SPEC-1-Universal Room Correction (Genesis Redux)

Background

Goal. Deliver a functionally equivalent, non-infringing room correction application inspired by ARC Genesis's "Measure → Target → Review" workflow. The product must avoid proprietary code, assets, trade dress, and brand identifiers, while preserving user-favored ergonomics and adding broad hardware interoperability.

Problem. High-quality room correction workflows are locked behind vendor ecosystems (Anthem, Audyssey, Dirac). Users want the same expert workflow independent of AVR/processor brand and want to ingest measurements from common tools and export filters to open hardware.

Approach. Build a desktop application ("Genesis Redux") with a universal ingestion pipeline (.txt/.frd from REW and optional .mqx bridge), high-performance visualization, target-curve shaping, safe inversion with boost/dip constraints, and export to open DSP endpoints (e.g., MiniDSP biquads). Adopt a distinct dark UI theme ("Obsidian & Amber") and a neutral layout that avoids confusing brand association while maintaining efficiency of the original interaction model.

Non-infringement posture. - No reuse of Anthem code, artwork, fonts, sound assets, verbiage, or specific screen copy. - No 1:1 "trade dress" cloning; preserve workflow but change look-and-feel (colors, icons, control arrangement sensibly, naming). - Only implement algorithms derivable from public acoustics/DSP literature or user-provided specs; any reverse-engineering must respect applicable laws and license terms.

Scope for MVP. - Platforms: **Windows desktop only** (native binary). - Inputs: REW-style magnitude/phase text exports; optional Audyssey MultEQ-X ingestion via user-assisted bridge (no bundled decryption keys). - Outputs: Filter reports (PDF) and PEQ/biquad exports for MiniDSP and generic IIR hosts. - UX: Session/project model, quick RTA/measure, target shaping (room gain, tilt, bass shelf), prediction preview, and report generation.

Out-of-scope for MVP. Direct programming of closed AVR firmwares; mobile apps; cloud services beyond optional crash/telemetry (off by default).

Requirements

Audience & Usage - Personal, in-home use by a single user. No redistribution of vendor materials. - Windows 10/11 desktop only.

Constraints & IP Hygiene - No vendor branding, copy, artwork, or trade dress. Use generic signal-processing terms only (e.g., Parametric EQ, LPF, HPF, phase, delay, gain). - Implement algorithms and file I/O based on public documentation and user-provided data.

Device/Home Page semantics - Denon/Marantz only (MVP). HTTP discovery with manual IP fallback. - Read-only device state. Load once on connect; manual Refresh allowed; no background polling. - Sub

outputs may be **tactile**, and subs may be **global** (summed) or **individual**, with optional **directional** metadata when provided.

MoSCoW

Must Have (MVP) - Measurement workflow (single & multi-position), mic calibration import, SPL calibration. - Target curve editor: room-gain, tilt, shelves, manual PEQ; correction-range limits. - IIR filter solver with caps on boost/Q and headroom; per-channel delay/level; phase-aware smoothing. - Visualization: magnitude/phase, excess phase estimate, group delay; before/after overlay. - Audio I/O: WASAPI (shared/exclusive); optional loopback capture. - Export: generic PEQ/biquad & MiniDSP text; printable PDF report. - Safety: limiter/headroom reservation; safe sweep levels. - Usability: undo/redo, presets, autosave, dark theme (distinct from vendors).

Should Have - Optional ASIO support. - FIR export (WAV taps) for convolution engines. - Multi-sub alignment (delay/gain/polarity) and combined bass correction. - Generic crossover helpers (HPF/LPF suggestions); saved mic-position sets with weights.

Could Have - Seat-map mic guide, target library import/export (JSON), quick RTA, optional VST3 export.

Won't Have (initially) - Direct programming of closed AVR/processor firmwares. - Cloud features (except opt-in crash reports).

Method

Platform & I/O Choice (Windows-only)

- **Framework:** JUCE (C++20) for Windows-only desktop app.
- **Audio APIs:** Primary **WASAPI** (shared & exclusive). Optional **ASIO** later.
- **Loopback Capture:** Optional **WASAPI loopback** for measuring system playback.
- **DSP/Math:** JUCE DSP module for IIR/biquads; **FFTW** for FFT/RTA; **Eigen** for target fitting & solvers.
- **Packaging:** CMake + JUCE; MSI/MSIX installer.

AVR Discovery & Home Page (Denon/Marantz only for MVP)

Startup flow (HTTP-only) 1. **LAN discovery:** Use SSDP/UPnP and mDNS to find candidates. Filter for Denon/Marantz signatures (HTTP server/friendly-name patterns). 2. **If none found:** Prompt for **manual IP** entry; validate with a quick HTTP probe. 3. **Identify & fetch:** Over HTTP, read model, firmware, zones, **amp assign**, speaker presence, and sub count. Cache locally for fast app loads. 4. **Home Page render:** Show a Speaker Layout map (our own visual style) and a Summary card (model, firmware, zones, active speakers, sub count). If multiple AVRs are found, ask user to choose a default.

Protocol posture - **Vendors supported:** Denon & Marantz only (MVP). - **Transport:** HTTP/JSON or HTTP form endpoints as available. **No Telnet**. LAN-only; user may disable discovery and use manual IP only.

Speaker/assign mapping - Channel codes supported: `FL/FR, C, SL/SR, SBL/SBR, FHL/FHR, RHL/RHR, TFL/TFR, TML/TMR, TRL/TRR, CH, TS, FWL/FWR, SW1–SW4`. Display friendly names in UI. Sub

count up to 4. - **Sub roles & modes (read-only for MVP):** `role = subwoofer|tactile`, `scope = global|individual`, `directivity = omni|directional` when exposed by AVR.

UI behavior - If discovery finds multiple AVRs, **prompt** to pick a default (do not auto-choose). If none are found, show a **Manual IP** prompt and connect. - Manual IP is treated as a **fallback**; discovery still runs at startup unless the user disables it. - The **Speaker Layout** panel mirrors Denon/Marantz channel families but uses our own iconography/arrangement to avoid vendor trade dress. - **Sub panel** shows per-sub tiles with role (Sub/Tactile), scope (Global/Individual), and directivity (Omni/Directional) when reported by AVR. - A compact **Amp Assign** summary (e.g., 15.1ch, 13.1ch+ZONE2, Front B, Bi-Amp) is shown with a link to a read-only terminal map. - **Connection model:** Read-only; fetch once on connect and persist for the session (no background polling). Manual **Refresh** button re-queries the AVR.

Data model (excerpt)

```
Device{
  id, brand:"Denon|Marantz", model, firmware,
  network:{ ip, port:80, protocol:"http" },
  ampAssign:{ mode, zones, frontB, biAmp },
  speakers:[ {code:"FL", name:"Front Left", present:true}, ... ],
  subs:{
    count:1..4,
    items:[{ id:1, present:true, role:"subwoofer|tactile", scope:"global|
individual", directivity:"omni|directional" }, ...]
  }
}
```

Flow (PlantUML)

```
@startuml
start
:Start application;
:SSDP/mDNS discovery (Denon/Marantz only);
if (Any AVR found?) then (yes)
  :Prompt to pick default if >1;
  :HTTP probe -> fetch model/firmware/ampAssign/speakers/subs;
else (no)
  :Prompt for manual IP;
  :HTTP probe -> validate & fetch metadata;
endif
:Render Home page (Summary + Speaker Layout);
stop
@enduml
```

Core DSP & Data Model (Draft)

Architecture components

```
@startuml
skinparam shadowing false
[UI Shell] --> [Measurement Engine]
[UI Shell] --> [Target Editor]
[UI Shell] --> [Solver]
[UI Shell] --> [Blend Engine (L/C/R + Subs)]
[UI Shell] --> [Export Engine]
[UI Shell] --> [Device Service]
[Measurement Engine] --> [Audio I/O (WASAPI)]
[Measurement Engine] --> [FFT/RTA]
[Solver] --> [DSP Primitives]
[Target Editor] --> [Project Store]
[Blend Engine (L/C/R + Subs)] --> [Solver]
[Device Service] --> [Project Store]
[Export Engine] --> [Project Store]
[Project Store] --> [SQLite DB]
@enduml
```

Local storage (SQLite)

- `projects(id, name, created_at, sample_rate, ref_level_dbfs)`
- `channels(project_id, code, enabled, delay_ms, trim_db, f3_hz)`
- `measurements(id, project_id, channel_code, pos_index, fs, nfft, mag_path, phase_path)`
- `mic_cal(project_id, file_path, sensitivity_mvpa, offset_db)`
- `targets(project_id, json)` // room gain, tilt, shelves, per-band PEQ; per-channel mid-tilt
- `filters(project_id, channel_code, kind, json)` // IIR biquads or FIR taps
- `devices(project_id, json)` // snapshot of AVR Home page state
- `blend(project_id, json)` // L/C/R blend settings & results

Measurement

- Log sweep generation (20 Hz–22 kHz), adjustable level, fade in/out, deconvolution to impulse response.
- Windowing (Hann/Tukey), excess phase estimation, 1/24–1/3-oct smoothing options for display.
- Multi-position merging: weighted **log-power average**; time-alignment via peak or excess-phase ref. (*Matches your request for power averaging and psycho smoothing*)

Target curve & editing (speaker-preserving)

- Global controls: **room gain (dB)**, **tilt (dB/decade)**, **bass shelf (freq/Q/gain)**.

- **Per-channel** target synthesis with **F3-aware mask**: detect F3 from midband −3 dB crossing; never push target below each speaker's F3.
- **Mid-tilt** control preserves extension (tilts mid/highs while holding the LF knee at F3).
- Subwoofer target generator: LPF slope + selectable **House curve** presets (**Flat / House / Huge**) and amount slider.
- Correction range: default **20–500 Hz**, user extendable per channel.

L/C/R + Subs Blend Engine (new)

- **Goal:** Blend each of L, C, R to subs near **their own F3**, not a fixed XO, preserving natural extension and avoiding "neutering" mains.
- **Blend region per channel:** `[F3 .. F3 + 0.5 octave]`.
- **True acoustic summation:** complex add of main and sub responses (mag + phase) through the blend region.
- **Alignment search:** optimize **delay / polarity / gain** of subs to maximize summed response flatness/target match in each channel's blend band; null-aware (avoid boosting into deep cancellations).
- **Rules:** No boosts below F3; no boosts <20 Hz for subs; variable Q and spacing; per-channel intelligence (L/C/R blended independently; surrounds use their own F3).
- **Outputs:** per-channel PEQs for mains; shared or per-sub PEQs depending on configuration; predicted curves for **Raw / Target / Corrected / Acoustic Sum**.

Blend activity (PlantUML)

```
@startuml
start
:For channel in {L,C,R};
:Detect F3(channel);
:BlendBand = [F3, F3*sqrt(2)];
:Build F3-preserving target (mid-tilt applied);
:Compute complex sum M(f)+S(f) over BlendBand;
repeat
  :Search {delay, polarity, gain} for subs;
  :Evaluate error to target in BlendBand;
repeat while (error improves?)
:Lock alignment; generate constrained PEQs;
:Predict corrected + acoustic sum;
@enduml
```

Solver (IIR, MVP)

- Compute minimum-phase target within correction band.
- **Filter budget:** per-channel `num_biquads = min(device_capabilities.peq_per_channel, project_limit)`; if AVR reports capacity (e.g., D/M models), we auto-adapt. Default `project_limit=15` unless user lowers it.
- Speaker EQ constraints: `max_boost_speakers=+6 dB`, `max_cut=−20 dB`, `Q_max=12`.

- **Subs = CUT-ONLY** policy: never apply positive gain. Baseline is a **floor line** derived from the smoothed response (e.g., rolling 15th-percentile or lower-envelope spline). Targets are built **relative to this floor**:
- **Flat-to-Floor:** flatten response down to the baseline (only cuts).
- **House-to-Floor:** apply a user house curve (amount slider) referenced to the baseline (still cuts-only).
- **Null detection** prevents attempts to fill deep cancellations; **F3 mask** blocks sub-F3 boosts on mains; **sub-safe** rules avoid <20 Hz boosts (subs cut-only already enforces this).
- Headroom reservation (e.g., 6 dB) and post-gain normalization.
- Optional per-sub alignment then combined bass solve.

Analyzer & Views

- **Before/After** overlays: Raw, Smoothed, Target, Predicted, Corrected.
- **Blend View:** visualize XO highlight, acoustic sum, and per-channel blend band for L/C/R.

Safety

- Pink-noise pre-check to set safe sweep level.
- Output limiter on preview playback.

Export

- **Generic PEQ/biquad text**: lines of `f, Q, gain_dB` or full biquad `b0 b1 b2 a1 a2` at a given sample rate.
- **MiniDSP text**: per-channel numbered biquads, gain/trim, and delay.
- **Denon/Marantz PEQ sheets**: generate per-channel tables capped by the device's reported PEQ count; label any unused slots as passthrough.
- **PDF report**: before/after plots, target curve, L/C/R blend summary, and a Sub section showing the chosen floor and whether **Flat-to-Floor** or **House-to-Floor** was used.

Implementation

1) Project setup

- **Language/Framework:** C++20 + JUCE 7.
- **Build:** CMake presets (Debug/Release), LTO for Release, unit tests via Catch2.
- **Deps:** FFTW (DLL), Eigen (header-only), SQLite (amalgamation), nlohmann/json.
- **App skeleton:** `App`, `MainWindow`, feature modules: `DeviceService`, `MeasurementEngine`, `TargetEditor`, `BlendEngine`, `Solver`, `Exporters`, `ProjectStore`.

2) Device Service (Denon/Marantz)

- **Discovery:** SSDP + mDNS; filter to D/M signatures. Fallback: manual IP dialog; persist as fallback only.
- **HTTP probes:** model, firmware, zones, amp-assign, speaker presence, sub count. Cache snapshot in `devices(project_id)`.
- **Home UI:** Summary card + Speaker Layout map (our visuals) + read-only Amp Assign/terminal map.

3) Measurement Engine

- **Signal:** exponential sweep 20–22k Hz, default 4 s, −12 dBFS; inverse-filter deconvolution to IR.
- **I/O:** WASAPI shared/exclusive; optional WASAPI loopback.
- **Mic/SPL:** load .txt/.csv mic cal; SPL calibration wizard.
- **Storage:** write mag/phase arrays to fast binary blobs referenced by `measurements` rows.

4) Target Editor (Speaker-preserving)

- **Controls:** room-gain, tilt, shelves, per-band PEQ; correction range per channel.
- **Mid-tilt:** rotate around F3 (see pseudocode).
- **Sub targets:** presets (Flat/House/Huge) with amount slider; CUT-ONLY semantics enforced downstream.

5) Blend Engine (L/C/R + Subs)

- **Band:** `[F3, F3*√2]` per main channel.
- **Alignment search:** grid + Nelder-Mead (or golden-section for delay) on `{delay, polarity, gain}`; objective = min RMSE to target in band.
- **Complex sum:** use min-phase recon or measured phase where available.

6) Solver (IIR, constrained)

- **Budget:** `num_biquads = min(device_capabilities, project_limit)` (default limit 15). Trim extra filters by lowest improvement.
- **Constraints:** speakers: max boost +6 dB, max cut −20 dB, Q≤12; subs: CUT-ONLY; no boosts below F3; no boosts <20 Hz.
- **Headroom:** reserve 6 dB; normalize post-solve.

7) Exporters

- **Generic PEQ & biquad text**; **MiniDSP**; **Denon/Marantz PEQ sheets** capped by device capacity.
- **PDF report:** before/after, targets, filter tables, device snapshot, L/C/R blend summary, sub floor line.

8) QA & Validation

- Golden projects; unit tests for F3 detection, floor baseline, null detection; end-to-end compare predicted vs measured.

Key pseudocode

F3 detection (per channel)

```
resp = smooth(mag_db, method="1/6-oct")
midband = median(resp[200..1000 Hz])
for f descending from 200 Hz:
```

```
  if resp(f) <= midband - 3 dB:
     return f
```

Mid-tilt anchored at F3

```
target(f) = base_curve(f)
for f >= F3: target(f) += tilt_db_per_decade * log10(f/F3)
for f <  F3: target(f) = min(target(f), resp(f))  // preserve extension
```

Sub floor baseline (cut-only)

```
resp6 = smooth(mag_db, method="1/6-oct")
floor = rolling_percentile(resp6, p=15%)
// lower-envelope spline optional to smooth
```

Blend search (per L/C/R)

```
band = [F3, F3*sqrt(2)]
obj(delay, pol, gain) = RMSE(sum_complex(main, sub(delay,pol,gain)), target,
band)
(best_delay, best_pol, best_gain) = argmin obj
```

Milestones

1) **M0 – Foundations**: repo, build, window shell, SQLite store, JSON config, logging. 2) **M1 – Device & Home**: D/M discovery + manual IP; HTTP probes; Home page summary & map. 3) **M2 – Measure & View**: sweep capture, deconvolution, plots, mic/SPL cal, project save/load. 4) **M3 – Target & Blend**: F3 detect, mid-tilt, sub target presets, L/C/R blend engine. 5) **M4 – Solver & Export**: constrained IIR, device-aware PEQ budgeting, MiniDSP + D/M sheets, PDF. 6) **M5 – QA polish**: golden projects, performance passes, crash reporting (opt-in).

Gathering Results

Acceptance checks - **Home page** correctly shows model/firmware/amp-assign, speakers, subs (incl. tactile/ global/individual/directional flags when present). - **Measurement** repeatability within ±0.5 dB above 40 Hz across repeated sweeps at MLP. - **F3 detection** stable (±5 Hz) across smoothing choices. - **Blend** improves band RMSE vs. baseline by ≥30% for L/C/R without reducing main LF extension. - **Subs** meet policy: no positive gain; floor-anchored shaping. - **Exports** load into MiniDSP/D&M PEQ entries without truncation; counts never exceed device caps.

Post-production telemetry (optional, local view if telemetry off) - Aggregate timing for blend/solve steps; number of filters generated; common device caps; crash rates.

Need Professional Help in Developing Your Architecture?

Please contact me at sammuti.com :)