H-Mode Physical-Layer Technical Standard

As-Built Normative Specification for the HFEdge H-Mode Modem — H-Mode TS 2.1

Luke McConoughey, N2OSW

2026-07-25

Table of Contents

Document control	Value
Document ID	H-Mode TS 2.1
Revision date	2026-07-25
Status	Normative description of the implemented protocol
Supersedes	Previous technical specification (docs/HMODE_TECH_SPEC.md)
Applies to	H-Mode v2 with in-band configuration negotiation, and the bit-exact H-Mode-1.0 classic profile
Implementation	include/hfedge/modem/hmode*.hpp, src/modem/hmode_*.cpp
Configuration schema	docs/hmode_config.schema.json
Mode metadata string	H-MODE
Author	Luke McConoughey, N2OSW
Copyright	© 2026 Luke McConoughey, N2OSW. All rights reserved.

This is the H-Mode physical layer as it actually exists in the HFEdge source tree on the revision date - not a wish list and not a sketch of a future implementation. It supersedes the previous technical specification in full. Since that revision, the implementation has gained in-band loading-map signaling; in-band configuration negotiation using parameter and profile/fingerprint descriptors; validated mid-stream markers with timing re-centring and automatic re-acquisition; closed-loop carrier-frequency-offset (CFO) and sampling-clock-offset (SCO) tracking on all waveform families; mid-stream AFDM markers; a deterministic dual-layer enhancement gate; full complex-IQ OTFS (ISFFT) and AFDM (DAFT) paths; a generalized (non-classic) OFDM waveform with a deterministic resource plan; a versioned on-air profile registry with configuration fingerprints; and an exact sample-count duration model.

The words shall, shall not, should, and may carry their usual standards meaning here: requirement, prohibition, recommendation, and permission. They define software behavior and interoperability. They don't replace an operator's regulatory obligations.

---

1. Scope and architecture overview

H-Mode is a configurable multicarrier audio-passband modem for carrying HFEdge characters over HF SSB channels. For SDRs and IQ-capable transceivers, there's also an optional complex-IQ baseband path. The payload is a 6-bit character stream - Base36 plus markers, or Base64url - and it carries no payload forward error correction. That's deliberate. HFEdge is meant to measure the substitutions, erasures, and insertions produced by an ionospheric path, not hide them inside the modem. Every decoded character therefore comes out with a confidence value and six per-bit log-likelihood ratios (LLRs); the receiver can also report per-resource channel-state information (CSI).

The OFDM family appears in two forms - classic and generalized - alongside OTFS, AFDM, and Hybrid. They all share the same outer framing, acquisition, in-band signaling, and soft-output architecture:

Family	Enum / JSON name	Core transform	Role
Classic OFDM	ofdm with ofdm.classic_bins = true	128-point FFT, fixed H-Mode-1.0 bin plan	Bit-exact H-Mode-1.0 compatibility
Generalized OFDM	ofdm with ofdm.classic_bins = false	Configurable nfft/ncp OFDM symbol with a deterministic pilot/guard/data plan (§7.2)	Production single-carrier-per-unit waveform (ofdm_robust_v1, ofdm_standard_v1)
OTFS	otfs	Real-audio path: M×N time–frequency grid as Hermitian OFDM columns. IQ path: true ISFFT delay-Doppler grid with Heisenberg synthesis (§7.3)	Primary robust/balanced advanced waveform
AFDM	afdm	Real-audio path: Hermitian IFFT with diagonal chirps. IQ path: full complex DAFT with both chirps (§7.4)	Chirped waveform for doubly selective channels
Hybrid	hybrid	OTFS-family core with hierarchical packing; optional dual-layer sparse-OFDM enhancement on reserved bins (§7.5)	Highest-rate layered waveform (hybrid_v1)

Nothing needs to be recompiled just to retune the modem. Every parameter is supplied through the JSON-serializable HModeRuntimeConfig (§12). Waveform synthesis always runs at the native 8000 Hz rate; when io.sample_rate_hz differs, the transmit output is linearly resampled (§6.7).

1.1 Protocol layering

On the air, a transmission arrives in this order:

1. A **classic acquisition sync block** — three classic 144-sample symbols: preamble A, preamble B, header (§5.2). All waveform families emit this identically.
2. For advanced waveforms (everything except classic OFDM), optional **auxiliary signaling symbols** using the classic symbol format: three configuration-descriptor symbols when config_announce is enabled (§8), then three loading-map descriptor symbols when adaptive loading is signaled (§9).
3. **Data units**: classic data symbols, generalized-OFDM symbols, AFDM symbols, or OTFS frames.
4. **Mid-stream markers**: after every sync.period_data_units data units, advanced waveforms re-insert one preamble-A symbol (followed by three map symbols when adaptive signaling is active); the classic waveform re-inserts a full three-symbol sync block. Markers support timing re-centring, closed-loop CFO/SCO tracking, late join, and automatic re-acquisition (§10.4).

Put compactly, the auxiliary-symbol order at the start of an advanced transmission is A, B, header, config ×3 (if announced), map ×3 (if adaptive), data. At a mid-stream marker, it's A, map ×3 (if adaptive), data. Figure 1 illustrates the sequence.

Figure 1. *On-air framing: initial acquisition (sync block + config/map descriptors) and the recurring mid-stream marker context.*

2. Fixed physical constants (classic layer)

These are the fixed nuts and bolts in hmode_params.hpp. They govern the classic OFDM symbol and the acquisition sync block shared by every waveform family.

Constant	Value	Derivation / note
kHModeSampleRate	8000 Hz	Native generation rate (§6.7 for resampling)
kHModeNfft	128	Radix-2 FFT
kHModeNcp	16	Cyclic-prefix samples
kHModeSymbolSamples	144	N + CP
Symbol duration	18 ms	144 / 8000
Symbol rate	55.56 symbols/s	1 / 18 ms
Bin spacing	62.5 Hz	8000 / 128
kHModeDefaultCenterHz	1500 Hz	Informational; classic TX is real passband, no mixing
Data carriers	9	Bins {8, 11, 14, 20, 23, 26, 32, 35, 38} → 500–2375 Hz
Pilot carriers	4	Bins {5, 17, 29, 41} → 312.5 / 1062.5 / 1812.5 / 2562.5 Hz
PAPR-reserved carriers	2	Bins {44, 47} → 2750 / 2937.5 Hz
kHModeBitsPerChar	6	
kHModeCharsPerSymbolThroughput	3	9 data bins × 2 bits (QPSK) = 18 bits
kHModeCharsPerSymbolRobust	1	First 3 data bins only
kHModePadIndex	63	Base36 short-group pad (H-Mode-1.0); see hmode_pad_index()
kHModeDataSymbolsPerSync	16	Classic sync cadence default
Throughput rate	≈166.7 characters/s	3 × 55.56
Robust rate	≈55.6 characters/s	1 × 55.56
kHModePaprClipCrestDb	6.0 dB	Classic clip+filter crest target

Constant	Value	Derivation / note
kHModePreambleTag	"HMODE1"	Preamble data-bin content
kHModeConfigSymbols	3	Configuration-descriptor symbols (54 bits)
kHModeMapSymbols	3	Loading-map descriptor symbols (54 bits)
kHModeLoadingGroups	16	Loading-map quantization groups

A classic bin k is **active** only when assigned to pilots, data, or PAPR reservation. HModeFft::band_limit forces every other bin to zero, including DC and the upper half-spectrum. With the classic bin plan, active carrier centers run from 312.5 to 2937.5 Hz, a span of 2625 Hz. Transmit filtering, hardware response, and operator configuration determine the measured occupied bandwidth. Figure 2 shows the plan.

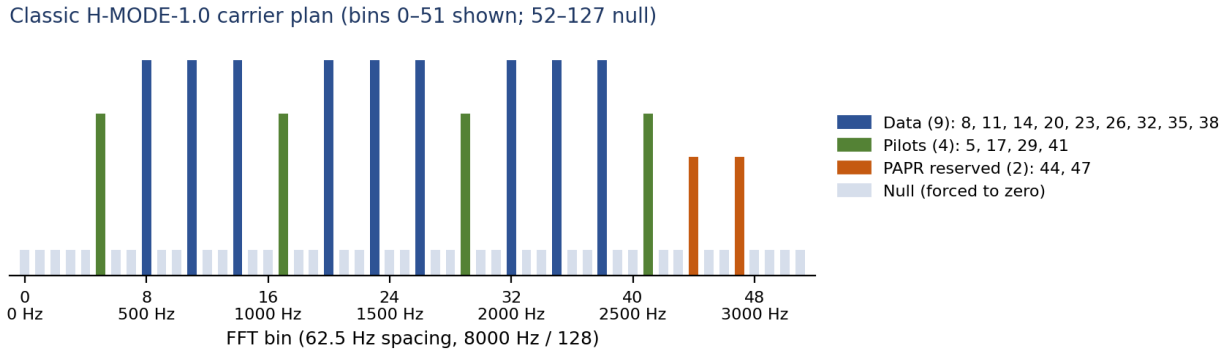

Figure 2. *Classic 128-bin carrier plan: 9 QPSK data carriers, 4 LFSR pilots, and 2 PAPR-reserved tones; every other bin is forced to zero.*

3. Alphabet and character coding

Each character carries 6 bits, MSB-first.

Base36 (HModeAlphabet::Base36, default). Indices 0–35 map to 0-9A-Z through the shared hfedge::base36 tables. Index **36** represents =, the HFEdge frame marker and a 6-bit extension to Base36. Index **63** pads unused character slots in the final classic packing group and is dropped by the receiver. During decode, values greater than 36 are reduced modulo 36 after pad removal.

Base64url (HModeAlphabet::Base64url). Indices 0–63 map through hfedge::kBase64UrlAlphabet (A–Za–z0–9-_), including _ at index 63. Because every 6-bit value is a payload codepoint, Base64url has **no in-band pad index** (hmode_pad_index returns −1). Classic Throughput short groups leave unused character carriers nulled; the receiver drops those slots by carrier-power threshold (5% of mean pilot power) rather than by codepoint, so _ is transported losslessly.

Unknown input characters encode as index 0. On loaded (advanced) paths, character bits are flattened into a continuous bit stream (§9.4); pad handling applies per emitted 6-bit group.

4. Modulation primitives

4.1 QPSK (2 bits)

hmode_qpsk_encode(b0, b1) → (I, Q) with I = +1/√2 if b0 = 0 else −1/√2, and Q = +1/√2 if b1 = 0 else −1/√2. Decoding derotates by a scalar phase reference φ:

```
x = Re(s)·cosφ + Im(s)·sinφ      llr0 = 2x     b0 = (llr0 ≥ 0) ? 0 : 1
y = Im(s)·cosφ − Re(s)·sinφ      llr1 = 2y     b1 = (llr1 ≥ 0) ? 0 : 1
```

4.2 Loaded constellations (0 / 1 / 2 / 4 bits per resource)

hmode_mod_bits(bits, value) in hmode_loading.cpp:

- **0 bits:** transmit a nulled resource.
- **1 bit (BPSK):** Re = +1 for bit 0, Re = −1 for bit 1.
- **2 bits (QPSK):** per §4.1; value bit 0 → b0, value bit 1 → b1.
- **4 bits (rectangular 16-QAM):** I,Q ∈ {±1, ±3}/√10, unit average power. Value bits: b0/b1 select I/Q sign (0 → +, 1 → −); b2/b3 select amplitude (0 → 3, 1 → 1). Demapper: sign bits threshold at 0; amplitude bits threshold at 2/√10; amplitude LLRs are llr_scale · (|x| − 2/√10). LLR sign convention matches classic QPSK (positive prefers bit 0).

Per-resource transmit power weighting: the encoder multiplies each resource by √power from the loading map. In the interoperable signaled mode, power is forced to 1.0 (§9.2). Figure 3 shows both constellations with their bit labels and decision thresholds.

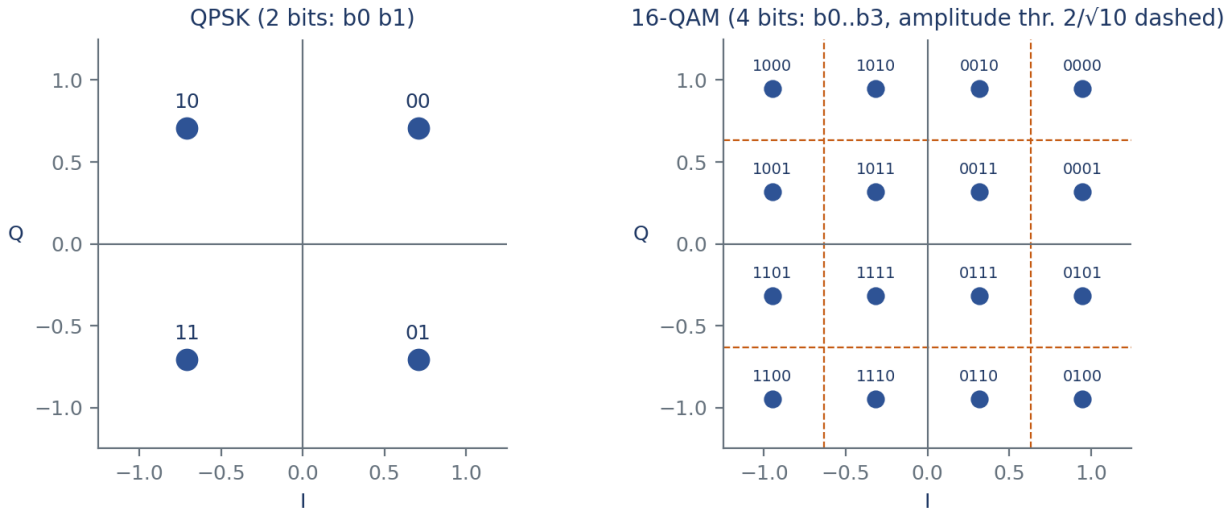

Figure 3. *H-Mode constellations with bit labels (b0 first): QPSK core and rectangular 16-QAM with the 2/√10 amplitude decision thresholds.*

Known limitation (normative caveat). The 4-bit demapper thresholds on absolute amplitude, but each received unit is RMS-normalized; without the data-aided equalizer providing an amplitude reference, 16-QAM decodes unreliably even on a clean channel. 16-QAM shall only be used where the data-aided tracker is enabled and warm (§14.2).

4.3 CRC

hmode_crc16 is CRC-16/CCITT-FALSE: polynomial 0x1021, init 0xFFFF, no reflection, no final XOR. The header symbol carries its low 6 bits; the configuration and loading-map descriptors each carry its low 12 bits.

4.4 Pilot sequence

Pilots are BPSK values from a 4-bit Fibonacci LFSR, taps at bits 3 and 0 (bit = ((s>>3) ^ s) & 1; s = ((s<<1)|bit) & 0xF), seeded 0x9. hmode_pilot_symbol(p, sym) advances the LFSR p + 4·sym steps from the seed and outputs the next bit as ±1 (1 → +1.0, 0 → −1.0). Every pilot value is regenerable from (pilot_index, symbol_index) alone — the sequence is stateless, public, and non-scrambling. Preamble pilots use symbol-index 0 values; preamble B negates them. Auxiliary symbols (header on advanced paths, config, map) and generalized-OFDM in-symbol pilots always use symbol-index 0, so the receiver needs no transmit symbol counter.

```
std::uint8_t lfsr_step(std::uint8_t& s) {   // 4-bit LFSR, taps 3/0
    const std::uint8_t bit = ((s >> 3) ^ s) & 1u;
    s = ((s << 1) | bit) & 0x0Fu;                  // seed 0x9
    return bit;
}
// hmode_pilot_symbol(p, sym): advance p + 4*sym steps; next bit -> +/-1
```

4.5 Classic symbol synthesis (bit-exact H-Mode-1.0)

hmode_synthesize_symbol(fd, out, fft, amplitude) performs, in order:

1. **IFFT** of the 128-bin frequency vector (inverse scales by $1/\sqrt{N}$; forward is unscaled).
2. **CP prepend**: final 16 time samples copied ahead of the 128-sample body (144 total).
3. **Soft clipping** of the whole block to a crest factor of 6 dB above block RMS.
4. **Re-filtering**: FFT the 128-sample body, zero all null bins (band_limit), inverse transform. The CP retains the clipped, pre-filter samples.
5. **RMS normalization** of the full 144-sample block to io.amplitude (default 0.5).

The transmitted signal is the real part of the complex baseband. No carrier mixing is applied: bin k lands at k · 62.5 Hz in the audio passband. Because only positive-frequency bins are populated, the pre-Re{·} complex signal is analytic; the receiver regenerates it as its correlation reference (§10.2).

---

5. Framing

5.1 Sync block (all waveforms)

A sync block is three consecutive classic 144-sample symbols (432 samples, 54 ms):

#	Symbol	Pilot bins	Data bins
1	**Preamble A**	hmode_preamble_pilot(i, false) = LFSR base values	QPSK of the first 2 bits of successive "HMODE1" tag characters; tag cursor advances after every second data bin
2	**Preamble B**	Negated A pilots	As A, complex-conjugated

#	Symbol	Pilot bins	Data bins
3	**Header**	hmode_pilot_symbol(i, symbol_index) (index 0 on advanced paths)	18 header bits, QPSK across the 9 data bins

5.2 Header payload (18 bits, MSB-first per field)

```
frame[7:0] | profile[1:0] | alphabet[1:0] | crc6[5:0]
```

frame is the low 8 bits of the modulator's 16-bit frame counter (increments per sync block). crc6 is the low 6 bits of CRC-16/CCITT-FALSE over the two packed header bytes {frame, (profile & 3) | (alphabet & 3) << 2}. On receive, a CRC-valid header updates the live profile and alphabet (values ≤ 1 only). The header value repeats modulo 256 and shall be treated as a continuity hint, never as a session identifier.

Advanced-path header decode. The advanced receiver decodes the header symbol (decode_header_symbol_advanced) with symbol-index-0 pilots after discarding preambles A and B, adopting the signaled profile and alphabet. (In the previous specification the advanced header was transmitted but ignored; this is no longer the case.)

5.3 Auxiliary signaling symbols

Config and map descriptors are carried on generic 18-bit classic symbols (hmode_pack_bits18_symbol): symbol-index-0 pilots on the 4 pilot bins, 18 payload bits QPSK-mapped across the 9 data bins. Integrity is checked at the descriptor level (12-bit CRC over the assembled 54 bits), not per symbol.

5.4 Classic data cadence

sync block → K data symbols → sync block → … with K = sync.period_data_units (default 16). Each Throughput data symbol carries 3 characters; Robust carries 1 character on the first 3 data bins. Short final Throughput groups: Base36 fills unused slots with pad index 63 (dropped at RX); Base64url nulls unused character carriers. Data-symbol pilots follow the LFSR sequence indexed by the running symbol index.

Worked example (test vector A): message ABC → 1 sync block (3 symbols) + 1 data symbol = 4 × 144 = **576 samples**.

5.5 Advanced-waveform framing

Advanced transmissions begin with one classic sync block, followed by the aux symbols of §1.1, then native data units:

- **Generalized OFDM**: one nfft + ncp symbol per unit.
- **AFDM**: one size + cp symbol per unit.
- **OTFS / Hybrid**: one frame of M · (N + cp) samples per unit (M OFDM columns).

After every sync.period_data_units data units, the transmitter inserts one preamble-A marker symbol (144 samples), followed by three map-descriptor symbols when adaptive signaling is active. **All three advanced families emit mid-stream markers**, including AFDM and generalized OFDM. (This is a protocol change relative to the previous specification, where AFDM emitted none; receivers predating the change cannot track current AFDM streams.)

5.6 Late join and mid-stream re-lock

The receiver distinguishes a full sync block from a bare mid-stream marker during acquisition by correlating preamble B one symbol after a preamble-A peak (§10.2). A bare marker acquisition consumes only the marker (plus map symbols when adaptive) and resumes data decoding, enabling late join into an in-progress transmission and re-lock after dropouts. A full-block acquisition resets the data-unit counter (data_units_decoded_ = 0), keeping warm-up-gated features in lockstep with the transmitter (§7.5).

6. Transmit processing common to all waveforms

6.1 OFDM column synthesis (advanced real-audio path)

synthesize_ofdm_column(freq, nfft, ncp, rolloff): IFFT, CP prepend, PAPR processing (§13) with band-edge tone-reservation bins when dual-layer is not active, then the optional Tukey window (rolloff > 0; OTFS columns use 0.05), then RMS normalization to io.amplitude. PAPR runs **before** windowing so the clip+filter stage band-limits the rectangular block's clean spectrum; filtering a windowed block would remove the window's spectral spread and distort in-band data. The real part is appended to the output stream.

6.2 IQ column synthesis

synthesize_ofdm_column_iq is identical except that no PAPR crest processing is applied — the IQ chain is linear because crest control is an audio-PA concern — and no Hermitian mirroring or audio band-limit is applied. When modulate_chars_iq() is used, the modulator captures the complex baseband sample-aligned with the real output.

6.3 IQ path gating

The complex-IQ path is selected by io.iq_path = true **on both ends**; it is deliberately not part of the config descriptor and must be set out-of-band. TX API: HModeModulator::modulate_chars_iq() → std::vector<std::complex<float>>; RX API: HModeDemod::feed_iq(). Acquisition sync blocks and markers are emitted as analytic complex baseband, so the same correlator locks unchanged.

6.4 Loading-map preparation at TX

Before emitting data, an advanced modulator with a non-Fixed loading algorithm (adaptive signaling, §9):

1. Optionally resamples an externally supplied per-resource SNR profile (set_snr_profile, nearest-index) to the resource count and runs the configured allocation algorithm.
2. Quantizes the resulting map into 16 group codes (§9.2). An all-zero map is replaced with a uniform map at max(bits_core, 1) bits.
3. **Applies the quantized map to itself** (apply_group_codes), so the transmitter transmits exactly what it signals.
4. Emits the map descriptor after the header/config symbols in every sync context (initial block and every mid-stream marker), incrementing a 4-bit sequence number when the map is re-prepared.

With algo = fixed, no map symbols are emitted and every resource carries bits_core.

6.5 Frame counter and symbol index

The modulator keeps a 16-bit frame counter (incremented per sync block; low 8 bits on air in the header) and a running symbol index that drives the classic pilot LFSR.

6.6 Capacity per unit

Per data unit, capacity in characters is loader.chars_capacity() = total_bits / 6, where total_bits sums map bits over active resources. The final unit is padded implicitly: unfilled stream bits are zero, and decode-side pad/short-group handling applies.

6.7 Output resampling

All synthesis runs at 8000 Hz. When io.sample_rate_hz differs, modulate_chars (and modulate_chars_iq) linearly interpolate the native stream to the target rate as a final step. On-air duration is therefore native_samples / 8000 regardless of the output rate. The receiver performs no sample-rate conversion and assumes input at the configured rate.

---

7. Waveform family details

7.1 Classic OFDM (ofdm, classic_bins = true)

Bit-exact H-Mode-1.0. Data symbols per §5.4; synthesis per §4.5. Profile robust (HModeProfile::Robust) sends 1 character per symbol on data bins {8, 11, 14} and decodes only those bins. The classic path never emits config or map descriptors (config_announce does not apply) and bypasses the configurable PAPR pipeline in favor of the fixed 6 dB clip+filter baked into synthesis.

7.2 Generalized OFDM (ofdm, classic_bins = false)

One nfft + ncp OFDM symbol per data unit with a **deterministic resource plan** derived identically at both endpoints by hmode_ofdm_plan(rt) from ofdm.{nfft, ncp, pilot_spacing, pilot_offset, guard_bins, papr_reserved_bins} and io.{sample_rate_hz, band_low_hz, band_high_hz}:

1. Active bins are the positive bins $k \in [1, \text{nfft}/2)$ whose analog frequency lies within the configured band (hmode_active_freq_bins; fallback $k \in [4, \text{nfft}/2 - 2)$ when the band selects none).
2. From the top of the active set: papr_reserved_bins tone-reservation bins, then guard_bins high guards; from the bottom: guard_bins low guards. The partition never consumes the last remaining bin.
3. Remaining bins interleave pilots and data: position j is a pilot when $j \bmod \text{pilot\_spacing} == \text{pilot\_offset} \bmod \text{pilot\_spacing}$ (pilot_spacing = 0 disables in-symbol pilots). A degenerate all-pilot plan is reclaimed as data.

Pilots are symbol-index-independent BPSK (hmode_pilot_symbol(i, 0)). Guard bins are transmitted null and feed receiver noise estimation; reserved bins remain free for tone reservation. DC and (real path) Nyquist are always null. The IQ path uses the identical plan.

TX: loader packs characters into the plan's data bins; pilots inserted; Hermitian mirror (real path) or analytic spectrum (IQ path); column synthesis per §6.1/§6.2 with ofdm.window_rolloff.

RX (demod_gofdm_frame): CFO derotation, FFT, SCO phase-ramp correction; **least-squares one-tap pilot gain equalization** (complex gain fit against the known BPSK pilots, applied to all data resources before the data-aided tracker); noise estimation from DC, Nyquist (real path), and guard bins; expected-stream pre-training, MMSE equalization once warm, decode, tracker update (§11).

7.3 OTFS (otfs) and the Hybrid core

Grid: M = otfs.delay_bins (default 8) × N = otfs.doppler_bins (default 64), CP otfs.cp (default 8) per OFDM column. One frame = one data unit:

```
samples_per_frame = M · (N + cp)        → 8 · 72 = 576 samples = 72 ms @ 8 kHz
```

Resource selection. Frequency bins k ∈ [1, N/2) whose analog frequency k · fs/N falls within [io.band_low_hz, io.band_high_hz] (defaults 300–3000 Hz) are usable; with defaults, fs/N = 125 Hz gives bins 3…24. The top two in-band bins are reserved (PAPR tone reservation, or the dual-layer enhancement bins when active), leaving 20 core data bins. Core resources occupy all M time rows enumerated m-major: nres = M × core_bins (160 with defaults; Figure 4).

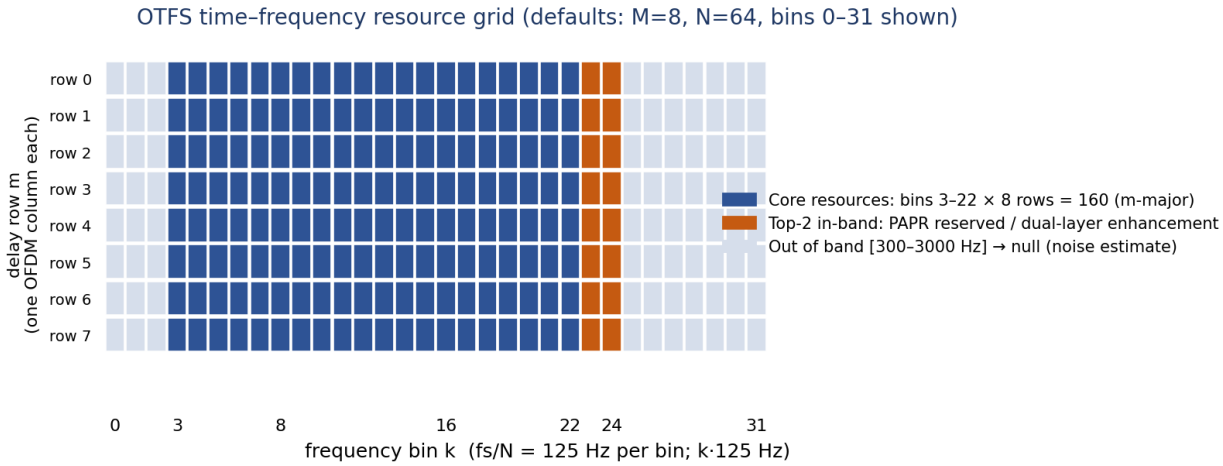

Figure 4. *OTFS real-audio resource grid at defaults: core resources fill bins 3–22 across all 8 delay rows; the top two in-band bins are PAPR-reserved or carry the dual-layer enhancement.*

Real-audio TX: the loader packs up to chars_capacity() characters into the resource vector; resources are placed directly onto the M×N time–frequency grid at (m, fb); each of the M rows is band-limited, Hermitian-mirrored, and synthesized per §6.1 with Tukey 0.05. A full symplectic ISFFT is **not** applied on the real path — transmitting Re{IFFT} of a complex delay-Doppler grid would destroy it.

Complex-IQ TX (io.iq_path): the resource vector is treated as a true delay-Doppler grid; hmode_isfft (row IFFTs over Doppler, column FFT/IFFT pair over delay; row-major X[m·N+n]) maps DD → TF, then Heisenberg synthesis emits one complex OFDM column per delay row using all N bins, no mirror, no band-limit, no PAPR.

RX: per column: CFO derotation, N-point FFT, SCO ramp (real path) or signed-bin fractional-timing ramp (IQ path). Real path gathers the M × core_bins positive-frequency slots; IQ path applies hmode_sfft (TF → DD) and extracts from the DD grid. Out-of-band positive bins (real) or unused DD bins (IQ) feed the null-bin noise tracker. Decode and tracker interaction per §11. Capacity at fixed QPSK over 160 resources: 320 bits (53 characters) per 72 ms frame ≈ 736 characters/s before overhead.

7.4 AFDM (afdm)

Block: N = afdm.size (default 128, radix-2), CP afdm.cp (default 16). Chirp parameters c1 (default 0) and c2 (0 → auto 1/(2N), IQ path only).

DAFT (hmode_daft): forward out = Λ_c2 · F · Λ_c1 · in with Λ_c = diag(e^{+jπ c n^2}); inverse out = Λ_c1^H · F^{-1} · Λ_c2^H · in. The inverse chirp order is reversed relative to forward (conjugate transpose of the

composition); an earlier implementation applied the forward order in the inverse and was corrected when the complex path was connected.

Real-audio path (io.iq_path = false): N/2 − 1 resources on bins 1…N/2−1 (hmode_afdm_resource_count); optional diagonal chirp e^{−jπ(c1+c2)k²} per bin (undone at RX; auto-c2 is not applied because a complex time chirp would break Re{·} invertibility); Hermitian mirror; plain IFFT; CP; PAPR (§13); RMS normalization. RX: CFO derotation, FFT, SCO ramp, chirp removal, then the common data-aided path. DC and Nyquist feed noise estimation.

Complex-IQ path (io.iq_path = true): full DAFT with both chirps and auto c2; data on every non-DC bin, nres = N − 1; linear chain (no PAPR). RX applies the inverse DAFT.

samples_per_frame = N + cp (144 with defaults). AFDM emits mid-stream preamble-A markers on the same cadence as OTFS (§5.5).

7.5 Hybrid (hybrid)

The Hybrid waveform is the OTFS-family path with **hierarchical** core/enhancement packing forced on (loading.hierarchical = true is set by both modulator and demodulator). Two enhancement mechanisms exist:

Same-grid hierarchical enhancement. With an adaptive map, resources loaded above bits_core carry enhancement bits packed after all core bits (§9.4). The receiver **always** decodes enhancement bits: the transmitter packs them unconditionally per the signaled map, so any receiver-local SNR gate would drop bits from the shared stream and desynchronize every character past the core budget. Poor SNR appears as low LLRs, not dropped bits.

Dual-layer enhancement (loading.dual_layer = true). The two highest in-band bins — otherwise PAPR-reserved — carry a sparse OFDM enhancement layer at a fixed bits_enhance_max map across all M rows (nenh = M × 2). Tone reservation is disabled on those bins while dual-layer is active. The transmitter begins packing enhancement characters once units ≥ data_aided.warmup_units; the receiver decodes the enhancement layer once data_units_decoded_ ≥ warmup_units, where data_units_decoded_ mirrors the TX unit counter exactly (reset on full-block acquisition, incremented per frame). Both gates are deterministic and switch on at the same frame. warmup_units is **not** part of the config descriptor and shall match at both ends out-of-band.

The transmitter drains a single character stream into core then enhancement per frame; the receiver appends enhancement characters after core characters, preserving stream order. When the expected-stream tracker is active, it consumes core **and** enhancement characters (consuming only the core count would drift the equalizer training reference).

Decoder behavior note. The receiver decodes the enhancement layer's full capacity every warm frame; frames where the transmitter packed fewer (or no) enhancement characters yield trailing low-confidence garbage characters after the true message ends, as does final-frame core padding. Higher layers delimit messages (HFEdge frame markers) rather than relying on stream truncation.

8. In-band configuration descriptor

With config_announce = true (the default), an advanced transmission announces its digital configuration in-band: 54 bits carried by kHModeConfigSymbols = 3 classic QPSK symbols, sent once immediately after the initial sync block's header. Disable config_announce only when compatibility with a pre-negotiation receiver requires it. Both ends must agree on the setting, and the advanced-waveform bitstream is different when the setting changes.

8.1 Common structure

Payload is 42 bits followed by crc12 = low 12 bits of CRC-16/CCITT-FALSE over the payload packed MSB-first into 6 bytes (last 6 bits zero). All fields are MSB-first. The first 2 bits are the version; Figure 5 shows all three layouts.

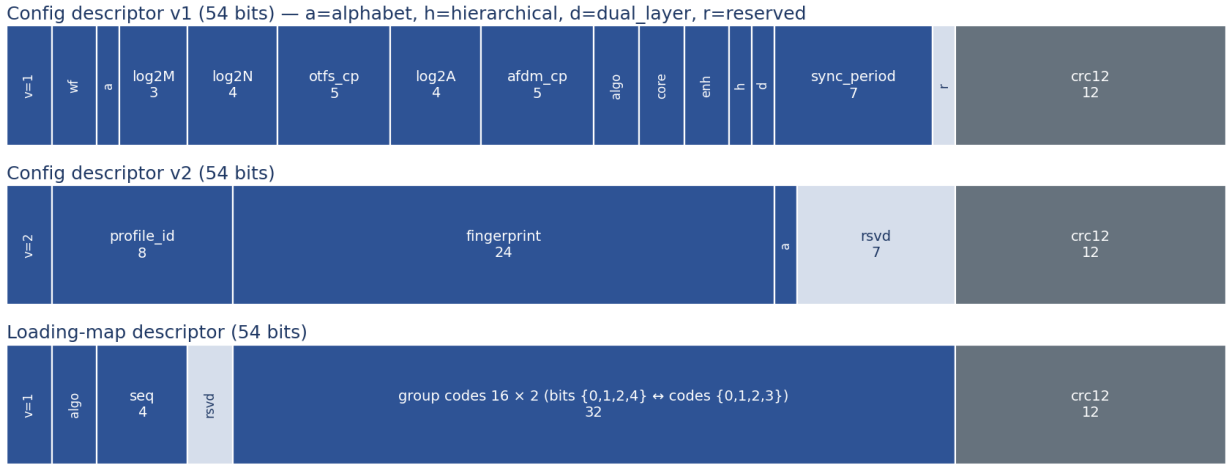

Figure 5. *Bit layouts of the three 54-bit in-band descriptors (all MSB-first; crc12 = low 12 bits of CRC-16/CCITT-FALSE over the 42-bit payload).*

8.2 Version 1 — parameter descriptor

Emitted when the configuration's profile_name is **not** a registered on-air profile but the parameters are representable:

```
version(2)=1 | waveform(2) | alphabet(1) | log2M(3) | log2N(4) |
otfs_cp(5)
| log2_afdm(4) | afdm_cp(5) | algo(2) | core_code(2) | enh_code(2)
| hier(1) | dual(1) | sync_period(7) | reserved(1) | crc12(12)
```

core_code/enh_code use the 2-bit loading code of §9.2. Generalized OFDM reuses the OTFS field slots: log2N ← log2(nfft), otfs_cp ← ncp; on decode with waveform = ofdm the receiver sets ofdm.classic_bins = false and leaves otfs/afdm untouched. AFDM chirps (c1/c2) and all io/band settings remain out-of-band.

Representability: all transform dimensions radix-2 within field range, both CPs ≤ 31, sync.period_data_units ≤ 127, and the waveform is not classic OFDM. An unrepresentable configuration is announced as a valid-CRC **version-0 "no info" descriptor**, keeping the on-air symbol count deterministic; the receiver keeps its configuration.

8.3 Version 2 — profile/fingerprint descriptor

Emitted whenever profile_name is in the on-air profile registry (§12.4):

```
version(2)=2 | profile_id(8) | fingerprint(24) | alphabet(1) |
reserved(7) | crc12(12)
```

On decode, the receiver looks the ID up in its **local** registry, builds a candidate configuration by adopting the registered profile's PHY definition (waveform, ofdm/otfs/afdm blocks, loading, sync, data_aided, band edges) over its local hardware I/O settings, applies the signaled alphabet, and verifies that its own fingerprint of the candidate equals the received fingerprint.

Decode results (HModeDescriptorResult): Invalid (CRC failure / version 0 / malformed — configuration untouched), Params (v1 overlaid), Profile (v2 adopted), UnknownProfile and FingerprintMismatch (**decode refused**).

8.4 Configuration fingerprint

hmode_config_fingerprint(cfg) is the low 24 bits of CRC-32 (IEEE) over a canonical little-endian serialization (format version 2) of every decode-relevant PHY field: waveform, alphabet, iq_path, quantized sample_rate_hz/band_low_hz/band_high_hz (1 Hz), the full ofdm block (window_rolloff quantized ×1000), otfs dims/cp, afdm size/cp/c1/c2 ($\times 10^6$), loading algo/bits_core/bits_enhance_max/hierarchical/dual_layer, sync.period_data_units, and data_aided.warmup_units (fingerprint format version 2 — warm-up gates the dual-layer enhancement on both ends, so a mismatch silently desynchronizes the character stream). io.amplitude, center_hz, and PAPR settings are excluded (TX-local; they do not affect the decoder's resource plan). Both endpoints shall compute identical fingerprints for on-air compatibility; the demodulator also uses fingerprint comparison as its config-changed test when applying a decoded descriptor.

8.5 Receiver application and refusal

On a successful descriptor decode whose fingerprint differs from the current configuration, the receiver applies the signaled configuration via configure_processing() — resizing FFTs, loader, and tracker **without** touching sync lock or buffered samples — and invalidates any held TX map. The receiver auto-configures waveform, alphabet, dimensions, loading, and sync period mid-acquisition; wrong initial dims/waveform/algo/alphabet all recover the payload.

On UnknownProfile or FingerprintMismatch, the receiver sets config_refused(): data units are **consumed but not decoded** (counters still advance, marker cadence maintained) until a later matching descriptor arrives. last_descriptor_result() exposes the outcome.

---

9. Adaptive bit/power loading and in-band map signaling

9.1 Loader model

HModeLoader owns a per-resource map {bits ∈ {0,1,2,4}, power, core_layer}. configure() sizes the map and resets it to bits_core; reset_fixed(bits) sets a uniform map; update_from_snr runs the configured algorithm; maybe_update_from_snr enforces loading.update_period_sec (and is a no-op returning the fixed map under algo = fixed); apply_group_codes applies a signaled 16-group code map with power forced to 1.0.

9.2 Map quantization and 2-bit codes

Bits {0, 1, 2, 4} ↔ codes {0, 1, 2, 3}. hmode_loading_quantize_groups splits the resource map into kHModeLoadingGroups = 16 equal index spans and encodes each group **conservatively** as the minimum bits of any resource in it.

9.3 Loading-map descriptor (TX → RX)

54 bits over kHModeMapSymbols = 3 aux symbols:

```
version(2)=1 | algo(2) | seq(4) | reserved(2) | codes(16×2) | crc12(12)
```

Transmitted after the config symbols in the initial sync block and after preamble A at **every** mid-stream marker (including OTFS resync markers). The receiver applies a CRC-valid descriptor via apply_group_codes when it has no map yet or the 4-bit sequence number changed; on CRC failure it keeps the previous map (the next marker re-signals it). Applying the TX map makes TX and RX bit-exact: the receiver's local SNR never drives the decode map while signaling is active, and the demodulator reconfigures its loader only on a resource-count change (map persists across frames).

Under algo = fixed no map descriptor exists; both ends use the uniform bits_core map. Without config negotiation (config_announce = false), each end determines whether map symbols are present from its **own** loading algorithm — another reason both ends must share the configuration.

Known limitation. A mid-stream map **update** (new sequence number) with Chow or waterfilling loading corrupts one character at the final sync-block boundary (pre-existing boundary bug). Frozen v1 registry profiles avoid it by using fixed loading; interactive balanced/high_speed are affected (§15.3).

9.4 Allocation algorithms (loading.algo)

The bit-count rule bits_for_snr(snr, margin_db, max_bits) with eff = SNR_dB − margin_db: below 0 dB → 0 bits; 0–3 dB → 1; 3–8 dB → 2; ≥ 8 dB → 4 when max_bits ≥ 4, else min(max_bits, 2).

- **fixed** — every resource at bits_core, power 1. Default and the interoperable production mode.
- **chow** (Chow-class margin-adaptive approximation) — bits per the rule; power = max(0.25, $\log_2 \mathrm{SNR}_i$ − mean($\log_2$SNR) + 1); core_layer = (bits ≤ bits_core) when hierarchical.
- **fischer** (Fischer–Huber-class) — bits per the rule; power = max(0.1, $\log_2 \mathrm{SNR}_i$ − water + 1) with water = margin/10 + $\log_{10}$(min SNR); all core.
- **waterfilling** — excess = $\log_{10}\mathrm{SNR}_i$ − $\log_{10}$(min SNR) − margin/10; bits 1/2/4 at excess > 0 / 0.9 / 1.8 decades (4 only when max_bits ≥ 4); power = max(0.05, excess + 1); core_layer = (bits ≤ 2).

Signaled operation quantizes whatever the algorithm produced and forces power to 1.0, so the power values above affect only unsignaled component-level use.

9.5 Bit packing

encode_chars flattens characters into an MSB-first 6-bit stream; resource values consume stream bits LSB-first. Non-hierarchical: resources fill in map order with load.bits each. Hierarchical: pass 1 packs min(bits, bits_core) core bits across all resources; pass 2 packs the remaining enhancement bits (bits − bits_core) on resources loaded above core, ORed above the core bits. decode_chars reverses the order, drops 6-bit groups equal to the alphabet pad index (Base36 only), and emits characters with analog demapper LLRs. include_enhancement = false consumes core bits only (used nowhere in the current receiver paths — enhancement is always decoded).

9.6 SNR feedback loop

The intended closed loop is: far-end receiver measures per-resource SNR (tracker.snr_lin(), exported via CSI/occupancy telemetry) → higher layer feeds it to the transmitting side's HModeModulator::set_snr_profile() → next transmission signals and uses the adapted map. The over-the-air feedback transport is out of scope of this standard. When the receiver's tracker is warm on a non-signaled (fixed/no-map) link, maybe_update_from_snr may adapt the local map for telemetry only.

10. Receiver architecture (HModeDemod)

10.1 Input and state

feed_audio maps each real sample to {s, 0} (no mixing); feed_iq accepts complex baseband. Two sync states: **Hunt** and **Locked**. All timing is maintained in absolute sample counts. Figure 6 summarizes the overall processing flow (samples_seen_) so CFO derotation is phase-continuous across buffer erasures.

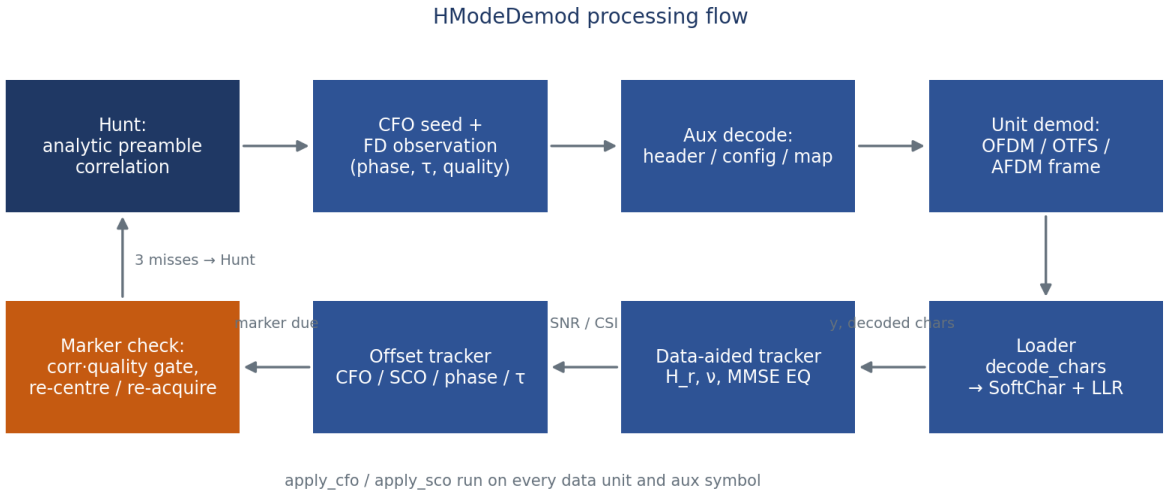

Figure 6. *Receiver processing flow: acquisition seeds the offset tracker, auxiliary symbols negotiate configuration and the loading map, and validated markers close the CFO/SCO tracking loop or trigger re-acquisition.*

10.2 Acquisition (Hunt)

The receiver synthesizes its own **analytic preamble-A reference** (144 complex samples via §4.5 at the configured amplitude, positive-frequency bins only) and computes the normalized analytic cross-correlation

$$\mathrm{corr}(i) = |\Sigma\, rx[i+n]\cdot\mathrm{conj}(ref[n])| \;/\; \mathrm{sqrt}(\Sigma rx^2 \cdot \Sigma|ref|^2 / 2)$$

coarse-stepped by 4 samples with ±4 fine refinement. This magnitude is invariant to carrier phase and degrades gracefully under CFO (a real·real correlation would scale with cos(phase) and could vanish). Lock requires corr > sync.acquire_threshold (default 0.35). On a peak:

1. **Coarse CFO seed** from the phase step between the correlation's two symbol halves (unambiguous to ≈ ±55 Hz), averaged with the same measurement on preamble B when present; seed_cfo() hard-sets the estimate.
2. **Block-vs-marker discrimination** (advanced): preamble B is correlated one symbol later; b_corr ≥ track_threshold ⇒ full sync block, else bare mid-stream marker.
3. **Frequency-domain preamble observation** (preamble_fd_observation): FFT of the known preamble after CFO derotation gives (a) the residual carrier phase as the k=0 intercept of the weighted mean of arg(rx·conj(expected)) (subtracting the slope·$\bar{k}$ term so timing is not corrected twice), (b) fractional timing τ from adjacent-active-bin conjugate products (wrap-immune phase-vs-bin slope; τ = −slope·N/2π), and (c) a coherence quality |Σr| / Σ|r| ≈ 1 for a true preamble.
4. **Bare-marker gate**: a bare-marker acquisition additionally requires corr · quality ≥ acquire_threshold; otherwise the receiver keeps hunting (the analytic correlator's noise floor is higher than the legacy real correlator's).
5. The phase residual is folded into the tracker's constant phase term (adjust_phase) — giving the pilotless OTFS/AFDM QPSK grids an absolute constellation orientation — and τ hard-sets the timing estimate. On a

full block, a second FD observation on preamble B one symbol later refines CFO through observe_marker_phase with gain 1 (±27.8 Hz unambiguous).

6. State is initialized per §5.6; classic mode consumes the preambles as sync-block symbols; advanced modes skip A+B and schedule header/config/map decoding.

While hunting, the oldest reference-length chunk is dropped once the buffer exceeds 8 reference lengths.

10.3 Auxiliary symbol decoding and the acquisition→marker gap

classic_aux_fft FFTs one classic symbol at the current offset with CFO/SCO correction applied and returns the pilot phase reference (symbol-index 0). Each aux symbol's pilot mean phase is chained as a **fine residual-CFO observation** (observe_marker_phase, gain 0.5, when the distance from the previous phase anchor is ≤ 4 symbols; plain adjust_phase otherwise). This closes the correction gap between acquisition and the first mid-stream marker, during which early data units would otherwise integrate the raw seed error.

Header → config ×3 → map ×3 are decoded in that order per §5, §8, §9.

10.4 Mid-stream marker validation, tracking, and re-acquisition

When a marker is due (pending_marker_check_, armed every sync.period_data_units units), the receiver correlates preamble A over ±sync.track_search (default 8) samples around the expected position, keeping that much history across buffer erasures. Validation requires **both** corr ≥ sync.track_threshold (default 0.25) **and** corr · quality ≥ track_threshold, where quality is the FD coherence of §10.2(3) — the product separates a genuinely degraded marker (e.g. 0.44 · 0.67 ≈ 0.29) from noise that clears one metric only (0.29 · 0.52 ≈ 0.15).

On a validated marker: timing **re-centres** on the peak (sync_offset_ = best_idx, recovering sample slips and slow SCO drift); the FD phase residual drives a closed-loop fine CFO update (gain 0.5) with phase re-anchoring; fractional τ blends at α = 0.5; and the integer-plus-fractional timing drift since the previous validated marker, divided by the sample distance, is blended into the SCO estimate (ppm). The marker (144 samples) is then consumed.

A single miss falls back to the legacy blind 144-sample skip on current timing (deep-fade tolerance). sync.max_marker_failures (default 3) consecutive misses with sync.auto_reacquire = true re-enter Hunt — clearing pending state but **keeping** the receive buffer — followed by an immediate re-acquisition attempt over the buffered samples. Combined with bare-marker acquisition (§5.6), this recovers gross dropouts, sample insertions/deletions, and supports late join.

10.5 Classic tracking and decode (per 144-sample symbol)

1. On the first symbol of each periodic sync block, the receiver runs the same FD closed-loop preamble observation as advanced markers (CFO via observe_marker_phase when the anchor distance ≤ 20 symbols, else adjust_phase; τ blend 0.5).
2. CP-strip, CFO derotation, 128-pt FFT, SCO phase ramp.
3. **Phase reference**: mean of arg(rx/expected) over the 4 pilots (preamble pilots during sync symbols 0–1, LFSR pilots otherwise). Pilot ratios also feed observe_pilots (§10.6) on non-header symbols.
4. Sync-block symbol 2 is header-decoded (CRC-checked; updates profile/alphabet).
5. Data symbols: QPSK decode of 9 (Throughput) or 3 (Robust) data bins against the pilot phase reference; per-bit LLRs scaled by llr_scale / 2.
6. **Data-aided refinement** (when enabled): null bins feed the shared noise floor; the tracker updates from the expected stream (regenerated via build_data_fd_from_chars) or decision-directed hard symbols; once warm (warmup_units), symbols are re-derived through the MMSE equalizer and bits/LLRs replaced.

7. Characters assemble 6 bits MSB-first; Base36 pad 63 dropped; Base64url null-carrier slots dropped by power threshold; confidence $\sigma(\text{mean } |\text{LLR}|)$ (stable logistic, floored 10^{-6}, capped 1.0) emitted with the 6 LLRs and running unit index.
8. **Adaptive sync period** (sync.adaptive_period): confidence EWMA ($0.9 \cdot \text{prev} + 0.1 \cdot \text{mean}$) expands the expected period toward sync.period_max above adapt.expand_threshold (0.85) and contracts toward sync.period_min below adapt.contract_threshold (0.55), one unit per symbol; adaptation runs only when adapt.enable is also set. The adapted period is not signaled; classic factory profiles disable adaptation.

10.6 Offset tracking (HModeOffsetTracker)

State: cfo_hz, sco_ppm, fractional timing timing_tau (samples), and a constant carrier phase term. EWMA constant $\alpha = 0.85$ for CFO/SCO blends.

- apply_cfo(time, abs_sample, fs): derotates by $-2\pi \cdot \text{cfo} \cdot t - \text{phase\_offset}$ at absolute sample positions; applied to every unit, aux symbol, and FD observation on all paths (classic, OTFS, AFDM, gOFDM, both real and IQ).
- apply_sco_freq(freq): per-bin phase ramp $e^{+j2\pi k \cdot \tau / N}$ correcting integrated residual timing; the IQ OTFS path applies the equivalent signed-bin ramp.
- observe_pilots (classic): mean pilot phase progression per symbol → CFO EWMA; least-squares phase-vs-bin slope → residual τ, with outlier rejection ($|\tau| > 6$ samples, instantaneous clock error > 500 ppm) and **integration** of the residual (timing_tau += $0.15 \cdot \tau$, clamp ±8) — the residual is measured on the already-corrected spectrum, so EWMA-toward would decay the correction.
- observe_marker_phase(residual, abs, dist, fs, gain): closed-loop update — residual phase over a known sample distance is pure residual-CFO integration; $\Delta\text{cfo} = \text{gain} \cdot \text{residual} / (2\pi \cdot \text{dist}/\text{fs})$ and the phase term is re-anchored (including the $-2\pi \cdot \Delta\text{cfo} \cdot t$ compensation) so alignment at the marker is undisturbed. Gain 1 at acquisition refinement, 0.5 in tracking.
- observe_cfo / observe_sco / observe_timing(τ, α) / adjust_phase / seed_cfo: blends, hard-set, and phase-fold operations as described above. Phase-anchoring operations reset the incremental pilot-phase chain (a phase step would otherwise corrupt it).
- observe_channel_phase_m4: data-driven CFO from the modulation-stripped ×4 domain (summed 4th powers of PSK resources; phase difference wrapped in the ×4 domain and divided by 4, avoiding the ±π branch cut). **Gated off by default** (data_aided.cfo_from_data = false): the observable is confounded by the per-unit mixer-phase alias (units start at arbitrary absolute samples) and reads as false CFO even on a clean channel. Marker/aux-based tracking supersedes it.

The operational design target is **±20 Hz** residual carrier after transceiver AFC and ±100 ppm sampling-clock error; the conformance suite (§14.1) verifies both, with final CFO estimates within ≈0.2 Hz on advanced paths. Estimates are exported via cfo_hz(), sco_ppm(), and occupancy telemetry.

10.7 Data-aided channel tracker (HModeDataAidedTracker)

Per-resource single-tap channel estimate H_r, initialized 1+0j.

- **Update:** $H_r \leftarrow \alpha \cdot H_r + (1-\alpha) \cdot (y_r / x_r)$ with $\alpha = \text{clamp}(\text{forgetting}, 0, 0.999)$ (default 0.92), skipping near-zero known symbols; per-resource residual $\nu_{\text{res},r} \leftarrow \beta \cdot \nu_{\text{res},r} + (1-\beta) \cdot |y_r - H_r x_r|^2$ with $\beta = 0.5\alpha + 0.4$. Each call counts one warm-up unit; steady state at units_seen ≥ warmup_units.
- **Null-bin noise floor:** observe_null_bins EWMA-updates a shared ν_{null} from unused bins (classic null bins; OTFS out-of-band/unused-DD bins; AFDM DC/Nyquist; gOFDM DC/Nyquist/guards).
- **Effective noise:** $\nu_{\text{eff}} = 0.65 \cdot \nu_{\text{null}} + 0.35 \cdot \nu_{\text{res},r}$ when both exist, else whichever exists (floor 10^{-8}).

- **Equalizer (MMSE-ish, iterative):** $\hat{x} = H^{*} \cdot y / (|H|^2 + \nu)$ seeded from ν_eff, refined between iterations by $\nu \leftarrow 0.85\nu + 0.15 \cdot |y - H\hat{x}|^2$; equalizer_iterations (default 2) passes.
- **Reference selection:** with an expected stream installed (set_expected_stream), advanced paths pre-train from the regenerated reference before decoding and consume decoded characters (core + enhancement); without one, decision-directed re-encoding of decoded characters updates the tracker after decode. require_expected_stream = true suppresses decision-directed updates entirely.
- **CSI:** snr_lin() = $|H_r|^2 / \nu$_eff (floored 10^{-6}); csi_snapshot() returns {resource_id, snr_lin, |H|, arg H, noise_var} per resource.

Operators may disable the tracker (data_aided.enable = false) for blind operation; the classic profile does so.

10.8 Config-refused consumption

While config_refused() is set, demod_unit advances all counters and marker scheduling but decodes nothing (§8.5).

---

11. Soft-metric interface

Per decoded character, via the DemodBase callbacks:

```
SoftChar { char c; float confidence ∈ (0,1]; float llr[6]; uint32
ofdm_index; }
HModeResourceCsi { size_t resource_id; float snr_lin; float mag; float
phase; float noise_var; }
```

confidence = σ(mean|llr|). ofdm_index is the running demodulated-unit counter (symbols for classic including sync symbols; frames/symbols for advanced waveforms). CSI emission requires emit_resource_csi = true and a registered callback. hmode_resource_occupancy() (and the modulator/demodulator occupancy APIs and set_occupancy_callback) export the configured/live resource map {resource_id, delay_row, freq_bin, bits, power, core/enhance} plus occupancy fraction, total bits, and current CFO/SCO; dual-layer enhancement entries are always included at the fixed bits_enhance_max map.

12. Configuration

12.1 Structure and field reference

HModeRuntimeConfig is JSON round-trippable via hmode_config_to_json / hmode_config_from_json / _load_file / _save_file; parsing is merge-style (absent fields keep prior values) and ends with validation (§12.2). Full field set and defaults:

Group	Field	Default	Notes
root	waveform	ofdm	ofdm / otfs / afdm / hybrid
	profile	throughput	throughput / robust (classic v1 mapping)
	alphabet	base36	base36 / base64url
	profile_name	"classic"	Registry membership selects the v2 descriptor (§8.3)

Group	Field	Default	Notes
	llr_scale	2.0	LLR output scaling
	emit_resource_csi	false	Enables CSI callback
	log_level	warn	off/error/warn/info/debug/trace
	config_announce	true	In-band config descriptor (§8). **Not** JSON-persisted; set programmatically; both ends must agree
io	sample_rate_hz	8000	TX resamples when ≠ 8000 (§6.7)
	center_hz	1500	Informational (no mixing)
	amplitude	0.5	Per-block RMS target, range [0, 1]
	iq_path	false	Complex-IQ TX/RX path (§6.3); out-of-band, both ends
	gpsdo	false	Declarative; no signal-processing effect
	band_low_hz / band_high_hz	300 / 3000	Resource band for OTFS/gOFDM; column band-limit
ofdm	nfft / ncp	128 / 16	Radix-2; classic uses 128/16
	classic_bins	true	True ⇒ bit-exact H-Mode-1.0; false ⇒ generalized OFDM
	window_rolloff	0.05	Tukey rolloff (classic forces 0)
	pilot_spacing / pilot_offset	8 / 4	gOFDM plan (§7.2); spacing 0 disables pilots
	guard_bins / papr_reserved_bins	2 / 2	gOFDM plan; validated ≤ 16 / ≤ 8
otfs	delay_bins (M) / doppler_bins (N) / cp	8 / 64 / 8	Radix-2 required
afdm	size / cp / c1 / c2	128 / 16 / 0 / 0	c2 = 0 → auto 1/(2·size) (IQ path only)
sync	period_data_units	16	Data units between sync contexts
	period_min / period_max	8 / 32	Classic adaptive bounds
	acquire_threshold	0.35	Acquisition correlation threshold
	adaptive_period	true	Classic RX-side adaptation (§10.5)
	track_threshold	0.25	Marker validation threshold (§10.4)
	track_search	8	Marker search half-window, samples
	max_marker_failures	3	Consecutive misses before re-acquisition
	auto_reacquire	true	Return to Hunt after repeated marker misses
data_aided	enable	true	
	warmup_units	4	Steady-state / dual-layer gate; must match both ends
	pilot_density_steady	0.0	Declarative (no steady-state pilots implemented)
	equalizer_iterations	2	
	forgetting	0.92	EWMA α

Group	Field	Default	Notes
	require_expected_stream	false	Suppress decision-directed tracker updates
	cfo_from_data	false	Data-driven CFO estimator (§10.6); off by default
loading	algo	fixed	fixed/chow/fischer/waterfilling
	bits_core / bits_enhance_max	2 / 4	
	hierarchical	true	Forced true for Hybrid
	dual_layer	false	Hybrid reserved-bin enhancement (§7.5)
	update_period_sec / snr_margin_db	0.5 / 3.0	Allocation gate and margin
papr	method	iterative_clip_tr	§13
	crest_db / iterations / slm_candidates / ace_extend	6.0 / 2 / 4 / 0.15	
bonding	enable / channel_center_hz	false / [1500.0]	Declarative (single instance per channel)
adapt	enable	true	Classic period adaptation requires this AND sync.adaptive_period
	expand_threshold / contract_threshold	0.85 / 0.55	Classic confidence thresholds

12.2 Validation

hmode_config_validate fails closed; hmode_config_from_json returns its result. Rules: ofdm.nfft, otfs.delay_bins, otfs.doppler_bins, afdm.size shall be non-zero powers of two; each CP ≤ its transform length; 0 ≤ band_low < band_high < sample_rate/2 with finite positive sample rate; generalized OFDM additionally requires guard_bins ≤ 16, papr_reserved_bins ≤ 8, and a non-empty data-bin plan; amplitude ∈ [0, 1]; period_data_units ≥ 1, period_min ∈ [1, period_max], acquire_threshold ∈ (0, 1]. The companion JSON Schema (docs/hmode_config.schema.json, draft 2020-12, additionalProperties: false) mirrors these constraints for external tooling.

12.3 Interactive factory profiles

hmode_config_named_profile(name); aliases in parentheses; unknown names fall back to classic.

Profile (aliases)	Waveform	Key deltas from defaults
classic (hmode1, 1.0)	Classic OFDM	data_aided off; fixed QPSK non-hierarchical; ClipFilter 6 dB ×1; adapt off; sync 16 fixed; rolloff 0
robust	OTFS 8×64 cp 8	Robust profile; sync 8 fixed; data-aided (warmup 4, eq iters 3, forgetting 0.90); fixed QPSK; ClipFilter 6.0 dB; adapt on, contract 0.45
balanced (default)	OTFS 8×64 cp 8	robust + Throughput; sync 16 adaptive [8, 24]; Chow 2−4 hierarchical, margin 3 dB, update 0.5 s; ClipFilter 6.0 dB
high_speed (highspeed, throughput)	Hybrid, dual_layer	balanced + sync 32 adaptive [8, 32]; waterfilling 2−4 hierarchical, margin 1.5 dB, update 0.25 s; SelectiveMapping ×8; pilot_density_steady 0.05

hmode_config_from_legacy(HModeConfig) maps the v1 struct (fs, amplitude, profile, alphabet, center) onto the classic profile (classic / classic_robust).

Caution. high_speed as defined ships SelectiveMapping PAPR and adaptive waterfilling, both of which carry known end-to-end defects (§15.3); the frozen hybrid_v1 registry profile is the interoperable variant. Interactive balanced retains Chow loading and is exposed to the mid-stream map-update boundary bug.

Example: a minimal JSON file selecting the frozen OTFS registry profile (omitted fields keep their defaults; the file must still pass validation):

```
{
  "profile_name": "otfs_robust_v1",
  "waveform": "otfs",  "alphabet": "base36",
  "io":   { "sample_rate_hz": 8000, "amplitude": 0.5,
            "band_low_hz": 300, "band_high_hz": 3000 },
  "otfs": { "delay_bins": 8, "doppler_bins": 64, "cp": 8 },
  "sync": { "period_data_units": 8, "acquire_threshold": 0.35,
            "track_threshold": 0.25, "auto_reacquire": true },
  "loading": { "algo": "fixed", "bits_core": 2 }
}
```

Integration sketch — the complete public path from a named profile to decoded characters:

```
// End-to-end use of a registered profile (TX and RX must both validate).
auto cfg = hmode_config_named_profile("otfs_robust_v1");
HModeModulator tx;  tx.configure(cfg);
std::vector<float> audio = tx.modulate_chars(msg);  // native 8 kHz

HModeDemod rx(DemodConfig{}, cfg);
rx.set_char_callback([](char c) { putchar(c); });
rx.feed_audio(audio);                          // or feed_iq(...)
```

12.4 On-air profile registry (v1 tiers)

Registered profiles are frozen, versioned definitions negotiated by ID + fingerprint via the version-2 descriptor (§8.3). Any change to a fingerprinted field requires a new _v2 name. hmode_profile_registry() / hmode_profile_id(name):

ID / Name	Waveform / dimensions	Loading / sync / PAPR	Notes
1 | ofdm_robust_v1	gOFDM nfft 256 / ncp 32, band 300–2800, pilot spacing 4 offset 2, guards 2, reserved 2	Loading: Fixed QPSK non-hier. Sync: 8 fixed PAPR: ClipFilter 6.0	Dense pilots, data-aided on
2 | ofdm_standard_v1	as ID 1; band high 3000, pilot spacing 8 offset 4	Loading: Fixed QPSK Sync: 16 fixed PAPR: ClipFilter 6.0	Moderate pilot density
3 | otfs_robust_v1	OTFS 8×64 cp 8	Loading: Fixed QPSK Sync: 8 fixed PAPR: ClipFilter 6.0	= robust frozen
4 | otfs_balanced_v1	OTFS 8×64 cp 8	Loading: Fixed QPSK non-hier. Sync: 16 fixed (pinned) PAPR: ClipFilter 6.0	Chow disabled (map-update boundary bug, §15.3)
5 | hybrid_v1	Hybrid 8×64 cp 8, dual_layer	Loading: Fixed QPSK hierarchical Sync: 32 fixed (pinned)	SLM and adaptive maps disabled (§15.3)

ID / Name	Waveform / dimensions	Loading / sync / PAPR	Notes
		PAPR: ClipFilter 6.0	
6 | afdm_v1	AFDM 128 / cp 16, c1 = c2 = 0 (auto)	Loading: Fixed QPSK Sync: 8 fixed PAPR: ClipFilter 6.5	Extra crest headroom for the size-128 DAFT symbol

All six pass noise-free loopback and v2 descriptor negotiation in the conformance suite; fingerprints and IDs are asserted unique.

12.5 Duration estimation

hmode_duration_seconds_cfg(profile, cfg) is an **exact sample-count model** of HModeModulator::modulate_chars, verified bit-exact against the modulator across all waveform families (AFDM markers included) and loading modes:

- Classic: whole data symbols (3 chars Throughput / 1 Robust) + 3-symbol sync blocks (initial + every period).
- Advanced: 3-symbol acquisition + config symbols (if announced) + map symbols (if adaptive); per-unit capacity from the quantized signaled map (bits_core per resource; 1 bit when bits_core = 0 under adaptive signaling); whole frames of M(N+cp) / (size+cp) / (nfft+ncp) samples; mid-stream markers of 1 + map symbols classic symbols every period (all advanced families, AFDM included); Hybrid dual-layer enhancement capacity counted only for units ≥ warmup_units. Duration = native samples / 8000 Hz, independent of io.sample_rate_hz.

If the on-air structure changes (marker cadence, aux symbols, warm-up gating), the estimator shall be updated in lockstep and re-verified against modulate_chars(...).size().

---

13. PAPR pipeline (hmode_papr_process)

Operates in place on one time-domain block (CP + body); the body (last nfft samples) is processed and the **CP is rebuilt from the modified body** afterward, so the block remains a true cyclic extension. All internal FFT round trips are normalized (forward_norm scales by 1/√N, pairing with the 1/√N inverse). papr.method:

Method	Behavior
none	Pass-through (stats only)
clip_filter, tone_reservation, ace	One pass of the combined clip+filter routine
selective_mapping	SLM: slm_candidates deterministic phase rotations e^{j2πc/C} applied to the lower half-spectrum; lowest-crest candidate wins. **Not decode-transparent — see §15.3**
iterative_clip_tr	papr.iterations (default 2) passes of clip+filter; tone reservation is applied on the final pass only (reserved tones added earlier would be re-clipped by the next pass into unfiltered in-band distortion)

Clip+filter routine: threshold = RMS · 10^(crest_db/20) clamped to [RMS, peak]; magnitude clip; normalized FFT; scale non-zero bins by 1 + ace_extend (active constellation extension); band filter — when a sample rate is supplied, on **folded** absolute frequency min(k, N−k)·fs/N outside [lo, hi] (preserving Hermitian pairs so the signal stays real; used by the gOFDM/OTFS columns and the AFDM path), else the classic null-bin band_limit; IFFT. No re-clip follows the filter (a post-filter clip would inject unfiltered in-band distortion; iterative schemes re-clip at the start of the next pass instead). Finally, when reserved bins are supplied, the clipping residual

(clipped input minus filtered block) is transformed and **projected onto the reserved bins only** — including their Hermitian mirror partners so the time signal stays real — and added back (tone reservation). hmode_crest_db reports peak/RMS before/after via HModePaprStats.

Defaults: crest 6.0 dB, 2 iterations, 4 SLM candidates, ACE extend 0.15. The classic path bypasses this pipeline (§7.1); the IQ paths are linear and never apply it. The AFDM real path applies it without reserved bins and passes the sample rate with a band of [df/2, fs/2 − df/2] (df = fs/size), so the filter keeps every AFDM data bin and nulls only DC and Nyquist — band-matched for any afdm.size. **Distortion note:** with tone reservation now genuinely confined to the reserved bins, clipping distortion is no longer re-added in-band; deep iterative clipping (iterative_clip_tr at crest targets below ≈8 dB) accumulates in-band EVM per pass — each pass re-derives its threshold from the filtered, lower-RMS body and clips deeper — and can flip occasional QPSK symbols even noise-free. Profiles therefore use a single clean ClipFilter pass (§12.3/§12.4).

---

14. Conformance and verification

14.1 Automated test coverage (all green at the revision date)

The CTest suite comprises 15 targets; those exercising this standard:

Suite	Coverage
test_hmode (Catch2, 43 cases / 677 assertions)	Classic identity and 576-sample ABC vector, 40 dB AWGN zero-CER, Watterson initial acquisition, RMS determinism, Base64url short groups, header/CRC/LFSR primitives, JSON round trip + unsafe-dimension rejection, OTFS loopback and marker alignment, 16-QAM mapper identity, loading component behavior and update gate, tracker SNR response, AFDM prefix recovery, occupancy telemetry, dual-layer Hybrid 124-character full-recovery with enhancement bits asserted in TX and RX occupancy, registry consistency (unique IDs/names/fingerprints, v2 emission), noise-free loopback of all four advanced v1 profiles, v2 negotiation from a gOFDM receiver, warmup-units fingerprint sensitivity, AFDM duration exactness with markers, adapt.enable period freeze
test_adaptive_loading_e2e	In-band map signaling end-to-end (Chow/Fischer/waterfilling), TX=RX map equality, hybrid hierarchical same-grid test (core 1 / enh 2) with full payload recovery
test_config_negotiation_e2e	Wrong dims / waveform / algo / alphabet at RX all recover the payload; announce-off matched-config operation
test_resync_e2e	Clean multi-unit head+tail, 5-sample deletion and insertion, deletion under adaptive map signaling, gross dropout (2.5 zeroed symbols + 100-sample cut) → Hunt → mid-stream relock, late join (first 40% dropped), classic regression
test_offset (42 cases)	Passband loopback with analytic-heterodyne frequency shift and Catmull-Rom clock resample: classic/OTFS/AFDM × CFO {0, ±5, ±12, ±20 Hz} × SCO {0, ±50, ±100 ppm} and combos; final CFO estimate within ≈0.2 Hz on advanced paths
test_iq	ISFFT/SFFT and DAFT round trips, OTFS/AFDM IQ loopback (exact payload), chirps, capacity, ±CFO, adaptive map, AWGN, hybrid dual-layer IQ, mid-stream deletion, late join, real-path regressions

14.2 Test vectors

docs/HMODE_TEST_VECTORS.md records the canonical classic vectors (e.g. ABC → 576 samples). Implementers should reproduce the classic vector bit-exactly and the advanced-path suites behaviorally.

14.3 What is not yet established

There's still honest work left to do. No calibrated sensitivity curves (CCIR Good/Poor CER-vs-SNR sweeps) have been published, so the AWGN and Watterson cases are implementation checks, not link-budget guarantees.

Confidence is σ(mean|LLR|), not a calibrated probability. Over-the-air SNR feedback transport (§9.6) and multi-channel bonding remain out of scope.

15. Implementation status and known limitations

15.1 Fully implemented

In-band loading-map signaling; in-band configuration negotiation (v1 parameters, v2 profile/fingerprint, refusal semantics); advanced header decode; validated mid-stream markers with timing re-centring, SCO observation, blind-skip fallback, and auto-reacquisition; late join; closed-loop CFO/SCO/phase/fractional-timing tracking on classic, OTFS, AFDM, and generalized OFDM; mid-stream AFDM and gOFDM markers; deterministic dual-layer warm-up gating with proven enhancement-payload recovery; complex-IQ OTFS (ISFFT) and AFDM (DAFT) TX/RX; generalized OFDM with pilot LS equalization; the v1 profile registry; exact duration estimation.

15.2 Reserved / declarative surfaces

bonding.*, io.gpsdo, data_aided.pilot_density_steady, and log_level do not affect signal processing. Classic sync-period adaptation runs only when **both** sync.adaptive_period and adapt.enable are set. io.center_hz is informational (no mixing on any path).

15.3 Known defects and constraints (normative cautions)

Priority	Issue	Effect	Mitigation
High	SelectiveMapping PAPR applies an unsignaled phase rotation that is not undone at RX	OTFS/Hybrid demod corrupted from the start when SLM is active	Do not use SLM on advanced waveforms; all registry profiles use ClipFilter; interactive high_speed is affected as defined
High	Mid-stream adaptive map **update** (Chow and waterfilling) corrupts one character at the final sync-block boundary	One-character error per map change on adaptive links	Use fixed loading (all v1 registry profiles) until fixed; static adaptive maps (no mid-stream update) are unaffected
Medium	16-QAM amplitude thresholds require the data-aided equalizer's amplitude reference	4-bit loading decodes badly without a warm tracker	Enable data-aided tracking wherever bits_enhance_max = 4 may engage
Medium	config_announce, io.iq_path, AFDM c1/c2, and io/band settings are not signaled in-band	Mismatch can desynchronize the link	Coordinate out-of-band; warmup_units is now fingerprinted (format v2) and pinned by v2 profile adoption
Medium	Dual-layer RX decodes full enhancement capacity every warm frame	Trailing low-confidence garbage characters after message end	Delimit messages at a higher layer
Low	Classic adaptive sync period is receiver-local and unsignaled	Period disagreement if enabled against a fixed-cadence TX	Classic factory profiles disable it
Low	GUI/SessionEngine profile exposure depends on application wiring	v2 profiles require the runtime-config API where the GUI has not been rebuilt	Use hmode_config_named_profile / validated JSON directly

15.4 Compatibility statements

- The classic profile is bit-exact H-Mode-1.0 in both directions.

- Advanced bitstreams with config_announce = true are **not** decodable by pre-negotiation receivers (three extra aux symbols), and current AFDM streams are not trackable by pre-marker receivers. Set config_announce = false and avoid AFDM only when interoperating with such builds.
- The version-0 descriptor guarantees a deterministic symbol count even for unrepresentable configurations.
- Fingerprint format version 2 (adds warmup_units) changes every profile fingerprint relative to format-1 builds: a format-1 receiver rejects format-2 version-2 descriptors with FingerprintMismatch (and vice versa), refusing decode rather than desynchronizing — upgrade both ends together.
- The tone-reservation correction and the PAPR-before-window reorder change the transmitted waveform of advanced profiles relative to earlier draft builds at the sample level; receivers are unaffected (the change is TX-side shaping within the same crest budget), but bit-exact TX regression vectors from those builds no longer match.

---

16. Operating and regulatory considerations

Nothing in H-Mode is hidden. It applies no encryption or undisclosed scrambling. The pilots, HMODE1 preamble tag, header and descriptor CRCs, profile registry, fingerprint serialization, and character mappings are all documented here and reproducible from public state. HFEdge carries station identification in the transparent character stream.

On the air, the operator still owns the signal. The classic carrier plan spans 2625 Hz between its lowest and highest active carrier centers; advanced waveforms occupy the configured [band_low_hz, band_high_hz]. Actual occupied bandwidth depends on filtering, drive level, hardware, and runtime band settings. The operator is responsible for measuring the transmitted signal, choosing settings appropriate to the authorized emission, providing required identification, and following the rules of the station and jurisdiction. This is an engineering specification, not a legal determination.

Appendix A. Descriptor bit layouts (summary)

Header (18 bits): frame[7:0] | profile[1:0] | alphabet[1:0] | crc6[5:0] — crc6 = low 6 bits of CRC-16/CCITT-FALSE over {frame, (profile&3)|((alphabet&3)<<2)}.

Config v1 (54 bits): ver(2)=1 | waveform(2) | alphabet(1) | log2M(3) | log2N(4) | otfs_cp(5) | log2_afdm(4) | afdm_cp(5) | algo(2) | core_code(2) | enh_code(2) | hier(1) | dual(1) | sync_period(7) | rsvd(1) | crc12(12).

Config v2 (54 bits): ver(2)=2 | profile_id(8) | fingerprint(24) | alphabet(1) | rsvd(7) | crc12(12).

Config v0 (54 bits): 42 zero payload bits + valid crc12 (“no info”).

Loading map (54 bits): ver(2)=1 | algo(2) | seq(4) | rsvd(2) | group_codes(16×2) | crc12(12) — codes map {0,1,2,3} → {0,1,2,4} bits.

All descriptor CRCs are the low 12 bits of CRC-16/CCITT-FALSE over the 42 payload bits packed MSB-first into 6 bytes (final 6 bits zero).

Appendix B. Glossary of acronyms

This glossary defines every acronym and initialism used in this standard, in the sense used here.

Term	Meaning in this standard
ACE	Active constellation extension — PAPR step expanding constellation points before filtering
API	Application programming interface — the public C++ header-level interface exposed by HFEdge
AFC	Automatic frequency control in the transceiver; reduces residual carrier offset before acquisition
AFDM	Affine Frequency Division Multiplexing — chirped-DAFT waveform family
AWGN	Additive white Gaussian noise (test channel, not an HF model)
BPSK / QPSK / QAM	Binary / quadrature phase-shift keying; quadrature amplitude modulation (rectangular 16-QAM here)
CCIR	International Radio Consultative Committee — “CCIR Good/Poor” name conventional HF channel-model categories (not yet characterized by the automated suite)
CCITT	International Telegraph and Telephone Consultative Committee — CRC-16/CCITT-FALSE is the checksum parameterization used throughout
CER	Character error rate
CFO / SCO	Carrier-frequency offset / sampling-clock offset
CP	Cyclic prefix
DC	Direct current — the zero-frequency FFT bin, always nulled on real-audio paths
CRC	Cyclic redundancy check (CRC-16/CCITT-FALSE; CRC-32/IEEE for fingerprints)
CSI	Channel-state information (per-resource SNR, gain, noise)
DAFT	Discrete affine Fourier transform
DD / TF	Delay-Doppler / time-frequency domains of the OTFS grid
EVM	Error vector magnitude — deviation of received/transmitted constellation points from their ideal positions
EWMA	Exponentially weighted moving average
FD	Frequency domain
FFT / IFFT	Fast Fourier transform / inverse FFT — radix-2 implementation; configured dimensions must be powers of two
FEC	Forward error correction (deliberately absent from the payload)
gOFDM	Generalized OFDM (classic_bins = false)
GUI	Graphical user interface — the HFEdge desktop application
HF	High frequency — conventionally the 3–30 MHz radio spectrum; H-Mode targets HF SSB operation
ID	Identifier (e.g. the on-air profile ID of §12.4)
IEEE	Institute of Electrical and Electronics Engineers — names the CRC-32 polynomial used by the configuration fingerprint
I/Q	In-phase and quadrature complex-signal components — the optional complex baseband path (io.iq_path)
ISFFT / SFFT	(Inverse) symplectic finite Fourier transform
JSON	JavaScript Object Notation — serialization format of HModeRuntimeConfig
LFSR	Linear-feedback shift register (pilot sequence)
LLR	Log-likelihood ratio (sign prefers bit 0 when positive)

Term	Meaning in this standard
LS	Least squares — the gOFDM one-tap pilot gain fit (§7.2)
LSB / MSB	Least- / most-significant bit — character streams flatten MSB-first; resource values consume bits LSB-first (§9.5)
MMSE	Minimum mean-square error (single-tap iterative equalizer)
OTFS	Orthogonal time frequency space
PA	Power amplifier — crest processing exists for the audio-PA chain; the IQ chain stays linear
PAPR	Peak-to-average power ratio
PHY	Physical layer
PSK	Phase-shift keying — generic term covering the BPSK/QPSK constellations
RMS	Root mean square — block amplitude normalization and crest-factor reference
RX / TX	Receive / transmit (also the receiver/transmitter side of the link)
SDR	Software-defined radio — target of the complex-IQ path
SLM	Selective mapping (PAPR; currently not decode-transparent)
SNR	Signal-to-noise ratio (linear per resource unless stated in dB)
SSB	Single sideband — the intended HF transceiver audio channel
TOC	(Document) table of contents
XOR	Exclusive OR — the Boolean operation in the pilot LFSR feedback