# Avalanche Noise Oscilloscope — 100% Accurate Simulation Fix Guide

This document consolidates every required fix to ensure the oscilloscope displays authentic NGSpice transient waveforms that react to real circuit physics, not synthetic math. It includes parsing fixes, correct device model selection, accurate circuit topology for avalanche behavior, proper transient noise injection, and analysis settings. Apply these patches and checks to achieve reliable, reproducible results.

## Contents

1. Quick Fix Checklist

2. Fix #1 — Robust SI Value Parsing (Resistors/Capacitors)

3. Fix #2 — Correct Transistor Model Selection

4. Fix #3 — Q1 Topology & Physics (Avalanche Reality)

5. Fix #4 — Replace Hardcoded Sine with Transient Noise

6. Fix #5 — Ensure True Transient Analysis (.tran) & Step Control

7. Fix #6 — Include Sane Device Models (.model cards)

8. Fix #7 — Cache Safety & Netlist Hashing

9. Sanity Checks (OP + Transient)

10. Mapping to Reported Issues

## 1) Quick Fix Checklist

- Implement SI unit parsing so values like “15kΩ”, “1µF”, “100µF” are numeric.

- Fix transistor model selection (BC549 vs BC337).

- Adjust Q1 topology/physics to produce realistic breakdown behavior (zener-assisted).

- Replace hardcoded sine stimulus with NGSpice transient noise at the correct node.

- Preserve your .tran analysis and set step sizes appropriately.

- Add realistic device .model cards for BJTs and zeners.

- Ensure cache is keyed to the netlist/graph to prevent stale waveforms.

## 2) Fix #1 — Robust SI Value Parsing (Resistors/Capacitors)

File: server/utils/units.ts (new)

export function parseSI(raw?: string): number {  
 if (!raw) return NaN;  
 const s0 = String(raw).trim();  
 const s = s0  
 .replace(/[, ]+/g, "")  
 .replace(/[ΩΩOhm|ohm|F|H|V|A]$/i, "")  
 .replace(/Ω|Ω|ohms?/gi, "");  
 const m = s.match(/^([-+]?\d\*\.?\d+(?:e[-+]?\d+)?)([a-zA-Zµμ]\*)$/);  
 if (!m) return NaN;  
 const num = parseFloat(m[1]);  
 const sufRaw = m[2] || "";  
 const suf = sufRaw;  
 const l = suf.toLowerCase();  
  
 const map: Record<string, number> = {  
 "": 1, "k": 1e3, "K": 1e3,  
 "M": 1e6, "meg": 1e6, "MEG": 1e6,  
 "g": 1e9, "G": 1e9,  
 "m": 1e-3,  
 "u": 1e-6, "µ": 1e-6, "μ": 1e-6,  
 "n": 1e-9, "p": 1e-12, "f": 1e-15  
 };  
 const factor =  
 suf in map ? map[suf] :  
 l in map ? map[l] :  
 (l.startsWith("k") ? 1e3 :  
 l.startsWith("meg") ? 1e6 :  
 l.startsWith("m") ? 1e-3 :  
 l.startsWith("u") || suf.includes("µ") || suf.includes("μ") ? 1e-6 :  
 l.startsWith("n") ? 1e-9 :  
 l.startsWith("p") ? 1e-12 :  
 1);  
  
 return num \* factor;  
}  
  
export const parseOhms = parseSI;  
export const parseFarads = parseSI;

Update server/utils/simple-spice.ts:

// Old  
// const r7 = parseFloat(component.value || "1000");  
// const c1 = parseFloat(component.value || "1e-6");  
  
// New  
import { parseOhms, parseFarads } from "../utils/units";  
const r7 = parseOhms(component.value);  
const c1 = parseFarads(component.value);  
// Repeat for R8, R9, C2, C4, etc.

## 3) Fix #2 — Correct Transistor Model Selection

File: server/utils/simple-spice.ts (around previous line ~150)

function pickBjtModel(component: { model?: string, ref?: string }) {  
 const m = (component.model || "").toUpperCase();  
 if (m.includes("BC549")) return "BC549";  
 if (m.includes("BC337")) return "BC337";  
 if ((component.ref || "").toUpperCase() === "Q4") return "BC549";  
 return "BC337";  
}  
  
const model = pickBjtModel(component);

## 4) Fix #3 — Q1 Topology & Physics (Avalanche Reality)

The original setup reverse-biases the B–E junction (C→GND, B→GND, E via R7 to +V). BE breakdown typically occurs ≈6–8 V, so with 20 V, Q1E will clamp ~6–8 V, not near +V. At 5 V supply you won’t reach avalanche; use a zener assist or raise supply.

Recommended modification (zener clamp for realistic breakdown & noise headroom):

\* Zener-assisted breakdown at emitter node  
R7 +V Q1E 15k  
Q1 0 0 Q1E BC337 ; C B E  
  
\* Emulate BE breakdown via diode model (better supported in SPICE)  
DNOISE 0 Q1E DZ6V8 ; use DZ4V7 if running from 5 V

## 5) Fix #4 — Replace Hardcoded Sine with Transient Noise

File: server/utils/simple-spice.ts (remove SIN at node 2)

\* Remove:  
\* VNOISE 2 0 SIN(0 0.001 1MEG 0 0)  
  
\* Add transient noise at the correct node (Q1E by name)  
VNOISE Q1E 0 TRNOISE(NA=1mV NT=1e-7 SEED=1)

Ensure your code maps the net name (e.g., "Q1E") to the right node label before emission.

## 6) Fix #5 — Ensure True Transient Analysis (.tran) & Step Control

.control  
set filetype=ascii  
tran 0.1us 50ms uic  
wrdata tran.csv time v(Q1E) v(NOISE) v(SCOPE\_OUT)  
.endc

Choose a timestep ≤ NT in TRNOISE (here NT=100 ns, step=0.1 µs). Do not strip or override this .tran block in your NGSpice interface.

## 7) Fix #6 — Include Sane Device Models (.model cards)

\* --- BJT models ---  
.model BC337 NPN (IS=1e-14 BF=200 VAF=100 NF=1 RB=100 RE=1 RC=1  
+ CJE=5p VJE=0.7 CJC=3p VJC=0.3 TF=0.3n TR=10n KF=2e-14 AF=1)  
  
.model BC549 NPN (IS=1e-14 BF=300 VAF=150 NF=1 RB=150 RE=1 RC=1  
+ CJE=6p VJE=0.7 CJC=3p VJC=0.3 TF=0.25n TR=8n KF=5e-15 AF=1)  
  
\* --- Zener models ---  
.model DZ6V8 D (IS=1e-14 N=1 BV=6.8 IBV=10u RS=10 KF=0 AF=1)  
.model DZ4V7 D (IS=1e-14 N=1 BV=4.7 IBV=10u RS=10 KF=0 AF=1)

## 8) Fix #7 — Cache Safety & Netlist Hashing

// client/src/utils/circuit-simulation.ts  
import { createHash } from "crypto";  
  
function simKey(components: any[], analysis?: object) {  
 const payload = JSON.stringify({ components, analysis });  
 try {  
 return createHash("sha1").update(payload).digest("hex");  
 } catch {  
 return payload; // fallback in browser  
 }  
}  
  
const key = simKey(components, analysisOptions);  
if (cache?.key === key && cache.success) {  
 return convertNgSpiceToSimulationResults(components, cache);  
}  
const fresh = await updateNgSpiceSimulation(components, analysisOptions);  
cache = { ...fresh, key };  
return convertNgSpiceToSimulationResults(components, fresh);

## 9) Sanity Checks (Operating Point + Transient)

- With +20 V and DZ6V8: V(Q1E) ≈ 6–8 V; I(R7) ≈ (20–~7)/15k ≈ 0.86 mA.

- With +5 V and DZ4V7: V(Q1E) ≈ 4.3–4.8 V; lower headroom → lower noise amplitude.

- If BE-only (no zener) at 5 V: no avalanche—expect minimal noise (physics).

## 10) Mapping to Your Reported Issues

- Resistor Value Parsing Error → Implement parseSI/parseOhms (Section 2).

- Wrong Transistor Model Selection → Use pickBjtModel + .model cards (Sections 3 & 7).

- Capacitor Value Parsing Error → Use parseFarads (Section 2).

- Q1 Circuit Topology Issue → Use zener-assisted breakdown; correct expectations (Section 4).

- Hardcoded Noise Source → Replace with TRNOISE at Q1E (Section 5).

- Missing Avalanche Physics → Covered by zener model + transient noise + proper .tran (Sections 4–6).

After applying these fixes, the oscilloscope should display time-varying noise that changes with component values and topology, reflecting authentic circuit behavior under NGSpice.