Encoding Pipelines

English

Зачем Разделены Архив, Монтаж, Хранение И Доставка

Это четыре режима раздела КОДИРОВАНИЕ. Они похожи, потому что все строят команды кодирования FFmpeg, но цель у них разная.

- АРХИВ - сохранить мастер с минимальными потерями или без потерь.
- МОНТАЖ - подготовить файл для монтажной программы.
- ХРАНЕНИЕ - получить компактный качественный файл для библиотеки.
- ДОСТАВКА - сделать файл для просмотра, отправки или заливки.

Если выбирать только по codec, легко ошибиться. Выбирай по назначению результата.

Общие Поля

Во многих encode-разделах повторяются:

- `Фильтр файлов источника`;
- `Декодировать через`;
- `Кодировать через`;
- `Аудио`;
- `MP3 / LAME`;
- `Битрейт аудио`;
- `Контейнер`;
- `Preset кодера`;
- `Разрешение`;
- `Pixel format`;

- `CRF`;
- `CQ/QP`;
- `Тюнинг кодирования`;
- `Перезаписывать`;
- `Тест: первый файл`;
- `Только показать команду`.

Tooltips над этими полями объясняют смысл без открытия документации.

Архив

Назначение: долговременные мастер-форматы.

Профили:

- `FFV1 MKV + FLAC`;
- `x264 lossless MKV`;
- `FFV1 + x264 lossless`.

Цели:

- `FFV1`;
- `x264 lossless`.

Когда выбирать:

- нужно сохранить технический master;
- качество важнее размера;
- файл будет источником для будущих transcode;
- материал должен пережить повторные операции без накопления потерь.

Контейнер по смыслу обычно MKV, потому что он хорошо держит FFV1/FLAC и служебные потоки.

Архив без потерь либо сохраняет данные источника как есть, либо отказывает — тихого понижения точности нет:

- x264 lossless держит YUV 8 и 10 бит; источник 12 бит, RGB или с альфой отклоняется строкой `[SKIP]` с подсказкой, что его сохранит FFV1, и прогон не зелёный. Пиксельный формат, выбранный руками, — выбор человека, он не отклоняется.

- FFV1 держит почти всё, вплоть до 16 бит, RGB и альфы; формат, которого нет в его списке, отклоняется так же.
- FLAC держит целочисленный звук; PCM с плавающей точкой (и 64-битный) копируется как есть — MKV его принимает.

Монтажные Кодеки

Назначение: intermediate/mezzanine для NLE.

Профили:

- `ProRes 422 MOV`;
- `ProRes LT MOV`;
- `ProRes Proxy MOV`;
- `DNxHR HQ MXF`;
- `ProRes + DNxHR`.

Цели:

- `ProRes Proxy`;
- `ProRes LT`;
- `ProRes 422`;
- `ProRes HQ`;
- `DNxHR HQ`;
- `DNxHR HQX`.

Когда выбирать:

- исходник тяжёлый для монтажа;
- нужен intra-frame codec;
- нужно отдать материал в монтаж;
- нужно временно уйти из long-GOP/H.264/H.265.

Рекомендации:

- Proxy/LT - для лёгкого монтажа и черновиков.
- ProRes 422/HQ - для качественного intermediate.
- DNxHR HQ/HQX - для Avid/Resolve/Windows workflows.

- MOV чаще удобен для ProRes, MXF - для broadcast/NLE pipelines.

Хранение

Назначение: компактные качественные файлы для библиотеки или передачи.

Профили:

- `x264 CRF 14`;
- `x264 1080p CRF 14`;
- `HEVC Main10 CRF 14`;
- `SVT-AV1 CRF 14`.

Цели:

- `x264 / H.264`;
- `x265 / HEVC`;
- `AV1` (на CPU — SVT-AV1).

Когда выбирать:

- нужен хороший файл для личной библиотеки;
- нужно уменьшить размер без жёстких upload-требований;
- важна читаемость на локальном оборудовании;
- можно позволить более медленный encode ради качества.

CRF:

- меньше = крупнее и чище;
- больше = меньше и хуже;
- `14` - очень качественная отправная точка.

Доставка / Публикация

Назначение: файлы для просмотра, заливки и передачи.

Профили:

- `Review H.264`;
- `Upload H.264`;

- `Upload HEVC 10-bit`;
- `SVT-AV1 delivery`.

CPU/NVENC/AMF/QSV выбираются отдельно в блоке `Кодер`, чтобы профиль описывал сценарий вывода, а не аппаратный backend.

Цели:

- `x264 / H.264`;
- `x265 / HEVC`;
- `AV1` (на CPU — SVT-AV1).

Когда выбирать:

- файл должен смотреться без специальных tools;
- нужен upload-friendly encode;
- нужно использовать hardware encode;
- важна скорость.

Hardware encode использует `CQ/QP`, software encode обычно использует `CRF`.

Decode Backend

`Декодировать через`:

- `Auto`;
- `CUDA`;
- `QuickSync`;
- `AMD/D3D11VA`;
- `dav1d AV1`.

Decode backend отвечает за чтение входного потока. Он не равен encode backend. Например, можно декодировать AV1 через dav1d и кодировать H.264 через x264 CPU.

`dav1d AV1` - не «ускоритель вообще», а принудительный декодер AV1 (`-c:v libdav1d`). На источнике другого кодека FFmpeg такой файл прочитать не может, поэтому preflight проверяет кодек первого файла Source и останавливает операцию с понятным сообщением. Для смешанного Source выбирай `CPU/Auto`.

Где живут кадры: `-hwaccel_output_format`

Аппаратный декодер отдаёт кадры либо в видеопамять, либо в системную. Управляет этим `-hwaccel_output_format`, и поведение по умолчанию у трёх стеков разное:

- `CUDA`: без явного формата кадры уходят в системную память. NVIDIA рекомендует `-hwaccel cuda -hwaccel_output_format cuda` именно чтобы этого избежать.
- `QuickSync`: по умолчанию FFmpeg оставляет кадры в видеопамяти и печатает предупреждение `defaulting hwaccel_output_format to qsv ... DEPRECATED`. Любой программный фильтр после этого падает с `Impossible to convert between the formats`.
- `AMD/D3D11VA`: по умолчанию кадры возвращаются в системную память; для полного GPU-пути AMD рекомендует `-hwaccel_output_format d3d11`.

Поэтому проект задаёт формат сам:

- Полный GPU-конвейер (`-hwaccel_output_format cuda` / `qsv` / `d3d11`) включается, когда декодер и кодер одного вендора, `Pixel format` оставлен как `auto` и нет программных фильтров - масштабирования, LUT или HDR-to-SDR. В терминале появляется строка `Decode frames stay in GPU memory ...`.
- Если что-то требует кадров в системной памяти, для QuickSync явно запрашивается формат загрузки (`nv12`, а для 10-битного источника `p010le`) - иначе фильтры не работают вовсе. CUDA и D3D11VA в этом случае формат не навязывают, чтобы не фиксировать разрядность за FFmpeg.

Смешанные пары (например CUDA-декод и CPU-кодирование) полностью рабочие: аппаратный декодер ускоряет чтение, а кодирование идёт на CPU.

Encode Backend

`Кодировать через`:

- `Auto`;
- `CUDA NVENC`;
- `QuickSync`;
- `AMD AMF`;
- `CPU`.

Рекомендации:

- CPU - качество и предсказуемость.

- NVENC/QSV/AMF - скорость, особенно для delivery/review.
- Auto - хорошо, если profile already knows what it wants.

Кодер применяется ко всем трём семействам таргетов: H.264, HEVC и AV1. Цель `AV1` кодируется через SVT-AV1 на CPU и через `av1_nvenc`, `av1_qsv` или `av1_amf` на соответствующем железе. Регуляторы качества при этом всегда родные для backend: `CRF` и `SVT-AV1 preset` для CPU, `CQ/QP` и `NVENC preset` / `QSV preset` / `AMF quality` для аппаратных кодеров.

Перед hardware encode запускай `Hardware Capabilities`.

Audio Mode

Общее поле `Аудио`:

- `Родной поток`;
- `AAC override`;
- `MP3`;
- `FLAC`;
- `PCM 16-bit`;
- `PCM 24-bit`;
- `PCM float`;
- `Без аудио`.

Для delivery обычно AAC. Для archive - source или FLAC. Для монтажных задач - source или PCM.

Не каждый режим влезает в любой контейнер, и это проверяется до запуска:

- `MXF` принимает только PCM 16-bit, PCM 24-bit или отсутствие аудио;
- `MOV` не умеет FLAC (`flac only supported in MP4`) - для FLAC выбирай MP4 или MKV, для MOV подойдёт PCM.

Недоступный режим гасится в форме, а при переключении контейнера на MXF или MOV уже выбранный несовместимый режим сам переходит на `PCM 24-bit`. Backend проверяет то же самое, поэтому CLI и профиль тоже не могут собрать невозможную пару.

MP3 / LAME

При `Аудио = MP3` появляется та же шкала пресетов LAME, что и в разделе АУДИО, - одна на всю программу:

- `VBR V0` (около 245 kbps) - значение по умолчанию, `-q:a 0`;
- `Insane 320` - `-b:a 320k`;
- `По битрейту` - берёт значение поля `Битрейт аудио`.

Поле `Битрейт аудио` при MP3 показывается только для третьего варианта, потому что первые два его не читают. Значение `384 kbps` для MP3 недоступно: потолок MPEG-1 Layer III - 320 kbps, и LAME молча опускает до него любое большее число. Ограничение живёт в общем контракте, поэтому и GUI, и CLI (`AUDION_MP3_LAME_PRESET`) кодируют MP3 одинаково.

Чересстрочный источник

Источник из двух полей (25i, 29,97i, 50i) в МОНТАЖЕ, ХРАНЕНИИ, ДОСТАВКЕ и ЦВЕТЕ по умолчанию становится прогрессивным. Поле `Чересстрочный источник` — в блоке «Дополнительно»:

- `Прогрессив x2` — по умолчанию: bwdif делает кадр из каждого поля, частота вдвое выше, движение целиком (50i → 50p);
- `Прогрессив` — цельный кадр из каждого кадра, частота прежняя (50i → 25p);
- `Оставить поля` — поля и их порядок сохраняются, для чересстрочного эфира; x264 кодирует их как поля.

Папка настроек говорит, что сделано: `x264_crf14_slow_progressive_2x`. Прогрессивный источник не трогается. АРХИВ поля сохраняет всегда — это архив без потерь.

Pixel Format

Варианты:

- auto;
- yuv420p;
- yuv420p10le;
- yuv422p;
- yuv422p10le.

Практически:

- yuv420p - максимальная совместимость;
- yuv420p10le - HEVC/AV1 10-bit delivery/storage;
- yuv422p/yuv422p10le - монтажные/профессиональные сценарии.

Pixel format применяется и к аппаратным кодерам: выбранный формат переводится в родной формат кодера (`yuv420p10le` → `p010le`, для QSV `yuv420p` → `nv12`). Если выбранный кодер физически не умеет запрошенный формат, операция останавливается с явным сообщением, а не пишет молча другой формат:

- H.264 на NVENC/QSV/AMF - только 8-bit. Для 10-bit выбирай HEVC/AV1 или CPU x264.
- 4:2:2 на QSV/AMF недоступен; на NVENC он есть только у нового железа.

`auto` оставляет согласование формата за FFmpeg — это по-прежнему поведение по умолчанию.

Encode Tuning

Флаги:

- Grain;
- Film;
- Animation;
- Fast decode;
- Zero latency.

Не включай tuning “на всякий случай”. Он должен соответствовать материалу или delivery-требованию.

Это значения `tune` для x264/x265, поэтому блок показывается только при `Кодер = CPU`. У NVENC, QSV и AMF своя шкала качества (`NVENC preset`, `QSV preset`, `AMF quality`), и список tuning они не читают.

Test On First File

`Тест: первый файл` особенно важен для кодирования. Он проверяет:

- найден ли input;

- работает ли backend;
- совместим ли container;
- корректен ли audio mode;
- не сломан ли LUT/decode path;
- хватает ли прав на Назначение.

Для большого Источника это обязательная привычка.