

— A2L-AWARE BINARY EDITING

Hexx Technical Brief

A2L-aware ECU binary and hex editor with interactive 3D map editing and automated checksum recomputation.

A2L-Aware Decoding

Physical Unit Editing

Synchronized 3D Surface

Checksum Auto-Recompute

Editing raw hex bytes risks silent binary corruption.

Manipulating ECU memory images without functional symbol context leads to broken tables and invalidated checksums.

ANONYMOUS BYTES

Manual Address Math

Plain hex editors display undifferentiated byte streams, forcing engineers to manually calculate memory offsets and endianness conversions.

NO LIMIT ENFORCEMENT

Raw Byte Nudging

Editing raw integers lacks physical axis breakpoints, engineering units, and upper/lower safety limits, allowing invalid values into flash.

FLASH CORRUPTION

Broken Checksums

Failing to recompute complex flash block CRCs results in bricked ECUs that fail internal bootloader verification checks.

A2L-aware ECU binary and hex editing.

Hexx bridges the gap between raw flash memory and engineering meaning, decoding binary dumps into named calibration characteristics.

SYMBOL DECODING

ASAM A2L Description Pairing

Pairs any Intel HEX, Motorola S19, or BIN image with an ASAM A2L file. Memory addresses automatically resolve into named calibration tables, scalars, and curves.

PHYSICAL UNITS & 3D MAPS

Engineering Cell Editor

Inspect and edit calibration maps in physical engineering units (°C, mbar, RPM). Interactive 3D wireframe plots reflect edits immediately in underlying bytes.

Checksum Protection: Automatically detects checksum algorithms and CRC regions in the ECU image, recalculating them upon saving changes.

Engineering tools for flash memory inspection.

INTERACTIVE 3D SURFACES

Visual Map Manipulation

- **Real-Time Smoothing:** Rotate, inspect, and reshape 3D calibration maps visually with automatic interpolation.
- **Axis Breakpoint Scaling:** Inspect non-linear axis points and verify boundary limits before flashing.
- **Multi-Format Viewing:** Seamlessly switch between byte-trace hex rows, 8/16/32-bit representations, and physical units.

REFERENCE DIFFING

Base vs Modified Comparison

- **Visual Map Diffs:** Side-by-side comparison highlights altered characteristics, shifted addresses, and modified cells.
- **Batch Updates:** Apply verified parameter patches across entire memory ranges while preserving untouched sectors.
- **Flash Safety Checks:** Pre-flash integrity verification checks address alignment and data record completeness.

Exhaustive ECU flash binary formats.

FILE FORMAT	EXTENSION	ENGINEERING CONTENT	OPERATION MODE
Intel HEX	.hex	Extended linear and segmented address records	Read / Edit / Write
Motorola S-Record	.s19, .s28, .s37	16, 24, and 32-bit address flash records	Read / Edit / Write
Raw Binary	.bin	Unstructured raw binary microcontroller memory dumps	Read / Edit / Write
ASAM A2L	.a2l	ASAP2 symbol descriptions, conversion rules, and memory layouts	Read (Symbol decoding)

What we do not claim.

Clear operational parameters for embedded engineering.

NOT A FLASHING DRIVER

No Direct Bootloader Flashing

Hexx is an offline binary and map editor. It does not replace physical CAN flash adapters or hardware bootloader flashing tools (e.g. Vector, Kvaser, ETAS hardware).

PROPRIETARY CRYPTOGRAPHY

Standard Checksum Algorithms

Hexx recalculates standard CRC and checksum blocks. Proprietary OEM cryptographic signatures or encrypted bootloader layers must be signed with official OEM signing tools.

How to evaluate Hexx on your binary drop.

STEP 1

Pair HEX & A2L

Load an Intel HEX or S19 file alongside its corresponding A2L description. Confirm automatic table decoding.

STEP 2

Test 3D Map Editing

Adjust curve values in physical engineering units and observe instantaneous synchronized byte updates.

STEP 3

Verify Checksums

Export the updated binary image and verify calculated checksum integrity against reference test flash images.

STANDALONE TRIAL

Request a Hexx Desktop Evaluation Build

contact@vehiclelab.in · <https://vehiclelab.in/docs/hexx-brief>