An Automated Synthesis Tool for Generating Noise-Immune Sub-Threshold Circuits

Amrita Mazumdar

Faculty Mentor: Professor Iris Bahar

Distributed Research Experience for Undergraduates (DREU)

Abstract

- Nanoscale circuits operating at sub-threshold voltages are increasingly susceptible to the impact of random telegraph signal (RTS) and thermal noise, resulting in **soft errors** that compromise a circuit's reliability.
- This work presents a **low-power, area-efficient error correction technique** and **an automated tool** to synthesize noise-immune circuits.
- The tool uses **two novel techniques** to selectively apply reinforcement using invariant relationships **to correct noise-induced signal errors**.
- Simulations demonstrate our synthesized circuits provide **better noise immunity than standard CMOS technology** in tests with limited area and power overhead.

Problem Statement

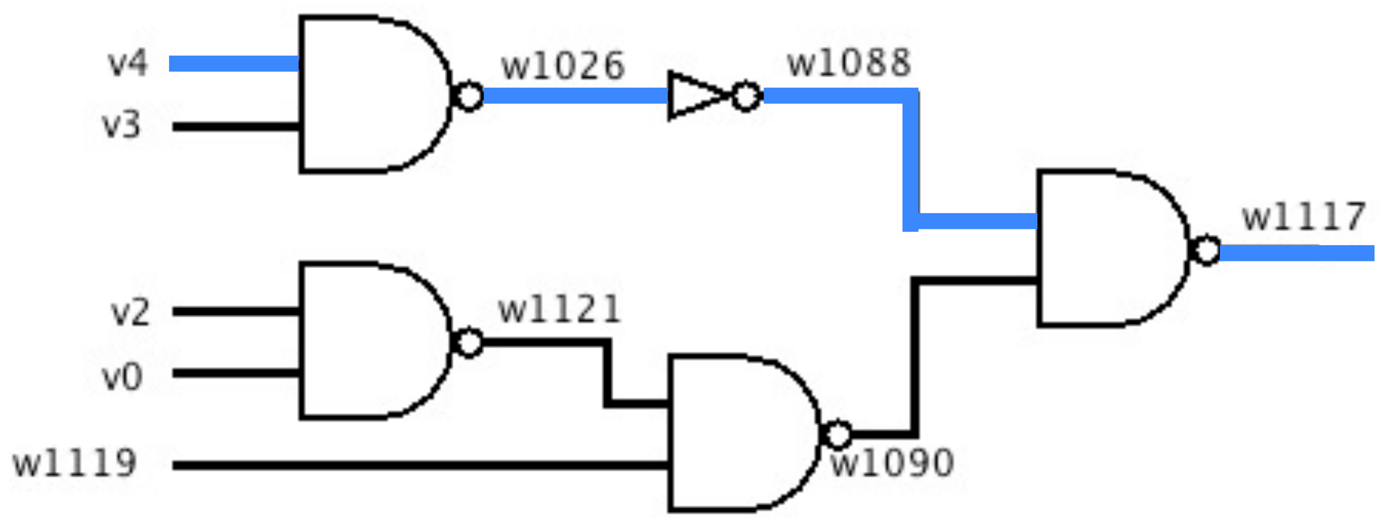
- How can we utilize Schmitt trigger logic and invariant relationships in a circuit to **increase a circuit's noise tolerance** and **reduce soft errors**?
- Can we **automate this process** to generate noise-immune, low-power circuits?

Background

- As CMOS technology shrinks in accordance with Moore's Law, nanoscale circuits are required to functionally operate at sub-threshold voltages.
- At such low power, circuits become much more susceptible to the impact of random telegraph signal (RTS) and thermal noise, and produce soft errors that compromise a circuit's reliability.
- Techniques to combat the effect of soft errors must also be low power and area-efficient, so as to not exceed the constraints of nanoscale circuit design.

Logical Implications

- **Logicial implication - invariant relationship** between two nodes in a circuit



Logical Implications		
Implicant		Implicand
w1121 (0)	→	w1090 (1)
w1090 (0)	→	w1119 (1)
v4 (0)	**→**	**w1117 (1)**
w1026 (1)	→	w1117 (1)
w1090 (0)	→	w1117 (1)

- Large circuits typically have many implications existing between various nodes.
- If a circuit violates an implication, an error must have occurred either at the second node or the logic in between the two. We can thus **use implications to easily detect and correct errors** within the circuit.
- Prior work demonstrates these strong error detection capabilities of implications.

Schmitt Trigger Gates

- Schmitt trigger gates use additional **transistors to reinforce the output** of a gate.

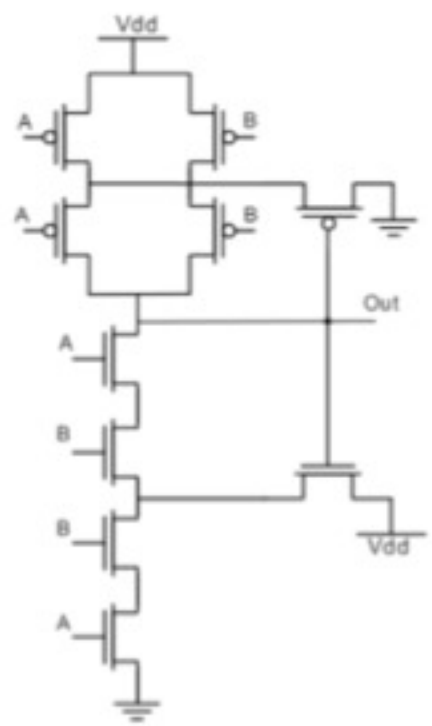


Figure 1:
Original Schmitt
trigger gate
(10 transistors)

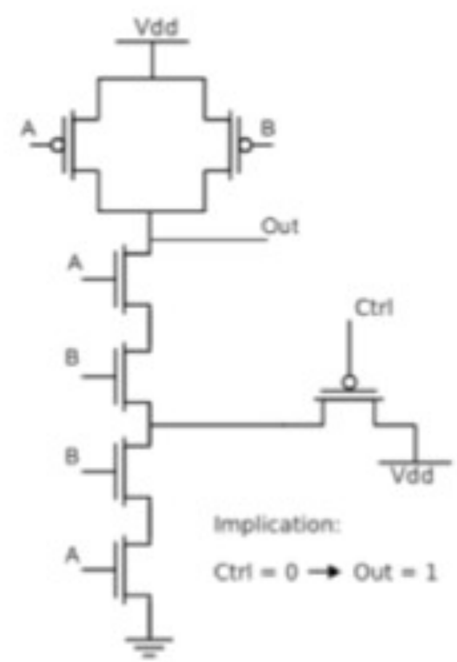


Figure 2:
Modified Schmitt
trigger gate for
implication
reinforcement
(7 transistors)

- Schmitt trigger gates have **higher noise margins**, but come with increased power and area overhead.

- We use a **modified Schmitt trigger gate** that reinforces a node according to a given invariant relationship, rather than feedback from the gate's inputs.

Methodology

Step 0: Test Circuits and Simulation Tools

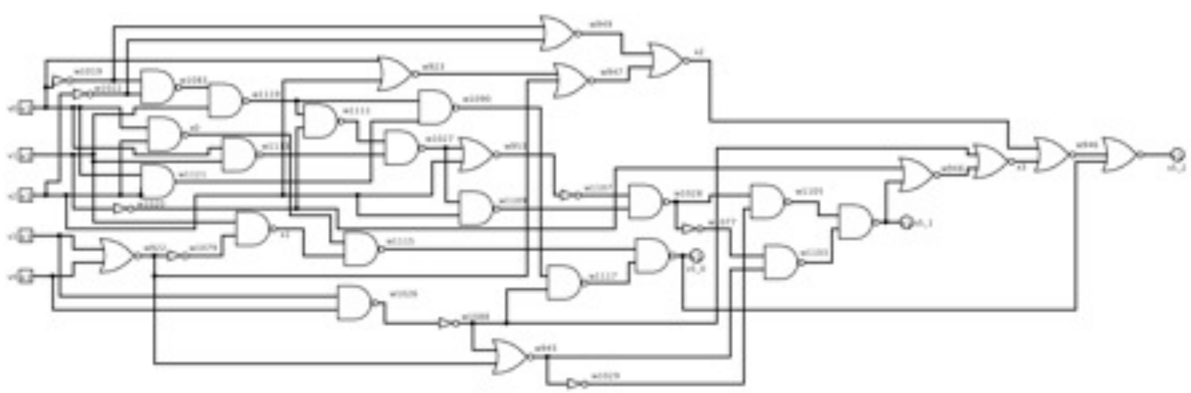


Figure 3:
RD53, one test circuit from MCNC

- Test circuits were taken from the MCNC benchmark set and simulated using 22nm FDSOI transistors.
- Mentor Graphics FastScan was used for logical, ATPG, and fault simulation.
- RTS and thermal noise were generated for circuit simulations using MATLAB.
- Circuit simulations were conducted using SPICE on Brown University's large-scale compute cluster, Oscar.

Methodology

Step 1: Generating Implications

- Prior work developed a **workflow to generate implications** from a circuit's Verilog netlist.
- A logical simulation is done to **generate outputs** for many of a circuit's input patterns.
- The output vectors are then parsed for **possible invariant relationships** between nodes in the circuit.
- We use the zChaff SAT solver to **validate these possible implications.**
- The **result is a list of implications** to be parsed and evaluated to best reinforce the circuit.

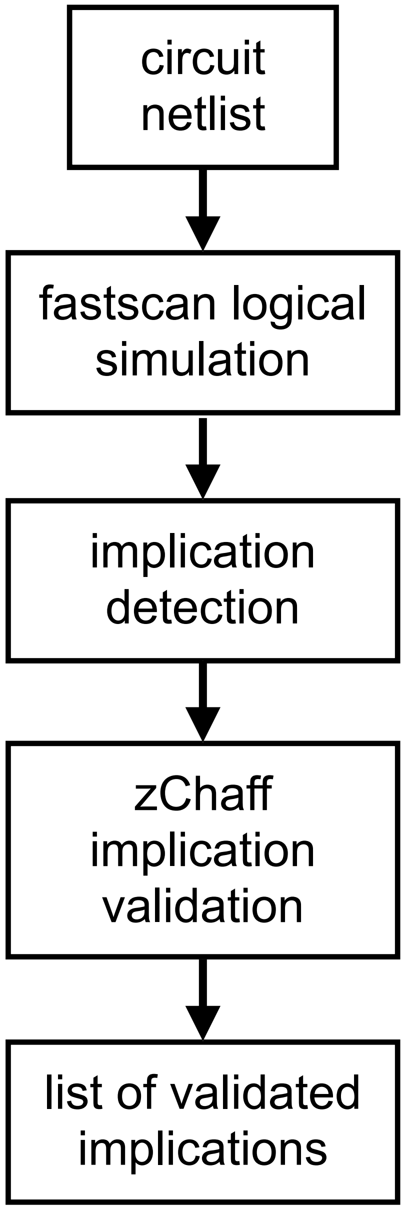


Figure 3: Implication Generation Workflow

Methodology

Step 2a: Building Implication Chains

- One technique evaluated was building **chains of implications** from an input through to an output.
- Implications are placed to reinforce nodes in a "chain" to reduce chances of failure before an output.
- **Chains are ranked** based on the probability of their implications being activated and the distance between implicand and implicant.
- Top ranking chains are output and then simulated to demonstrate noise suppresion potential.

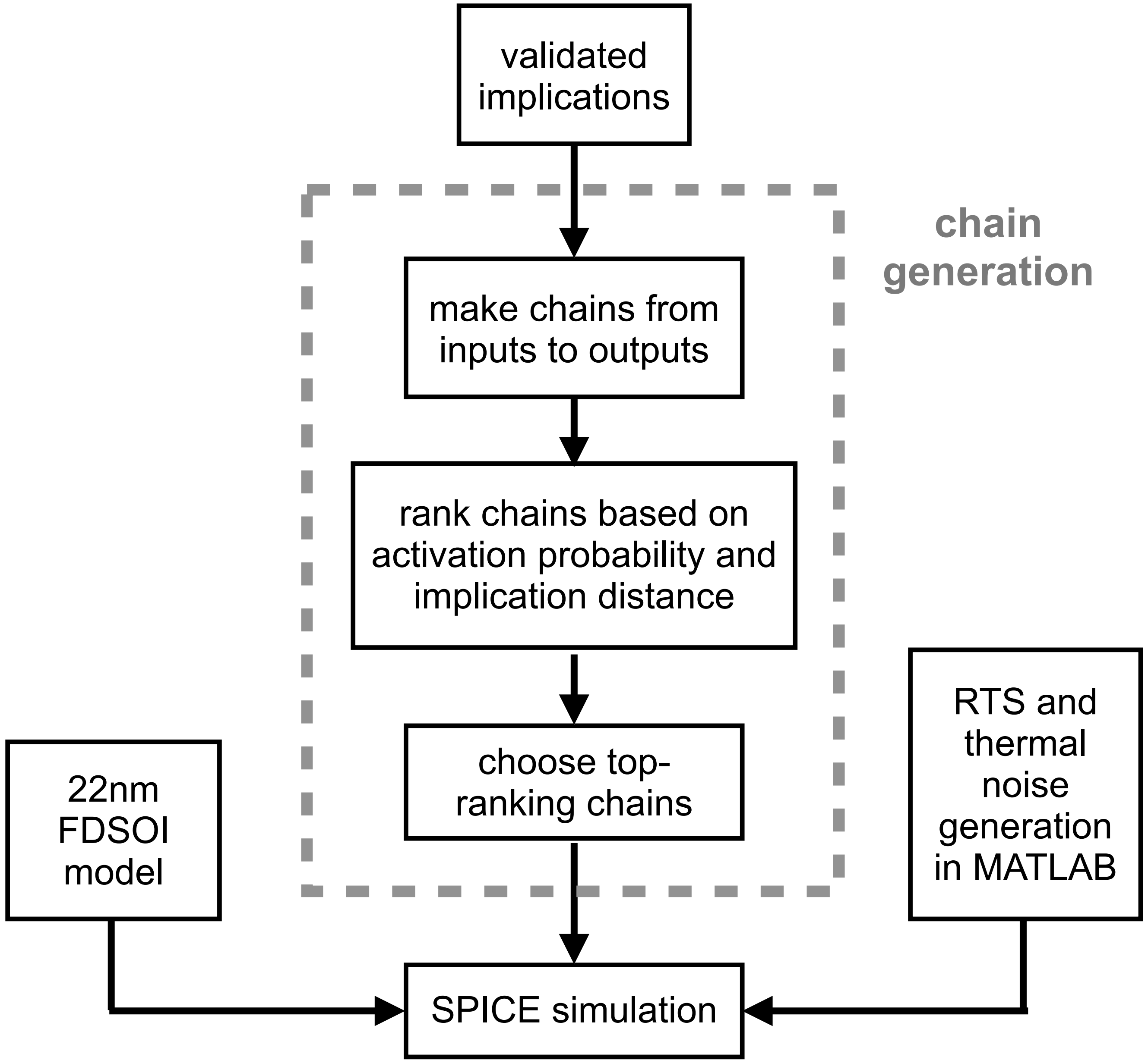


Figure 4: Chain Building Workflow

Methodology

Step 2b: High-Fault Node Reinforcement

- Another strategy implemented was to find **nodes with a high probability of failure** and reinforce those specifically.
- Automatic test pattern generation (ATPG) and fault simulation are conducted on each output to **determine the most failure-prone nodes** in an output's fan-in cone.
- The logical simulation from Step 1 is used to **find steady, or low-fault, nodes** to use as implicants.
- These lists of high-fault implicants and low-fault implicands are ranked together to **choose the optimal supporting implications** for an output.

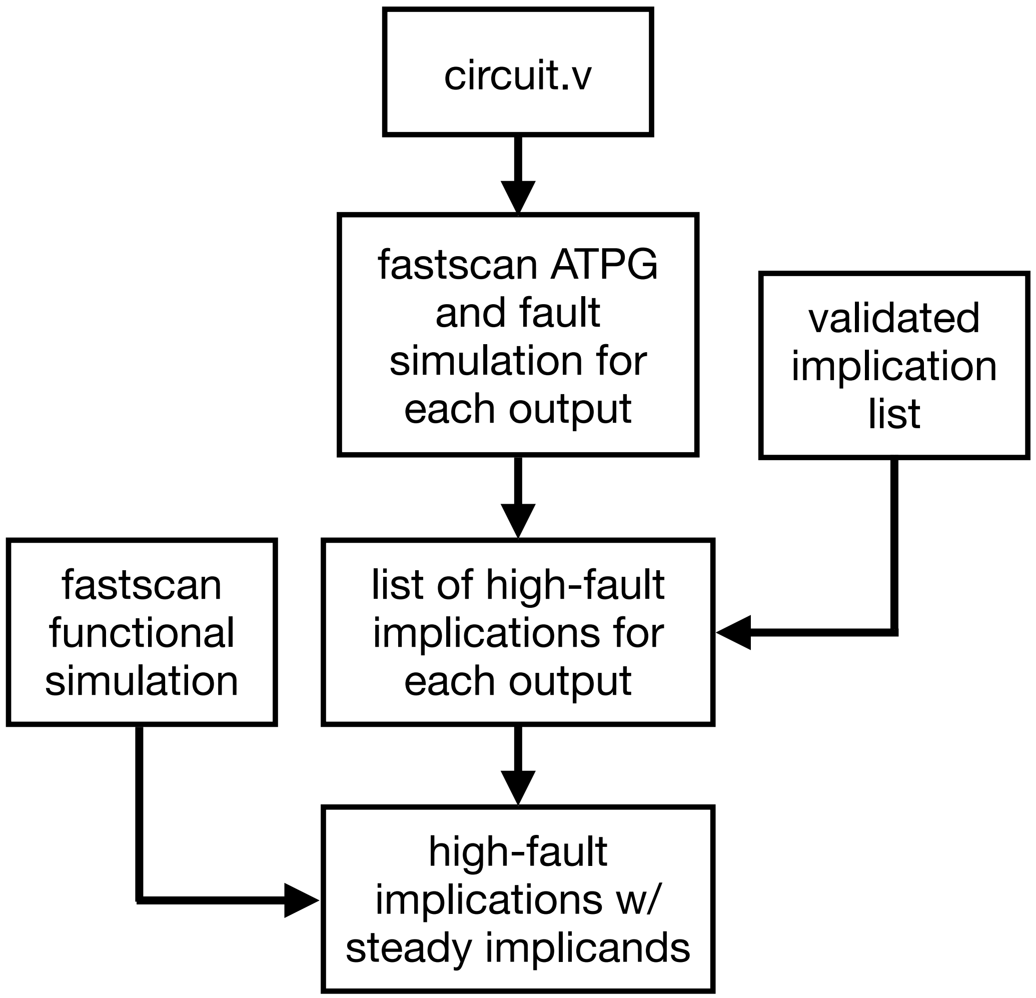
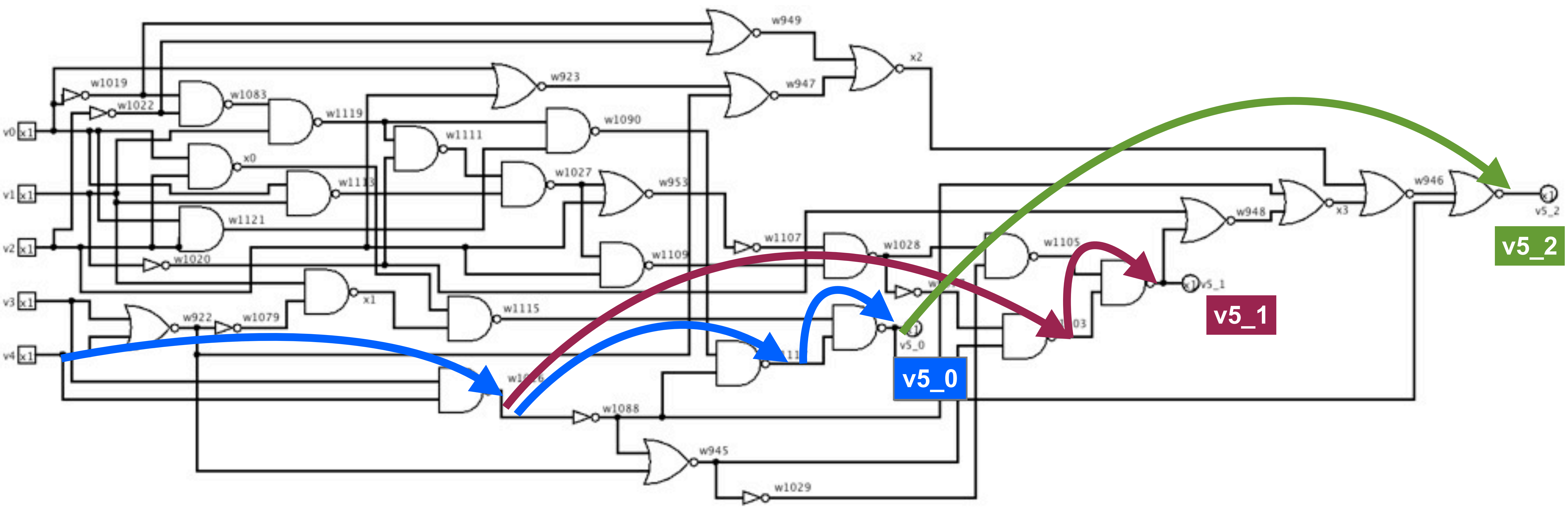


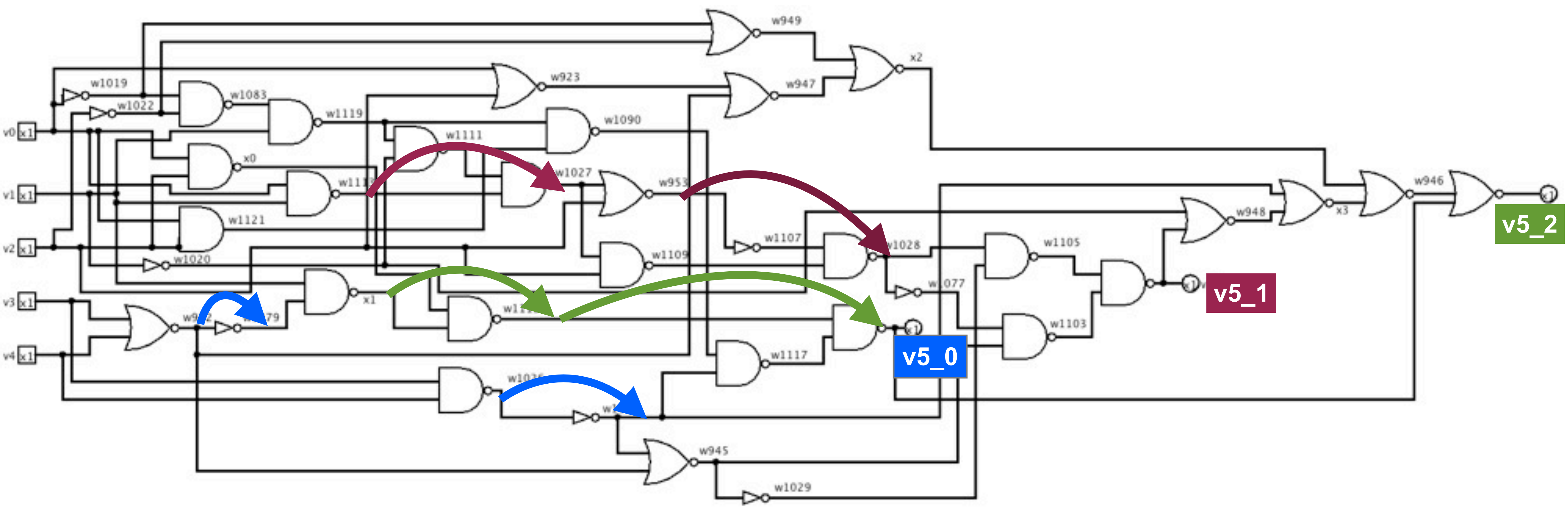
Figure 4: Chain Building Workflow

RD53 – Implication Chains



- Schmitt trigger gates have **higher noise margins**, but come with increased power and area overhead.
- We use a **modified** Schmitt trigger gate that reinforces a node according to a given invariant relationship, rather than feedback from the gate's inputs.

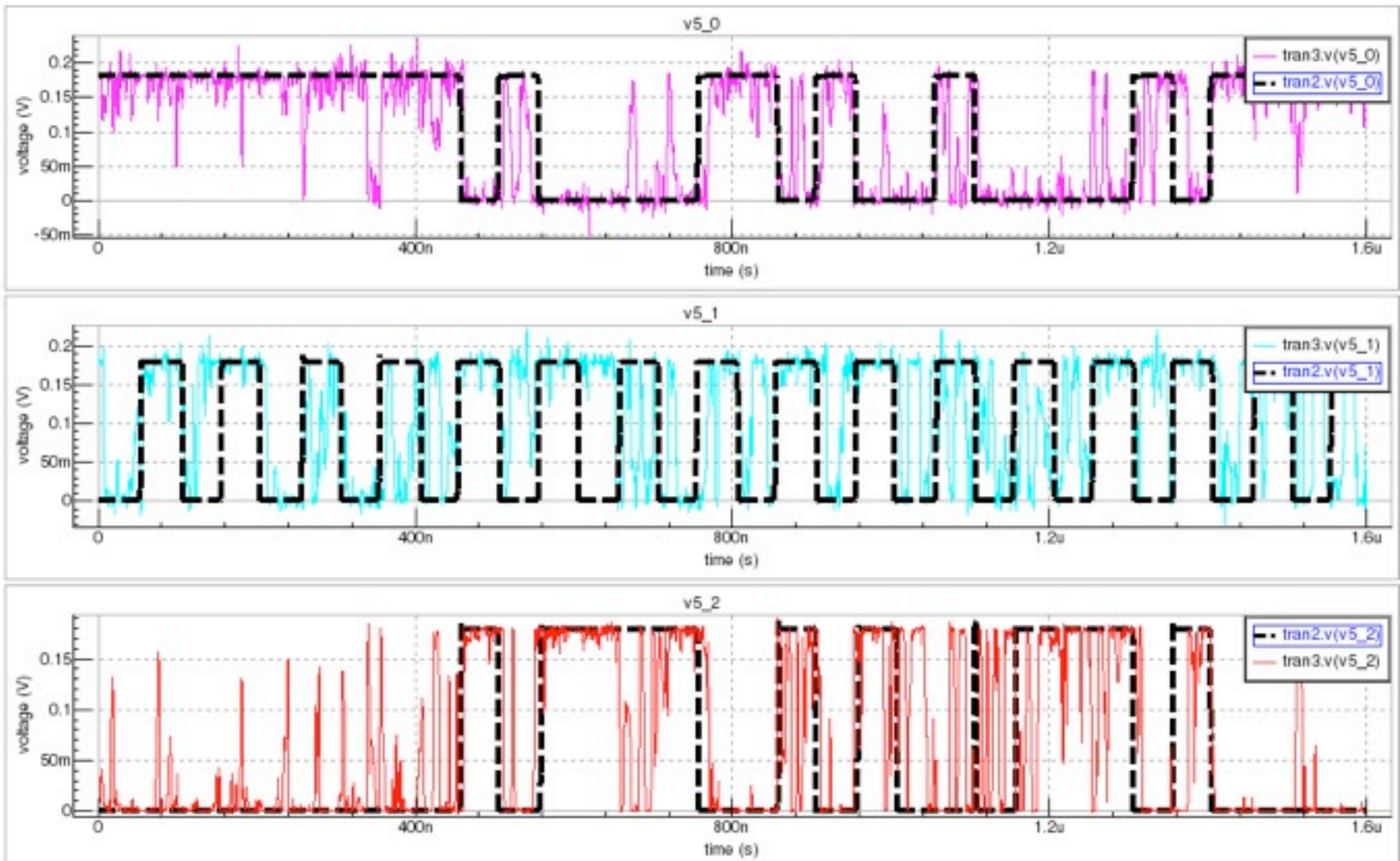
RD53 – Fault Reinforcement

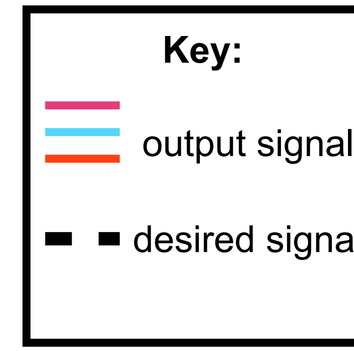


- Fault reinforcement trials were run weighing various combinations of an implication’s activation probability, implicant steadiness, and failure probability of the implicand.
- Simulation results demonstrated that considering only implicant steadiness and high-fault implicands gave the best noise tolerance. Incidentally, these scenarios produced the most “chain-like” implication sets.

Simulation Results

Reference Results - No Implications

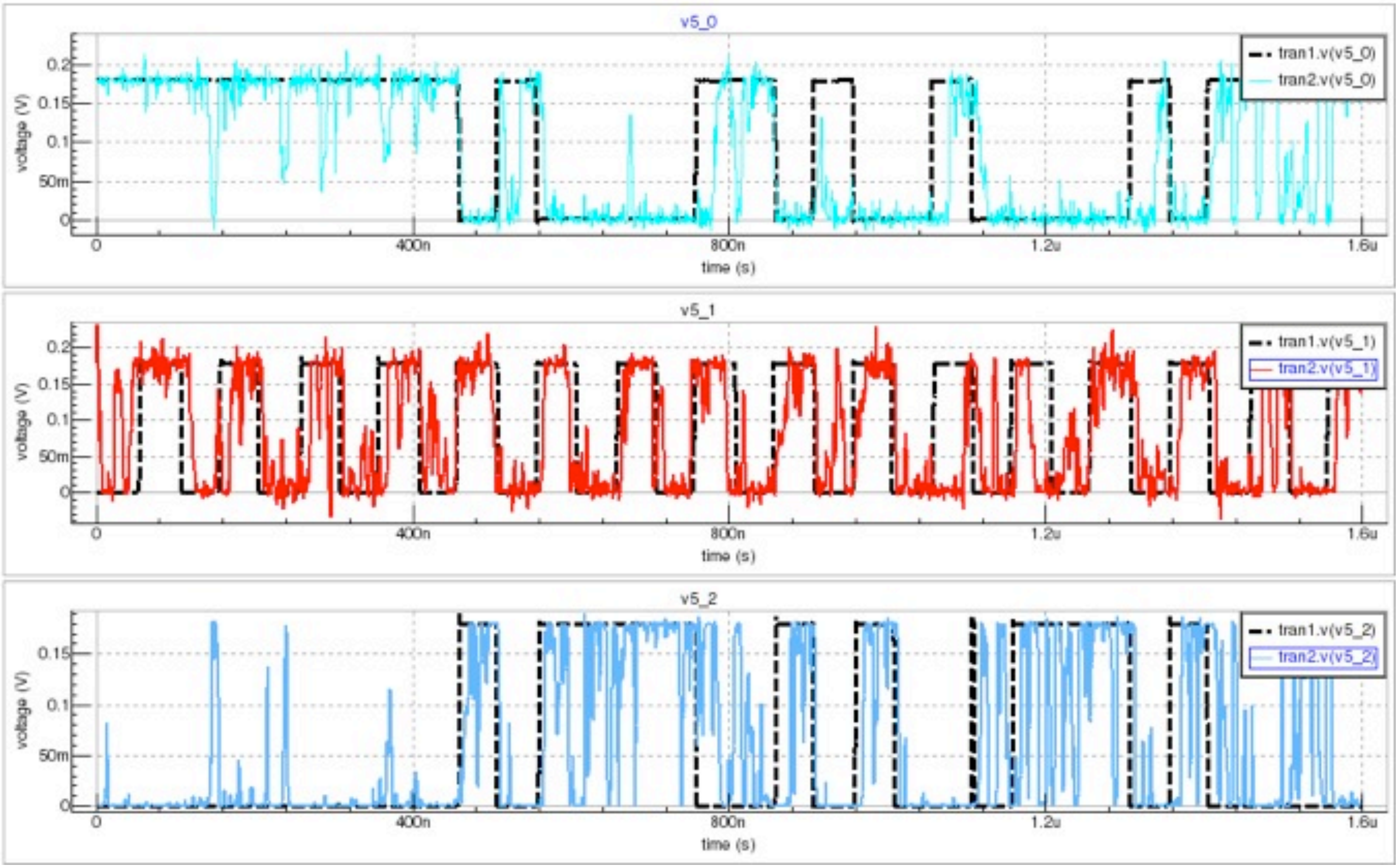


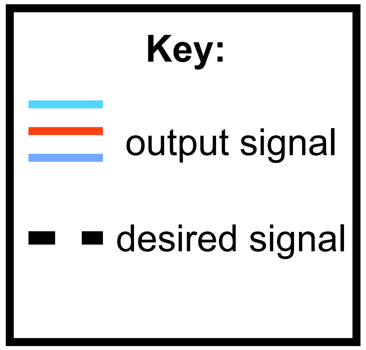


- 80mV of thermal noise and 50mV RTS noise were injected into the original signals.
- The output of the circuit was latched to generate a typical output vector for the noise-injected circuit.
- Errors can be observed at the beginning of v5_0, from the middle on in v5_1, and from the middle on in v5_2.

Simulation Results

High-fault Node Reinforcement Results

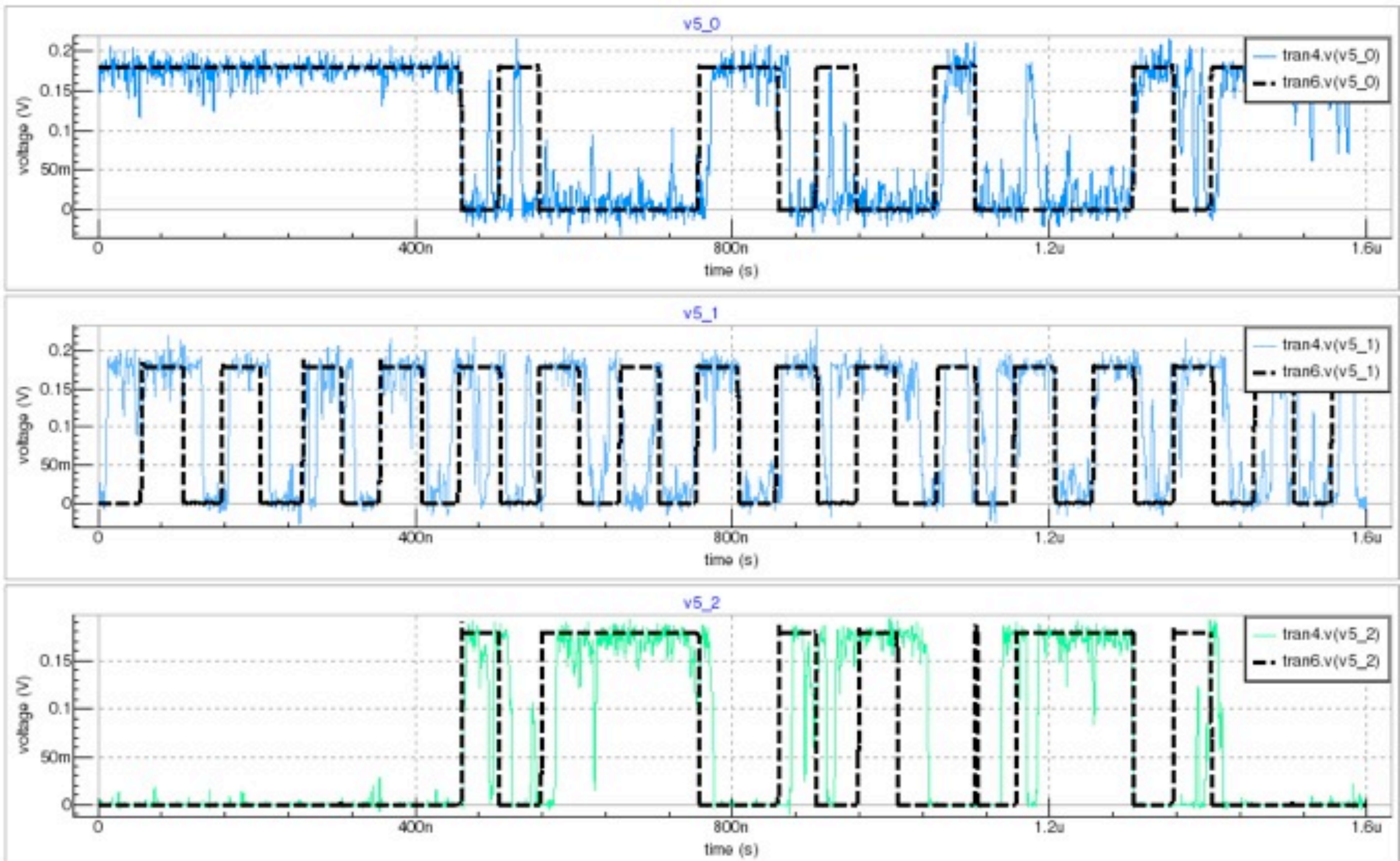


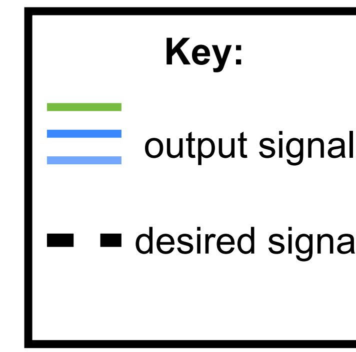


- Errors have been significantly mitigated in v5_1, at the beginning of v5_2, and during the middle of v5_0.
- General noise has been significantly reduced in v5_0 and v5_1, but remains for the most part in v5_2.

Simulation Results

Implication Chain Results





- Errors are suppressed in all output waveforms, with slight noise remaining in more error-prone sections.
- General noise has also been significantly reduced, notably in the beginning section of v5_0 and v5_2.

Conclusions

- Our results demonstrate that **we can successfully use Schmitt trigger logic and implications to increase a circuit's noise immunity**.
- Our tool currently implements **two strategies for selecting implication sets**, chain building and high-fault node reinforcement.
- Simulation results demonstrate **chain building to be a more effective and efficient technique than high-fault node reinforcement** for noise reduction and error mitigation.

Future Work

- We hope to use our observations from the high-fault implication trials to improve the chain-selection algorithm for improved error correction and noise suppression.
- While simulations have been conducted on a few additional MCNC benchmarks, we intend to extend our testing to larger circuits to examine error mitigation in larger-scale circuit design.