_s16le` — те же сэмплы, порядок байт, который требует контейнер.
- **MPEG-TS не принимает PCM.** Материал с AVCHD-карты со звуком PCM в TS не переложить; MOV возьмёт как есть.
- **MOV не принимает FLAC и Opus**, хотя MP4 принимает оба.

4. Точность реза

4.1 Кадры

Проверено на восьми синтетических источниках и одном камерном. Во всех случаях число кадров совпало с планом до единицы.

Источник	Запрос	Кадров	Расхождение
ProRes 25p	3 → 9	150	0
HEVC long-GOP 29.97	12 → 30	600	0 (после добора по декодируемым)
H.264 all-intra 23.976	3 → 9	144	0
29.97 drop-frame	5 → 15	330	0
59.94 drop-frame	3 → 9	360	0
MPEG-TS (старт 1.44 с)	2 → 8	300	0
4 канала PCM	2 → 8	150	0
Canon HEVC 4:2:2 10 бит	3 → 10	166	0
ARRI ProRes 4444 XQ 24,000	2 → 8	144	0
VFR (30 → 17.6)	4 → 16	по времени	+1.07 с, объявлено

4.2 Синхрон звука

Метка есть в обеих дорожках одновременно: белый кадр в начале каждой секунды и щелчок в тот же момент. Отклонение считается **от источника**, а не абсолютное.

Контейнер	Изменение против источника
MP4	0...2 мс
MOV	0...5 мс
MKV	1...3 мс
MXF	0...5 мс (после перезаписи PCM; до неё было 9...18)

На камерном файле звук найден в исходнике **дословно**: кусок начинается на 48 сэмплов (1 мс, сороковая доля кадра) после точки реза.

4.3 Метаданные

Проверено на файле с поворотом, двумя дорожками и тегами проекта:

Что	MP4	MOV	MKV
Поворот (<code>rotation=90</code>)	да	да	да
Язык дорожек (<code>rus</code> , <code>eng</code>)	да	да	да
Теги проекта (title, artist)	да	да	да
Камерные теги (make, model, brands)	да	да	частично
Таймкод как дорожка <code>tmcd</code>	да	да	нет (текстовый тег)

Камерные теги переносит `-movflags use_metadata_tags`; без него MP4-мультиплексор пишет свои `isomiso2mp41` и теряет `make`, `model` и бренд вроде `mp42hvc1CAEP`.

5. Грабли FFmpeg, найденные измерением

Каждая стоила отдельного разбора; ни одна не очевидна из документации.

№	Грабля	Проявление	Как обойдено
1	<code>-ss</code> до <code>-i</code> сбрасывает время	<code>-to</code> после него считает от нового нуля	хвост уезжает как длительность/кадры, не как абсолютная точка
2	<code>round(fps)</code> на дробных частотах	4 минуты на 29.97 превращаются в 240.27 с	частота живёт как <code>Fraction</code> от <code>ffprobe</code> до <code>ffmpeg</code>
3	Округление конца «к ближайшему»	теряется кадр, стоящий на 29.9966 при запросе 30.0	конец округляется вверх
4	<code>-t</code> при копировании	перебор на 1–3 кадра на потоке с В-кадрами	хвост задаётся <code>-frames:v</code>
5	<code>-frames:v</code> точен только с точным <code>-ss</code>	при <code>-ss</code> мимо ключевого кадра кусок короче на зазор	seek всегда на вычисленный ключевой кадр

№	Грабля	Проявление	Как обойдено
6	HEVC с B-пирамидой	муксер пишет на 2–4 кадра меньше, чем просили	недобор измеряется и рез повторяется один раз с поправкой
7	<code>-avoid_negative_ts make_zero</code>	видео уезжает на 0.1 с от звука	не используется
8	<code>start_time</code> $\neq 0$ (MPEG-TS 1.44 с)	все точки смещаются на эту величину	смещение снимается с ключевых кадров и возвращается при seek
9	<code>avg_frame_rate</code> на VFR	счёт кадров покрывает не тот отрезок (12 с $\rightarrow$ 16 с)	VFR определяется и режется по времени, с предупреждением
10	<code>-n</code> на существующий файл	«already exists», код возврата 0	цель проверяется до запуска
11	PCM в MXF при <code>-c copy</code>	384 лишних сэмпла в начале (8 мс)	PCM переписывается сэмпл-в-сэмпл (LE)
12	<code>-map_metadata 0</code> в MP4	камерные теги и бренды теряются	<code>-movflags use_metadata_tags</code>
13	<code>`ran=mono</code>	<code>c0=c1`</code> на моно	пишет тишину, код возврата 0
14	MXF + сжатый звук	«Could not write header», код 4294967274	отклоняется до запуска с объяснением
15	Пакет $\neq$ кадр	муксер пишет 600 пакетов, декодер отдаёт 597: хвостовые B-кадры ссылаются за границу реза	недобор ловится по длительности потока и добирается по декодируемым кадрам
16	Камерный RAW в MXF	<code>codec_name</code> пуст, размеры нулевые, «dimensions not set», файл нулевой длины	отклоняется до запуска: у FFmpeg нет декодера X-OCN и ARRIRAW
17	Предупреждение ломает разбор JSON	<code>stderr</code> подмешивается в <code>stdout</code> , и <code>json.loads</code> падает на строке «could not resolve file descriptor strong ref»	пробы читаются с разделёнными потоками

6. Как всё это меряется

Команды, которыми получены числа выше. Их же стоит использовать при следующей проверке — иначе результаты не сравнить.

Кадры. Считать пакеты, а не декодировать: результат тот же, время — в сорок раз меньше (0.14 с против 5.7 с на четырёх минутах 59.94).

```
ffprobe -v error -select_streams v:0 -count_packets -show_entries  
stream=nb_read_packets -of csv=p=0 FILE
```

Ключевые кадры вокруг точки, без сканирования всего файла:

```
ffprobe -v error -select_streams v:0 -skip_frame nokey -read_intervals  
"START%+WINDOW" -show_entries frame=pts_time -of csv=p=0 FILE
```

Синхрон. Нужен источник с меткой в обеих дорожках: белый кадр и щелчок в один момент. Видео читать через `signalstats` с `-fps_mode passthrough` — иначе фильтр продублирует кадры под свои часы и сдвинет времена:

```
ffmpeg -v info -i FILE -map 0:v:0 -vf  
"signalstats,metadata=print:key=lavfi.signalstats.YAVG" -fps_mode passthrough -f null  
-
```

Звук — по сэмплам, а не через `silencedetect`: детектор работает окнами по 20 мс, а мерить приходится единицы миллисекунд.

```
ffmpeg -v error -i FILE -map 0:a:0 -ac 1 -ar 48000 -f s16le -
```

Побитовая проверка копирования. Найти звук куска внутри звука источника: если он там есть дословно, звук скопирован, а позиция даёт точное смещение реза.

Совместимость. Записать одну секунду `-c copy` в контейнер и **прочитать обратно**.

Партнёрский поток подбирать под контейнер: MXF не примет AAC, TS не примет PCM, и без этого ячейка покажет отказ партнёра, а не проверяемого кодека.

Ловушки самих измерений

Три ошибки, на которых я уже спотыкался — стоят того, чтобы их не повторять:

1. **Сравнивать «первую найденную вспышку» с «первым найденным щелчком».** Если один поток открывается меткой, а другой нет, детекторы описывают разные события — получается «рассинхрон 107 мс», которого нет.
2. **Мерить синхрон детекторами FFmpeg.** `blackdetect` работает целыми кадрами, `silencedetect` — окнами; вместе они дают 40 мс погрешности, то есть больше, чем измеряемая величина.
3. **Проверять совместимость по коду возврата.** Файл может записаться и не читаться обратно; TS «принимает» ProRes ровно так.

6.1. Камерный RAW: чего инструмент не может

Два формата, которые FFmpeg не открывает **ни в какой версии** — проверено и на 8.0.1, и на 9.0.1:

Файл	Что внутри	Что видит ffprobe
Sony BURANO X-OCN LT	X-OCN в MXF OP1a	поток без имени кодека, <code>width=0</code> , звук <code>pcm_s24le</code> читается
ARRI Alexa Mini <code>.mxf</code>	ARRIRAW в MXF	то же самое

Попытка простого копирования даёт «dimensions not set» и файл нулевой длины, поэтому такие исходники отклоняются до запуска с прямым объяснением. Резать их можно в родном ПО производителя (Sony Catalyst, ARRI Reference Tool) или после транскода.

Тот же материал в **MOV** режется полноценно: ARRI ProRes 4444 XQ (2944×2160, 12 бит, 5 дорожек PCM) прошёл кадр в кадр, вместе с таймкодом 02:00:11:11 → 02:00:13:11.

7. Чего мы не проверяли

Список открытый — чтобы никто (включая автора) не решил, что матрица закрыта:

- камеры, кроме Canon HEVC и ARRI ProRes: Sony XAVC, Panasonic, BMPCC, GoPro с телеметрией, AVCHD прямо с карты;

- файлы **длиннее** часа (час проверен: 50,7 ГБ, рез кадр в кадр на 58-й минуте и на последних секундах — см. `MEASUREMENTS_RU.md`);
- нехватка места на диске в середине записи;
- источник на сетевом пути или занятый другой программой;
- две операции одновременно;
- звук 96 кГц и 32 бита float в реальном материале;
- звук, идущий за последним кадром на границу пакета: после добора хвоста он уходит вперёд на 0,1 с и об этом сообщается, но на слух это не проверялось.