# Audion Media Tools

Русский · User Guide · Measurements · Decisions

**Contents**



A portable shell for real work with video and audio: download, inspect, transcode, prepare for editing, package for upload, put into an archive, take audio apart carefully.

## Why It Exists

FFmpeg does all of this — and that is the problem. The right command for an editing codec, for an archival master, and for an upload all look different, run to a line and a half of flags, and half those flags are learned the hard way: the container will not hold that stream, picking a channel flipped the byte order in the audio, the camera metadata was lost in the remux.

This program keeps that experience in presets and moves the controls into a window: lists, buttons, checkboxes, remembered paths, an operation terminal, and the visible command. The spirit of the old command-line version is intact — what changed is how you use it.

## The Central Decision: Newer Is Not Better

Every FFmpeg build is compiled against a particular version of the NVIDIA hardware-encoding headers, and each one demands its own minimum driver. Install the newest build on an older driver and hardware encoding does not get faster — it stops working.

| FFmpeg build | NVENC headers | NVIDIA driver required |
|---|---|---|
| 9.0.1 | n13.1.15.0 | **610.0** |
| 8.0.1 | n13.0.19.0 | **570.0** |
| 7.1.1 | n13.0.19.0 | **570.0** |
| 7.1 | n12.2.72.0 | 551.76 |

The third row matters most: 7.1.1 was built with the same headers as 8.0.1 and needs the same 570.0 — "rolling back one version" on an older driver gains nothing. Only 7.1 helps.

So the installer picks a build to match your driver rather than taking the latest.

**8.0.1 ships by default** — a choice, not a forgotten update. Editing machines today mostly run drivers somewhere between 571 and 609; the 610 branch is on very few. Both 8.1 and 9.x require it: shipping them would mean advertising NVIDIA acceleration and denying it to most of the people promised it. 8.0.1 has everything these programs use and runs on the drivers people actually have.

With no NVIDIA card none of this matters: the latest build is installed and encoding runs on the processor.

### What each path can do

All three hardware paths are supported equally fully: NVIDIA, Intel, and AMD each cover H.264, HEVC, and AV1, with their own native quality controls and their own decode probes. The differences in the table below are limits of the encoders themselves, not gaps in the program.

| path | H.264 | HEVC | AV1 | 10-bit | 4:2:2 |
|---|---|---|---|---|---|
| CPU (x264 / x265 / SVT-AV1) | yes | yes | yes | yes | yes |
| NVIDIA NVENC | 8-bit | yes | `av1_nvenc` | HEVC and AV1 | newer hardware only |
| Intel QuickSync | 8-bit | yes | `av1_qsv` | HEVC and AV1 | no |
| AMD AMF | 8-bit | yes | `av1_amf` | HEVC and AV1 | no |

**H.264 on any hardware path is 8-bit only.** If you need 10, use HEVC, AV1, or x264 on the processor.

Decoding is a separate stack: on a machine without hardware encoding, hardware decoding may work perfectly well. It is checked by its own probes: `CUDA decode`, `QuickSync decode`, `AMD/D3D11VA decode`, and `dav1d` for AV1.

Quality controls always belong to the path: `CRF` and the SVT-AV1 preset on the processor, `CQ/QP` and the NVENC, QuickSync, or AMF presets on hardware. Only x264 and x265 read the fine-tuning list, which is why it appears solely when the processor is selected.

**Run "Hardware Capabilities" before the first run on a new machine.** The result is cached and the profiles are checked against it. A missing path is a normal answer, not a fault: it means the driver or the hardware is absent. The cache belongs to the machine — if it is from another one or missing, run the diagnostics again.

## Why You Can Trust This

Cut accuracy, audio sync, and the limits of FFmpeg here are not claimed — they are measured. The body of measurements rests on one rule: **nothing in it is taken from documentation.** Every cell in every table is an actual attempt, every number a measurement. Where a measurement turned out to be wrong, that is recorded too: a separate section collects the cases where a plausible conclusion had to be withdrawn.

## Where this started

Since version 19.1 Shutter Encoder reads a fractional rate as an integer: 29.97 becomes 30. The 1.001 error accumulates — 1.8 seconds of drift per half hour, four seconds per hour. It is still there in 20.2, and issue #402 remains open.

Here the count runs on an exact fraction:

| rate | fraction | request | correct | via `round(fps)` |
|---|---|---|---|---|
| 23.976 | 24000/1001 | 50 s | **1199** | 1200 |
| 29.97 | 30000/1001 | 4 min | **7193** | 7200 |
| 59.94 | 60000/1001 | 4 min | **14386** | 14400 |

The quantities at stake: a frame is 33 ms at 29.97, an audio packet 8 ms, a sample 20 microseconds. A one-frame shift is what an editor later hunts for by hand.

## Verified on what cameras actually write

Not on generated clips — on footage from real bodies, from Sony to ARRI. The frame discrepancy is zero throughout:

| source | request | frames | discrepancy |
|---|---|---|---|
| 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 starting at 1.44 s | 2 → 8 | 300 | 0 |
| 4 channels of PCM | 2 → 8 | 150 | 0 |
| Canon HEVC 4:2:2 10-bit | 3 → 10 | 166 | 0 |
| ARRI ProRes 4444 XQ 24.000 | 2 → 8 | 144 | 0 |

One finding stands out: an Alexa Mini shooting a true 24.000 declares `24/1` — so reading that file as 23.976 **creates** the 1.001 error rather than removing it. Five frames over four minutes, exactly what other tools are criticised for. The rate is taken from the file as written; one that cannot be read is refused rather than replaced with a guess.

### Sync measured, not assumed

A marker present in both streams at the same instant: a white frame at the top of every second and a 20 ms click at the same moment. Deviation from the source's own offset:

| container | change |
|---|---|
| MP4 | 0…2 ms |
| MOV | 0…5 ms |
| MKV | 1…3 ms |
| MXF | 9…18 ms |

Only MXF shifts anything — its muxer aligns audio to its own edit units, under half a frame, and that is stated as a warning. An earlier warning about MKV was disproved by precise measurement and removed rather than left in as a plausible-sounding caution.

### Seventeen FFmpeg quirks — all compensated

None of them is obvious from the documentation, each cost a separate investigation, and each is worked around in code. Among them: `-ss` before `-i` resets the clock; an end point rounded "to nearest" loses the frame sitting at 29.9966; HEVC with a B-pyramid writes two to four frames fewer than asked; `avg_frame_rate` on variable frame rate turns a 12-second request into 16; `-n` on an existing file answers "already exists" with **exit code 0**, so a run that wrote nothing reads as a success; PCM copied into MXF gains 384 extra samples at the head.

The full list with symptoms is in the measurements, section 10.

The checklist shows what is run before a release. Both documents are in Russian.

## Principles

**Verified on a real file, not a synthetic one.** A generated clip carries neither camera tags nor big-endian audio — and those are exactly what two defects hid behind. The verification matrix is built on footage from actual cameras.

**The form shows what survives, not what is removed.** A list of what will be in the result reads unambiguously; a list of exclusions does not.

**What the container cannot hold is refused before the run, not halfway through.** An incompatible codec and container pairing is blocked in the window and checked again by the service.

**A stopped run leaves nothing that cannot be opened.** An unfinished file does not pretend to be ready.

**A refused overwrite is a skip, not a success.** The report says so.

**Results go into an operation folder, not into the filename.** `Source\Day1\clip.mov` and `Source\Day2\clip.mov` do not collapse into one flat folder and fight over a name: the subfolder structure is reproduced in the output.

## What Is Inside

| page | about |
|---|---|
| Download | a link or a list, format profiles, resolution, subtitles, threads |
| Audio streams | extraction, resampling, normalisation, packaging; audio into video without re-encoding the picture |
| Trimming | frame-accurate or on keyframes, with marks remembered per file |
| Remux, frame rate, colour tables | changing the container, working with frame rate, applying lookup tables |
| Archive | long-term masters: FFV1, lossless x264 |
| Editing codecs | ProRes and DNxHR in MOV or MXF |
| Storage | compact and good: x264, x265, SVT-AV1 |
| Delivery | review, sending, upload |
| Diagnostics | what hardware is present, what it can do, what to encode with |

## Next

- User Guide — working through the pages, the workbench, reports.

- Measurements — the numbers everything rests on.
- Checklist — what is run before a release.
- Decisions — why it works this way, dated and verified.

---

# Technical Reference

## Running

```
launcher_gui.cmd
```

If the environment is not built:

```
builder_main.cmd
```

The builder has separate entries for installing FFmpeg (from two sources), yt-dlp, and 7-Zip into the portable `Tools\` folder.

The window opens at `http://127.0.0.1:8080/`.

## The Workbench

The top row sets the paths for every page:

| button | what it does |
|---|---|
| Source | choose the folder with the originals |
| Add file… | choose a single file as the source |
| Target | choose the output folder |
| Reset | restore the project `Source\` and `Transcoded\`, touching no files |
| Delete | clear the current source and target after confirmation |
| List | print the names of the current source files to the terminal |

Plus `Probe Source` — one inspection pass with a short parameter line.

One shared vocabulary across all Audion projects. In Russian: **Источник**, **Добавить файл...**, **Назначение**, **Сбросить**, **Удалить**, **Список**. The words `Destination`, `Clear`, «Цель», and «Очистить» are not used for these controls.

The source is walked recursively and the subfolder structure is preserved in the output. Results are written into an operation and profile folder:

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

A manually chosen target simply replaces `Transcoded`; the internal structure stays the same.

## Defaults

| what | value |
|---|---|
| quality (CRF/CQ) | 14 |
| audio | 384 kbps; M4A has a 256 profile |
| resampler | SoX/libsoxr at full quality (`precision=33`) |
| download archive | off |

Pixel format is not chosen on the editing-codec page — the target defines it: ProRes is always 10-bit 4:2:2, DNxHR HQ is 8-bit, DNxHR HQX is 10-bit. To change bit depth you pick a different target, not a separate field.

MXF offers 16- or 24-bit PCM or no audio at all; incompatible choices are blocked in the window and checked by the service. FLAC is unavailable for MOV — that container does not hold it; use MP4 or MKV.

## Reports

After audio operations an `audio_report.md` and `.json` are written: the source stream, input parameters, processing mode, loudness, and the output path.

## Deliberately Out of Scope

WebM is currently an input, download, and remux format — not a separate target for bulk encoding. If a VP9 or AV1 delivery in WebM is needed, it is better added as its own profile with explicit quality, container, and audio parameters than bolted onto the existing ones.