An Unconditional Stable General Operator Splitting Method for Transistor Level Transient Analysis

Zhengyong Zhu, Rui Shi, Chung-Kuan Cheng
Department of CSE
University of California, San Diego

Ernest S. Kuh
Department of EECS
University of California, Berkeley

Outline

I. Introduction

II. Operator Splitting Algorithm

III. Stability Analysis

IV. Partitioning

V. Experimental Results

VI. Conclusion

I. Operator Splitting- A Simple Example

Initial value problem (IVP) of ordinary differential equation (ODE)

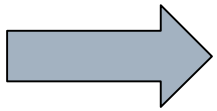
$$\frac{\delta u}{\delta t} = Lu$$

where L is a linear/nonlinear operator and can be written as a linear sum of m subfunctions of u

$$Lu = L_1 u + L_2 u + \cdots + L_m u$$

Suppose $U_1, U_2, \cdots, U_m$ are updating operators on u from time step n to n+1 for each of the subfunctions, the operator splitting method has the form of:

$$u^{n+1} = U(u^n, h)$$



$$u^{n+(1/m)} = U_1(u^n, h/m)$$

$$u^{n+(2/m)} = U_2(u^{n+(1/m)}, h/m)$$

$$\cdots$$

$$u^{n+1} = U_m(u^{n+(m-1)/m}, h/m)$$

I. Previous work on Operator Splitting

- First used to solve PDE equation
- Previous Work
  - Namiki and Ito (IEEE Tran. Microwave Theory and Technology 1999)
    - adopted its special form, the alternating direction implicit (ADI) to simulate a 2D EM wave. Unconditional Stable.
  - Zheng (IEEE Tran. Microwave Theory and Technology 2001)
    - extend to 3D problem
  - Chen (ICCAD 2001, DATE 2003)
    - 2D-ADI, 3D-ADI for power network analysis
- Limitation
  - Split along geometric direction.

I. Operator Splitting on Circuit Simulation

- We generalize the operator splitting to graph based modeling
- No geometry or locality constrains
- Convergence
  - A-stable: independent of time step size
  - Consistence : local truncation error

II. Formulation - Forward Euler

Circuit Equation for RLC circuits:

$$\begin{bmatrix} C & 0 \\ 0 & L \end{bmatrix} \begin{bmatrix} \dot{V} \\ \dot{I} \end{bmatrix} + \begin{bmatrix} G & -A^T \\ A & R \end{bmatrix} \begin{bmatrix} V \\ I \end{bmatrix} = \begin{bmatrix} U \\ 0 \end{bmatrix}$$

Forward Euler Formulation

$$V(t+h) = V(t) + \dot{V}(t)h,\ I(t+h) = I(t) + \dot{I}(t)h$$

$$\begin{bmatrix} \frac{C}{h} & 0 \\ 0 & \frac{L}{h} \end{bmatrix} \begin{bmatrix} V(t+h) \\ I(t+h) \end{bmatrix} = \begin{bmatrix} \frac{C}{h} - G & A^T \\ -A & \frac{L}{h} - R \end{bmatrix} \begin{bmatrix} V(t) \\ I(t) \end{bmatrix} + U(t)$$

where

C: capactiance matrix L: inductance matrix

R: resistance matrix G: conductance matrix

A: incidence matrix h: time step

II. Formulation – Backward Euler

Circuit Equation for RLC circuits:

$$\begin{bmatrix} C & 0 \\ 0 & L \end{bmatrix} \begin{bmatrix} \dot{V} \\ \dot{I} \end{bmatrix} + \begin{bmatrix} G & -A^T \\ A & R \end{bmatrix} \begin{bmatrix} V \\ I \end{bmatrix} = \begin{bmatrix} U \\ 0 \end{bmatrix}$$

Backward Euler Formulation $V(t+h) = V(t) + \dot{V}(t+h)h,\ I(t+h) = I(t) + \dot{I}(t+h)h$

$$\begin{bmatrix} \frac{C}{h} + G & -A^T \\ A & \frac{L}{h} + R \end{bmatrix} \begin{bmatrix} V(t+h) \\ I(t+h) \end{bmatrix} = \begin{bmatrix} \frac{C}{h} & 0 \\ 0 & \frac{L}{h} \end{bmatrix} \begin{bmatrix} V(t) \\ I(t) \end{bmatrix} + U(t+h)$$

where

C: capactiance matrix L: inductance matrix

R: resistance matrix G: conductance matrix

A: incidence matrix h: time step

II. Splitting Formulation

Split the circuit resistor branches into two partitions, we have

$$A = A_1 + A_2$$

$$G = G_1 + G_2$$

$$R = R_1 + R_2$$

Each partition has a full-version of capacitors and inductors.

II. Splitting Formulation

General Operator Splitting Iteration:

$$\begin{cases} \begin{bmatrix} \frac{2C}{h}+G_1 & -A_1^T \\ A_1 & \frac{2L}{h}+R_1 \end{bmatrix} \begin{bmatrix} V(t+\frac{h}{2}) \\ I(t+\frac{h}{2}) \end{bmatrix} = \begin{bmatrix} \frac{2C}{h}-G_2 & A_2^T \\ -A_2 & \frac{2L}{h}-R_2 \end{bmatrix} \begin{bmatrix} V(t) \\ I(t) \end{bmatrix} + U(t+\frac{h}{2}) \\ \begin{bmatrix} \frac{2C}{h}+G_2 & -A_2^T \\ A_2 & \frac{2L}{h}+R_2 \end{bmatrix} \begin{bmatrix} V(t+h) \\ I(t+h) \end{bmatrix} = \begin{bmatrix} \frac{2C}{h}-G_1 & A_1^T \\ -A_1 & \frac{2L}{h}-R_1 \end{bmatrix} \begin{bmatrix} V(t+\frac{h}{2}) \\ I(t+\frac{h}{2}) \end{bmatrix} + U(t+h) \end{cases}$$

Alternate Backward and forward integration on partitions

III. Convergence Study

General Operator Splitting

A numerical integration method

Alternate Backward and forward integration on partitions

Two conditions for convergence

Global Condition — The single step errors do not grow too quickly (stability)

Local Condition — One step errors are small (consistency)

III. Stability Analysis

A Stable

Theorem: The operator splitting approach is stable independent of time step h

How to prove?

Derive iterative matrix, and prove its maximum eigenvalue no larger than 1

III. Stability Analysis- Iterative Matrix

$$\text{Let } P_1 = \begin{bmatrix} G_1 & -A_1^T \\ A_1 & R_1 \end{bmatrix}, P_2 = \begin{bmatrix} G_2 & -A_2^T \\ A_2 & R_2 \end{bmatrix}, S = \begin{bmatrix} \frac{2C}{h} & 0 \\ 0 & \frac{2L}{h} \end{bmatrix}, \text{and } X = \begin{bmatrix} V \\ I \end{bmatrix}$$

$$A = A_1 + A_2 \quad G = G_1 + G_2 \quad R = R_1 + R_2$$

The operator splitting approach can be simplified as:

$$\begin{cases} (P_1 + S)X(t + \frac{h}{2}) = -(P_2 - S)X(t) + U(t + \frac{h}{2}) \\ (P_2 + S)X(t + h) = -(P_1 - S)X(t + \frac{h}{2}) + U(t + h) \end{cases}$$

Actually an iterative update on X:

$$X_{(k+1)} = \Lambda X_{(k)}$$

$$\Lambda = (P_2 + S)^{-1}(P_1 - S)(P_1 + S)^{-1}(P_2 - S)$$

III. Stability Analysis-2

The operator splitting approach converges if the maximum Eigenvalue of the iterative matrix no larger than 1

$$\rho(\Lambda) <= 1$$

$$\Lambda = (P_2 + S)^{-1}(P_1 - S)(P_1 + S)^{-1}(P_2 - S)$$

If we can prove $\|\Lambda\| <= 1$, then $\rho(\Lambda) <= 1$

We pick the norm: $\|x\|_{S^{-1}} = (x^T S^{-1} x)^{1/2}$

Where S is a positive definite matrix

IV. Partitioning

- Objective
  - Minimize the overall nonzero fill-ins
  - Guarantee DC path for every nodes
  - Hint:
    - Tree structure generate no nonzero fill-ins in LU factorization.
    - High Degree nodes generates more nonzero fill-ins during LU factorization

- Linear Circuits (talk about circuits with transistors later)
  - Bipartition of neighbors for each node
  - Basic idea
    - In the LU decomposition process, non-zero fill-in will be introduced among neighbors of the pivot. Reduce the number of neighbors for all nodes will be beneficial to decrease the number of non-zero fill-ins.
    - Avoid loop, make tree structure as much as possible
  - Check DC path, reassign partition if necessary

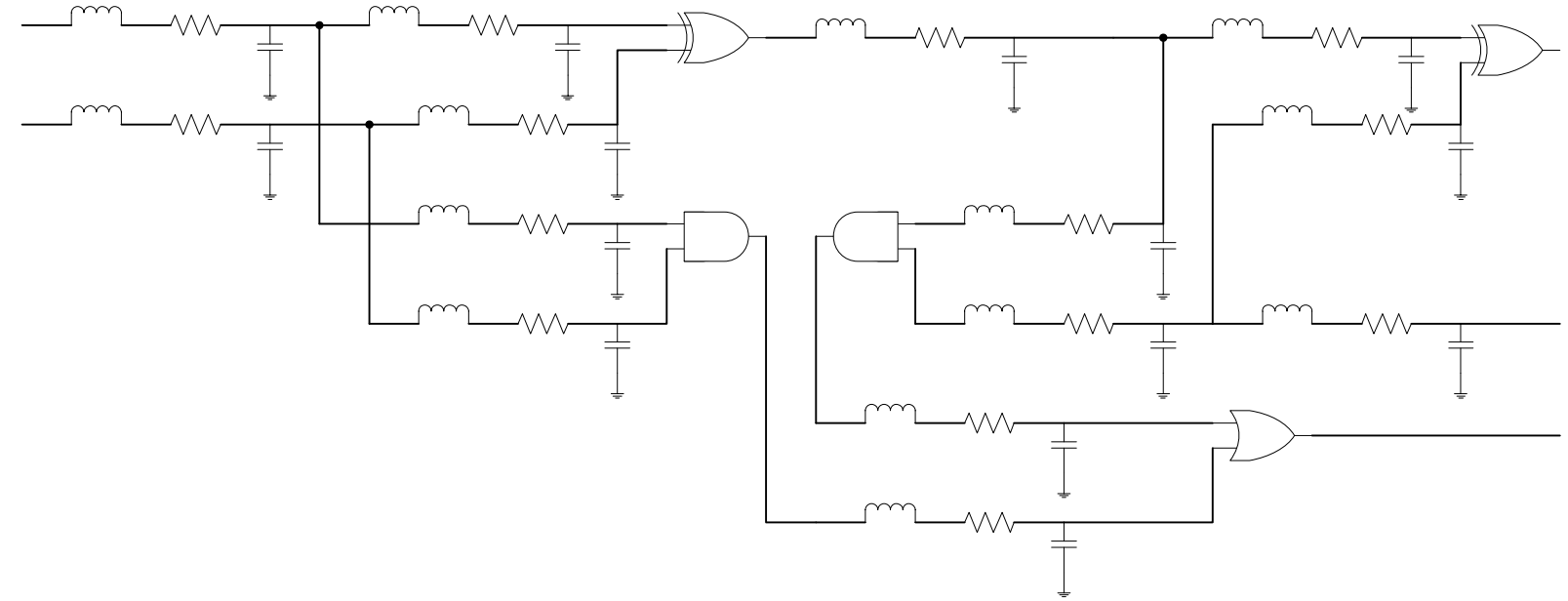
IV. Partitioning -Nonlinear circuits

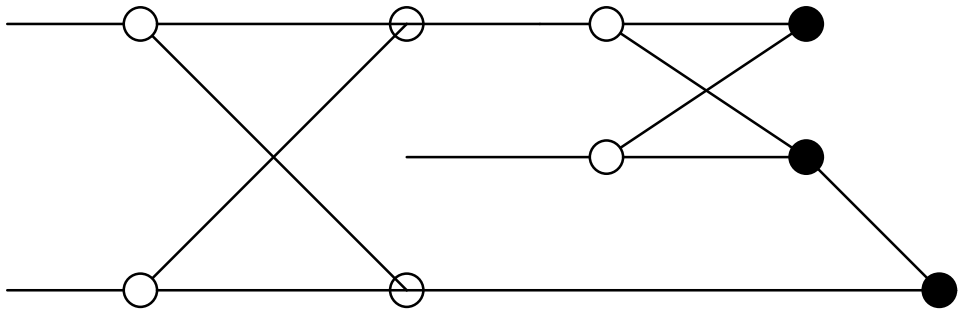
- Duplicate transistors into both partition
  - Taking into account the nonlinearity of transistor gates
- Regard each gate as super node
  - apply the splitting algorithm on super node structure
- Super Node in LU decomposition
  - During factorization, eliminate the internal nodes of each super node first
  - Nonzero fill-ins are confined inside the gate, reduce the number of unnecessary nonzero fill-ins.

IV. Partitioning Algorithm

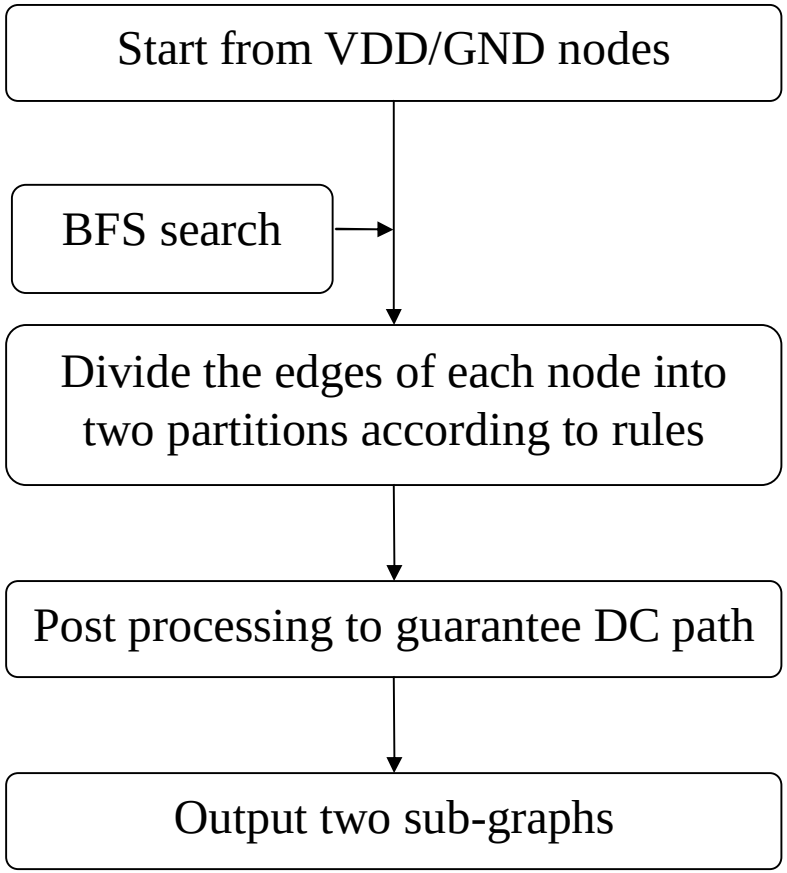
- Objective:
  - Split the circuit into two partitions
  - Minimize the total number of non-zero fill-ins for the matrix decomposition in circuit simulation
- Basic idea:
  - Split the circuits into two partitions with structures close to tree or forest
  - Decrease the degree of all the nodes in two partitions

IV. Undirected graph representation





IV. Partitioning Algorithm



IV. Partitioning Algorithm

- Branch rule: the edges in one branch belong to the same partition. If one branch is broken into two partitions, the broken branch node will result in much iteration for simulation.
- Degree rule: the edges of node whose degree is two belong to the same partition. The line structure wouldn't cause many non-zero fill-ins and it will be propitious to provide DC path in the sub-graphs.
- Loop rule: the loop will be avoided in each sub-graph if possible. Edges loop in the sub-graphs will potentially introduce certain number of non-zero fill-ins.
- Balance rule: the edges for each node in the graph will be divided into two sub-graphs evenly. Thus each sub-graph will be much simpler than the original graph.

IV. # of nonzero fill-ins

Test Case:

A small ASIC Design

Spice matrix :

Dimension: 10,286

The number of non-zero elements: 46,655

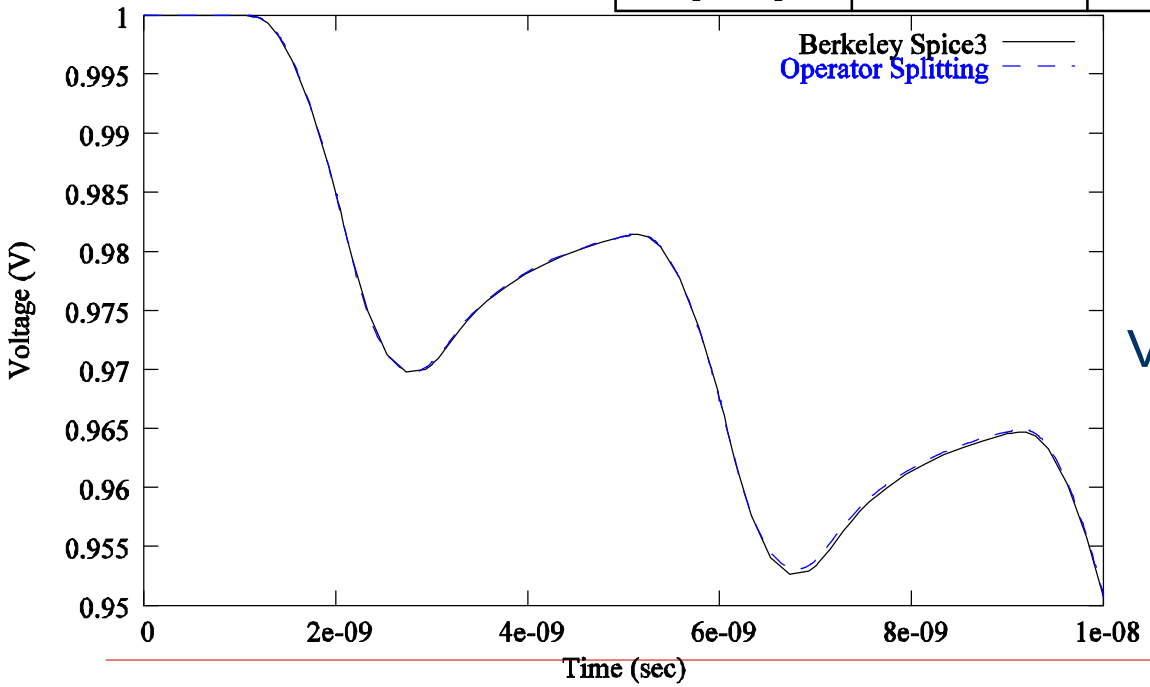
The number of non-zero fill-ins: 90,960

	Sub-matrix1		Sub-matrix2		Total # non-zero fill-ins
	# non-zero elements	# non-zero fill-ins	# non-zero elements	# non-zero fill-ins	
Operator Splitting	38,572	2,618	42,020	10,040	12,658

V. Experimental Results-1

Power Network & Gate Sinks

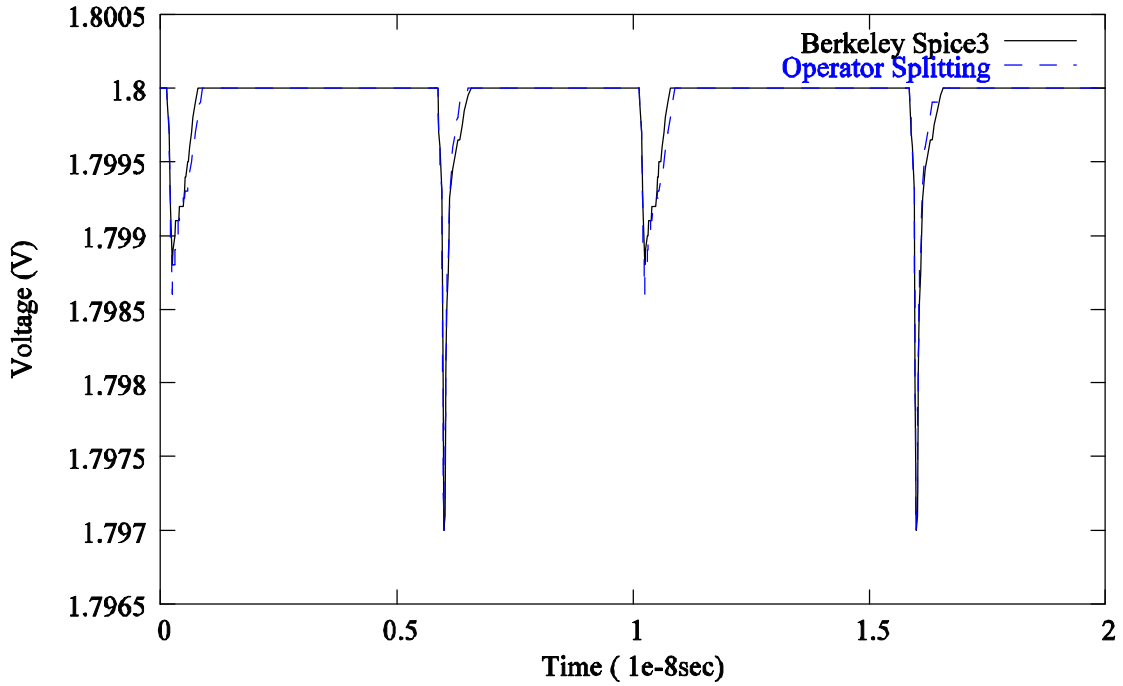
Examples	Circuit1	Circuit2	Circuit3	Circuit4
# Nodes	11,203	41,321	92,360	160,657
# Transistors	74	513	1,108	2,130
Simulation Period	10ns	10ns	10ns	10ns
SPICE3(sec)	602.44	8268.92	39612.32	N/A
Operator Splitting	74.64	305.38	681.18	1356.21
Speedup	8.1x	27.1x	58.2x	N/A



Voltage Drop of Circuit3

V. Experimental Results-2

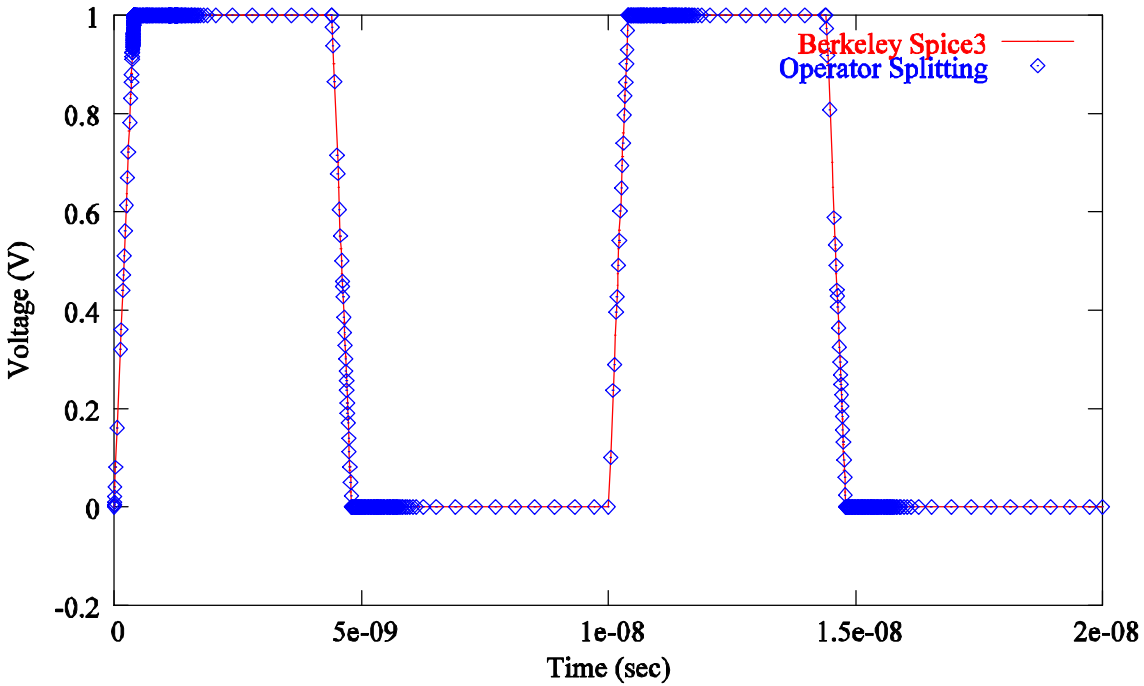
- RLC Power/Clock network case.
  - 29110 nodes, 720 transistor devices
  - Spice3 Runtime: 12015 sec.
  - Our Run time: 649.5 sec. 18.5x



V. Experimental Result-3

Two 1K and 10K cell designs

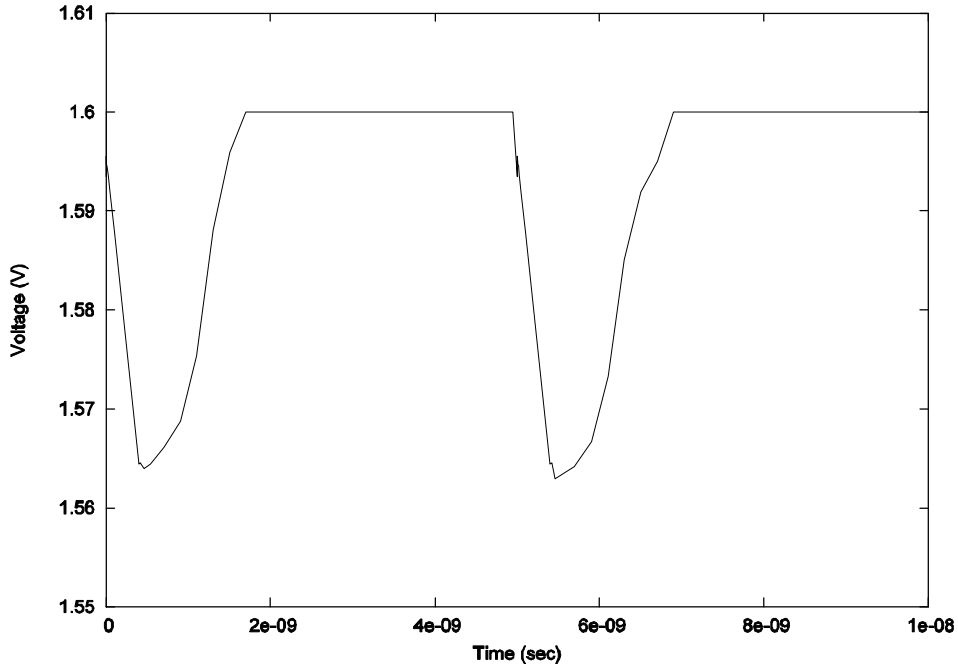
Bottle neck:
Nonzero fill-ins
Device Evaluation Time



	# of Nodes	# of transistor	Spice3 runtime(s)	Our runtime (s)	Speedup
1k Cell	10,200	6,500	2121	415.9(s)	5.1x
10k Cell	123,600	69,000	44293	3954.7(s)	11.2x

V. Large Power Ground Network

- 600,000 nodes
- Irregular RC network
- 10ns Transient Simulation: 4083sec



VI. Conclusion

- Moore's Law demands an extraordinary fast circuit simulator with guaranteed accuracy.
- Accuracy
  - No trade off for speedup
  - Rigorous convergence check
  - Spice Level Accuracy
- Performance
  - Orders of magnitude speedup over SPICE
  - The larger the linear network, the higher the speedup
- Expandable
  - Possible to combine with some other techniques
    - Parasitic reduction
    - Fast device evaluation
    - Simplified linearization process