Audion Media Tools — принятые решения

English · О программе · Руководство · Замеры

Содержание

- Точность реза
- Синхрон
- Контейнеры, метаданные, пределы
- Работа и форма
- Отказы и честность
- Версии
- Окно

Что решено и почему — с датами и с числами, на которых решение стоит. Где измерение потом опровергло вывод, это тоже записано.

Точность реза

Разделение решает каждый стык один раз — 14.09.2026

Разделение было двумя независимыми резами — «убрать конец» до точки и «убрать начало» от неё, — и к ключевому кадру прыгало только начало второй части. Части встречались, лишь если точка сама стояла на ключевом кадре. Замер на 750 кадрах с ключевым каждую секунду: точка 10,2 с положила 5 кадров в обе части, точка 10,7 с оставила 7 кадров ни в одной.

Теперь `plan_split` один раз кладёт каждую точку на ближайший ключевой кадр, и этот кадр — общий стык: одна часть кончается перед ним, следующая с него начинается. Одна точка IN даёт две части, IN и OUT — три. Длина части считается через `round`, а не `ceil`: метка ключевого кадра уже стоит на границе кадра, а на 23,976 кадр 240 читается как 10,010010 с — 240,00024

кадра, и `ceil` положил бы его в обе части. Пять случаев проверены хэшами каждого декодированного кадра и сырыми сэмплами: части подряд — это исходник.

Звуковой файл режется по сэмплу, пурист может копировать пакеты — 14.09.2026

У звука нет ключевых кадров, и точке некуда прыгать: она становится ближайшим сэмплом, кусок декодируется и режется `atrim` по номерам сэмплов, а записывается как был или в формат из набора «Аудиопотоков». Декодирование здесь ничего не стоит: кусок бит в бит равен тому же отрезку полностью декодированного исходника, замерено на WAV, FLAC, MP3, AAC и Vorbis. Без потерь для сжатого источника — WAV или FLAC; «Как есть» кодирует его ещё раз и говорит об этом.

Чекбокс «Точная резка» можно снять. Тогда пакеты сжатого звука копируются нетронутыми, точка уходит к границе пакета, а стык обходится в то, что измерено для кодека, — 34 мс у MP3, 21 мс у AAC и Vorbis, около полусекунды у Opus, — и журнал это пишет. Два дефекта контейнера найдены и убраны, а не описаны: части MP3 несли заголовок с задержкой кодера исходника и сдвигали звук после каждого стыка, а склейка AAC проигрывала праймер, который прятал список правок. Кодеки и контейнеры, которые не мерили, в этом режиме не предлагаются.

`time_reference` в Broadcast WAV сдвигается вместе с резом, как таймкод видео. FFmpeg без просьбы теряет чанк `bext`, а четыре его поля отдаёт под именами, которых сам же при записи не читает, — раздел передаёт их под нужными. iXML FFmpeg записать не может вовсе, и журнал говорит, что он остаётся.

Частоту берут у файла, не угадывают — 26.08.2026

`ffprobe` сообщает точную дробь, и ничего из неё не выводится. Файл, снятый на 23,976, объявляет `24000/1001`; снятый на честные 24,000 — `24/1`. Оба читаются как написано.

Это отменяет решение от 25.08, по которому каждая целая частота NTSC читалась как её дробный близнец — на вере, будто контейнеры округляют. **Они не округляют.** ARRI Alexa Mini, снимающая настоящие 24,000 (260 кадров на 10,8333 с, измерено), объявляет `24/1`, и множитель превращал верную частоту в неверную: пять кадров расхождения за четыре минуты — та самая ошибка ×1,001, ради предотвращения которой опция и существовала, только направленная в другую сторону.

Галочка убрана из панели и из обоих входов. Частота, которую не удалось прочитать, отвергается, а не заменяется догадкой.

Начало прыгает на ключевой кадр, конец считается кадрами — 25.08.2026

Начало копируемого куска может быть только ключевым кадром, поэтому оно прыгает — и настоящая точка печатается вместе с расстоянием до запрошенной. У конца такого ограничения нет: он идёт как `-frames:v N`, где N берётся из точной дроби.

`-t` измерен и отвергнут: на потоке с B-кадрами он перебирает на два-три кадра. `-avoid_negative_ts make_zero` отвергнут тоже — вместе с перемоткой до `-i` он запускает видео на 0,1 с позже звука.

Точки реза — это реальное время, таймкод остаётся меткой — 25.08.2026

Точка вводится и показывается реальным временем (`00:01:28,4`), а не таймкодом. На частотах NTSC они расходятся по определению: `00:04:00:00` без пропуска кадров на 29,97 — это кадр 7200, то есть 240,24 секунды настоящего времени. Инструмент, принимающий одно за другое, уезжает на `1001/1000` — измеренный дефект Shutter Encoder, с которого началась эта работа: он держится с версии 19.1, на 20.2 никуда не делся, четыре секунды на час.

Таймкод в операции остаётся, но только как метка, которую надо сдвинуть. Каждый перевод между секундами и кадрами идёт через точную дробь.

Рез идёт только по ключевым кадрам — 24.08.2026

Видео копируется, рез ложится на ближайший ключевой кадр. Покадровая точность — она требует перекодирования всего файла — намеренно вне области.

Обещание инструмента: быстро и без потерь. Кнопка, которая иногда работает две секунды, а иногда двадцать минут — в зависимости от галочки, которую никто не читает, — это обещание ломает. Покадровый рез умеет дюжина других программ; быть быстрым и честным об этом — более полезная ниша.

Следствие: интерфейс показывает **настоящую** точку реза, а не запрошенную — ближайший ключевой кадр со смещением словами («00:01:28,4, −1,6 с»). Тихая поправка хуже медленного реза.

Движок отвечает матрице источников, а не одному ролику — 25.08.2026

Рез проверен на россыпи камерных источников, а не на том единственном файле, на котором писался: ProRes с PCM в MOV, HEVC с длинной группой кадров, H.264 без межкадрового сжатия,

29,97 с пропуском кадров, переменная частота, четыре звуковых канала, MPEG-TS и имя с пробелами и кириллицей. Пять дефектов, которые могла показать только такая россыпь:

Кадр, стоящий чуть внутри конечной точки, принадлежит куску. Конечный индекс округляется вверх, а не к ближайшему: на 29,97 тридцатая секунда — это кадр 899,1, а кадр 899 стоит на 29,9966 — внутри. Округление к ближайшему молча его теряло.

`-frames:v` точен только тогда, когда перемотка попала в настоящий ключевой кадр. В любом другом положении FFmpeg считает кадры от предыдущего ключевого и отбрасывает лишние — кусок выходит короче ровно на этот зазор. Служба всегда перематывает на вычисленный ключевой кадр.

HEVC с пирамидой B-кадров пишет на два-четыре кадра меньше запрошенного. Недостача устойчива для потока, но не следует из `has_b_frames`, поэтому она измеряется, а не угадывается: кадры считаются, и если каких-то нет — рез повторяется один раз с запросом, поднятым на величину недостачи. Запрос `N + d` даёт ровно `N`.

Переменная частота запрещает арифметику кадров. `avg_frame_rate` — среднее, и счёт по нему покрывает не тот отрезок: двенадцать запрошенных секунд вышли шестнадцатью. Такой источник узнаётся по расхождению `r_frame_rate` и `avg_frame_rate`, режется по времени и помечается в журнале.

Отсчёт времени не всегда начинается с нуля. MPEG-TS обычно стартует с 1,44 с. Оператор вводит позиции от первого кадра, FFmpeg считает по собственной шкале потока. Начало вычитается из найденных ключевых кадров и прибавляется обратно при перемотке — в службе, в командном запуске и в предпросмотре.

Синхрон

Синхрон измеряется, а не предполагается — 25.08.2026

Рез, сдвинувший один поток и не сдвинувший другой, — отказ, который обнаружится последним: на монтаже, а не в журнале. Поэтому он проверяется маркером, существующим в обоих потоках в один и тот же миг: белый кадр в начале каждой секунды и щелчок 20 мс там же.

Прочитать эти маркеры правильно вышло со второй попытки. `blackdetect` и `silencedetect` работают целыми кадрами и окнами, что кладёт под измерение порог в 40 мс, а сравнение «первой замеченной вспышки» с «первым замеченным щелчком» сравнивает разные события, если один поток открывается маркером, а другой нет. Честный способ читает видео через

`signalstats` с `fps_mode passthrough` — чтобы ни один кадр не дублировался под часы фильтра — и связывает каждую вспышку с её собственным щелчком по настоящим отметкам времени файла.

Измеренное так, относительно собственного смещения источника:

контейнер	расхождение с источником
MP4	0…2 мс
MOV	0…5 мс
MKV	1…3 мс
MXF	9…18 мс

Покрыты все режимы — убрать начало, убрать конец, оставить середину, разделить надвое — на AAC, на 24-битном PCM и на ProRes, включая путь с выбором канала, который пересобирает звук. Сдвигает только MXF: его муксер равняет звук по собственным единицам монтажа, это меньше половины кадра и объявлено предупреждением.

Прежнее предупреждение утверждало, что MKV гонит звук вперёд на длину звукового пакета. **Точное измерение это опровергло** — MKV сдвигает начало обоих потоков вместе, а это не рассинхрон, — и предупреждение снято, а не оставлено правдоподобной страшилкой.

Контейнеры, метаданные, пределы

Настоящий камерный файл нашёл то, чего не показали синтетические — 26.08.2026

Canon `MVI_*.MP4` — HEVC Rext 4:2:2 10 бит UHD на 23,976, звук LPCM `pcm_s16be`, таймкод 15:58:57:17 — прошёл через все режимы. Кадры везде точны (242 / 240 / 166 и разделение 144 + 170, покрывающее все 314 кадров источника), таймкод сдвинулся правильно, а звук нашёлся внутри источника дословно: кусок начинается на 48 сэмплов — одну миллисекунду, сороковую долю кадра — после точки реза, скопирован, а не пережат.

Два дефекта, которые мог показать только настоящий файл:

Терялись камерные метаданные. Источник несёт производителя, модель и бренд в отдельном блоке; `-map_metadata 0` их не переносит, и муксер писал обобщённое, теряя остальное. По

плану раздела камерные метаданные положено сохранять — добавлен `-movflags use_metadata_tags`, и после этого всё сверено тег за тегом.

Выбор канала менял формат сэмплов. `pcm_s16be` выходил как `pcm_s16le` — те же сэмплы с переставленными байтами и другой меткой в файле. Оба контейнера принимают PCM с обратным порядком байтов, поэтому порядок источника сохраняется.

Ни то, ни другое не всплыло бы на сгенерированных роликах: `lavfi` пишет прямой порядок байтов и никаких камерных тегов вовсе.

Чего матрица ещё не покрывает — 26.08.2026

Проверено с тех пор: две звуковые дорожки (камера плюс рекордер) обе переживают рез со сдвинутым таймкодом; метки HDR и HLG проходят без изменений; источник с открытой группой кадров режется в точное число кадров; таймкод 59,94 с пропуском кадров сдвигается верно; HEVC с LPCM в MP4 — то, что на самом деле пишут Sony и Panasonic — сохраняет `hvc1`, сохраняет `ipcm` и попадает в кадр.

Предупреждение про PCM в MP4 теперь срабатывает только при смене контейнера. Камера, снимающая HEVC с LPCM, ровно такой MP4 и пишет, и предупреждать о нём на каждом файле — шум, а не сведения.

Не проверено — и лучше сказать прямо, чем делать вид, что матрица полна: файлы длиннее часа и ряд редких сочетаний.

MXF переписывает PCM вместо копирования — 26.08.2026

Измеренное посэмплово: причина расхождения не в единицах монтажа муксера, а в том, что копирование PCM в MXF несёт лишние 384 сэмпла в начале — восемь миллисекунд.

Контейнер, который не удержит поток, отвергается до запуска — 25.08.2026

MP4 не держит ProRes, и FFmpeg говорит об этом фразой «Could not find tag for codec» и файлом нулевой длины. Известные отказы проверяются по кодеку источника заранее и сообщаются как пропуск — с причиной и с перечнем контейнеров, которые подойдут.

MPEG-TS не несёт указателя, поэтому перемотка в нём интерполируется по позициям байтов: точна в середине файла и способна промахнуться за конец рядом с ним. Это объявлено предупреждением, а результат заметно короче запланированного — или такой, что не читается вовсе — считается отказом, а не успехом со странной длительностью.

Канал, которого в файле нет, отвергается — 26.08.2026

Запрос несуществующего канала прежде давал тишину и код возврата 0.

Работа и форма

Что нашёл аудит 2.6.0 в обрезке — 28.09.2026

Пять дефектов за аудит и повторную проверку. Каждый воспроизведён на настоящем FFmpeg до исправления, и каждый теперь держит тест, который на старом коде падает.

Субтитры собираются для реза заново, а не копируются через него. Сначала MP4 с субтитрами не обрезался вовсе («Error selecting an encoder»: поток назначался без кодека). Явный кодек и копирование это починили, а повторная проверка измерила, что копия ошибается на каждой границе (TRIM-5): рез по числу кадров терял реплику через OUT, поиск оставлял реплике из-до IN полную длину, а склейка `Вырезать` делала из реплики 0,5–1,5 с одну микросекунду и сдвигала следующую на секунду раньше — и тест, смотревший только на наличие дорожки, проходил. Теперь каждая реплика вынимается в SRT на часах самого файла (`-copyts` минус начало картинки), обрезается по сохраняемым кускам, сдвигается на шкалу результата и вмешивается вторым входом в тот же вызов, что копирует картинку. MP4 и MOV держат mov_text, Matroska — SRT (замерено: чужой вид ни один не принимает); субтитры картинкой не обрезать без перерисовки, они остаются и называются. Тесты теперь читают реплики обратно — текст, начало и конец — и на копирующей версии падают.

Прогон говорит, что случилось. Служба возвращала счётчики без `ok`, а исполнитель читает только `ok` — упавшая запись, отвергнутые точки и отмена показывались зелёным статусом с кодом 0. Результат, оставленный из-за выключенной перезаписи, — пропуск по просьбе, а не ошибка.

Отмена убирает только то, что остановила. Раньше она проверяла файл следующего, не начатого куска и удаляла его, если тот не открывался, — старый файл при выключенной перезаписи.

Плеер отдаёт точки только своего файла. Стрелки меняют файл в разделе, а не в mpv, и `Взять A и B` записывал петлю старого дубля в новый. Файл спрашивается у самого mpv. При открытии другого файла A и B сбрасываются: это настройки плеера, и замерено — петля 1–3 с переживала обычный `loadfile`.

Строки — варианты, а не одна монтажная линейка — 28.09.2026

В разделе обрезки появился `+`, который добавляет ещё одну строку IN и OUT. Строки сознательно не куски одной линейки, как список сегментов в монтажке: каждая строка `Оставить` или `Вырезать` — отдельный прогон по тому же дублю и отдельный файл, `_keep01` или `_cut01`, с номером строки на экране. Между строками нечего согласовывать, поэтому они пересекаются как угодно: один момент входит в два клипа, а в каждом файле `Вырезать` из целого дубля вырезан свой кусок. Одна заполненная строка — ровно прежний прогон, без суффикса: так целы и врапперы, и старое поведение.

`Разделить` остаётся одной линейкой: N точек дают N + 1 частей, до 30 точек. Точки из плеера берутся только вперёд — позиция на предыдущей точке или позади неё ничего не пишет, — иначе случайно отведённый ползунок перемешал бы части. `+` сбрасывает A и B в плеере, а взятие точек — нет: пока строка открыта, её петля играет, и неверная точка слышна — ради этого петля и существует. Две точки на одном ключевом кадре останавливают прогон и называются в сообщении; склеивать их молча рассматривали и не стали — куски короче секунды сплиту не нужны.

Спиннеры `▲▼` не разбивают таймкод на поля по секциям: тогда его нельзя было бы скопировать и вставить целиком. Поле остаётся одной строкой, а стрелка меняет секцию, где стоит курсор. Младшая секция шагает кадром файла, а не миллисекундой: миллисекунда при 25p на рез не влияет.

Рез идёт по одному файлу — 25.08.2026

Папка дублей не делит точки реза: у них разные ключевые кадры, разная хлопушка, разный уход из кадра. Поэтому режется один файл за раз, и файл выбирается шагами — значок с текущим именем и стрелка по бокам — а не из списка.

Обёртки командной строки тоже режут один файл: тот, на который указывает `AUDION_TRIM_FILE`, или первый в `input`, и обёртка называет, какой. Резать папку по одной паре чисел никогда не было полезным поведением — только простым.

Отказы и честность

Прерванный прогон не оставляет того, что нельзя открыть — 26.08.2026

Отмена реза через полчаса работы над часовым файлом убила FFmpeg правильно и доложила об отказе правильно — и оставила 6 ГБ в папке результата под тем именем, которое носил бы

готовый кусок. Муксер не успел записать указатель, поэтому файл не открывается ничем, но взгляд на папку говорит, что рез там есть.

Прежнее правило удаляло только пустые файлы — на том основании, что частичный результат может пригодиться. Это верно, пока частичный файл открывается. Проверка теперь на читаемость: что открывается — остаётся, и журнал говорит, докуда дошло; что не открывается — удаляется, с причиной и размером.

Отказ перезаписать — это пропуск, а не успех — 25.08.2026

FFmpeg отвечает на `-n` при существующей цели фразой «already exists. Exiting.» и кодом возврата **0**. Прогон, не записавший ничего, читается как успех, а следующая проверка меряет предыдущий файл. Путь реза проверяет цель сам и докладывает пропуском.

Так же проверяют цель кодирование, звук, упаковка, субтитры, частота и цвет: файл, который уже лежит в Назначении, пропускается до запуска FFmpeg, строкой `[SKIP]` в журнале и счётом в пропущенных, а не в записанных.

Версии

FFmpeg 9.0.1 проверен, а не принят на веру — 26.08.2026

Вложенная сборка перешла на 9.0.1, и каждая таблица переснята на ней: матрица контейнеров и кодеков совпала с матрицей 8.0.1 ячейка в ячейку, резы на дробных частотах легли на те же кадры, все 75 обёрток отработали без правок. Из восьми параметров, исчезнувших между версиями, здесь не используется ни один.

Какая сборка на самом деле поставляется — решает установщик по версии драйвера NVIDIA; на большинстве машин это ветка 8.x. Поэтому рабочими держатся обе, а документация называет сборку, на которой сняты числа, а не версию поставки.

Рез нацелен на камерный материал, а не на сборку фильмов — 24.08.2026

Рез сделан для камерных оригиналов: MOV и MP4, H.264, H.265 и ProRes, звук PCM или AAC, одна-две дорожки. Не для фильмов с пятью языковыми дорожками, субтитрами и AC-3 — это сборка контейнера, и другие инструменты делают её хорошо.

Следствие для набора параметров: каналы вместо дорожек (камера пишет пушку и петличку двумя моно-каналами), камерные метаданные сохраняются, а не срезаются, и никакой

нормализации громкости — она уничтожает запас по уровню и связь между дублями. Нормализация принадлежит отдельному профилю для записи совещаний.

И то, что терять нельзя: **таймкод**. По нему монтажёр синхронизирует камеры между собой и с полевым рекордером. Рез обязан сдвинуть его правильно, а `tmcd`, как известно, молча исчезает при перепаковке MOV в MP4 — поэтому, если перенести его не удаётся, сказать об этом надо до запуска, а не после.

Окно

Окно — лицо консольной команды. Кнопка окна и `media_cli.cmd run` — одно и то же: окно запускает операцию через команду, и первая строка прогона в журнале — готовая команда, которую можно повторить. Форма каждой операции строится из её описания в манифесте, поэтому окно, консоль и документы говорят одно.

Терминал в главной роли. Вывод FFmpeg — часть работы: строки операции идут в цветах служб, под полосой хода — родная строка FFmpeg. Итог прогона держится постоянно, а не всплывающим сообщением, которое легко пропустить.

Формы раздельные. У каждого раздела и режима своя форма, и она помнится между запусками; общие на всё окно — только кодер и декодер: это выбор машины, а не задачи.

Действие называет свой объект. Кнопка стоит у своего поля, и подсказка её называет: «Добавить файлы LUT», «Открыть папку LUTs», а не «Добавить» и «Открыть»: на одном экране бывает несколько полей с файлами.

Внешний ввод переносится к себе. Кнопки добавления копируют выбранное во входную папку: безопаснее, чем долгие операции по сетевым путям и съёмным дискам.

Кодировка `.cmd` — условие сборки. Все файлы `.cmd` проекта — в UTF-8 без метки порядка байтов и со строгими CRLF; проверка встроена в сборку окружения и есть отдельным сценарием `install\Check-CmdEncoding.cmd`.