ATTI Forward White-Box Modeling: Technical White Paper

Analog Topology-based Transient Integration

Prepared by: M-VAVE Audio Algorithm Team Version: V1.1 — Principles and Validation Requirements Date : September 2026 Category: Audio Algorithms / Circuit Simulation

Abstract — ATTI (Analog Topology-based Transient Integration) is a white-box circuit modeling framework for guitar effects. It follows a forward modeling approach: circuit connections, device models, input signals, and control parameters define circuit equations, which are solved at discrete time points to obtain node voltages, associated states, and the output response. This paper uses the term "vector topology" for the data representation that organizes device connections and solver indices, and "vector-topology transient solving" for the process that combines this representation with numerical integration. The architecture builds on established circuit simulation methods, with engineering emphasis on circuit representation, reuse of solver structures, and audio processing constraints. This paper describes its mathematical foundations, design boundaries, and validation requirements. The supported scope, accuracy, and real-time performance of a specific product must be confirmed by a version-specific test report.

1. Modeling Objectives and Terminology

The response of a guitar effect depends jointly on nonlinear devices, energy-storage elements, interstage loading, feedback networks, and control settings. White-box modeling explicitly describes these relationships to construct a digital model linked to the circuit structure. Its accuracy depends on circuit documentation, device models, parameter calibration, and numerical solver configuration. A white-box approach alone does not guarantee an exact match to a particular physical unit.

"White-box" means that the model explicitly uses circuit structure and device relationships; "forward" means predicting the circuit response from a given model and input conditions. Here, forward refers to the direction of the modeling problem, as distinguished from identification, which estimates parameters from measurements. In circuits with feedback and mutual loading, all parts must satisfy coupled constraints and generally cannot be computed independently in sequence from input to output.

Model construction: Circuit documentation and device models → Circuit equations and index structures
Runtime processing: Input, controls, and history → Coupled solution for the current time step → State update and audio output

In this paper, "SPICE-like" refers to methods such as netlist description, assembly of device equations, and numerical transient solving. It does not imply the use of a particular SPICE software kernel or compatibility with all of its syntax and device models. "Vector topology" is this paper's name for an engineering data representation. If "topological integration" appears in related materials, it is only shorthand for the solving process described above, not a separate theory of numerical integration.

2. Circuit Representation and Solver Architecture

2.1 Circuit Equations and Unknowns

SPICE-like methods assemble circuit equations from device port relationships and Kirchhoff's laws. With Modified Nodal Analysis (MNA), the unknowns typically include nonreference node voltages, any necessary branch currents, and internal variables introduced by the models. Not every branch current needs to be an independent unknown. Dynamic circuits can be described generally using differential-algebraic equations. [1]

General implicit form: $F(x, \dot{x}, t, u, p) = 0$
x is the vector of circuit unknowns, $\dot{x}$ its time derivative, t time, u the input and external excitations, and p the device and control parameters. Purely algebraic constraints contain no corresponding derivative terms. The vector of unknowns and the set of independent dynamic states need not have the same dimension. This expression identifies the problem class; it does not, by itself, constitute an algorithmic innovation of ATTI.

After time discretization, the current unknowns are solved at each internal time step so that device relationships and circuit constraints hold simultaneously within specified tolerances. The accepted history is then committed. Feedback, reverse coupling, and loading effects should be included in the joint equations or in a partitioned formulation whose equivalence has been validated.

2.2 The Vector Topology Data Representation

Vector topology organizes circuit connections, device instances, and their mappings to solver variables. "Vector" denotes indexed collections of data; it does not imply that these collections share physical dimensions or that SIMD parallel processing is used. At the architectural level, this paper distinguishes the following information:

1. **Node and port indices**
   Record reference nodes, internal nodes, input and output ports, and their mappings to equation unknowns.
2. **Device and branch records**
   Record device types, terminal lists, parameters, and branch reference directions. Multiterminal devices such as transistors and operational amplifiers connect through terminal mappings and their model equations; they cannot all be reduced to a single two-terminal element.
3. **Dynamic states and history**
   Store the charge, flux linkage, or equivalent voltage and current history required by the integrator. The storage format depends on the device model and discretization method.
4. **Control and parameter mappings**
   Map knob, power-supply, and switch settings to parameters, external excitations, or topology events. Continuous parameter changes and switching events that alter connections are handled separately.
5. **Constraints and solver configuration**
   Record boundary conditions, initialization rules, equation structure, and error settings. Different data items have their own types, units, and lifetimes.

2.3 Established Methods and Engineering Contributions

Indexed representations, matrix assembly, numerical integration, and nonlinear iteration are established circuit simulation methods. ATTI's engineering design focuses on organizing these methods into a solver architecture for effects processing. Potential optimizations to evaluate include precompiling connectivity, reusing sparse structures, reducing repeated parameter calculations, and partitioning the problem while preserving port coupling.

The actual benefits of these approaches should be demonstrated through controlled comparisons using the same models, error requirements, and hardware conditions. Model simplification or approximate partitioning should be accompanied by a statement of applicability and error. Data reordering or caching alone is not evidence of improved accuracy, real-time capability, or originality. This paper does not provide implementation data sufficient to quantify these optimization benefits.

2.4 Internal States and Diagnostics

A white-box representation allows the model's internal node voltages, device currents, and dynamic states to be recorded, helping diagnose clipping, bias changes, and feedback responses. The ability to record internal states is distinct from "observability" in control theory, which concerns reconstructing all states from external measurements alone. Retaining states also does not imply that every audible difference can be uniquely attributed to a particular component.

Nonlinearity and memory are different properties. A static nonlinear relationship can have no internal memory. Capacitors, inductors, and device models that include charge storage, thermal processes, or other dynamic mechanisms introduce the corresponding history dependence. Mechanisms absent from a model do not appear automatically through white-box solving.

3. Transient Solving and the Audio Timeline

3.1 Discretization of Dynamic Elements

Numerical integration converts dynamic device relationships into discrete relationships between current unknowns and history terms. Backward Euler, the trapezoidal rule, and Gear/BDF methods are common choices, with different accuracy, damping, and stability characteristics. The specific method and order should be documented in the version configuration. [1] The following example explains discretization; it does not indicate that the product has selected this integrator.

Example: Backward Euler companion relationship for a linear capacitor
For constant capacitance C, the passive sign convention gives i = C · dv/dt. With time-step size h:
i = (C/h) · v − (C/h) · v
The current step can be represented by a conductance C/h and a current source determined by the accepted voltage history. The history is updated after the current step is accepted.

Consider an RC circuit in which an input voltage source drives, through a resistor R, a capacitor C connected to ground, with no additional output load. The output node voltage satisfies (u − v)/R = C(v − v)/h, giving:

v = [h · u + RC · v] / (h + RC)
This recurrence illustrates how the input, history, and circuit parameters jointly determine the current output. When nonlinear devices are added, their current or charge relationships must be incorporated into the node equations. This is an illustrative example, not a measured ATTI result.

3.2 Nonlinear Solving and State Commit
The discretized nonlinear equations can be written as r(x) = 0, with history terms, current input, and parameters given for that solve. Common solution methods include Newton–Raphson and its damped variants, which use the residual and Jacobian matrix to correct the unknowns at each iteration. [1] These are general solver foundations. The method used by a specific ATTI version must be confirmed by its implementation documentation.

1. **Prepare the current step**
   Read the input, events, and accepted history; form an initial guess and the discrete relationships for dynamic devices.
2. **Assemble and solve**
   Assemble the coupled equations according to terminal mappings and iterate over nonlinear unknowns. Linear cases can be solved directly.
3. **Check acceptance criteria**
   Check equation residuals and changes in unknowns using absolute and relative tolerances appropriate to their physical dimensions. Check time-discretization error as well when necessary.
4. **Commit or fall back**
   Commit history only for accepted time steps. On failure, apply the version-defined budgets and fallback strategy, and record failure and protection events.

Convergence means that the discrete equations satisfy specified tolerances. It does not, by itself, establish accuracy relative to the continuous circuit or a physical unit. Device operating regions are determined by the equation solution; “reaching a reasonable operating region” cannot replace convergence criteria. If damping, step limiting, or regularization is used, its effect on the target response should also be checked.

3.3 Internal Step Size and Fixed Audio Sample Rate
The audio interface delivers data at a fixed sample rate f , while the internal solver may use a different time grid. A fixed integer oversampling factor M corresponds to an internal step size h = 1/(M f) and requires defined input interpolation, output low-pass filtering, and decimation. If adaptive time stepping is used, reconstruction from nonuniform internal time points to the audio sample grid, filtering, and error control must also be defined.

Reducing the step size may improve time-discretization accuracy, but increases computation. Real-time processing must finish within audio deadlines; delaying output delivery cannot provide unlimited solving time. Internal step size, substep count, iteration count, and retry count should all have explicit budgets. This discussion does not claim that ATTI has implemented adaptive time stepping or dynamic oversampling.

3.4 Antialiasing, Control Events, and Output Processing
Nonlinearity generates additional frequency components. Oversampling and appropriate filtering reduce aliasing; a smaller integration step alone cannot replace a complete antialiasing design. Discontinuities caused by knob changes and bypass switching require parameter smoothing, crossfading, or suitable state transitions. Their time constants and effects on the response should be documented separately.

The audio interface must specify the conversion between digital levels and model port voltages, input source impedance, output loading, and DC handling. If the product adds DC filtering or protective limiting, their locations and trigger conditions should be documented, and validation should record the circuit model output and final product output separately. Protection activation changes the waveform and cannot serve as evidence of circuit-model agreement.

4. Module Structure and Real-Time Design Requirements

The table below describes this paper’s architectural modules and the engineering information that must be confirmed for each. It is not a list of implemented features.

Module	Responsibility	Information to Confirm for Each Version
Circuit input and parsing	Convert supported circuit descriptions into device instances and terminal connections	Supported netlist syntax subset, model list, handling of unsupported items
Topology and equation organization	Build variable indices, device mappings, and solver structures	Matrix representation, cache invalidation conditions, topology event handling
Initial state	Establish a DC operating point or specified power-on state	Initialization method, handling of inconsistent initial conditions and failures
Transient and nonlinear solving	Solve the current step and update history	Integrator, tolerances, iteration limits, substep and retry budgets
Audio interface	Perform level mapping, sample-rate conversion, and output processing	Sample rates, filter configuration, latency, protection conditions
Diagnostics and runtime monitoring	Record errors, processing times, and exceptional even	Logging overhead, deadline-miss statistics, recovery after failure

4.1 Structure Reuse and Coupling Boundaries
When topology is unchanged, indices and matrix sparsity structures can be prepared in advance. Whether numerical factorizations can be reused depends on whether matrix coefficients change. Knob adjustments and nonlinear iterations may invalidate relevant caches. Modular solving must also preserve port impedances, feedback, and interstage loading, or explicitly state the conditions under which approximations apply. Splitting a circuit into modules does not automatically guarantee equivalence to a full-system solution.

Potential Optimizations to Evaluate
- Precompile connectivity and variable mappings to reduce repeated parsing
- Reuse sparse structures and numerical results while their validity conditions hold
- Eliminate suitable linear unknowns while preserving coupling
- Move nonessential logging and resource allocation out of the audio processing path

Boundaries Requiring Validation
- The model’s applicable level, frequency, temperature, and control ranges
- Strong feedback, ill-conditioned equations, and nodes lacking a reference connection
- State handling at power-on, during abrupt parameter changes, and after topology changes
- Errors introduced by approximations, caching, and scheduling changes

4.2 Real-Time Budgets and Failure Recovery
For an audio interface with B samples per block, the block duration is B/f□. The actual processing budget must account for other tasks on the device and retain a safety margin. Average CPU usage alone is insufficient to support a claim of reliable real-time operation. Reports also need peak block processing times within the test scope, deadline-miss counts, and stress conditions. An observed peak is not a proof of worst-case execution time for all inputs.

Nonconvergence, nonfinite values, and exhausted budgets should have explicit recovery strategies. Depending on the application, a product may use bounded retries, degraded operation, controlled bypass, or muting, but switching transients, state recovery, and behavior under persistent failure should be validated. Protection mechanisms keep the output controlled; they do not replace passing accuracy or real-time results. The chosen strategies and thresholds must be listed in the version documentation.

5. Relationship to Other Modeling Methods

White-box, black-box, and gray-box describe how extensively a model uses knowledge of internal mechanisms. DSP is the general term for digital signal processing and encompasses digital implementations of these approaches. Methods such as WDF, state-space modeling, and MNA may also be combined. The table below outlines typical differences; it is not a ranking of sound quality or performance. Related circuit modeling and platform comparisons are discussed in references [2] and [3].

Method or Implementation	Model Basis	Dynamics and Controls	Typical Characteristics	Key Validation Questions
SPICE-like transient solving	Circuit connections and device equations	Represented through dynamic devices, parameters, and excitations	Supports interpretation in terms of node and device states; general-purpose implementations are often used for offline analysis	Model applicability, convergence, discretization error, and runtime overhead
Wave digital filters (WDF)	Circuit ports and wave-variable representations	Can represent energy storage, nonlinearity, and parameter changes	Suitable for modular circuit modeling; complex connections and multiple nonlinear devices require corresponding solver structures	Connection methods, nonlinear solving, and suitability for the target circuit
Nonlinear state-space models	State, input, and output relationships derived from circuits	Dynamic states are updated explicitly or implicitly; controls can enter the equations	Facilitate the organization of dynamic systems; implementation cost varies with the model and solution method	Equation derivation, feedback handling, discretization, and stability
Black-box behavioral models	Input-output measurements and system identification or training	Can include memory; conditioned models can accept control parameters such as knob settings	Do not require complete circuit documentation; internal variables generally do not directly correspond to physical nodes	Data coverage, generalization to unseen inputs and control combinations, and runtime cost
Models built from filters and nonlinear blocks	Target responses, measurements, or partial circuit derivations	Can include feedback, envelopes, and parameter automation	Flexible structure and cost; physical interpretability depends on how the model is constructed	Whether the selected structure covers the target dynamics, nonlinearity, and control range
Gray-box or hybrid models	Partial physical structure combined with measurement-based fitting	Determined by the combined structure and parameter design	Allow tradeoffs among physical modeling, data requirements, and computation	Module interfaces, identifiability, phase, and accumulated error
ATTI (the architecture in this paper)	SPICE-like device relationships and an indexed circuit representation	Designed to handle states, parameters, and events through coupled solving	A white-box approach focused on circuit organization, structure reuse, and audio processing constraints	Supported scope, optimization benefits, and real-time performance await confirmation in version-specific reports

Black-box models are not equivalent to recording playback or static lookup tables. Recurrent neural networks can represent history dependence, and conditioned architectures can model control parameters. Their accuracy and generalization still require validation for the target device. [4] White-box methods offer explicit representation of physical structure, but remain limited by missing model mechanisms, parameter errors, and numerical approximations. Different approaches should be compared using the same inputs, control ranges, and resource conditions.

6. Validation and Evidence for Release

6.1 Validation Levels

1. **Structure and basic equations**
   Check node references, port polarity, units, device mappings, and KCL/KVL consistency. Verify basic behavior using analytically solvable cases such as resistor networks and RC/RL dynamics. Numerical acceptance criteria should include appropriate tolerances.

2. **Numerical implementation**
   Compare against a reference solver using the same netlist, device parameters, initial conditions, input, and loading. Align time grids and examine changes as step sizes are reduced or tolerances tightened, distinguishing solver error from model differences. Repeatability supports regression checks; it is not equivalent to accuracy.

3. **Agreement with physical hardware**
   Record the hardware revision, power supply, knob settings, signal levels, source impedance, load, and measurement chain. Compare model output with measured responses. Separate calibration signals from independent validation signals, and document variation among units and measurement uncertainty.

4. **Audio and real-time operation**
   Test sine waves at multiple levels, frequency sweeps, multitone signals, band-limited transients, envelopes, and performance excerpts, covering control changes and abnormal inputs. Record audio errors, block processing times, nonconvergence, deadline misses, and protection activation. Listening comparisons should specify level matching and the test method.

Agreement with a reference SPICE solver supports implementation correctness for the same model. Agreement with physical hardware provides additional support for the validity of the selected models and parameters. Neither replaces the other. Additional filtering, limiting, or model fitting should also be included in the report to avoid conflating the circuit core response with that of the complete product.

6.2 Minimum Public Validation Case

A report supporting conclusions about accuracy and real-time performance should include at least one specific circuit and cover its representative operating range. The following are reporting requirements. Results not included in this version must not be treated as tests already passed.

Report Item	Minimum Required Information	Status in This Version
Target and model	Schematic or verifiable description, device model sources and versions, parameters, initial conditions, power supply, and loading	No specific validation circuit included
Solver configuration	Integrator, tolerances, step size, oversampling filters, iteration and fallback limits, and numerical precision	
Reference and measurements	Reference solver version, physical hardware and measurement chain, level calibration, and time-alignment method	No product configuration included No comparison records included
Audio error	Frequency response, waveforms, harmonics or intermodulation, aliasing, and dynamic response; metric definitions, units, and pass thresholds	No plots or numerical results included
Runtime performance	Hardware, software build, sample rate, buffer size, test duration, average and peak processing times, and exception counts	No benchmark results included
Reproducibility and applicability	Test inputs or generation rules, control sequences, version identifiers, known failure conditions, and report date	To be provided with the validation report

6.3 Scope of Conclusions in This Document

This version describes ATTI's modeling approach and engineering requirements. It does not provide an implementation code audit, a product feature list, or a quantitative test report. It therefore does not establish that a particular platform meets real-time requirements, quantify error relative to physical hardware, or demonstrate a performance advantage over other methods. Public validation may withhold internal code and device-tuning details, but should provide sufficient test conditions, metrics, and results to assess the conclusions.

Target applications include overdrive, distortion, fuzz, analog compression, filtering, and related feedback networks. The actual modeling scope depends on the supported devices and dynamic mechanisms. For example, an optical compressor requires an appropriate model of the optoelectronic response. Purely digital delay, convolution reverb, and similar effects can be implemented by their respective algorithm modules, with circuit models forming part of the overall signal chain.

Technical Positioning — ATTI builds its models on **circuit structure and device relationships** and predicts dynamic responses through **coupled transient solving**. Vector topology supports the organization and computation of circuit data. Its engineering value is demonstrated by verifiable feature boundaries, error results, and runtime performance.

References
[1] ngspice Project Team
Ngspice User's Manual, continuously updated online edition; see the sections on Convergence and Transient Analysis Options. Accessed: 2026-09-07. Product validation should separately specify the exact software and manual versions used.
[2] David T. Yeh; Julius O. Smith
Simulating guitar distortion circuits using wave digital and nonlinear state-space formulations. DAFx, 2008.
[3] Jatin Chowdhury
A Comparison of Virtual Analog Modelling Techniques for Desktop and Embedded Implementations. arXiv:2009.02833, 2020.
[4] Yen-Tung Yeh; Wen-Yi Hsiao; Yi-Hsuan Yang
Hyper Recurrent Neural Network: Condition Mechanisms for Black-Box Audio Effect Modeling. DAFx, 2024.

The references support the general technical background; they do not certify the ATTI implementation or endorse its performance. Design requirements, illustrative examples, and validation results yet to be supplied represent different forms of evidence. Product support claims and quantitative conclusions should be tied to an identified software version, hardware configuration, and test report.

CN

ATTI 正向白盒算法技术白皮书

基于模拟电路拓扑的瞬态积分

编制：M-VAVE 音频算法团队　　版本：V1.1 — 原理与验证要求　　日期：2026 年 9 月　　分类：音频算法 / 电路仿真

摘要 — ATTI（Analog Topology-based Transient Integration，基于模拟电路拓扑的瞬态积分）是面向吉他效果器的白盒电路建模框架。它采用正向建模路线：根据电路连接、器件模型、输入信号和控制参数建立电路方程，在离散时间点求解节点电压及相关状态，得到输出响应。本文将用于组织器件连接与求解索引的数据表示称为“向量拓扑”，将其与数值积分结合的求解流程称为“向量拓扑瞬态求解”。这一架构建立在已有电路仿真方法之上，工程重点是电路表示、求解结构复用和音频运行约束。本文说明其数学基础、设计边界及验证要求；具体产品的支持范围、精度和实时性能需要由版本测试报告确认。

1. 建模目标与术语

吉他效果器的响应受到非线性器件、储能元件、级间负载、反馈网络和控制设置的共同影响。白盒建模通过显式描述这些关系，建立能够关联到电路结构的数字模型。其准确性取决于电路资料、器件模型、参数标定和数值求解配置；采用白盒路线本身并不保证与某台实物完全一致。

“白盒”指模型显式利用电路结构与器件关系；“正向”指根据给定模型和输入条件预测电路响应。这里的正向是建模问题的方向，与从测量结果估计参数的辨识问题相对应。对于包含反馈和相互负载的电路，各部分需要满足耦合约束，不能一般性地按输入端到输出端的顺序独立计算。

建模流程： 电路资料与器件模型 → 电路方程及索引结构
运行流程： 输入、控制量与历史状态 → 当前时间步的耦合求解 → 状态更新与音频输出

本文中的“SPICE 类”表示沿用网表描述、器件方程组装和瞬态数值求解等方法，不据此声明使用某个 SPICE 软件内核，或兼容其全部语法与器件模型。“向量拓扑”是本文对工程数据表示的命名；“拓扑积分”如用于相关材料，仅作为上述求解流程的简称，不表示一种独立的数值积分理论。

2. 电路表示与求解架构

2.1 电路方程与未知量

SPICE 类方法依据器件端口关系和基尔霍夫定律组装电路方程。采用改进节点分析（Modified Nodal Analysis，MNA）时，未知量通常包括非参考节点电压、必要的支路电流及模型引入的内部变量；并非每条支路电流都必须作为独立未知量。动态电路可用微分代数方程作一般性描述。[1]

一般隐式表达： $F(x, \dot{x}, t, u, p) = 0$
x 为电路未知量向量，X 为其时间导数，t 为时间，u 为输入及外部激励，p 为器件与控制参数。纯代数约束不含相应导数项；未知量向量与独立动态状态不必具有相同维数。该表达说明问题类型，不单独构成 ATTI 的算法创新。

时间离散后，在每个内部时间步内求解当前未知量，使器件关系与电路约束在规定容差内同时成立，再提交已接受的历史状态。反馈、反向耦合和负载效应应包含在联合方程或经验证的等价分块中。

2.2 向量拓扑的数据表示

向量拓扑用于组织电路连接、器件实例及其与求解变量的映射。“向量”表示索引化的数据集合，不表示这些集合具有相同量纲，也不直接意味着采用 SIMD 并行计算。本文在架构层面区分以下信息：

1. **节点与端口索引**
   记录参考节点、内部节点和输入输出端口，以及它们与方程未知量的映射。
2. **器件与支路记录**
   记录器件类型、端子列表、参数和支路参考方向。晶体管、运放等多端器件通过端子映射及其模型方程接入，不能统一简化为一个二端元件。
3. **动态状态与历史量**
   保存积分器所需的电荷、磁链或等价电压电流历史量；具体存储形式由器件模型和离散方法决定。
4. **控制与参数映射**
   将旋钮、供电及开关设置映射为参数、外部激励或拓扑事件。连续参数变化与改变连接关系的开关事件分别处理。
5. **约束与求解配置**
   记录边界条件、初始化规则、方程结构及误差设置。其中不同数据具有各自的类型、单位与生命周期。

2.3 通用方法与工程贡献

索引化表示、矩阵组装、数值积分和非线性迭代属于已有电路仿真方法。ATTI 的工程设计关注如何将这些方法组织为面向效果器的求解架构。可评估的优化方向包括预编译连接关系、复用稀疏结构、减少重复参数计算，以及保留端口耦合的分块处理。

这些方向的实际收益应通过相同模型、误差要求和硬件条件下的对照测试说明。若采用模型简化或近似分块，应报告适用范围与误差；数据重排或缓存本身不能作为精度提升、实时能力或独创性的证据。本文未提供足以量化上述优化收益的实现数据。

2.4 内部状态与诊断

白盒表示允许记录模型内部的节点电压、器件电流和动态状态，辅助定位削波、偏置变化和反馈响应。这里的“内部状态可记录”不等同于控制理论中仅由外部测量重建全部状态的“可观测性”。保留状态也不意味着每项听感差异都能唯一归因到某个元件。

非线性与记忆是不同性质。静态非线性关系可以没有内部记忆；电容、电感，以及包含电荷存储、热过程或其他动态机制的器件模型，才引入相应历史依赖。未纳入模型的机制不会因使用白盒求解而自动出现。

3. 瞬态求解与音频时间轴

3.1 动态元件的离散化

数值积分把动态器件关系转换为当前未知量与历史量之间的离散关系。后向欧拉、梯形法和 Gear/BDF 是常见选择，其精度、阻尼和稳定性特征不同；具体方法与阶数应在版本配置中列明。[1]下例用于解释离散化过程，不表示产品已选用该积分器。

示例： 线性电容的后向欧拉伴随关系
对恒定电容 C，采用被动符号约定，有 $i = C \cdot dv/dt$。时间步长为 h 时：
$i = (C/h) \cdot v - (C/h) \cdot v$
当前步可表示为电导 C/h 与由已接受历史电压确定的电流源。历史量在当前步被接受后更新。

以输入电压源经电阻 R 驱动对地电容 C、输出无额外负载的 RC 电路为例，输出节点电压满足 (u − v)/R = C(v − v)/h，因此：

v = [h · u + RC · v] / (h + RC)
该递推式展示输入、历史状态与电路参数如何共同决定当前输出。加入非线性器件后，需要将其电流或电荷关系并入节点方程。它是解释性算例，不是 ATTI 的实测结果。

3.2 非线性求解与状态提交
离散化后的非线性方程可记为 r(x⃞) = 0，历史量、当前输入和参数在该次求解中给定。常见求解方式为 Newton–Raphson 及其阻尼变体；每次迭代利用残差和雅可比矩阵修正未知量。[1]这属于通用求解基础，ATTI 具体版本采用的方法需由实现说明确认。

1. **准备当前步**
   读取输入、事件与已接受的历史量，形成初始猜测和动态器件的离散关系。
2. **组装并求解**
   按端子映射组装耦合方程，对非线性未知量进行迭代；线性情形可直接求解。
3. **检查接受条件**
   依据与量纲相适应的绝对及相对容差，检查方程残差和未知量增量；必要时同时检查时间离散误差。
4. **提交或回退**
   仅对接受的时间步提交历史状态。失败时按版本定义的预算和回退策略处理，记录失败及保护事件。

收敛表示离散方程满足指定容差，不能单独证明对连续电路或实物的准确性。器件工作区由方程解确定，不能用“落入合理工作区”替代收敛判据。阻尼、限步和正则化若被采用，还应检查其对目标响应的影响。

3.3 内部步长与固定音频采样率
音频接口按固定采样率 f交付数据，内部求解可以采用不同时间网格。固定整数过采样倍率 M 对应内部步长 h = 1/(M f)，需要定义输入插值、输出低通滤波及抽取。若采用自适应步长，还需定义非均匀内部时间点到音频采样网格的重建、滤波和误差控制。

减小步长可能改善时间离散精度，但会增加运算量。实时运行必须在音频截止时间内完成，不能通过延迟交付输出获得无限求解时间。内部步长、子步数、迭代次数和重试次数均应有明确预算；本文不据此声明 ATTI 已实现自适应步长或动态过采样。

3.4 抗混叠、控制事件与输出处理
非线性会产生额外频率成分。过采样及适当滤波用于降低混叠；减小积分步长不能单独替代完整的抗混叠设计。旋钮和旁路切换造成的不连续则需要参数平滑、交叉淡化或适当的状态转换，其时间常数及对响应的影响应另行说明。

音频接口需要规定数字电平与模型端口电压的换算、输入源阻抗、输出负载及直流处理。若产品加入额外的直流滤波或保护限幅，应说明位置和触发条件，并在验证中分别记录电路模型输出与最终产品输出。保护触发会改变波形，不能作为电路模型一致性的证据。

4. 模块划分与实时设计要求

下表给出本文的架构划分及各模块需要确认的工程信息，不作为已实现功能清单。

模块	职责	版本需确认的信息
电路输入与解析	将支持的电路描述转换为器件实例及端子连接	网表语法子集、模型清单、不支持项的处理
拓扑与方程组织	建立变量索引、器件映射和求解结构	矩阵表示、缓存失效条件、拓扑事件处理
初始状态	建立静态工作点或指定上电状态	初始化方法、不一致初值及失败处理
瞬态与非线性求解	求解当前步并更新历史状态	积分器、容差、迭代上限、子步与重试预算
音频接口	完成电平映射、采样转换与输出处理	采样率、滤波配置、延迟、保护条件
诊断与运行监测	记录误差、耗时和异常事件	记录开销、超时统计、失败后恢复方式

4.1 结构复用与耦合边界
在拓扑不变时，可预先建立索引和矩阵稀疏结构。是否能够复用数值分解，取决于矩阵系数是否变化；旋钮调整和非线性迭代可能使相关缓存失效。模块化求解还需保留端口阻抗、反馈和级间负载，或明确其近似条件。把电路拆成模块并不自动保证与整体求解等价。

可评估的优化方向
- 预编译连接关系与变量映射，减少重复解析
- 在有效条件内复用稀疏结构及数值计算结果
- 消去适合消元的线性未知量，保留耦合关系
- 将非必要日志和资源分配移出音频处理路径

需要验证的边界
- 模型适用的电平、频率、温度及控制范围
- 强反馈、病态方程和缺少参考连接的节点
- 上电、参数突变及拓扑改变时的状态处理
- 近似、缓存和调度变化引入的误差

4.2 实时预算与异常恢复
对于每块 B 个采样点的音频接口，块时长为 B/f。实际可用处理预算还需扣除设备上的其他任务并保留余量。报告平均 CPU 占用不足以支持稳定实时运行，还需要给出测试范围内的峰值块耗时、超时次数和压力条件。观察到的峰值也不等同于对所有输入的最坏执行时间证明。

未收敛、非有限数值和预算耗尽应具有明确的恢复策略。产品可根据场景选择有界重试、降级、受控旁路或静音，但应验证切换瞬态、状态恢复及持续失败行为。保护机制保证输出受控，不能代替精度或实时通过结果。具体选用策略及阈值由版本说明列出。

5. 与相关建模方法的关系

白盒、黑盒和灰盒描述模型对内部机理的利用程度；DSP 是数字信号处理的总称，涵盖这些路线的数字实现。WDF、状态空间与 MNA 等方法也可能组合使用。下表用于说明典型差异，不构成音质或性能排名。相关电路建模与平台比较见文献 [2]、[3]。

方法或实现	模型依据	动态与控制	典型特点	主要验证问题
SPICE 类瞬态求解	电路连接与器件方程	通过动态器件、参数及激励表达	便于关联节点和器件状态；通用实现常用于离线分析	模型适用性、收敛、离散误差及运行开销
波数字滤波器（WDF）	电路端口及波变量表示	可表达储能、非线性和参数变化	适合模块化电路建模；复杂连接与多个非线性器件需相应求解结构	连接方法、非线性求解及目标电路适配方程推导、反馈处理、离散化与稳定性
非线性状态空间模型	电路推导的状态、输入与输出关系	显式或隐式更新动态状态，控制量可进入方程	便于组织动态系统；实现成本随模型和求解方法变化	数据覆盖、未见输入与控制组合的泛化、运行成本
黑盒行为模型	输入输出测量与系统辨识或训练	可具有记忆；条件化模型可接收旋钮等控制参数	无需完整电路资料；内部变量通常不直接对应物理节点	所选结构是否覆盖目标动态、非线性与控制范围
滤波与非线性模块模型	目标响应、测量或部分电路推导	可包含反馈、包络及参数自动化	结构与成本灵活；物理解释程度取决于构建方式	模块接口、可辨识性、相位与累计误差
灰盒或混合模型	部分物理结构与测量拟合结合	由组合结构及参数设计决定	在机理建模、数据需求与计算量之间进行设计取舍	支持范围、优化收益和实时性能待版本
ATTI（本文架构）	SPICE 类器件关系及索引化电路表示	设计上通过耦合求解处理状态、参数和事件	属于白盒路线，工程关注电路组织、结构复用及音频运行约束	报告确认

黑盒模型不等同于录音回放或静态查表。循环神经网络可以表达历史依赖，条件化结构可以建模控制参数；其精度和泛化仍需针对目标设备验证。[4]白盒方法的优势在于物理结构的显式表达，但也受到模型缺项、参数偏差和数值近似的限制。不同路线应在相同输入、控制范围及资源条件下比较。

6. 验证与发布依据

下表给出本文的架构划分及各模块需要确认的工程信息，不作为已实现功能清单。

6.1 验证层次

1. **结构与基本方程**
   检查节点参考、端口极性、单位、器件映射及 KCL/KVL 一致性。以电阻网络、RC/RL 动态等可解析案例检查基本行为；数值接受条件应包含适当容差。

2. **数值实现**
   在相同网表、器件参数、初值、输入及负载下与参考求解器比较。对齐时间网格并检查步长或容差收紧后的变化，区分求解误差与模型差异。重复性是回归检查依据，不等于准确性。

3. **实物一致性**
   记录实物版本、供电、旋钮、电平、源阻抗、负载及测量链路。将模型输出与实测响应对照；校准所用信号与独立验证信号分开，并记录样机差异及测量不确定度。

4. **音频与实时运行**
   测试多电平正弦、扫频、多音、带限瞬态、包络和演奏片段，覆盖控制变化与异常输入。记录音频误差、块耗时、未收敛、超时及保护触发；听音比较需说明电平匹配与测试方法。

与参考 SPICE 求解一致，支持相同模型下的实现正确性；与实物一致，进一步支持所选模型和参数的有效性。两者不能相互替代。额外滤波、限幅或模型拟合也应纳入报告，避免混淆电路核心与完整产品的响应。

6.2 最小公开验证案例

支持精度与实时性结论的报告至少应包含一个具体电路，并覆盖其代表性工作范围。以下为报告要求，尚未附入本版本的结果不得视为已经通过的测试。

报告项目	最低记录内容	本版材料状态
目标与模型	电路图或可核对描述、器件模型来源和版本、参数、初值、供电与负载	未附具体验证电路
求解配置	积分器、容差、步长、过采样滤波、迭代和回退上限、数值精度	未附产品配置
参考与测量	参考求解器版本、实物及测量链路、电平标定、时间对齐方式	未附对照记录
音频误差	频响、波形、谐波或互调、混叠及动态响应；说明指标定义、单位和通过阈值	未附曲线和数值
运行性能	硬件、软件构建、采样率、缓冲大小、测试时长、平均和峰值耗时及异常计数	未附基准结果
复现与适用范围	测试输入或生成规则、控制序列、版本标识、已知失败条件和报告日期	待随验证报告提供

6.3 当前文档的结论范围

本版说明 ATTI 的建模路线与工程要求，未提供实现代码审计、产品功能清单或定量测试报告，因此不据本文声称特定平台的实时达标、与实物的误差水平，或相对其他方法的性能优势。公开验证可以保留内部代码与器件调校细节，但应提供足以评估结论的测试条件、指标和结果。

目标应用包括过载、失真、Fuzz、模拟压缩、滤波及相关反馈网络。实际可建模范围取决于所支持的器件和动态机制，例如光电压缩器需要适当的光电响应模型。纯数字延迟、卷积混响等可由对应算法模块实现，电路模型可作为整体信号链的一部分。

技术定位 — ATTI 以 电路结构与器件关系为建模依据，通过 瞬态耦合求解预测动态响应。向量拓扑服务于电路数据的组织与计算；工程价值由可核对的功能边界、误差结果和运行性能体现。

参考资料

[1] ngspice 项目组
Ngspice User's Manual，在线更新版；参见 Convergence 与 Transient Analysis Options 相关章节。访问日期：2026-09-07。产品验证应另行固定实际使用的软件及手册版本。
[2] David T. Yeh; Julius O. Smith
Simulating guitar distortion circuits using wave digital and nonlinear state-space formulations. DAFx, 2008.
[3] Jatin Chowdhury
A Comparison of Virtual Analog Modelling Techniques for Desktop and Embedded Implementations. arXiv:2009.02833, 2020.
[4] Yen-Tung Yeh; Wen-Yi Hsiao; Yi-Hsuan Yang
Hyper Recurrent Neural Network: Condition Mechanisms for Black-Box Audio Effect Modeling. DAFx, 2024.