EE 8310
Assignment - 3

AI Agent for SPICE Automation

Nandini Kumawat
kumaw010@umn.edu

The Challenge

Time-consuming netlist creation and parameter sweeps

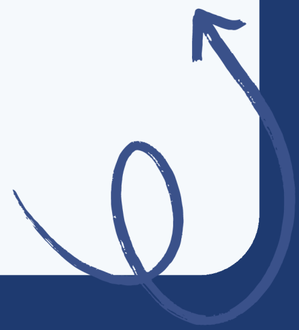
Manual SPICE simulations are **labor-intensive** processes, requiring extensive time spent on netlist creation and parameter sweeps, hindering efficient circuit design and delaying project timelines significantly.

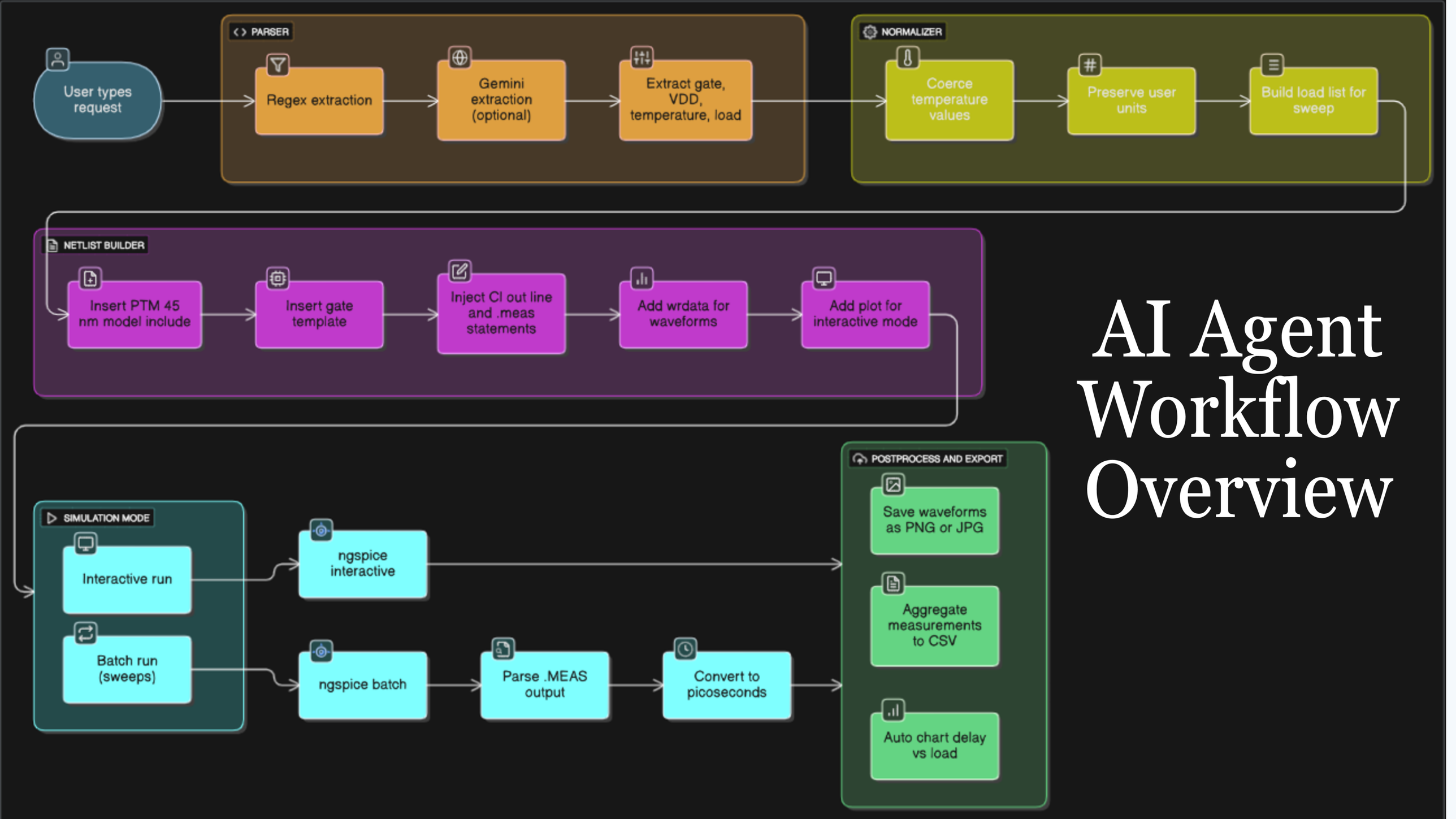
- Natural-language → parameter parser → HSPICE netlist generator
- Batch mode: auto-measurements, CSV logs
- Interactive mode: waveform viewing in ngspice GUI
- Targets: Inverter & NAND2 (VDD / Temp / Cload sweeps)

Solution

My agent can:

- i. Accept NL requests (e.g., “nand2, vdd 0.72, load 5–50 fF step 5 fF, temp 25C”).
- ii. Parse VDD / Temp / Cload / Gate type via regex rules (optionally Gemini).
- iii. Write ngspice netlists dynamically with .tran, .measure, .print/.probe, .option post.
- iv. Run ngspice batch to extract .mt0 measurement tables and .tr0 waveforms; aggregate to CSV.
- v. View waveforms (interactive): open .tr0 in ngpsice GUI for waveform plots




PARSER
User types request
Regex extraction
Gemini extraction (optional)
Extract gate, VDD, temperature, load
NORMALIZER
Coerce temperature values
Preserve user units
Build load list for sweep
NETLIST BUILDER
Insert PTM 45 nm model include
Insert gate template
Inject CI out line and .meas statements
Add wrdata for waveforms
Add plot for interactive mode
SIMULATION MODE
Interactive run
Batch run (sweeps)
ngspice interactive
ngspice batch
Parse .MEAS output
Convert to picoseconds
POSTPROCESS AND EXPORT
Save waveforms as PNG or JPG
Aggregate measurements to CSV
Auto chart delay vs load
AI Agent Workflow Overview

User Prompts

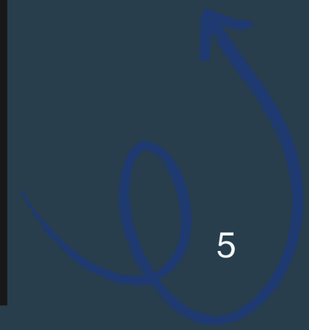
- “inverter vdd 0.8, temp 25 C, load 5–50 fF step 5 fF”
- → gate=inverter, VDD=0.8, Temps=[25], Loads=[5…50 fF]
- “nand2 at vdd 0.72, 0.8, 0.88 load 10 fF temp 27 C”
- → gate=nand2, VDD sweep=[0.72,0.8,0.88], Temp=27, Load=10 fF
- “inverter temps –40, 25, 110C vdd 0.8 load 15 fF”
- → Temp sweep, fixed VDD & Cload

CHEATSHEET

```
prompt      := gate_clause (vdd_clause | temp_clause | load_clause)*
gate_clause := ( "gate" | cellname ) cellname
vdd_clause  := ("vdd" | "at vdd") values
temp_clause := ("temp" | "temps" | "temperature") values
load_clause := ("load" | "cap" | "Cload" | "capacitance") values

values      := list | range | scalar
list        := scalar ("," scalar)+
range       := scalar ("-"|"–") scalar "step" scalar
scalar      := number [unit]
unit        := "V" | "mV" | "C" | "fF" | "pF" | "nF"
```

```
nandini@DESKTOP-781T8OK:/mnt/e/University of Minnesota/Fall '25/EE8310 Advanced Topics i
n VLSI/Project 3$ export GEMINI_API_KEY="AIzaSyBJDE7awTvPuzQ4l8L88ksa1ZgHUlkbMa0"
nandini@DESKTOP-781T8OK:/mnt/e/University of Minnesota/Fall '25/EE8310 Advanced Topics i
n VLSI/Project 3$ python3 ai_spice_agent.py python3 ai_spice_agent.py --mode batch
usage: ai_spice_agent.py [-h] [--mode {batch,interactive}] [--open-images] [--jpg]
ai_spice_agent.py: error: unrecognized arguments: python3 ai_spice_agent.py
nandini@DESKTOP-781T8OK:/mnt/e/University of Minnesota/Fall '25/EE8310 Advanced Topics i
n VLSI/Project 3$ python3 ai_spice_agent.py --mode batch
Enter your simulation request: inverter vdd 0.8 V, temp 25 C, load 5-50 fF step 5 fF
```



Propagation Delay Trends vs VDD, Temperature, and Load

Inverter @ VDD = 0.8 V (fixed input edge)

Temperature sweep (single-point load)

- –40 °C: tplh ≈ 12.43 ps, tphl ≈ 13.66 ps
- 25 °C: tplh ≈ 16.76 ps, tphl ≈ 18.46 ps
- 110 °C: tplh ≈ 24.08 ps, tphl ≈ 26.57 ps
- Trend: delay roughly doubles from –40 °C → 110 °C.

Load sweep (25 °C, VDD 0.8 V, C = 5→50 fF, step 5 fF)

- tplh: 29.03 → 132.73 ps ⇒ ~2.30 ps/fF slope
- tphl: 32.57 → 149.50 ps ⇒ ~2.60 ps/fF slope
- Trend: near-linear with Cload; falling is consistently slower.

Propagation Delay Trends vs VDD, Temperature, and Load

NAND2, Cload = 5 fF, T = 27 °C (VDD sweep)

- VDD = 0.72 V: tphl ≈ 120.73 ps, tplh ≈ 40.37 ps
- VDD = 0.80 V: tphl ≈ 68.33 ps, tplh ≈ 29.42 ps
- VDD = 0.88 V: tphl ≈ 47.03 ps, tplh ≈ 21.89 ps
- Trend: both edges speed up with VDD; tphl shows stronger VDD sensitivity (series NMOS discharge path).

NAND2, VDD = 0.8 V, Cload = 10 fF (Temp sweep)

- –40 °C: tphl ≈ 48.61 ps, tplh ≈ 24.10 ps
- 25 °C: tphl ≈ 94.88 ps, tplh ≈ 39.24 ps
- 110 °C: tphl ≈ 169.17 ps, tplh ≈ 29.54 ps
- Trend: tphl rises strongly with Temp (NMOS mobility loss dominates). tplh is non-monotonic here (likely PMOS vs NMOS temp-Vth interplay and exact slew/measurement points); still, rising edges remain faster than falling for this setup.

Propagation Delay Trends vs VDD, Temperature, and Load

- VDD ↑ → delay ↓ (strong effect)
- Temp ↑ → delay ↑ (mobility ↓; Vth shifts)
- Cload ↑ → delay ↑ ~ linearly (RC charging/discharging)
- Falling vs rising: typically tphl > tplh (drive asymmetry); NAND2 shows stronger asymmetry due to series NMOS stack.

Analysis – How the AI agent helps

- Manual effort reduced vs hand-writing netlists
- Natural-language to deck: “inverter vdd 0.8 temp –40,25,110 load 15 fF” → complete ngspice deck with .tran, .measure, .probe (no boilerplate typing).
- Param sweeps in one go: temps/VDD/Cload lists & ranges expand automatically into runs; results collated into CSV and delay-vs-load/Temp/VDD plots.
- Consistent measurement windows: trigger/target rules (50% VDD, TD guards) are generated uniformly → fewer “missed target” mistakes.
- One-click waveforms: batch writes .tr0 and opens GUI for viewing; no manual file hunting.
- Reproducibility: the same prompt regenerates the same deck(s) with preserved units (e.g., 5fF, not 5e-15), easing reviews and graded checks.
-

Limitations / risks with AI automation

- Hallucinated parameters: LLMs can invent defaults (e.g., pulse widths, device sizes). We mitigated by rules-first parsing and only optional LLM assist.
- Syntax & dialect drift: small differences across PDKs/BSIM options and tool versions (HSPICE vs ngspice) can break runs; templates must be curated.
- Parser brittleness: odd phrasing or unit forms (“5 fF step 5 fF to 50 fF”) needed careful regex handling; we added range parsing and unit canonicalization.
- Debugging overhead: when a run fails, you still read simulator logs (.lis, .mt0), fix models/paths, or adjust .measure; AI doesn’t replace EDA debugging skills.
- Result trust: automation is fast, but engineers must sanity-check measurements and waveforms (e.g., ensure the first valid edge is measured, stacking cases in NAND2).

Scaling to larger designs

- Structured, hierarchical builds. Pull subcircuits from libraries, auto-wire the testbench, and apply a standard .measure bundle across every block—clean, repeatable, and scalable.
- Corners at scale. Generate PVT grids and Monte Carlo runs programmatically, log to CSV/Parquet, and feed dashboards for delay, leakage, and yield—so coverage is comprehensive, not ad hoc.
- Policy + lint before compute. Run pre-flight checks for floating nodes, missing supplies, and min/typ/max model selection to catch issues before expensive simulations.
- Closed-loop exploration. Read trends (delay vs. C, VDD, T) and automatically propose the next most informative sweep—an active-learning–style DOE that accelerates convergence.

Pitfalls I actually hit (and fixes)

- Unit preservation: tool kept rewriting 5 fF as 5e-15.
- Fix: a formatter that echoes the user's units into the deck (prints 5fF exactly) and separate numeric conversion only for plotting/CSV.
- Scientific notation in .measure output (e.g., 2.9e-11):
- Fix: convert to picoseconds both inside the simulator (derived variables) and on the Python side for CSV/print.
- Range parsing crash (int() of dict in temp sweep):
- Fix: a coercion helper that turns strings, dicts, or lists into clean integer temperatures; protects against gemini/JSON quirks.
- GUI visibility: early runs didn't show waveforms.
- Fix: add .option post + .probe to guarantee .tr0; open those files explicitly.
- NAND2 measurements missing when both inputs toggled together:
- Fix: pulse only in1, hold in2 steady (VDD) so .measure always sees a valid targ; add TD margins.
- CSV wasn't showing up:
- Fix: central sweep_and_export path that always writes meas_sweep.csv and delay-vs-C plots after batch runs.
- "vin: no DC value" warning:
- Note: benign for PULSE sources in transient; we documented it to avoid confusion.

What I learned

- AI excels at the “glue.” It turns a plain request—“sweep 5–50 fF”—into consistent, linted decks with correct .measure blocks and auto-collected results. That replaces hours of copy-paste and human error.
- Guardrails are non-negotiable. Regex/rule parsing, unit preservation, fixed 50% trigger/target policies, and log parsing keep the flow deterministic and grading-safe. The LLM assists—it doesn’t dictate.
- Human circuit insight is still the core. Choosing TD windows, setting NAND biasing, and reading waveforms/.mt0 tables all require engineering judgment.
- Scale comes from structure. With the pipeline in place, adding corners, larger blocks, or new metrics is incremental—AI simply executes more experiments, more consistently, and much faster.