A Fast Approximation Technique for Power Grid Analysis

Mysore Sriram

Intel Technologies India Pvt. Ltd.

ASP-DAC 2011

Outline

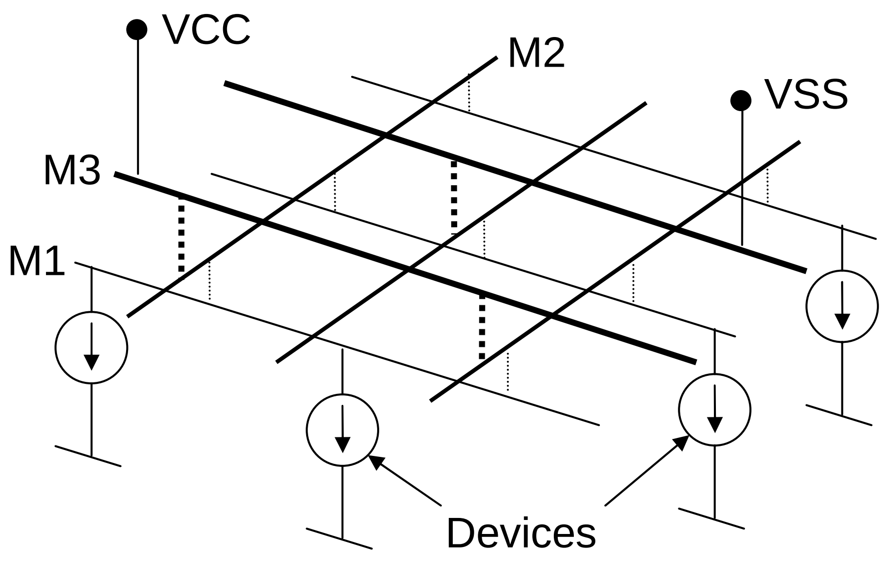
- Introduction
- Row-based iterative (RBI) solvers
- Heuristic analysis of RBI convergence behavior
- Fast approximate solver (FAS) algorithm
- Results
- Applications
- Summary

Introduction

- On-die power delivery is a growing challenge
  - Lower supply voltages make IR drop a big problem
  - Higher device density can cause electromigration issues
- Over-design is expensive
  - Direct impact to die area due to wire-limited designs
- Early verification of power delivery is critical
  - Late discovery of issues can be very expensive
- Growing need for automatic fixing of issues
  - Avoid long loops with verification flow

Problem Description

- Power grid modeled as R(LC)-network
- Devices represented as current sources
  - Tied to nodes on lowest layer
- Some nodes on top layer tied to voltage source
- Static analysis:
  - R-network
  - Time-invariant I_{src}
- Dynamic analysis:
  - RLC network
  - Time-varying I_{src}



Mathematical Formulation

- Static analysis:

$$\mathbf{GV} = \mathbf{I}$$

Where $\mathbf{G}$ = conductance matrix

$\mathbf{V}$ = vector of node voltages (to be computed)

$\mathbf{I}$ = vector of current sources (input)

- Transient analysis

$$\mathbf{GV}(t) + \mathbf{C}\, d\mathbf{V}/dt = \mathbf{I}(t)$$

$(\mathbf{G} + \mathbf{C}/h)\, \mathbf{V}(t) = \mathbf{I}(t) + \mathbf{C}/h\, \mathbf{V}(t-h)$ (Backward Euler)

where h = fixed time step, $\mathbf{C}$ = capacitance matrix

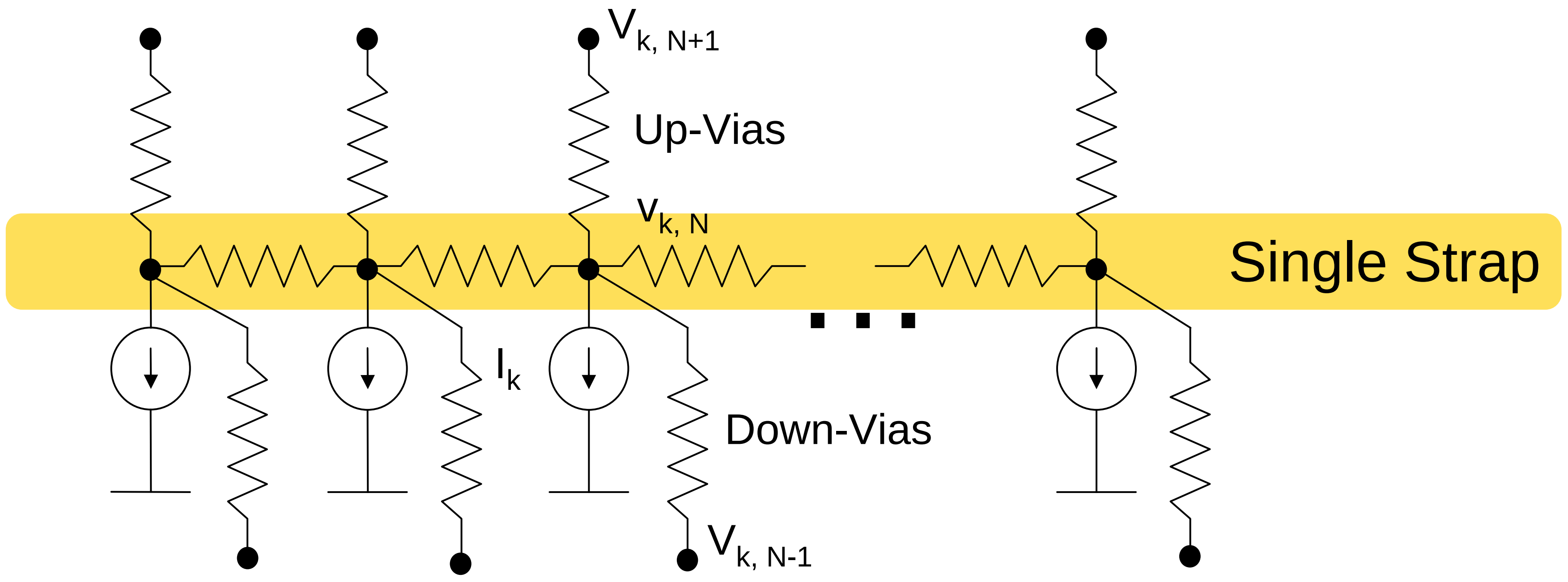
- Transient analysis = repeated solution of static case

Solution Techniques

- Matrix sizes can be O(10M) – O(100M) nodes
- Several solution techniques exist
  - Sparse-matrix techniques
  - Multigrid
  - Monte-Carlo approaches
  - Iterative solvers
  - Block-iterative solvers
- Our approach based on block-iterative RBI solver
  - RBI is 5X faster than multigrid on our real power grids
- Idea is to get an ***even faster*** approximate solution

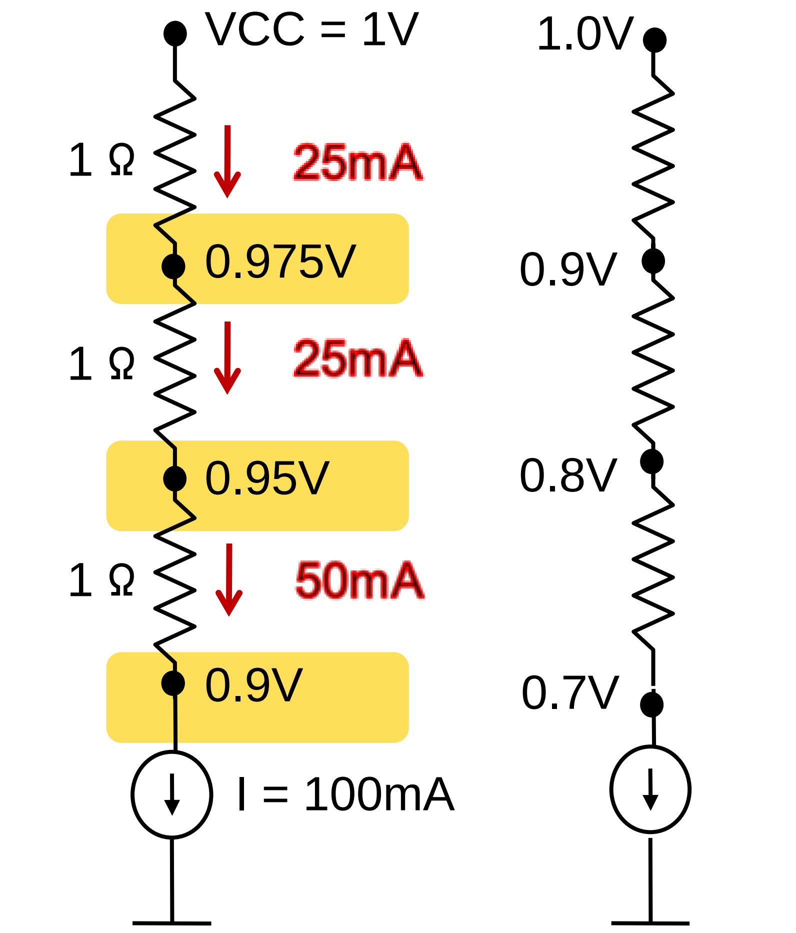
Row-Based Iterative (RBI) Solver

- Block-iterative procedure (Yu Zhong et al, ICCAD 2005)
  - Solve one strap at a time, keeping all other straps fixed
  - Iterate until convergence (guaranteed to converge)
- Each strap is a row of nodes
  - Simple ladder structure, can be solved in linear time



RBI Convergence

- Iterative solvers typically converge slowly
- One way of looking at the convergence problem is as "**current loss**" going from lower to higher layers
- If we can force all of the current to be propagated up, accuracy may be better

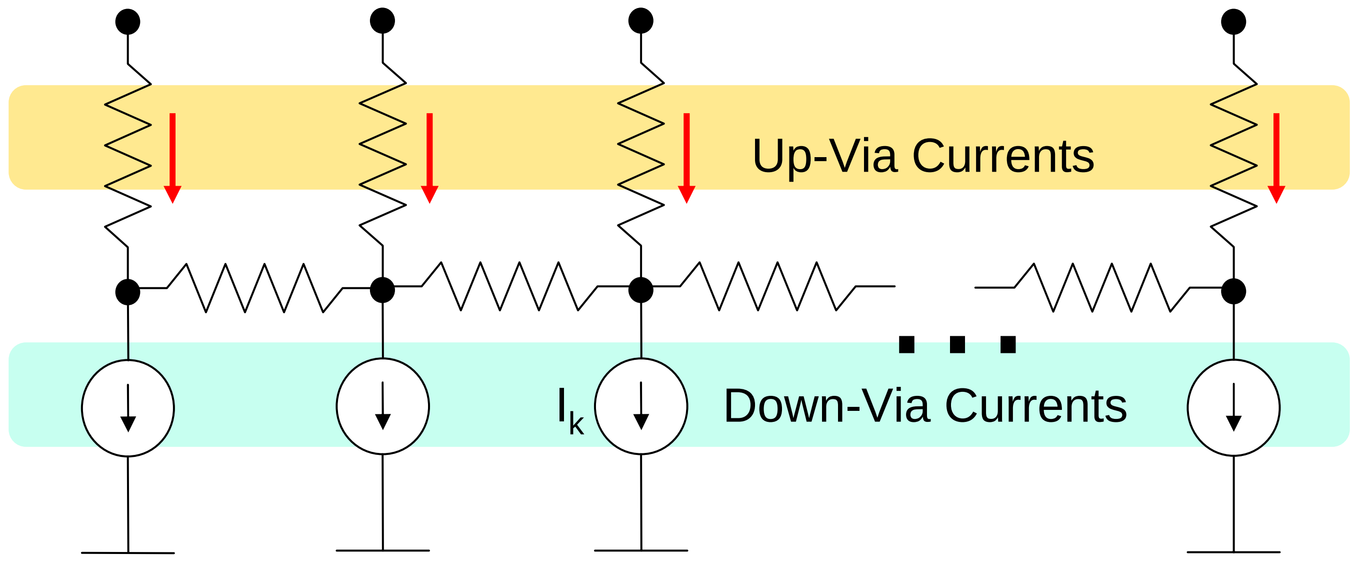


(a) First iteration (b) Solution

Current Propagation

- **Key idea**: prevent "current loss"
  - Compute currents in up-vias on layer K,
  - Then model down vias on layer (K+1) as current sources

Problem: how to distribute total current among up-vias



Upstream Resistance

- **Key idea**: distribute current in up-vias in proportion to how well the upper node is connected to V_{src}
- **Definition**: For a node N, $\mathbf{R_{upstream}}$(N) is the resistance of the least resistive path from N to V_{src}
- $\mathbf{R_{upstream}}$ can be computed for all nodes in the network by a simple breadth-first traversal of the network, starting from the voltage source node
- $\mathbf{R_{upstream}}$ is a useful indicator of power grid health
- Path of least resistance is also a useful debug tool

FAS Algorithm

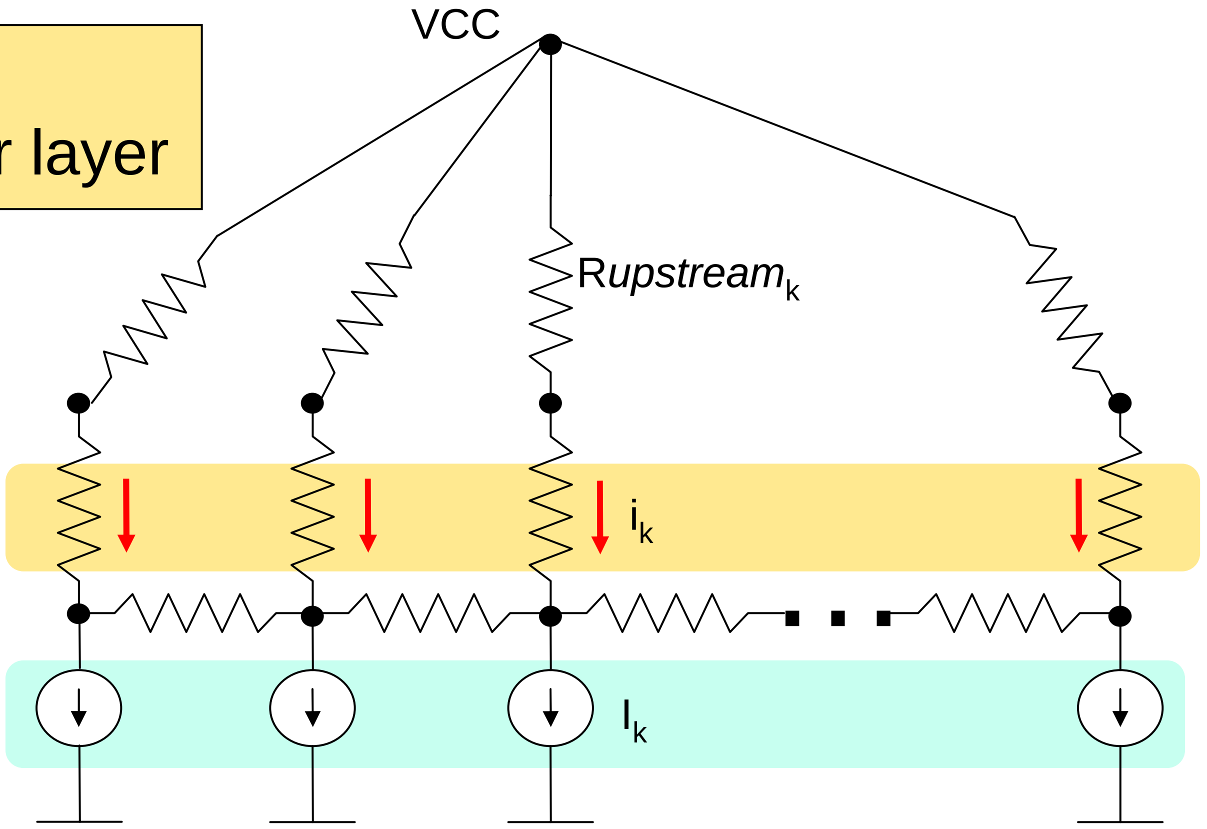
- Compute $\mathbf{R}_{\mathbf{upstream}}$ for every node N
- For layer = lowest to highest:
  - For each row on layer:
    - **Propagate_Currents_Up (row)**
- For layer = highest to lowest:
  - For each row on layer:
    - **Propagate_Voltages_Down (row)**
- Linear-time, linear-space algorithm
- No iterations
- Propagation steps trivially parallelizable

Propagate_Currents_Up (row)

- Each downvia replaced by current source
- Each upvia replaced by R(upvia)+ $\mathbf{R_{upstream}}$

Output: Currents propagated to upper layer

Input: Currents propagated from lower layer

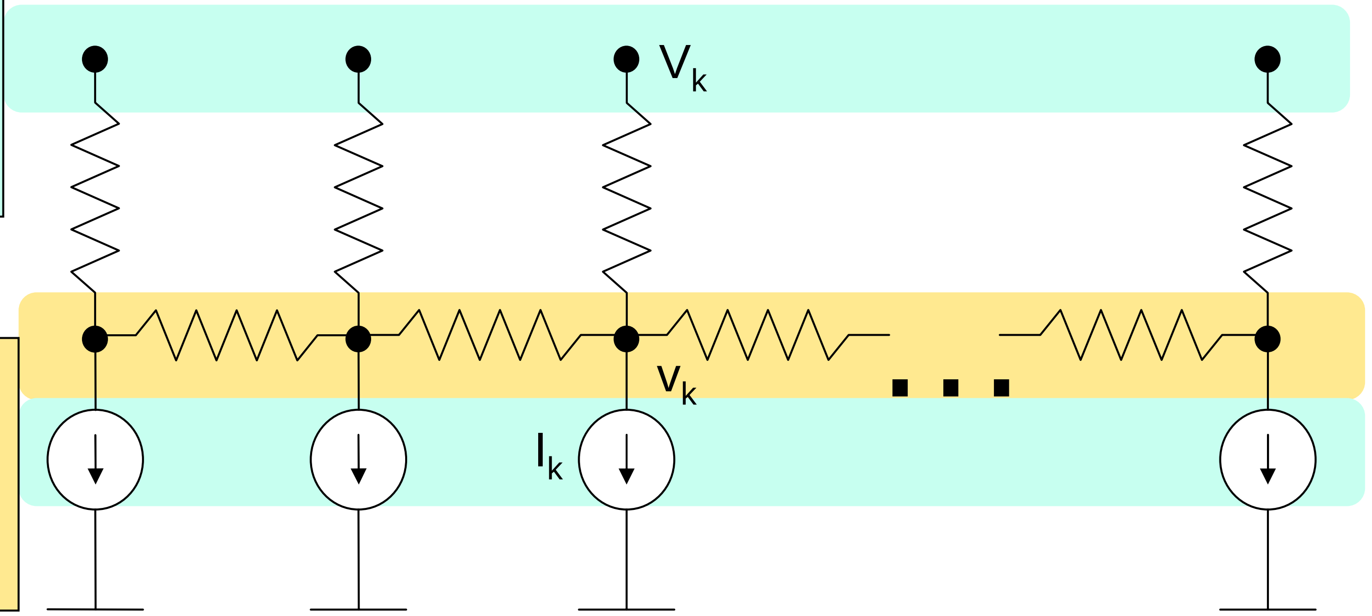


Propagate_Voltages_Down (row)

- Down-vias replaced by current sources
- Up-vias not modified

Input: Node voltages on upper layer

Output: Node voltages on current layer



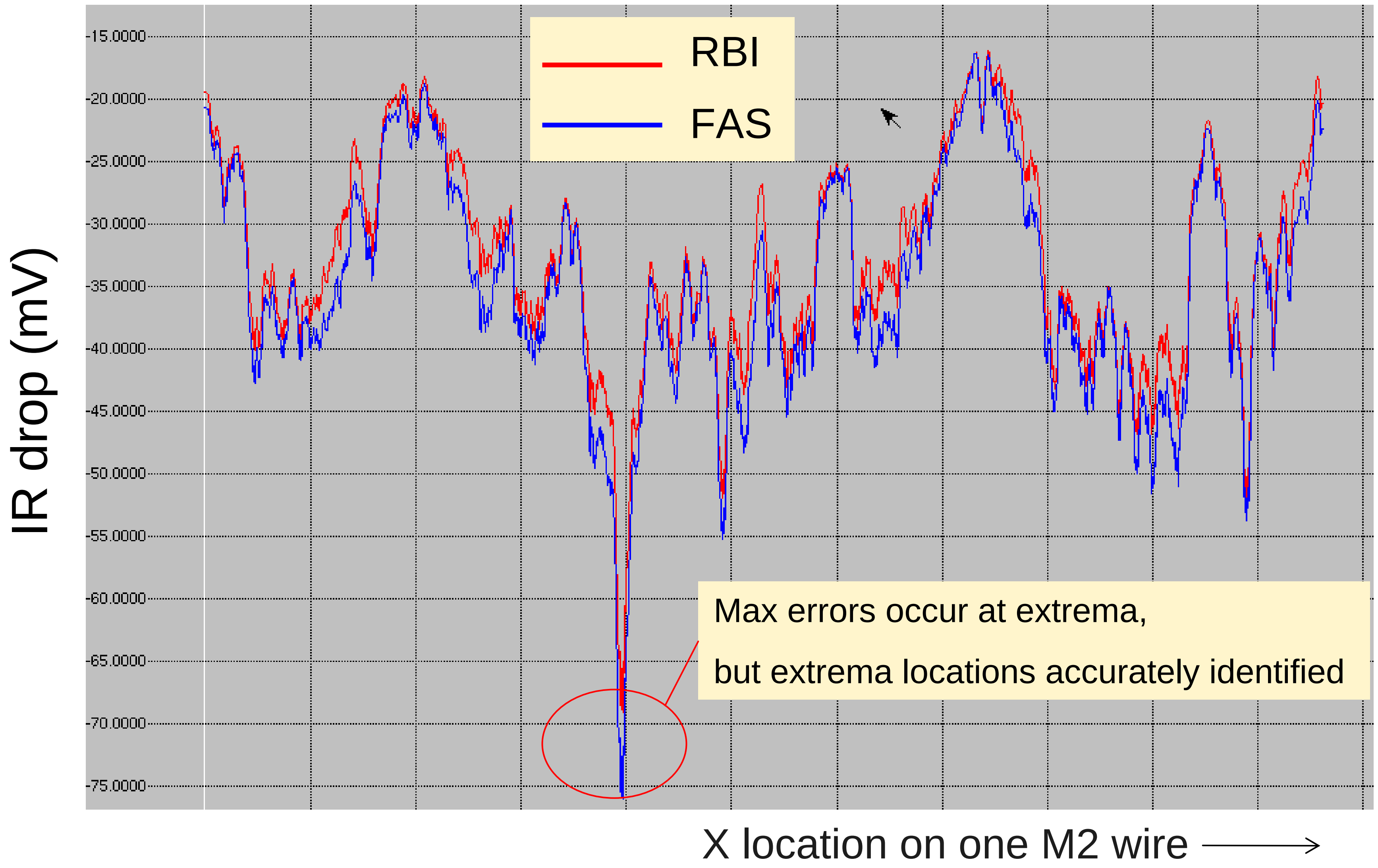
Results

- Real test cases from a 32 nm CPU design

Block	cbd1	cbd2	cbd3
Nodes	2.27M	4.49M	28.3M
RBI runtime (s)	10.8	21.5	130.2
FAS runtime (s)	2.12	4.01	22.8
Incremental recompute (s)	0.58	1.22	8.39
Memory (GB)	0.56	1.05	6.26
Avg. Error %	2.7%	3.6%	4.6%
Max Error %	10.3%	18.1%	14.5%

- Using 4 threads on a dual-core 3.2GHz Xeon CPU
- Speed ~ 1s/million nodes – 5X faster than RBI

Fine-Grained Fidelity of FAS

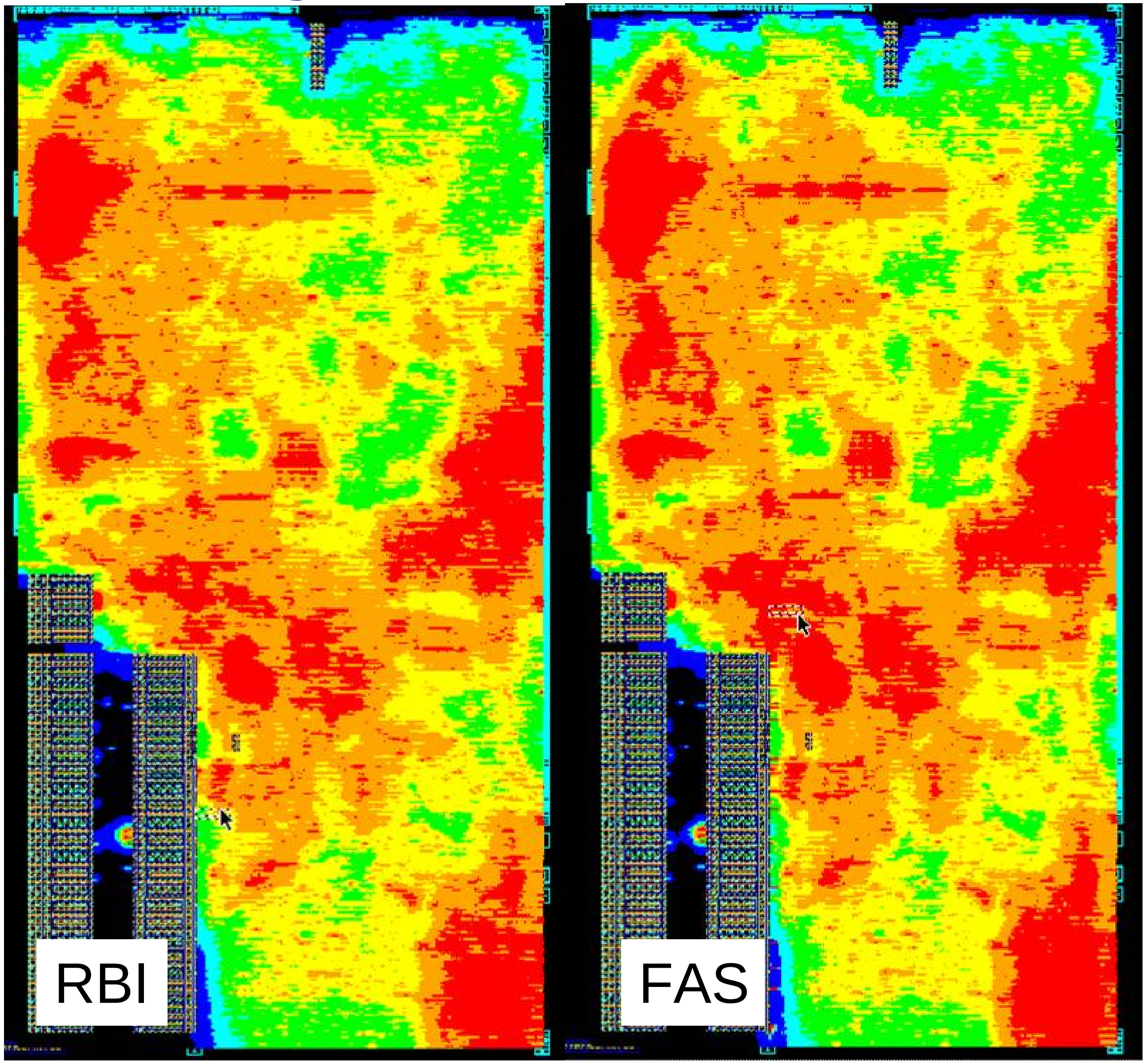


Max errors occur at extrema,

but extrema locations accurately identified

Fidelity of FAS

- FAS can quickly identify potential problem areas needing more accurate analysis



Applications

- FAS properties similar to Elmore Delay
  - Very fast and easy to compute
  - Incremental recomputes are even faster
  - Excellent fidelity, somewhat pessimistic
- Can be used in inner loop of optimization flows
  - Auto-fixing by strap addition
  - Placement optimization for IR drop
- Dynamic analysis
  - Transient analysis of RLC networks
  - Analysis of multiple switching scenarios

Summary

- VLSI power grid analysis is increasingly important as supply voltage becomes lower and integration density increases
- Need for fast early analysis to avoid late surprises
- FAS algorithm exploits well-formedness property of real VLSI grids to solve > 1M nodes/sec with reasonable accuracy
- High “fidelity” of results makes it good candidate for use inside optimization algorithms