Generated by AI - Google Gemini Advanced Deep Research with Gemini 2.5. Pro

Analysis of Software-Based Speaker Audio Enhancement in the MacBook Air A2337 (M1, 2020) via Asahi Linux Project Findings

1. Introduction

1.1. The Ascendance of Software DSP in Laptop Audio

The pursuit of thinner, lighter, and more portable computing devices has invariably led to significant constraints on the physical space available for audio components. Laptop manufacturers, particularly in the premium ultraportable segment exemplified by the Apple MacBook Air, face the challenge of delivering acceptable, and ideally high-quality, audio experiences from extremely small internal speakers, often referred to as microspeakers. These microspeakers, due to fundamental physical limitations related to their size and the cost constraints of mass production, inherently suffer from poor frequency response linearity, limited low-frequency extension (bass), and potential for distortion at higher volumes.[1]
To overcome these physical shortcomings, the industry has increasingly turned to sophisticated software-based Digital Signal Processing (DSP). Computing power has become relatively inexpensive compared to the cost and physical space required for high-performance speaker hardware.[1] Consequently, manufacturers across the board, including Apple, Google, and various Windows PC vendors, heavily leverage DSP algorithms running on the device's main processor or dedicated co-processors to correct speaker deficiencies, enhance perceived audio quality, and protect the hardware from damage.[1]
Techniques such as equalization, dynamic range compression, psychoacoustic bass enhancement, and advanced speaker protection models are now commonplace, transforming the raw output of physically limited speakers into a subjectively better listening experience.[2]

1.2. Focus: MacBook Air A2337 (M1, 2020) Audio Enhancements

This report specifically investigates the software-side audio enhancement strategies employed for the built-in stereo speakers of the Apple MacBook Air (M1, 2020). This model, identified by the model number A2337 and EMC identifier 3598 [4], holds significance as one of the first Mac computers to feature Apple's custom M1 System-on-Chip (SoC), marking the beginning of Apple's transition away from Intel processors.[4] While praised for its performance and efficiency, the audio capabilities of its integrated stereo speakers [6] rely substantially on software processing executed by the M1 chip itself. This analysis focuses exclusively on these

software measures as applied to the internal speakers.

1.3. Methodology: Leveraging Asahi Linux Reverse Engineering

Direct analysis of the proprietary audio processing algorithms within Apple's macOS is inherently difficult due to the closed-source nature of the operating system. Therefore, this report utilizes the findings of the Asahi Linux project as the primary source of information.[1] Asahi Linux is an ongoing community effort dedicated to reverse-engineering Apple Silicon hardware and developing open-source drivers and software support to enable Linux distributions to run effectively on these machines.[8]

The Asahi audio sub-project, in particular, has invested significant effort in understanding and enabling the audio hardware on Apple Silicon Macs, including the speakers.[1] Their work involves reverse-engineering the hardware interfaces, identifying necessary DSP steps, and implementing an open-source software stack to replicate or provide high-quality audio output comparable, and in some aspects aiming to be superior, to macOS.[1] By examining the architecture, algorithms, configuration data, and safety mechanisms developed by the Asahi audio team, it is possible to infer the likely strategies and capabilities implemented by Apple in macOS for the A2337. Key resources for this analysis include the asahi-audio [1] and speakersafetyd [9] GitHub repositories, public developer communications and discussions [2], and official project documentation and blog posts.[14]

1.4. Report Objectives and Scope

The central objective of this report is to provide a detailed technical description of the software-based audio enhancement techniques likely utilized for the internal speakers of the MacBook Air A2337, derived from the evidence gathered through the Asahi Linux project's reverse-engineering efforts. The analysis will cover:

- The software architecture employed by Asahi Linux to manage audio processing on Apple Silicon.
- The critical speaker safety mechanisms implemented to prevent hardware damage.
- The equalization strategies used to correct frequency response.
- The approach to dynamic range control and limiting.
- The handling of stereo imaging and psychoacoustic bass effects.
- Specific configuration details relevant to the A2337 model.

The scope is strictly limited to the software processing applied to the A2337's built-in speakers and the inferences that can be drawn from the public work of the Asahi Linux project. It does not involve direct disassembly or analysis of macOS software binaries.

2. The Asahi Linux Audio Architecture for Apple Silicon

To deliver a high-quality and safe audio experience on Apple Silicon hardware, the Asahi Linux project constructed a sophisticated audio stack leveraging modern Linux audio components and custom configurations tailored to the unique requirements of these devices.

2.1. Core Framework: PipeWire and WirePlumber

The foundation of the Asahi Linux audio system is built upon PipeWire and WirePlumber, the contemporary standard for audio and low-latency multimedia handling on Linux desktops.[1] PipeWire serves as the central multimedia server, managing audio streams between applications and hardware with low latency, while WirePlumber acts as the session and policy manager, responsible for configuring the audio graph, managing devices, and applying policies.[1] The Asahi audio setup requires relatively recent versions of these components (PipeWire 1.0 or above, WirePlumber 0.5.2 or above) to ensure compatibility with the necessary features.[1]

Notably, the Asahi development team collaborated closely with the upstream developers of PipeWire and WirePlumber. This collaboration aimed to design a flexible and modular solution capable of handling the complex DSP requirements not just of Apple Silicon, but potentially any hardware platform needing similar advanced audio processing. This upstream contribution signifies an effort to enhance the capabilities of the broader Linux desktop audio ecosystem, rather than creating an entirely isolated, Asahi-specific solution.[1] The architecture developed, therefore, has implications beyond Apple hardware, potentially paving the way for better support for devices with integrated DSP on other Linux systems.

2.2. Virtual Audio Devices and Plugin Chains

A core architectural concept employed by Asahi Linux is the creation of *virtual audio devices* within PipeWire.[1] These virtual devices (sinks for output, sources for input) are not directly tied to a single hardware interface but represent a processing pipeline composed of a chain of audio plugins.[1] For speaker output, a virtual stereo sink is presented to applications. When audio is sent to this sink, PipeWire routes it through a predefined chain of DSP plugins (for EQ, compression, etc.) before finally sending the processed audio to the actual hardware speaker outputs.[1] A similar process applies to microphone input.

This abstraction layer provides several benefits. It simplifies the user experience, as applications only interact with a standard stereo device, unaware of the complex processing happening underneath.[1] It also allows for centralized management and configuration of the DSP chain. WirePlumber plays a crucial role here by automatically detecting the specific Apple Silicon Mac model upon system startup. Based on this detection, it dynamically loads the appropriate virtual device configuration and associated resources, such as model-specific calibration data (impulse responses for EQ).[1] Furthermore, the system deliberately hides the raw, unprocessed hardware audio interfaces from the userspace environment. This prevents applications or users from inadvertently bypassing the necessary DSP and safety mechanisms, which could result in poor audio quality or even hardware damage.[1]

2.3. Key Dependencies and Components

The successful operation of the Asahi audio stack relies on several key software components working in concert:

- **PipeWire & WirePlumber:** The core audio server and session manager.[1]
- **speakersafetyd:** A critical userspace daemon providing real-time thermal protection for the speakers, preventing damage from overheating (detailed in Section 3).[1]
- **linux-asahi Kernel:** Specific versions of the Asahi-patched Linux kernel are required, containing the necessary hardware drivers for the audio interfaces (amplifiers, codecs) and crucial safety interlocks that interact with speakersafetyd.[1]
- **Bankstown:** Listed as a dependency, this is likely a library or utility involved in the DSP processing chain, although its exact function is not detailed in the provided resources.[1]
- **Triforce:** A component specifically developed for microphone processing, implementing beamforming algorithms to improve input quality from the built-in microphone arrays (discussed further in Section 6).[1]
- **LSP Plugins (Linux Studio Plugins):** This suite of open-source audio plugins, specifically the LV2 versions, provides the core DSP functionalities like equalization and potentially dynamic range compression within the PipeWire plugin chain.[1]

2.4. Insight: Modularity and Upstream Contribution

The architectural choices made by the Asahi audio team reflect a forward-thinking approach. By building upon the standard Linux audio frameworks of PipeWire and WirePlumber and contributing enhancements back upstream, they ensured that the solutions developed for Apple Silicon's unique challenges could potentially benefit the wider Linux community.[1] The problem of integrating complex, hardware-specific DSP and safety features is not unique to Apple devices. Other manufacturers are increasingly relying on similar techniques, often implemented through proprietary drivers or userspace blobs.[9] The Asahi model, utilizing PipeWire's plugin architecture and WirePlumber's dynamic configuration capabilities, provides a template for how such complex audio systems can be supported in an open-source environment. This modularity allows for easier adaptation to new hardware and promotes a more standardized approach to advanced audio processing on desktop Linux, moving beyond the limitations of traditional ALSA-based systems.

3. Speaker Safety and Protection (speakersafetyd)

Perhaps the most critical aspect of enabling speaker audio on Apple Silicon Macs under Linux is ensuring the physical safety of the speaker hardware. Apple's design pushes the small drivers to their thermal and excursion limits, necessitating a robust protection system.

3.1. The Necessity of Active Protection

Apple Silicon MacBooks, including the A2337 Air, are engineered to produce surprisingly loud and full sound from their compact speaker systems. This is achieved, in part, by driving the speakers with more power than they might traditionally be rated for continuous operation.[2] Apple utilizes sophisticated DSP within macOS to manage this, maximizing perceived volume and frequency response while carefully monitoring the speaker's state to prevent damage.[2] Attempting to use these speakers at anything beyond very low volumes without this protective

DSP layer is extremely risky. The Asahi Linux developers discovered early on that doing so could easily lead to permanent physical damage, such as destroying the tweeters or damaging the voice coils due to overheating or over-excursion.[2] Marcan, the project founder, confirmed personally damaging tweeters on an M2 MacBook Air during early testing before safety mechanisms were in place.[10] Furthermore, the raw, unprocessed audio output from these speakers sounds objectively poor and unbalanced, making it unsuitable for use even if it were safe.[11] Therefore, a reliable speaker protection system is not merely an enhancement but a fundamental requirement for enabling usable speaker output on these machines under Linux.

3.2. The speakersafetyd Daemon

To address this critical need, the Asahi project developed speakersafetyd, a dedicated userspace daemon responsible for implementing real-time speaker protection.[1] Written primarily in the Rust programming language for safety and performance [9], speakersafetyd represents what is believed to be the first open-source implementation of a "smart amplifier" protection model.[9] Its development was essential to mitigate the risk posed by hardware designs that rely on proprietary software for safety, a trend seen not only with Apple but increasingly with other vendors in the desktop and mobile space.[9]

3.3. Thermal Modeling Algorithm

The core of speakersafetyd's protective function lies in its thermal modeling algorithm.[9] Instead of applying a simple, static power limit (which would severely restrict volume and dynamic range) [2], the daemon dynamically estimates the temperature of the speaker components, primarily the voice coil and magnet assembly.

This estimation is based on a model derived from the speaker's electromechanical properties, characterized by Thiele/Small parameters, combined with measured thermal characteristics, such as the thermal time constants of the voice coil and magnet.[9] To feed this model with real-time data, speakersafetyd utilizes feedback signals provided by the audio amplifier hardware itself. Specifically, it relies on integrated speaker voltage and current sense (V/I sense) capabilities present in the amplifier chips used by Apple.[9] Analysis and component identification suggest that Apple Silicon Macs, likely including the A2337 generation, predominantly use Class-D amplifier chips from Texas Instruments (TI), such as the TAS2764 or the closely related TAS5770L variant, which is explicitly noted as being found in Apple Silicon Macs.[9] These TI chips feature digital audio inputs and integrated V/I sense outputs, making them well-suited for implementing this type of software-based monitoring and protection scheme.[22]

Based on the continuous V/I sense data, the daemon calculates the power being delivered to the speaker and uses the thermal model to predict the current temperature. If the estimated temperature approaches or exceeds predefined safe operating limits for the specific speaker driver, speakersafetyd intervenes by instructing the underlying ALSA kernel driver to reduce the hardware volume level applied by the amplifier. Once the modeled temperature drops

back into the safe range, the volume limit is relaxed, allowing the output level to increase again.[9] This dynamic adjustment allows the speakers to safely handle transient peaks at high power levels, which occur frequently in music and other dynamic audio content, while preventing sustained high power that would lead to overheating and damage.[3]

3.4. Kernel Interlocks and Configuration

To ensure robustness, the speaker protection system involves coordination between the userspace speakersafetyd daemon and the kernel-level ALSA driver. The kernel driver includes safety interlocks designed to prevent damage even if the userspace daemon fails.[10] If speakersafetyd is not running, crashes, or becomes unresponsive (e.g., due to high system load), the kernel driver automatically clamps the amplifier's hardware volume to a predetermined, conservatively safe (but very low) maximum level.[10] Additionally, the kernel enforces a basic high-pass filter to protect the delicate tweeters from damaging low-frequency energy.[26]

The accuracy of the thermal model is paramount for both safety and performance. This requires model-specific parameters, including the Thiele/Small parameters and thermal constants for the particular speaker drivers used in each Mac model.[9] While the specific file locations and formats for these parameters are not explicitly detailed in the reviewed materials, they are necessarily loaded and utilized by speakersafetyd, likely through configuration files tailored to the detected hardware (e.g., for the A2337/J313).[9] During the initial rollout of speaker support, the Asahi team implemented an additional safety measure: a hard limit reducing the maximum output signal level by 7dBFS within the DSP chain itself, intended to catch potential bugs or unexpected behavior in the protection system during early usage.[2] This conservative limit was planned to be removed as confidence in the system grew.[16]

3.5. Insight: Software vs. Hardware Protection Trade-offs

Apple's decision to implement this critical speaker protection primarily in software running on the main CPU [2], rather than fully utilizing potentially available hardware-based protection features within the amplifier chips (such as the integrated "Smart Amp" functionalities offered by TI [9]), has significant implications. While a software-based approach might offer Apple greater flexibility in tuning, potential cost savings on specialized hardware, or tighter integration with their OS-level audio processing, it fundamentally ties the hardware's safety and full performance potential to a specific, proprietary software environment.

This creates a substantial barrier for alternative operating systems like Asahi Linux, which must painstakingly reverse-engineer and reimplement this complex, safety-critical logic from scratch.[9] An error in this reverse-engineered implementation could lead to hardware destruction.[10] Furthermore, this design choice complicates third-party repair, as the correct functioning of the speakers depends on the interplay between the hardware components and the specific software environment they are running in. A protection system implemented primarily in hardware within the amplifier chip itself would likely be OS-agnostic, simplifying support for diverse operating systems and potentially making hardware replacement more

straightforward. Apple's software-centric approach, therefore, while potentially offering advantages within their closed ecosystem, inherently reduces the interoperability and robustness of the hardware when used outside of macOS.

4. Equalization (EQ) Strategy

Equalization plays a fundamental role in shaping the sound output of the MacBook Air A2337, compensating for the inherent limitations of its small speakers and defining its overall tonal character.

4.1. The Role of EQ in Microspeaker Systems

As established, the physical constraints of laptop speakers lead to significant deviations from an ideal, flat frequency response.[1] Microspeakers typically struggle to reproduce low bass frequencies, may exhibit resonances or sharp peaks and dips in the midrange, and can have uneven high-frequency output. Without correction, this results in sound that may be perceived as thin, muddy, harsh, or generally unbalanced. Aggressive equalization applied in the digital domain is therefore essential to counteract these deficiencies, smooth out the frequency response, and create a more pleasant and intelligible listening experience.[1]

4.2. Asahi Linux EQ Goals: Neutrality and Fidelity

The Asahi Linux project approaches equalization with a distinct philosophy compared to what is inferred about Apple's strategy in macOS. The stated goal for Asahi audio is to achieve a *mostly flat* frequency response across the audible spectrum when measured from a typical listening position.[1] This emphasis on neutrality aims to ensure that the speakers faithfully reproduce the source material as accurately as possible, minimizing coloration introduced by the playback system itself.[1]

This contrasts with the perceived sound signature of macOS, which Asahi developers describe as likely tuned for an exaggerated curve, often resembling a "smiley-face" EQ or following trends like the Harman target curve.[1] Such tunings typically boost low bass and high treble frequencies relative to the midrange, a common practice in consumer audio products designed to create a subjectively "exciting" or "rich" sound. However, the Asahi team views this approach as detrimental to accuracy, potentially making the sound "muddier" or reminiscent of early "Beats headphones".[1] Consequently, Asahi Linux explicitly does *not* aim to replicate the exact macOS sound profile, instead prioritizing objective fidelity as the baseline.[13] Test notes for the J31x (14" MacBook Pro) reinforce this, stating goals like linear midrange response, clear highs, and *diminished* sub-bass (below ~80-100Hz) to prevent unwanted resonances and vibrations, rather than artificial boosting.[19]

4.3. Implementation: Impulse Responses and Filters

The mechanism chosen by Asahi Linux to implement this model-specific EQ is based on *impulse response* files.[1] For each supported Mac model, including the A2337, a unique impulse response file is created based on acoustic measurements of that specific hardware. This file

effectively encodes the precise filter characteristics needed to correct the speaker's frequency response deviations and achieve the target neutral profile.[1]
These impulse response files are loaded by PipeWire at runtime and applied using a convolver plugin within the audio processing chain.[19] The use of impulse responses strongly indicates that the primary filter type employed for equalization is Finite Impulse Response (FIR) filtering.[1] FIR filters are well-suited for high-precision equalization tasks because they can be designed to have a linear phase response. Linear phase is crucial for audio fidelity as it ensures that all frequencies pass through the filter with the same time delay, preserving the waveform's shape and avoiding phase distortion artifacts. While Infinite Impulse Response (IIR) filters can be more computationally efficient for certain types of EQ tasks, the reliance on measured impulse responses points towards FIR as the core method for achieving the desired frequency correction in the Asahi stack. The process involves measuring the speaker response (e.g., using Room EQ Wizard - REW), determining the target curve, calculating the necessary correction filter, and generating the FIR impulse response file.[19]

4.4. A2337 Specifics

The MacBook Air (M1, 2020), model A2337 (with the internal Apple codename J313), is explicitly listed as a supported device within the asahi-audio project configuration for both speaker output and microphone input.[1] This confirms that a dedicated, model-specific EQ profile, stored as an impulse response file, exists for the A2337 and is automatically loaded and applied by the PipeWire/WirePlumber system when running Asahi Linux on this hardware.

4.5. Insight: Calibration Complexity

The requirement for distinct impulse response files for every supported Mac model underscores the significant acoustic variability between different hardware designs and the intricate nature of DSP calibration.[1] Even if core components like the M1 chip or specific amplifier models are shared across devices, subtle differences in the speaker drivers themselves, their mounting, the internal chassis volume and shape, porting (if any), and grille design can drastically alter the system's overall acoustic signature.
Achieving a consistently high level of audio fidelity necessitates meticulous measurement and calibration for each unique hardware configuration.[19] This involves specialized equipment (like calibrated microphones) and software (like REW) to capture the raw response and sophisticated filter design techniques to generate the corrective impulse response.[19] This model-specific calibration process represents a substantial and ongoing workload for the Asahi team, mirroring the likely extensive internal calibration efforts undertaken by Apple during product development. It highlights that software DSP is not a generic overlay but a critical tuning step tailored to the specific physical characteristics of each device like the A2337.

5. Dynamic Range Control (DRC) and Limiting

Dynamic range control encompasses techniques like compression and limiting, used to

manage the difference between the loudest and quietest parts of an audio signal. In the context of laptop speakers, DRC plays a role in maximizing perceived loudness, preventing distortion, and ensuring intelligibility.

5.1. macOS Approach

Based on the analysis and contrasting goals stated by the Asahi project, macOS is understood to employ Dynamic Range Compression (DRC) as a standard part of its audio processing for the built-in speakers.[1] The likely purposes of this DRC in the Apple ecosystem are manifold:

- **Increased Perceived Loudness:** By reducing the peak-to-average ratio of the audio signal, DRC can make the overall sound seem louder without increasing the absolute peak level, which is beneficial for small speakers with limited headroom.
- **Peak Control/Clipping Prevention:** Limiting or compressing loud transients prevents the audio signal from exceeding the digital or analog limits of the system, avoiding harsh clipping distortion.
- **Improved Consistency:** DRC can help maintain a more consistent volume level, making quiet passages more audible and preventing loud passages from being overwhelming, which can be useful in noisy environments or for late-night listening.

The specific parameters and aggressiveness of the DRC used in macOS on the A2337 are proprietary, but the Asahi project's commentary suggests it contributes to the overall "processed" sound signature they aim to differ from.[1]

5.2. Asahi Linux Approach and Status

Consistent with their goal of a more neutral and less processed sound, the Asahi Linux audio stack appears to take a more conservative or currently incomplete approach to DRC compared to the inferred macOS implementation.[1] Based on available documentation regarding the DSP chain's status [16]:

- An **input compressor** is implemented within the DSP plugin chain. The snippets do not detail its specific configuration or algorithm (e.g., whether it's a simple peak limiter, a multi-band compressor, or something else). Its primary role might be basic overload protection at the input stage of the DSP chain.
- A **final limiter or compressor stage**, typically placed at the very end of the processing chain just before output to the hardware, is explicitly stated to be **missing** in the current implementation as of the documentation date.[16]

The absence of this final dynamic control stage has potential consequences. Audio signals containing significant energy in frequency ranges that are heavily boosted by the model-specific EQ (the Asahi documentation specifically mentions the ~200Hz region as potentially problematic for the J31x model) could exceed the system's headroom and result in audible distortion or digital clipping before the speakersafetyd thermal protection engages.[16] The documentation acknowledges this as a current limitation to be addressed.[16]

5.3. Interaction with speakersafetyd

It is crucial to distinguish between the dynamic range processing within the audio DSP chain

(EQ, compressors, limiters acting on the *signal*) and the protection provided by speakersafetyd (acting on the *hardware* based on thermal modeling). The DSP chain shapes the audio waveform according to its programmed parameters (EQ curves, compression thresholds, etc.). speakersafetyd, operating independently based on V/I sense feedback and thermal models, acts as a safety override, reducing the overall hardware volume level when the physical speakers are at risk of overheating, regardless of the signal processing applied by the DSP chain.[9] While a well-designed final limiter in the DSP chain would prevent signal clipping, speakersafetyd prevents thermal damage, addressing two different potential failure points.

5.4. Insight: Phased Rollout and Complexity

The explicit acknowledgment of a missing final limiter/compressor stage [16] strongly suggests an incremental development strategy for the Asahi audio stack. Initial efforts understandably focused on establishing basic hardware enablement (drivers), implementing the critical safety layer (speakersafetyd), and providing fundamental sound quality improvements through equalization.[1]

Implementing effective DRC and limiting, especially in conjunction with aggressive EQ and a thermal protection system, is a non-trivial task. Poorly tuned compression or limiting can introduce audible artifacts (like pumping or distortion) that may be more objectionable than the issues they aim to solve. Finding the right balance between maximizing loudness, minimizing distortion, ensuring intelligibility, and seamlessly interacting with the thermal protection requires careful design and tuning. The current state likely reflects a deliberate decision to prioritize the foundational elements, with the more nuanced dynamic processing planned as a subsequent refinement phase once the core system is stable and well-understood. This phased approach is common in complex software engineering projects, allowing developers to tackle intricate problems step-by-step.

6. Other DSP Techniques (Stereo Image, Bass Enhancement, Microphones)

Beyond core equalization and dynamics management, modern audio systems often employ additional DSP techniques to enhance the listening experience, particularly psychoacoustic effects aimed at overcoming hardware limitations or improving spatial perception.

6.1. Stereo Widening and Spatialization

Apple actively markets features like "Wide stereo sound" and support for "Dolby Atmos playback" for the A2337's built-in speakers.[6] This strongly implies that macOS incorporates spatialization algorithms designed to create a broader, more immersive soundstage than would typically be expected from two small speakers placed close together in a laptop chassis. Techniques like cross-talk cancellation, phase manipulation, or head-related transfer function (HRTF) filtering might be employed to achieve this effect. Dolby Atmos processing, when applied to stereo speakers, typically involves downmixing the multi-channel Atmos

stream and applying spatial cues to simulate a more enveloping sound field.[29] User discussions confirm the presence of a noticeable "wide" effect in macOS, though some find it unnatural or problematic for critical listening like audio mixing.[31]
The Asahi Linux project, however, adopts a different stance. Their stated goal is to provide a "simpler, natural-sounding stereo system" that prioritizes accurate reproduction of the source material's inherent stereo image.[1] Asahi developers explicitly state they avoid hard-coding specific spatial effects like those potentially used in macOS's "spatial audio" processing, preferring to offer a neutral baseline.[16] They believe this provides a more objective representation and allows users to apply their own preferred spatialization effects via third-party plugins if desired.[13] Test notes for the 14-inch MacBook Pro (J31x), which features a more complex six-speaker array, indicated that simple stereo routing without additional cross-mixing was sufficient for good stereo separation in that specific case, further suggesting a focus on straightforward stereo reproduction.[19]

6.2. Psychoacoustic Bass Enhancement

Another area of divergence concerns low-frequency reproduction. Psychoacoustic bass enhancement techniques aim to create the *perception* of bass frequencies that the small speakers cannot physically generate effectively. This is often done by manipulating the harmonic content of the audible bass range, tricking the human auditory system into perceiving fundamental tones that aren't actually present.[16]
macOS is understood to utilize such psychoacoustic bass processing.[1] The Asahi project, however, made a conscious decision *not* to implement these techniques.[1] Their reasoning is twofold: firstly, they identified what they perceive as a bug in Apple's implementation that introduces audible artifacts; secondly, they find that such processing contributes to an unnatural, "colored" sound profile.[1] Instead of creating artificial bass, Asahi's approach focuses on accurately reproducing the bass frequencies that *are* physically achievable by the speakers, within the constraints imposed by the EQ and safety systems.[16] As mentioned previously, the EQ strategy involves carefully managing the actual low-end response, including attenuating sub-bass below ~80-100Hz to prevent unwanted chassis vibrations and midrange muddiness, rather than attempting to artificially extend the perceived bass response.[19]

6.3. Microphone Beamforming (Triforce)

While the primary focus of this report is speaker enhancement, the DSP work for the microphone array provides relevant context regarding the overall audio processing capabilities. The MacBook Air A2337 features a three-microphone array.[6] These microphones are Pulse Density Modulation (PDM) types, which are inherently omnidirectional and sensitive, making them prone to picking up ambient noise.[20]
To achieve usable input quality, Apple employs beamforming techniques in macOS. Asahi Linux replicates this functionality using a custom component named Triforce within its PipeWire DSP chain.[1] Triforce implements advanced beamforming algorithms (specifically, a

Minimum Variance Distortionless Response (MVDR) beamformer is mentioned) that process the signals from the three microphones.[20] By analyzing the phase and amplitude differences between the microphones, the beamformer can enhance sounds originating from a specific direction (typically directly in front of the laptop, where the user would be) while attenuating noise coming from other directions.[20] This complex signal processing is deemed essential for obtaining clear microphone input from the built-in array.[20]

6.4. Insight: Philosophical Differences in Audio Reproduction

The contrasting choices made by Asahi Linux developers regarding spatial processing and bass enhancement, when compared to the inferred macOS approach, highlight a fundamental difference in audio reproduction philosophy. Apple appears to prioritize maximizing the subjective "wow factor" and compensating heavily for hardware limitations through perceptual tricks (psychoacoustic bass, spatial widening), even if this introduces coloration or deviates significantly from the original source recording. Their goal seems geared towards creating an immediately impressive sound for the average consumer listening to typical media content.

Asahi Linux, conversely, champions a philosophy rooted in fidelity and neutrality.[1] Their DSP efforts focus primarily on *correcting* the inherent flaws of the hardware (via EQ) and ensuring its *safe operation* (via speakersafetyd), aiming to deliver a baseline sound that is as faithful as possible to the original source material within the physical constraints of the speakers. Subjective enhancements like spatial effects or exaggerated EQ are left to the user's discretion, typically via additional user-configurable plugins.[13] This approach aligns more closely with traditional high-fidelity audio principles, prioritizing accuracy over artificial enhancement. The reverse-engineering process thus reveals not only the technical capabilities but also these underlying design philosophies.

7. Configuration and Calibration Specifics for MacBook Air A2337

The successful implementation of the Asahi Linux audio stack relies heavily on accurate hardware identification and the application of model-specific configuration and calibration data.

7.1. Confirmed Support

The MacBook Air model featuring the M1 chip, released in late 2020, is unambiguously identified by the model number A2337 and EMC number 3598.[4] This specific model is explicitly listed as a supported device within the asahi-audio project's documentation, covering both speaker output and microphone input functionalities.[1] Further hardware identification relevant to the audio subsystem includes the audio daughterboard part number, which is cited as 820-01929-A (or variations like 820-01929-05) across multiple sources.[7] This board houses the headphone jack and potentially interfaces with the main logic board for speaker signals. Apple's internal part number (APN) for this board assembly (depending on color, e.g., silver vs.

space gray/gold) appears as 923-03672 or 923-03673.[7] The associated flex cable connecting this board to the main logic board is often identified as 821-03452-A or 821-03452-01.[36]

7.2. Model-Specific Files

The Asahi audio architecture, managed by WirePlumber, relies on detecting the specific machine model (A2337 in this case) and loading corresponding configuration files.[1] These files define the structure of the PipeWire processing graph – the specific sequence of plugins (convolver for EQ, potentially compressors, etc.) that make up the virtual audio device for the A2337's speakers.

Central to this model-specific configuration is the impulse response file (.irs or similar format).[1] This file contains the unique set of filter coefficients meticulously calculated from acoustic measurements of the A2337 speaker system. It represents the core calibration data that corrects the frequency response anomalies specific to the A2337's drivers and enclosure, enabling the system to achieve the target neutral sound profile defined by the Asahi project.[1] Without loading the correct A2337-specific impulse response, the EQ would be incorrect, leading to suboptimal sound quality.

7.3. Amplifier Chip Identification

Accurate speaker protection via speakersafetyd depends on receiving voltage and current sense data from the speaker amplifier chips. While the Asahi documentation confirms the use of TI amplifiers with V/I sense capabilities in Apple Silicon Macs [9], pinpointing the exact chip in the A2337 requires cross-referencing with hardware analysis and component databases. The TI TAS2764 is frequently cited as the likely candidate family due to its features matching the requirements (digital input, Class-D, V/I sense, suitable power ratings).[22] Some repair resources specifically identify a TAS5770L variant as being used in Apple Silicon Macs [24], and one parts supplier lists a TAS5770AC0 chip specifically for the A2337.[21] These amplifiers are Class-D chips capable of driving the speakers efficiently while providing the necessary feedback for the thermal protection model.[22] It's also worth noting that other audio-related chips, potentially from Cirrus Logic (like the CS42L83A codec mentioned for audio jack functionality in some MacBooks [48] or other DAC/ADC components [49]), are part of the overall audio subsystem, but the speaker amplification stage with V/I sensing appears to be handled by TI components. These amplifier chips would be located on the main logic board (MLB) or potentially the audio daughterboard (820-01929-A) [7], connected to the speaker modules via internal cabling.

7.4. Insight: Hardware Consistency vs. Calibration Needs

While the foundational M1 SoC architecture [5] and potentially key components like the TI speaker amplifiers [9] might be shared across the initial generation of M1 devices (A2337 Air, A2338 Pro, Mac Mini), the Asahi project's methodology clearly demonstrates that this component sharing does *not* translate to identical audio behavior. The necessity of creating and applying unique EQ calibration files (impulse responses) for each distinct model [1]

highlights the profound impact of physical design variations on acoustic performance. Differences in the specific speaker drivers chosen, their placement within the chassis, the internal volume and acoustic properties of the enclosure, and even the design of the speaker grilles can lead to significantly different raw frequency responses. Therefore, the software calibration step – generating and applying the model-specific .irs file – is not merely a generic enhancement but a critical, bespoke tuning process. It is essential for correcting the unique acoustic fingerprint of each hardware implementation, like that of the A2337, to achieve the desired target sound profile (in Asahi's case, neutrality). This underscores that achieving optimal audio performance in tightly integrated systems requires careful co-design and calibration of both hardware and software elements.

8. Synthesis: Inferred Apple Software Audio Strategy for A2337

By analyzing the reverse-engineering efforts and explicit design choices of the Asahi Linux audio project, we can construct a plausible picture of the software-based audio enhancement strategy likely employed by Apple within macOS for the MacBook Air A2337's internal speakers.

8.1. Summary of Likely Apple Techniques

The evidence gathered suggests Apple's approach on the A2337 involves a multi-faceted DSP strategy aimed at maximizing perceived audio quality and loudness from the physically constrained hardware:

- **Aggressive Equalization:** Apple undoubtedly applies significant equalization. Unlike Asahi's goal of neutrality, Apple's EQ is likely tuned to compensate for speaker deficiencies while also shaping the sound towards a specific target curve preferred by consumers, potentially boosting bass and treble frequencies for a subjectively richer or more exciting sound.[1]
- **Dynamic Range Compression (DRC):** MacOS is inferred to use DRC to manage the dynamic range of audio signals. This helps increase perceived loudness, prevents digital and analog clipping at high volumes, and potentially improves audibility in various listening conditions.[1] The exact nature and intensity of this compression are proprietary.
- **Psychoacoustic Bass Enhancement:** To overcome the physical inability of the small speakers to reproduce very low frequencies, Apple likely employs psychoacoustic algorithms that manipulate harmonics to create the *illusion* of deeper bass than is actually being generated.[1]
- **Spatial Processing:** Features marketed as "Wide stereo sound" and support for "Dolby Atmos playback" indicate the use of spatial DSP techniques.[6] These likely involve algorithms designed to broaden the stereo image and potentially simulate surround or height channels from the built-in stereo speakers, aiming for a more immersive experience.[29]
- **Software-Based Thermal/Excursion Protection:** A critical component is a

sophisticated, proprietary protection system running in software on the M1 CPU.[2] This system likely uses real-time voltage and current feedback from the speaker amplifier chips (e.g., TI TAS2764/TAS5770L) to model the speakers' thermal state and potentially their excursion limits. It dynamically adjusts the output power/volume to prevent physical damage while pushing the speakers to their maximum safe performance limits.[2]

8.2. Contrast with Asahi's Philosophy and Implementation

The Asahi Linux project provides a valuable counterpoint, highlighting different philosophical goals and implementation choices:

- **Neutrality vs. Enhancement:** Asahi prioritizes objective fidelity and a neutral, flat frequency response baseline, avoiding the exaggerated EQ curves and psychoacoustic effects inferred in macOS.[1]
- **Accuracy vs. Artifacts:** Asahi developers specifically rejected psychoacoustic bass due to perceived bugs and artifacts in Apple's implementation, favoring accurate reproduction of achievable frequencies.[1] They also avoid hard-coding spatial effects, leaving such subjective enhancements to the user.[16]
- **Openness vs. Proprietary:** The Asahi speaker protection system (speakersafetyd) is open-source and developed through community reverse engineering [9], contrasting sharply with Apple's closed-source, integrated macOS solution.
- **Phased vs. Integrated:** Asahi's implementation appears phased, with core EQ and safety prioritized, while some dynamic processing elements (like a final limiter) were noted as pending implementation.[16] Apple's solution is presumably fully integrated within macOS.

8.3. Comparative DSP Approaches (macOS vs. Asahi Linux on A2337)

The following table summarizes the key differences between the inferred macOS approach and the documented Asahi Linux approach for speaker DSP on the A2337, based on the analysis presented:

DSP Aspect	Inferred macOS Approach (A2337)	Asahi Linux Approach (A2337)
EQ Goal	Perceptually pleasing curve (likely boosted bass/treble, e.g., Harman-like); compensates for hardware flaws.[1]	Mostly flat frequency response; neutrality; accurate reproduction; compensates for hardware flaws.[1]
EQ Implementation	Proprietary filters applied within macOS audio stack.	Model-specific impulse response files (.irs) applied via PipeWire convolver plugin (likely FIR filters).[1]
DRC/Limiting	Likely employs DRC/limiting for	Includes an input compressor;

	loudness maximization and distortion prevention.[1]	noted absence of a final limiter/compressor stage (as of documentation date).[16]
Psychoacoustic Bass	Likely uses techniques to enhance perceived bass beyond physical capabilities.[1]	Explicitly avoided due to perceived artifacts and coloration; focuses on accurate bass reproduction.[1]
Spatial Processing	Implements "Wide stereo sound" and Dolby Atmos virtualization for enhanced soundstage.[6]	Avoids hard-coded spatial effects; aims for natural stereo reproduction; user can add effects via plugins.[1]
Speaker Protection	Sophisticated, proprietary software model running on M1 CPU, using V/I sense feedback.[2]	Open-source speakersafetyd daemon using thermal modeling based on V/I sense feedback, plus kernel interlocks.[9]

8.4. Insight: The Symbiotic Relationship of DSP and Hardware

The analysis strongly indicates that the audio performance of the MacBook Air A2337 is not merely a function of its physical speaker components but is fundamentally reliant on a deep, symbiotic integration between the hardware and a complex software DSP stack executed on the M1 chip. The speakers themselves, while potentially of reasonable quality for their size, sound poor and are inherently unsafe to drive at high levels without the corrective and protective software layer.[2]

Apple designs these systems holistically, leveraging the processing power of Apple Silicon to implement DSP strategies that compensate for physical limitations and achieve a desired acoustic target. This includes not only shaping the frequency response and dynamics but also incorporating critical safety mechanisms that depend on specific hardware features like the V/I sensing capabilities of the chosen amplifier chips.[9] The effort required by the Asahi Linux project to reverse-engineer and reimplement these functions, particularly the speakersafetyd protection daemon, underscores the extent to which the software is essential for the basic usability and safety of the hardware.[9] The perceived audio quality of the A2337 is thus an emergent property of this tightly coupled hardware-software system, where the software DSP is arguably as crucial as the physical transducers themselves.

9. Conclusion

9.1. Summary of Findings

The investigation into the software-based audio enhancement measures for the MacBook Air A2337's speakers, primarily through the lens of the Asahi Linux project's reverse-engineering efforts, reveals a sophisticated reliance on Digital Signal Processing. Apple likely employs a

combination of aggressive, model-specific equalization (tuned for perceptual appeal rather than strict neutrality), dynamic range compression, psychoacoustic bass enhancement, and spatial processing ("Wide stereo sound," Dolby Atmos virtualization) to maximize the perceived quality and loudness from the device's compact speakers. Critically, this is underpinned by a proprietary, software-based thermal and excursion protection system running on the M1 CPU, utilizing real-time feedback from speaker amplifier chips (likely TI TAS2764/TAS5770L variants) to prevent hardware damage while pushing performance limits. Asahi Linux replicates the necessary functionalities using an open-source stack built on PipeWire/WirePlumber, employing FIR filters via impulse responses for EQ, an input compressor, and the vital speakersafetyd daemon for thermal protection, while deliberately opting for a more neutral sound profile that avoids psychoacoustic bass and hard-coded spatial effects.

9.2. Complexity of Modern Laptop Audio

This analysis underscores the immense complexity involved in achieving high-quality audio reproduction from modern, thin-and-light laptops like the MacBook Air A2337. It is no longer sufficient to simply consider the quality of the speaker drivers in isolation. The final acoustic output is a result of an intricate interplay between the physical hardware (drivers, enclosure acoustics, amplifier capabilities like V/I sensing) and a highly tailored software DSP stack running on powerful, efficient processors like the Apple M1. The integration of sophisticated, real-time speaker protection algorithms is not merely an enhancement but a fundamental requirement for enabling the hardware to perform safely at its intended levels.

9.3. Significance of Open-Source Reverse Engineering

The work of the Asahi Linux project is invaluable not only for enabling an alternative operating system on Apple Silicon hardware but also for providing deep technical insights into the functioning of these proprietary systems. Their efforts in reverse-engineering the audio stack, particularly the complex and safety-critical speaker protection mechanisms, demystify the techniques Apple uses and highlight the challenges faced when attempting to support such hardware outside its native ecosystem.[9] The development of open-source solutions like speakersafetyd and the contributions to upstream projects like PipeWire and WirePlumber demonstrate the potential for the Linux community to tackle these complex audio processing challenges. This work is ongoing, with DSP profiles and software capabilities expected to continue evolving and improving over time, further refining the audio experience on Apple Silicon Macs under Linux.[2]

Referenzen

1. AsahiLinux/asahi-audio: Userspace audio for Asahi Linux - GitHub, Zugriff am April 19, 2025, https://github.com/AsahiLinux/asahi-audio
2. A better title for this would be "Asahi introduces advanced speaker DSP to Linux... | Hacker News, Zugriff am April 19, 2025, https://news.ycombinator.com/item?id=38239696

3. Speaker Support in Asahi Linux | Hacker News, Zugriff am April 19, 2025, https://news.ycombinator.com/item?id=38239503
4. MacBook Air 13" Retina Late 2020 (M1) Repair - iFixit, Zugriff am April 19, 2025, https://www.ifixit.com/Device/MacBook_Air_13%22_Late_2020
5. Apple MacBook Air "M1" 8 CPU/7 GPU 13" Specs - EveryMac.com, Zugriff am April 19, 2025, https://everymac.com/systems/apple/macbook-air/specs/macbook-air-m1-8-core-7-core-gpu-13-retina-display-2020-specs.html
6. MacBook Air (M1, 2020) - Technical Specifications - Apple Support, Zugriff am April 19, 2025, https://support.apple.com/en-us/111883
7. A2337 Audio Board for 2020 Apple M1 MacBook Air 13" - tekdep, Zugriff am April 19, 2025, https://tekdep.com/product/a2337-audio-board/
8. Asahi Linux - GeeksforGeeks, Zugriff am April 19, 2025, https://www.geeksforgeeks.org/asahi-linux-2/
9. AsahiLinux/speakersafetyd: Rust speaker safety daemon ... - GitHub, Zugriff am April 19, 2025, https://github.com/AsahiLinux/speakersafetyd
10. Asahi Linux speaker safety daemon written in Rust by marcan (experimental) - Reddit, Zugriff am April 19, 2025, https://www.reddit.com/r/linux/comments/11lv2ko/asahi_linux_speaker_safety_daemon_written_in_rust/
11. Is speaker support still not ready? : r/AsahiLinux - Reddit, Zugriff am April 19, 2025, https://www.reddit.com/r/AsahiLinux/comments/1613qav/is_speaker_support_still_not_ready/
12. Linux on Apple Silicon with Alyssa Rosenzweig [audio] - Hacker News, Zugriff am April 19, 2025, https://news.ycombinator.com/item?id=42021237
13. fine tuning speakers : r/AsahiLinux - Reddit, Zugriff am April 19, 2025, https://www.reddit.com/r/AsahiLinux/comments/18mgwvf/fine_tuning_speakers/
14. Updates galore! November 2022 Progress Report - Asahi Linux, Zugriff am April 19, 2025, https://asahilinux.org/2022/11/november-2022-report/
15. Blog - Asahi Linux, Zugriff am April 19, 2025, https://asahilinux.org/blog/
16. SW:Speakers | Asahi Linux Wiki, Zugriff am April 19, 2025, https://leo3418.github.io/asahi-wiki-build/swspeakers/
17. New in Fedora Asahi Remix, Zugriff am April 19, 2025, https://asahilinux.org/2024/01/fedora-asahi-new/
18. The first Asahi Linux Alpha Release is here!, Zugriff am April 19, 2025, https://asahilinux.org/2022/03/asahi-linux-alpha-release/
19. asahi-audio/test_notes/j31x.md at main - GitHub, Zugriff am April 19, 2025, https://github.com/AsahiLinux/asahi-audio/blob/main/test_notes/j31x.md
20. Progress Report: Linux 6.14, Zugriff am April 19, 2025, https://asahilinux.org/2025/03/progress-report-6-14/
21. TAS5770AC0 Audio IC for Macbook Air 13.3" M1 A2337 Ori - deviceparts, Zugriff am April 19, 2025, https://www.deviceparts.com/tas5770ac0-audio-ic-for-macbook-air-133-m1-a2337-ori.html
22. TAS2764 data sheet, product information and support | TI.com, Zugriff am April 19,

2025, https://www.ti.com/product/TAS2764
23. TAS2764 Digital Input Mono Class-D Audio Amplifier With Speaker IV Sense - Texas Instruments, Zugriff am April 19, 2025, https://www.ti.com/lit/ds/symlink/tas2764.pdf
24. Series comparison - Patchew, Zugriff am April 19, 2025, https://patchew.org/linux/20250215-apple-codec-changes-v1-0-723569b21b19@gmail.com/diff/20250227-apple-codec-changes-v3-0-cbb130030acf@gmail.com/
25. TAS2764 | Buy TI Parts | TI.com, Zugriff am April 19, 2025, https://www.ti.com/product/TAS2764/part-details/TAS2764YBHR
26. Is the speaker popping/burning out thing an actual thing? + YouTube + How to contribute : r/AsahiLinux - Reddit, Zugriff am April 19, 2025, https://www.reddit.com/r/AsahiLinux/comments/wy2gv9/is_the_speaker_popping_burning_out_thing_an_actual/
27. MacBook Air (Retina, 13-inch, 2020) - Technical Specifications - Apple Support, Zugriff am April 19, 2025, https://support.apple.com/en-us/111991
28. MacBook Air (Apple silicon) - Wikipedia, Zugriff am April 19, 2025, https://en.wikipedia.org/wiki/MacBook_Air_(Apple_silicon)
29. Spatial Audio on the new MacBook Pro is absolutely incredible : r/apple - Reddit, Zugriff am April 19, 2025, https://www.reddit.com/r/apple/comments/qglz8q/spatial_audio_on_the_new_macbook_pro_is/
30. Dolby Atmos Speaker Setup, Zugriff am April 19, 2025, https://www.dolby.com/about/support/guide/dolby-atmos-speaker-setup/
31. Help needed- I've tried everything. My Macbook Air (M1, 2020-macOS ventura) is INCAPABLE of producing stereo output. System-wide, everything comes out in Mono. I've played with the settings, reset my core driver, deleted preference files, no budge. : r/mac - Reddit, Zugriff am April 19, 2025, https://www.reddit.com/r/mac/comments/10owxco/help_needed_ive_tried_everything_my_macbook_air/
32. How can I turn off or on wide stereo sound for MacBook Air M1 2020?, Zugriff am April 19, 2025, https://discussions.apple.com/thread/253858662
33. 2020 Macbook Air M1 fake stereo audio - Apple Support Communities, Zugriff am April 19, 2025, https://discussions.apple.com/thread/253608071
34. How to disable wide soundstage DSP on Macbook Air M1 (Macos Sonoma)? - Ask Different, Zugriff am April 19, 2025, https://apple.stackexchange.com/questions/467524/how-to-disable-wide-soundstage-dsp-on-macbook-air-m1-macos-sonoma
35. New A1932 A2179 A2337 Headphone Audio Jack Board 820-01124-A 820-01992-A 820-01929-A For MacBook Air 13" 2018 2019 2020 Year - AliExpress, Zugriff am April 19, 2025, https://www.aliexpress.com/item/1005005398155282.html
36. Laptop A2337 I/o Power Audio Jack Board With Flex Cable 820-01929-a 821-03452-01 For Macbook Air Retina 13 - AliExpress, Zugriff am April 19, 2025, https://www.aliexpress.com/i/1005005119168553.html

37. Audio Board - 2020 A2337 13 MacBook Air (820-01929) | eBay, Zugriff am April 19, 2025, https://www.ebay.com/itm/166074968835
38. A2337 Audio Board with IPD Cable for MacBook Air (M1, 2020) 820-01929-A, 923-03672, 923-03663, 821-03452-A, CPQ204904 - Amazon.com, Zugriff am April 19, 2025, https://www.amazon.com/A2337-820-01929-923-03672-923-03663-821-03452/dp/B0DLTNY57Y
39. MacBook Air 13" (A2337, Late 2020) Audio Daughterboard - iFixit, Zugriff am April 19, 2025, https://www.ifixit.com/products/macbook-air-13-a2337-late-2020-audio-daughterboard
40. MacBook Air 13" A2337 Late 2020 MGN63LL/A Genuine Audio Board 820-01929-a, Zugriff am April 19, 2025, https://www.gotlaptopparts.com/products/macbook-air-13-a2337-late-2020-mgn63ll-a-genuine-audio-board-820-01929-a
41. MODEL M1 - 2 Channel Wireless Streaming Amplifier with 100W, eARC and HEOS Built-in | Marantz, Zugriff am April 19, 2025, https://www.marantz.com/en-us/product/amplifiers/model-m1/300836.html
42. Apple MacBook Air 13" A2337 M1 2020 Audio Board Replacement OEM - eBay, Zugriff am April 19, 2025, https://www.ebay.com/itm/265708160859
43. 923-03673 MacBook Air 13 M1 2020 A2337 Audio Board, Silver 820-01929-A - Usedmac, Zugriff am April 19, 2025, https://usedmac.com/product/923-03673/
44. A2337 Audio Board for for 2020 Apple M1 MacBook Air 13” 923-03672 - DV Warehouse, Zugriff am April 19, 2025, https://www.dvwarehouse.com/audio-board-for-macbook-air-13-2020-m1-chip-a2337.html
45. MacBook Air 13" A2337 - 2020 - Beetstech, Zugriff am April 19, 2025, https://beetstech.com/store/apple-parts/macbook-air-13-inch-a2337-2020
46. New A2337 Audio Jack Board For Macbook Air 13" A2337 Headphone Jack Board 820-01929-A Late 2020 EMC 3598 - AliExpress, Zugriff am April 19, 2025, https://www.aliexpress.com/i/1005008313376171.html
47. Genuine Apple MacBook Air 13" M1 2020 A2337 OEM Audio Board Space Gray / Gold, Zugriff am April 19, 2025, https://www.ebay.com/itm/175990186016
48. (CS42L83A) Audio Jack Codec IC - MacBook Pro A1706, A1708 - Beetstech, Zugriff am April 19, 2025, https://beetstech.com/product/audio-jack-codec-ic-cs42l83a
49. Any measurements of Macbook M1 base model's DAC? | Audio Science Review (ASR) Forum, Zugriff am April 19, 2025, https://www.audiosciencereview.com/forum/index.php?threads/any-measurements-of-macbook-m1-base-models-dac.40295/