Audion Media Tools

English · Руководство · Консольная команда · Замеры · Решения

Содержание

- Зачем это сделано
- Главное решение: FFmpeg — по драйверу
  - RAW камер
    - ProRes RAW на видеокарте
  - Что умеет каждый путь
- Почему этому можно верить
  - С чего всё началось
  - Проверено на том, что снимают камеры
  - Синхрон измерен, а не предположен
  - Семнадцать неточностей FFmpeg — компенсированы
- Принципы
- Что внутри
- Дальше
- Техническая часть
  - Запуск
  - Папки
  - Значения по умолчанию
  - Отчёты
  - Перед запуском
  - Что проверить после прогона
  - Что осталось за рамками

Портативная программа для настоящей работы с видео и звуком: скачать, проверить, перекодировать, подготовить к монтажу, упаковать для заливки, положить в архив, аккуратно

разобрать звук, нарезать камерные файлы без потерь.

Зачем это сделано

Всё это умеет FFmpeg — и в этом же проблема. Правильная команда для монтажного кодека, для архивной мастер-копии и для заливки на площадку выглядит по-разному, занимает полторы строки ключей, и половина этих ключей выясняется опытом: что контейнер не держит выбранный поток, что смена канала перевернула порядок байтов в звуке, что метаданные камеры потерялись при перепаковке.

Программа хранит этот опыт в пресетах и даёт ему окно: разделы и режимы, пресеты, память путей, терминал с родным выводом FFmpeg и видимая команда. Каждая кнопка окна — операция консольной команды `media_cli.cmd`, поэтому та же работа идёт из пакетного файла или у ИИ-агента, которого попросили словами.

Главное решение: FFmpeg — по драйверу

Каждая сборка FFmpeg собрана под конкретную версию заголовков аппаратного кодирования NVIDIA, и каждая требует своего минимума драйвера. Поставьте самую свежую на драйвер постарше — аппаратное кодирование не ускорится, а перестанет работать. Поэтому установщик подбирает сборку по вашему драйверу.

драйвер NVIDIA	**какую FFmpeg ставит программа**
видеокарты NVIDIA нет	самую новую стабильную сборку Gyan, не ниже 9.0.2
610.88 (LTS) и новее	самую новую стабильную сборку Gyan, не ниже 9.0.2
570.0 – 610.87	8.0.1 (заголовки NVENC n13.0.19.0) — доступны не все операции
551.76 – 569.xx	7.1 (заголовки NVENC n12.2.72.0) — доступны не все операции
ниже 551.76	никакую: установщик останавливается и говорит почему

610.88 — LTS-драйвер, под который собраны программы. С ним или без видеокарты NVIDIA ставится самая новая стабильная FFmpeg — выйдет 9.1, будет 9.1 — и никогда не старше 9.0.2. Если самая новая сборка не прошла пробу NVENC на этом драйвере (новая FFmpeg может потребовать драйвер новее), установщик возвращается к 9.0.2. Ниже 610.88 ставится прежняя ветка, при которой работает NVENC, и установщик, окно перед скачиванием и бейдж FFmpeg в заголовке говорят, что доступны не все операции — те, которым нужна FFmpeg 9.x. Сборка

подбирается при установке: сменился драйвер — FFmpeg, поставленная снова (SETTINGS → КОМПОНЕНТЫ → ОБНОВИТЬ), подберётся под него.

Для ProRes RAW нужна 9.x. В мае 2026 декодер ProRes RAW в FFmpeg исправили: теперь он разбирает кривую линеаризации, записанную в файле, и это устраняет искажения тонов и цвета, которые видны на кадре. Исправление вошло в 9.0; 8.0.1 и более ранние декодируют ProRes RAW с этими искажениями. Кроме того, 9.x обрезает кадр по рабочей области сенсора и передаёт с каждым кадром цветовые параметры камеры. Проявляет ProRes RAW программа сама, на видеокарте, — о нём и другом RAW камер ниже.

7.1.1 старому драйверу не поможет: она собрана теми же заголовками, что и 8.0.1, и требует те же 570.0 — «откатиться на версию назад» не даёт ничего. Помогает только 7.1.

Нет видеокарты NVIDIA — драйвер не важен: ставится самая новая сборка, кодирует процессор, Intel или AMD.

RAW камер

FFmpeg не проявляет RAW камер. ProRes RAW он читает как сигнал сенсора — и 9.x без баланса белого и цветовой матрицы камеры, фильтра, который их применяет, в FFmpeg нет. Поэтому ProRes RAW проявляет cineraw на видеокарте (ниже); пока он не установлен, КОДИРОВАНИЕ, ЧАСТОТА и ЦВЕТ ProRes RAW не берут, а НАРЕЗКА, УПАКОВКА и АУДИО берут как есть. Остальной RAW FFmpeg не читает вовсе — ни картинку, ни копию потока. Источник только из RAW отклоняется до начала, среди других видео такой файл не удаётся сам, остальные делаются; строка предпроверки называет программу и её страницу. Программы ставит человек, с сайтов производителей и на их условиях: Media Tools их не скачивает и не несёт.

RAW	файлы	чем открыть
ProRes RAW, ProRes RAW HQ	`.mov`	DaVinci Resolve, Final Cut Pro
ARRIRAW, ARRIRAW HDE	`.mxf`, `.ari`, `.arx`	ARRI Reference Tool, DaVinci Resolve
RED R3D	`.r3d`	REDCINE-X PRO, DaVinci Resolve
Nikon N-RAW	`.nev`	DaVinci Resolve, REDCINE-X PRO
Blackmagic RAW	`.braw`	Blackmagic RAW Player, DaVinci Resolve
Sony RAW, X-OCN	`.mxf`	Sony RAW Viewer, DaVinci Resolve

RAW	файлы	чем открыть
Canon Cinema RAW, Cinema RAW Light	`.rmf`, `.crm`	Canon Cinema RAW Development, DaVinci Resolve

В любой из них — перевод в ProRes 4444 или DNxHR 444, и этот файл уже идёт в программу.

ProRes RAW на видеокарте

ProRes RAW программа проявляет сама — как снято: баланс белого и цветовая матрица камеры из файла, мягкий завал светов для SDR, BT.709. Проявляет cineraw из libcraw (BSD-2-Clause) на видеокарте с Vulkan 1.4, которая читает изображения без заданного формата: NVIDIA RTX подходит, встроенная графика Intel — нет. Готовых сборок у libcraw нет, поэтому cineraw собирается из исходников при сборке программы и приходит вместе с ней; видеокарта для сборки не нужна — её проверяет программа, когда проявляет. СЕРВИС → «Установить проявку ProRes RAW» пересобирает его, например под другой коммит libcraw: переносной MSYS2 только на время сборки (в Windows ничего не ставится), FFmpeg 9.0.2, урезанная до чтения MOV и декодера ProRes RAW (LGPL), и libcraw проверенного коммита. Окно перед загрузкой называет файлы и размер.

загрузка для сборки	около 185 МБ: MSYS2 41 МБ, компилятор и инструменты 127 МБ, исходники libcraw 16 МБ
время сборки	5 минут (Ryzen 9 5900X)
на диске	`Tools\cineraw` — 3,5 МБ программ и 33 МБ исходников
скорость, 4.2K 23,976	5 кадров в секунду в ProRes 4444 (RTX 5070, Ryzen 9 5900X): минута съёмки — около 5 минут

КОДИРОВАНИЕ → RAW проявляет ProRes RAW Источника в ProRes 4444 или DNxHR 444 для монтажа, H.264 или H.265 10 бит для просмотра (кодер — процессор или NVENC) со звуком и таймкодом источника и метками BT.709 в файле. КОДИРОВАНИЕ, ЧАСТОТА и ЦВЕТ берут ProRes RAW, проявив его сначала в ProRes 4444 во временной папке; результат назван и разложен как источник. Клип режется на куски, их проявляют несколько cineraw разом: чтение ProRes RAW идёт на процессоре и занимает ядро на кусок, видеокарта проявляет кадр за 3 мс. Ни один кадр не ложится на диск — cineraw отдаёт его программе через канал Windows, и стыки кусков точны до кадра.

Что умеет каждый путь

Все три аппаратных пути поддержаны одинаково полно: NVIDIA, Intel и AMD закрывают H.264, HEVC и AV1, каждый со своими родными регуляторами качества и своими пробами декодирования. Различия в таблице ниже — это пределы самих кодировщиков, а не пробелы в программе.

путь	H.264	HEVC	AV1	10 бит	4:2:2
CPU (x264 / x265 / SVT-AV1)	да	да	да	да	да
NVIDIA NVENC	8 бит	да	`av1_nvenc`	HEVC и AV1	только новое железо
Intel QuickSync	8 бит	да	`av1_qsv`	HEVC и AV1	нет
AMD AMF	8 бит	да	`av1_amf`	HEVC и AV1	нет

H.264 на любом аппаратном пути — только 8 бит. Нужно 10 — берите HEVC, AV1 или процессорный x264.

Декодирование — отдельный стек: на машине без аппаратного кодирования аппаратный декод может работать прекрасно. Проверяется своими пробами: `CUDA decode`, `QuickSync decode`, `AMD/D3D11VA decode` и `dav1d` для AV1.

Регуляторы качества всегда родные для пути: `CRF` и предустановка SVT-AV1 для процессора, `CQ/QP` и предустановки NVENC, QuickSync или AMF для железа. Список тонкой настройки читают только x264 и x265, поэтому он и показывается лишь при выборе процессора.

Перед первым запуском на новой машине — СЕРВИС → «Возможности оборудования». Результат ложится в кэш, и профили сверяются с ним. Отсутствие пути — нормальный ответ, а не поломка: значит нет драйвера или самого железа. Кэш привязан к машине: если он чужой или его нет, проверку прогоняют заново. SETTINGS показывает, что прошло проверку, рядом с процессором, видеокартой и её драйвером.

Почему этому можно верить

Точность резки, синхрон звука и пределы FFmpeg здесь не заявлены, а измерены. Свод замеров держится на одном правиле: **ни одно утверждение не взято из документации** — каждая ячейка таблицы это реальная попытка, каждое число замер. Где измерение оказалось неверным, это тоже записано: отдельный раздел собирает случаи, когда правдоподобный вывод пришлось отменить.

С чего всё началось

Shutter Encoder с версии 19.1 читает дробную частоту как целую: 29,97 становится 30. Ошибка ×1,001 копится — 1,8 секунды рассинхрона на каждые полчаса, четыре секунды на час. На 20.2 дефект на месте, обращение #402 открыто.

Здесь счёт идёт точной дробью:

частота	дробь	запрос	правильно	через `round(fps)`
23,976	24000/1001	50 с	**1199**	1200
29,97	30000/1001	4 мин	**7193**	7200
59,94	60000/1001	4 мин	**14386**	14400

Величины, за которые идёт разговор: кадр — 33 мс на 29,97, звуковой пакет — 8 мс, сэмпл — 20 микросекунд. Смещение на один кадр монтажёр потом ищет руками.

Проверено на том, что снимают камеры

Не на сгенерированных роликах — на материале с настоящих аппаратов, от Sony до ARRI. Расхождение по кадрам везде ноль:

источник	запрос	кадров	расхождение
ProRes 25p	3 → 9	150	0
HEVC long-GOP 29,97	12 → 30	600	0
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

Отдельная находка: Alexa Mini, снимающая честные 24,000, объявляет `24/1` — и читать такой файл как 23,976 значит **создать** ошибку ×1,001, а не устранить её. Пять кадров за четыре минуты, ровно то, за что критикуют чужую нарезку. Частота берётся у файла как есть; ту, что не прочиталась, честнее отклонить, чем заменить догадкой.

Синхрон измерен, а не предположен

Маркер, существующий в обоих потоках в один момент: белый кадр в начале каждой секунды и щелчок 20 мс там же. Отклонение от собственного смещения источника:

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

Сдвигает только MXF — его муксер равняет звук по своим единицам монтажа, это меньше половины кадра и объявлено предупреждением. Прежнее предупреждение про MKV точное измерение опровергло, и оно снято, а не оставлено правдоподобной страшилкой.

Семнадцать неточностей FFmpeg — компенсированы

Ни одна не очевидна из документации, каждая стоила отдельного разбора и каждая обойдена в коде. Среди них: `-ss` до `-i` сбрасывает отсчёт времени; конец, округлённый «к ближайшему», теряет кадр, стоящий на 29,9966; HEVC с B-пирамидой пишет на 2–4 кадра меньше запрошенного; `avg_frame_rate` на переменной частоте превращает 12 запрошенных секунд в 16; `-n` на существующий файл отвечает «уже существует» и **кодом возврата 0**, из-за чего пустой прогон читается как успех; PCM при копировании в MXF получает 384 лишних сэмпла в начале.

Полный перечень с проявлениями — в замерах, раздел 10. Список проверок показывает, что именно прогоняется перед выпуском.

Принципы

Проверено на настоящем файле, а не на сгенерированном. Синтетический ролик не несёт ни тегов камеры, ни звука с обратным порядком байтов — а именно на них ломались две вещи,

которых иначе никто бы не увидел. Матрица проверок построена на файлах с камер.

Режим назван по тому, что останется. «Оставить», «Вырезать», «Разделить» — список того, что будет в результате, читается однозначно; список исключений — нет.

Что контейнер не удержит — отвергается до запуска, а не в середине. Несовместимое сочетание кодека и контейнера блокируется в форме и проверяется службой ещё раз.

Остановленный прогон не оставляет того, что нельзя открыть. Файл пишется под своим временным именем и получает настоящее только целым; незавершённый файл не притворяется готовым.

Операция не меняет Источник. Результат ложится в Назначение; файлы Источника остаются как были.

Результат кладётся в папку операции, а не в имя файла. `input\Day1\clip.mov` и `input\Day2\clip.mov` не сваливаются в одну плоскую папку и не спорят за имя: структура подпапок повторяется в результате.

Что внутри

Лента окна, у каждого раздела — свои режимы:

раздел	режимы	о чём
АУДИО	извлечь · звук в видео · конвертировать	звук из видео, звук внутри видео с копией картинки, звуковые файлы; ресемплинг, громкость в LUFS, каналы
КОДИРОВАНИЕ	архив · монтаж · хранение · доставка	FFV1 и x264 без потерь; ProRes и DNxHR в MOV или MXF; x264, x265, SVT-AV1; просмотр и заливка
УПАКОВКА	видео · аудио · видео и аудио · субтитры	другой контейнер без перекодирования; поворот камеры, снятой боком; потоки субтитров отдельно
НАРЕЗКА	оставить · вырезать · разделить	покадровая нарезка камерных файлов без потерь; несколько вариантов с одного дубля, разделение до 30 точек; звук без потерь (WAV, FLAC, ALAC) — по сэмплу, сжатый — по своим пакетам без перекодирования или по сэмплу в режиме «Точно»
ЧАСТОТА	varispeed · conform	в 23,976 и весь рабочий ряд
ЦВЕТ	LUT	LUT с гаммой перед ним; HDR в SDR

раздел	режимы	о чём
СКАЧАТЬ	лучшее MP4 · надёжное MP4 · своё видео · для загрузки · своё аудио · субтитры	ссылка, плейлист или список ссылок через yt-dlp

СВЕДЕНИЯ говорят, что за файлы в Источнике (ffprobe). СЕРВИС держит проверки и установщики: инвентаризацию, самопроверку, проверку профилей, возможности оборудования, тест матрицы пресетов, установку 7-Zip, FFmpeg, yt-dlp и mpv.

Дальше

- Руководство — окно, порядок работы, нарезка с плеером, отчёты.
- Консольная команда — `media_cli.cmd`: любая операция из консоли или сценария.
- Операции — каждое поле, вариант и пресет; Обёртки — 95 готовых задач из `Scripts\`.
- ИИ-агент — для Claude Code, Codex и подобных.
- Замеры — цифры, на которых всё держится.
- Проверки — что прогоняется перед выпуском.
- Решения — почему сделано так, с датами и числами.
- Журнал изменений — что принесла каждая версия.

Техническая часть

Запуск

`Start.exe` в корне программы открывает окно. Ничего не устанавливается: окно, его PowerShell, Python, FFmpeg, yt-dlp, mpv и 7-Zip живут в папке программы.

Нет окружения Python или инструмента — его собирает `builder_main.cmd`: меню с окружением, проверками и установщиком для каждого инструмента (FFmpeg, yt-dlp с Deno, mpv, 7-Zip). Каждый установщик до скачивания называет файлы, источник и размер.

Папки

папка	что в ней
`input\`	Источник по умолчанию: файлы для работы; `input\urls.txt` — список ссылок
`output\`	Назначение по умолчанию: каждый результат — в папке своей операции
`LUTs\`	LUT раздела ЦВЕТ (`.cube`, `.3dl`, `.lut`)
`logs\`, `report\`	журнал каждого прогона и его отчёт
`Scripts\`	95 готовых задач — обёрток консольной команды

Источник обходится рекурсивно, структура подпапок сохраняется в результате. Результат пишется в папку операции и профиля:

```
output\Remux\remux_mkv_to_mp4_video_audio\Day1\clip.mp4
```

Другое Назначение просто заменяет `output`, внутренняя структура остаётся прежней. Скачанное ложится в `output\Download`.

Значения по умолчанию

что	значение
качество (CRF/CQ)	14
звук	384 kbps; для M4A есть профиль 256
ресемплер	SoX/libsoxr, полное качество (`precision=33`)
архив скачиваний	выключен
перезапись	выключена

Пиксельный формат в режиме МОНТАЖ не выбирается — его задаёт сама цель: ProRes всегда 10 бит 4:2:2, DNxHR HQ — 8 бит, DNxHR HQX — 10 бит. Чтобы поменять разрядность, выбирают другую цель, а не отдельное поле.

Для MXF доступны PCM 16 или 24 бита либо вывод без звука; несовместимые варианты блокируются в форме и проверяются службой. FLAC недоступен для MOV — этот контейнер его не держит, нужен MP4 или MKV.

Отчёты

Каждый прогон пишет журнал в `logs\`, а `batch_report.md` и `.json` — в `report\`. АУДИО добавляет `audio_report.md` и `.json`: исходный поток, параметры входа, режим обработки, громкость, путь результата.

Перед запуском

Каждая медиа-операция проверяет себя до того, как стартует FFmpeg или yt-dlp: Источник существует, Назначение есть или может быть создано, файлы нужного вида найдены, выбранный LUT существует, цель упаковки удержит потоки, кодер есть в этой сборке FFmpeg, аппаратный путь прошёл проверку оборудования. Что не так — говорят строки `[PREFLIGHT]`, и ничего не запускается. Предпроверка не заменяет внимания, но отсекает самые дорогие ошибки до первого кадра.

Что проверить после прогона

Законченный прогон — ещё не принятый результат. Что просит посмотреть каждая работа:

работа	на что смотреть
архив	размер до большого прогона; хранить контрольные суммы; проверять декодирование с места хранения, а не только с рабочего диска
монтаж	частоту кадров, пиксельный формат, порядок полей, формат сэмплов звука и таймкод в той монтажке, которая примет файлы
хранение, доставка	значения по умолчанию — отправная точка: попробовать трудное движение, зерно, градиенты, субтитры и пики звука на коротком куске, на том плеере или площадке, для которых файл; аппаратный кодер может выглядеть иначе процессорного
упаковка	каждый файл разбирается, а не судится по расширению; что контейнер не удержит, пропускается с причиной. Поворот пишется в контейнер — матрица отображения, которую читает монтажка, — и ничего не перекодируется: файл 960x540 сохраняет байты, а декодер отдаёт 540x960. MP4, MOV и MKV его хранят, MXF — нет
частота	длительность, метки времени, синхрон, число кадров и каденцию на движении — а не одно целевое число; папку со смешанными частотами лучше разделить
цвет	какой LUT выбран и его версию; диапазон, передаточную функцию, первичные цвета, матрицу, клиппинг и нейтральность в цветоуправляемом просмотрщике. Превью 18 % серого показывает направление гаммы, а не доказательство. Путь LUT с пробелами, апострофами, запятыми или скобками программа экранирует для FFmpeg сама

работа	на что смотреть
звук	выбранный поток — по языку, раскладке и кодеку, а не по номеру; PCM и FLAC надёжнее в MOV или MKV, чем в MP4; синхрон, длительность и громкость после прогона
скачивание	список ссылок до пачки; неудачные ссылки — по одной: доступ, вход, отсутствующий формат и сеть лечатся по-разному

Аппаратные кодеры получают пиксельный формат в своей записи (`yuv420p10le` становится `p010le`, QuickSync берёт `nv12` вместо `yuv420p`); чего кодер не умеет — 10 бит H.264 на железе — отклоняется до запуска, а не пишется в другом формате.

Тишина прогона — не зависание: строка хода, рост файла и журнал скажут больше. Одно окно ведёт одну операцию за раз; две копии программы могут работать рядом, но делят процессор, видеокарту и диск и не должны писать один и тот же файл.

Что осталось за рамками

Программа не пытается заменить монтажную программу вроде DaVinci Resolve или Premiere, стать универсальным сборщиком команд FFmpeg, угадывать творческие решения, прятать риск перезаписи или постоянно разбирать большие папки ffprobe. Её сильная сторона — безопасные, повторяемые и понятные медиа-процедуры.

WebM пока входной, скачиваемый и перепаковываемый формат, но не отдельная цель массового кодирования. Понадобится доставка под VP9 или AV1 в WebM — её лучше добавить отдельным профилем с внятными параметрами качества, контейнера и звука, а не пристёгивать к существующим.