LIM – A General-Purpose Simulator for High-Speed Circuit Design

José E. Schutt-Ainé, UIUC
Thong Nguyen, Synclesis
Daniel Shaw, Synclesis

IBIS Summit
May 10, 2023
Aveiro, Portugal

Outline

- LIM vs MNA
- LIM Components
- LIM & IBIS
- LIM & Macromodeling
- LIM Format & Support

Commercial Simulators - Comparison

	Vendor_0	Vendor_1	Vendor_2	Vendor_3	Vendor_4	Vendor_5	LIM
Frequency Dependence	NO	YES	YES	NO	NO	YES	YES
Uses MNA	YES	YES	YES	YES	YES	YES	NO
BSIM-CMG	NO	YES	YES	YES	YES	NO	NOT YET

MNA: Modified Nodal Analysis

Why LIM?

- MNA has super-linear numerical complexity
- LIM has linear numerical complexity
- LIM has no matrix ill-conditioning problems
- Accuracy and stability in LIM are easily controlled
- LIM is much faster than MNA for large circuits

LIM: Leapfrog Method

$$\begin{tikzcd}
I_{ij}^{n} \arrow[dr] & & I_{ij}^{n+1} \arrow[dr] & & I_{ij}^{n+2} \arrow[dr] & \\
& V_i^{n+1/2} \arrow[ur] & & V_i^{n+3/2} \arrow[ur] & & V_i^{n+5/2}
\end{tikzcd}$$

$\longrightarrow$ *n: time*

$$V_i^{n+1/2} = \frac{\dfrac{C_i V_i^{n-1/2}}{\Delta t} + H_i^n - \displaystyle\sum_{k=1}^{N_a} I_{ik}^n}{\dfrac{C_i}{\Delta t} + G_i}$$

$$I_{ij}^{n+1} = I_{ij}^n + \frac{\Delta t}{L_{ij}}\left(V_i^{n+1/2} - V_j^{n+1/2} - R_{ij} I_{ij}^n\right)$$

Leapfrog method achieves second-order accuracy, i.e., error is proportional to Δt^2

VinC – Voltage in Current*

$$V_i^{n+1} = \frac{\dfrac{C_i V_i^{n-1}}{\Delta t} + H_i^{n+1/2} - \sum_{k=1}^{N_a} I_{ik}^{n+1/2}}{\dfrac{C_i}{\Delta t} + G_i} = \Gamma_i V_i^n + Z_{ni} H_i^{n+1/2} - Z_{ni} \sum_{k=1}^{N_a} I_{ik}^{n+1/2}$$

$$I_{ij}^{n+1/2} = I_{ij}^{n-1/2} + \frac{\Delta t}{L_{ij}} \left(V_i^{n+1} - V_j^{n+1} - R_{ij} I_{ij}^{n-1/2} \right)$$

Algebraic sum of all currents incident at node *i* except that of branch ***ij***

$$\Downarrow$$

$$I_{ij}^{n+1/2} = \frac{I_{ij}^{n-1/2} + \dfrac{\Delta t}{L_{ij}} \left(\Gamma_i V_i^n + Z_{ni} H_i^{n+1/2} - Z_{ni} \sum_{k=1,k\neq j}^{N_a} I_{ik}^{n+1/2} - \Gamma_j V_j^n - Z_{nj} H_j^{n+1/2} + Z_{nj} \boxed{\sum_{k=1,k\neq j}^{N_a} I_{jk}^{n+1/2}} - R_{ij} I_{ij}^{n-1/2} \right)}{\left[1 + Z_{ni} \dfrac{\Delta t}{L_{ij}} + Z_{nj} \dfrac{\Delta t}{L_{ij}} \right]}$$

VinC formulation achieves higher accuracy and stability

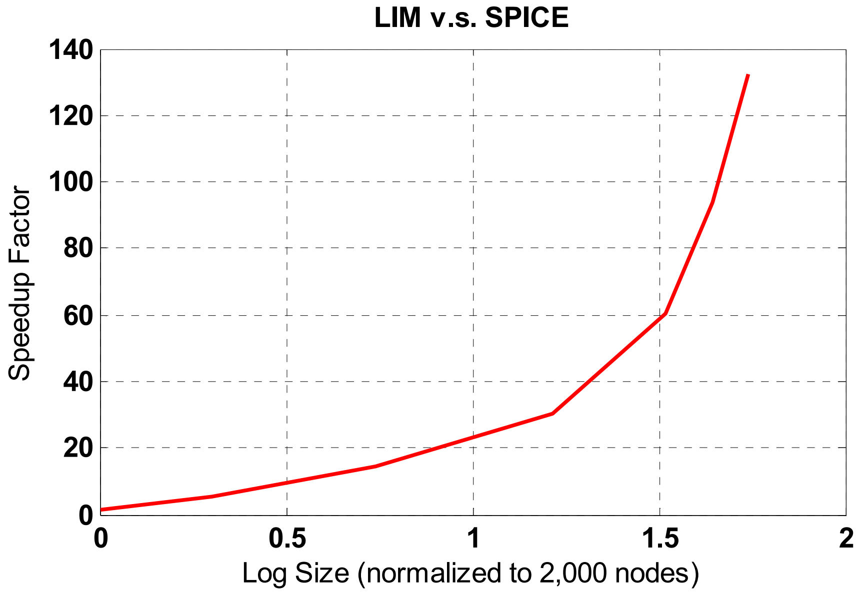
---

* K.H. Tan, P. Goh and M.F. Ain, "Voltage-in-current formulation for the latency insertion method for improved stability", *Electronics Letters*, vol. 52, no. 23, Nov. 2016, pp. 1904-1906.

LIM vs SPICE

Simulation Times

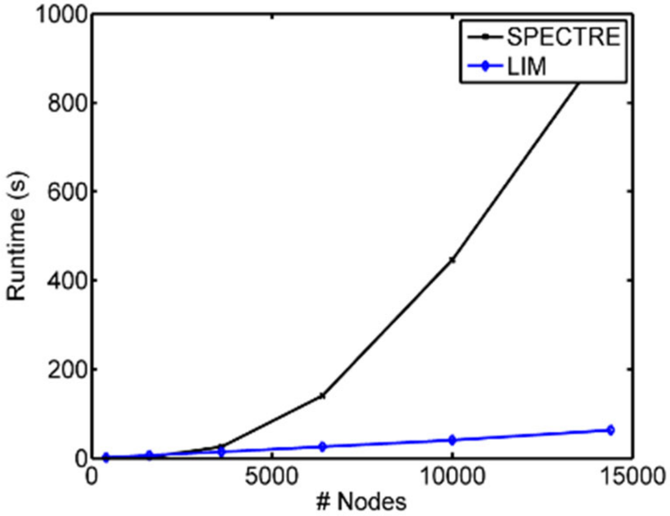
No of Nodes	20,000	30,000	40,000	50,000
SPICE (sec)	1224	2935	4741	7358
LIM (sec)	9	13	17	21
Speedup	136	225	278	350



Example – RLGC Grid

- Comparison of runtime for LIM and Vendor_3.

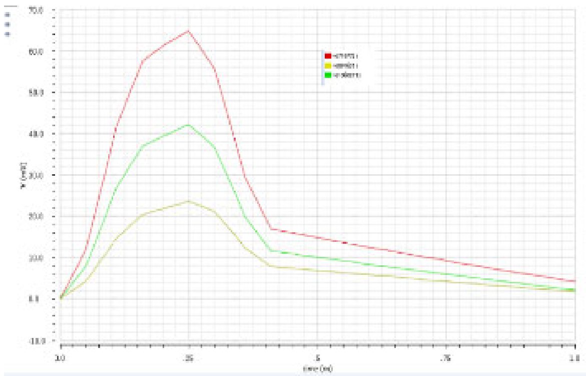
Circuit Size	# nodes	VENDOR_3 (s)	LIM (s)
20 × 20	400	0.15	1.53
40 × 40	1600	2.14	6.25
60 × 60	3600	25.73	14.23
80 × 80	6400	140.59	25.71
100 × 100	10000	445.77	40.64
120 × 120	14400	945.00	62.89



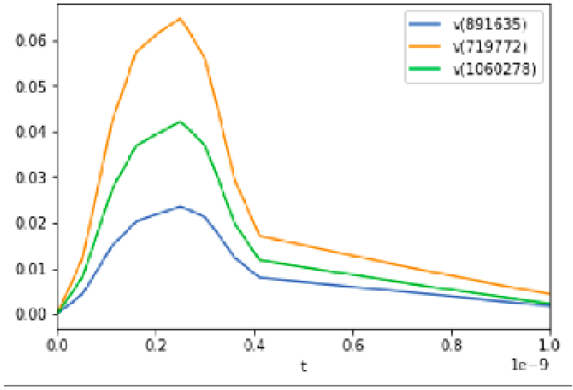
- LIM exhibits linear numerical complexity!
  - Outperforms conventional SPICE-like simulators.

LIM – Simulation of Large Circuits

VENDOR_3



VENDOR_0



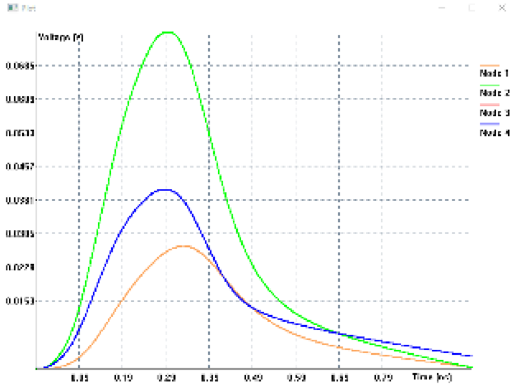
- 2,029,744 series resistors
- 380,742 shunt resistors
- 380,742 series capacitors
- 380,742 shunt capacitors
- 381 series inductors
- 835,858 series voltage sources
- 381 shunt voltage sources
- 761,484 shunt current sources

VENDOR_3: **1 day 20 hours 44 minutes**

VENDOR_0: **3 days 2 hours 57 minutes**

LIM: **2 hours 37 minutes 39 seconds**

LIM



Large Circuit Results*

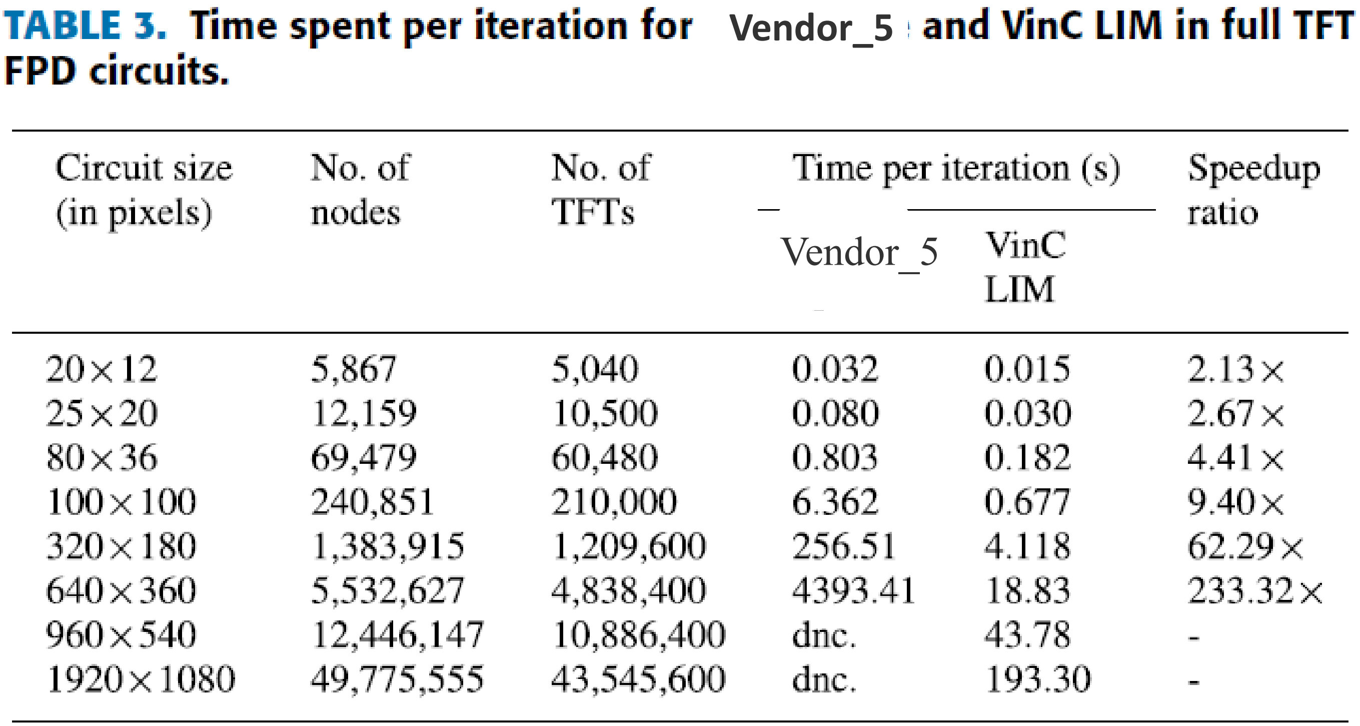
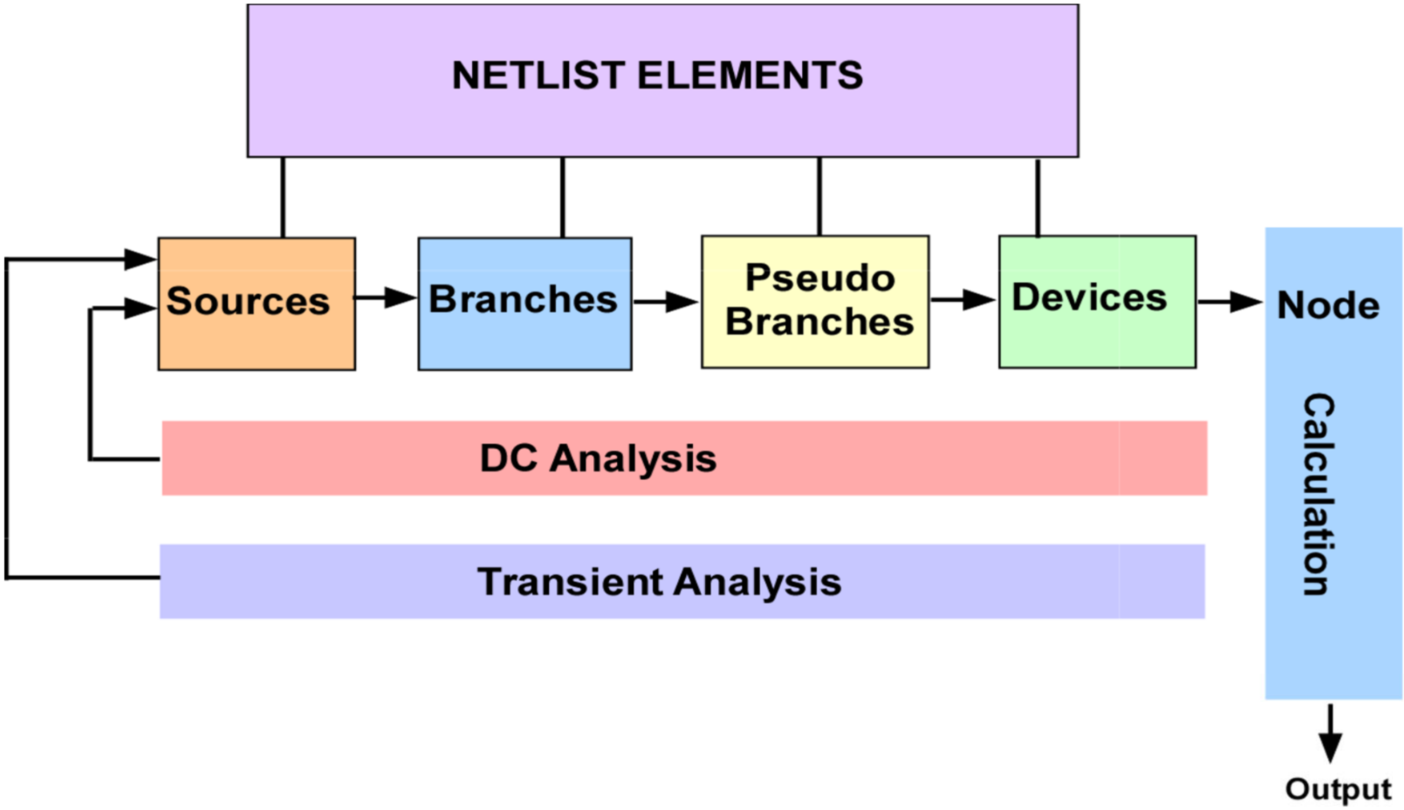


TABLE 3. **Time spent per iteration for Vendor_5 and VinC LIM in full TFT FPD circuits.**

Circuit size (in pixels)	No. of nodes	No. of TFTs	Time per iteration (s)		Speedup ratio
			Vendor_5	VinC LIM	
20×12	5,867	5,040	0.032	0.015	2.13×
25×20	12,159	10,500	0.080	0.030	2.67×
80×36	69,479	60,480	0.803	0.182	4.41×
100×100	240,851	210,000	6.362	0.677	9.40×
320×180	1,383,915	1,209,600	256.51	4.118	62.29×
640×360	5,532,627	4,838,400	4393.41	18.83	233.32×
960×540	12,446,147	10,886,400	dnc.	43.78	-
1920×1080	49,775,555	43,545,600	dnc.	193.30	-

*Wei Chun Chin, Andrei Pashkovich, José E. Schutt-Ainé, Nur Syazreen Ahmad, Patrick Goh,, "Thin-Film Transistor Simulations With the Voltage-In-Current Latency Insertion Method", *IEEE Access, Volume 9, 2021.*

Simulator Flow



LIM Simulator Development

TYPES OF ANALYSIS

DC Analysis

AC Small-Signal Analysis

Transient Analysis

Pole-Zero Analysis

Small-Signal Distortion Analysis

Sensitivity Analysis

Noise Analysis

Done
In Development

LIM Simulator Development

ELEMENTARY DEVICES
Resistors
Capacitors
Inductors
Mutual Inductors

INDEPENDENT SOURCES
Pulse
Sinusoidal
Exponential
Piece-Wise Linear
Pseudo-random Bit Sequence

DEPENDENT SOURCES
Voltage-Controlled Current Sources
Voltage-Controlled Voltage Sources
Current-Controlled Current Sources
Current-Controlled Voltage Sources

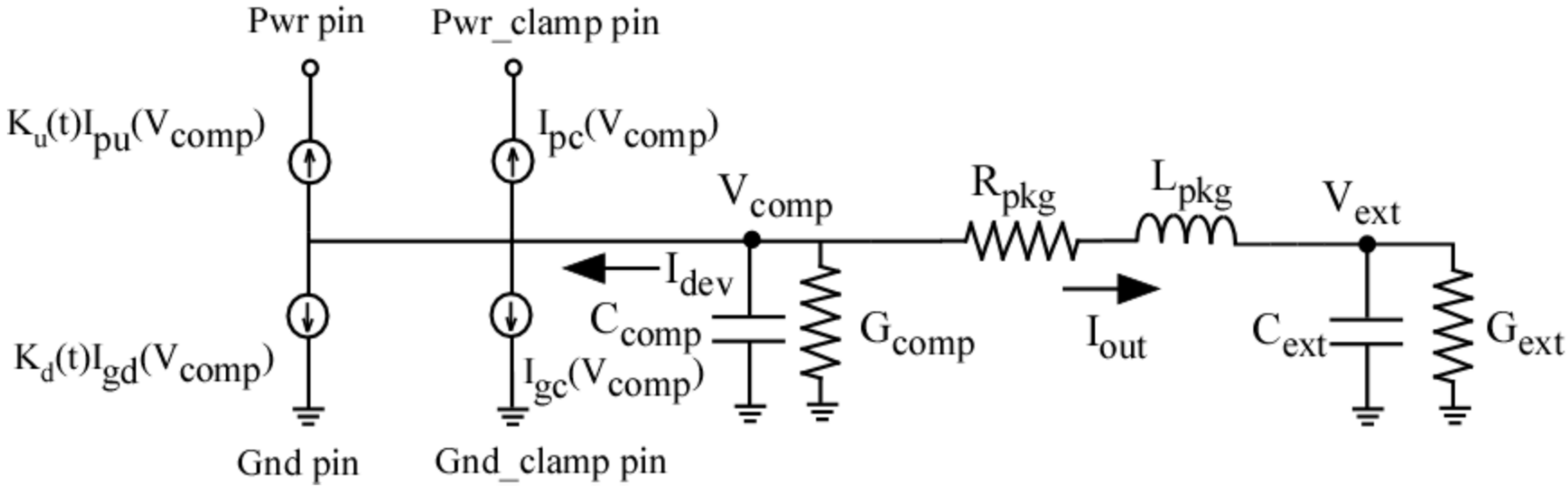
TRANSMISSION LINES
Lossless Single Transmission Lines
Lossy Single Transmission Lines
Lossless Multiconductor Transmission Lines
Lossy Multiconductor Transmission Lines

MACROMODELS
Model-Order Reduction
Convolution Model

DEVICES
Junction Diodes
Bipolar Junction Transistors (BJTs)
Junction Field-Effect Transistors (JFETs)
MOSFETs
MESFETs

Done
In Development

IBIS-LIM Formulation



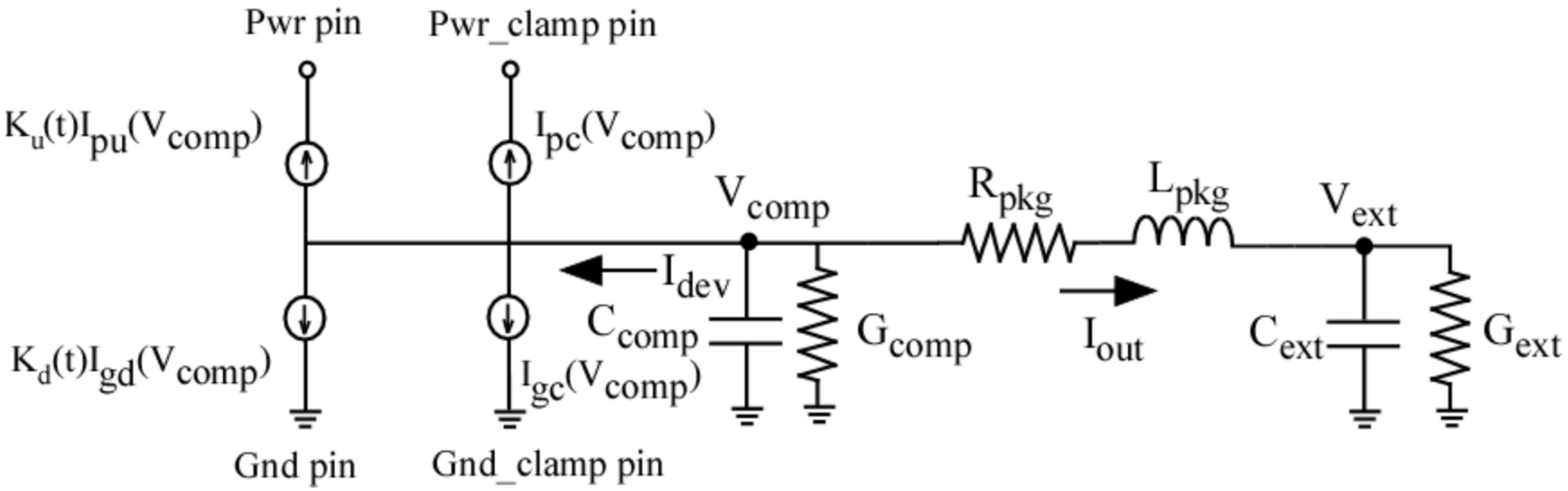
$$C_{ext}\frac{\left(V_{ext}^{n+1/2}-V_{ext}^{n-1/2}\right)}{\Delta t}+\frac{G_{ext}}{2}\left(V_{ext}^{n+1/2}+V_{ext}^{n-1/2}\right)=I_{out}^{n}$$

$$V_{ext}^{n+1/2}=\frac{I_{out}^{n}+\left(\frac{C_{ext}}{\Delta t}-\frac{G_{ext}}{2}\right)V_{ext}^{n-1/2}}{\left(\frac{C_{ext}}{\Delta t}+\frac{G_{ext}}{2}\right)}$$

$$V_{comp}^{n+1/2}-V_{ext}^{n+1/2}=L_{pkg}\frac{\left(I_{out}^{n+1}-I_{out}^{n}\right)}{\Delta t}+\frac{R_{pkg}}{2}\left(I_{out}^{n+1}+I_{out}^{n}\right)$$

$$I_{out}^{n+1}=\frac{\left(V_{comp}^{n+1/2}-V_{ext}^{n+1/2}\right)+I_{out}^{n}\left(\frac{L_{pkg}}{\Delta t}-\frac{R_{pkg}}{2}\right)}{\left(\frac{L_{pkg}}{\Delta t}+\frac{R_{pkg}}{2}\right)}$$

IBIS-LIM Solution



$$C_{comp}\frac{\left(V_{comp}^{n+1/2}-V_{comp}^{n-1/2}\right)}{\Delta t}+\frac{G_{comp}}{2}\left(V_{comp}^{n+1/2}+V_{comp}^{n-1/2}\right)=-I_{out}^{n}-I_{dev}^{n}$$

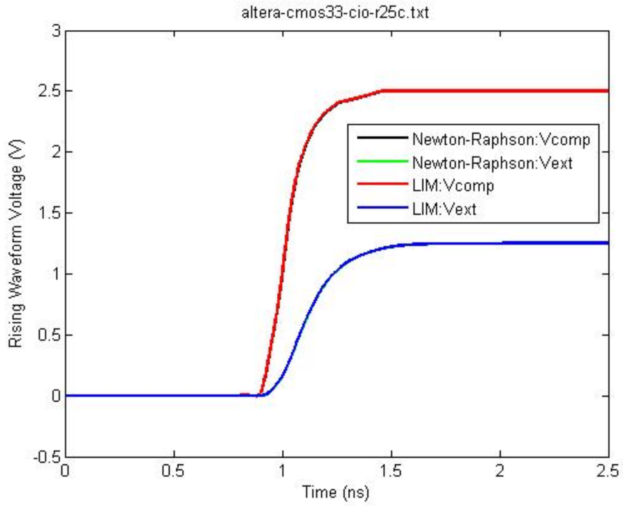
Explicit equations

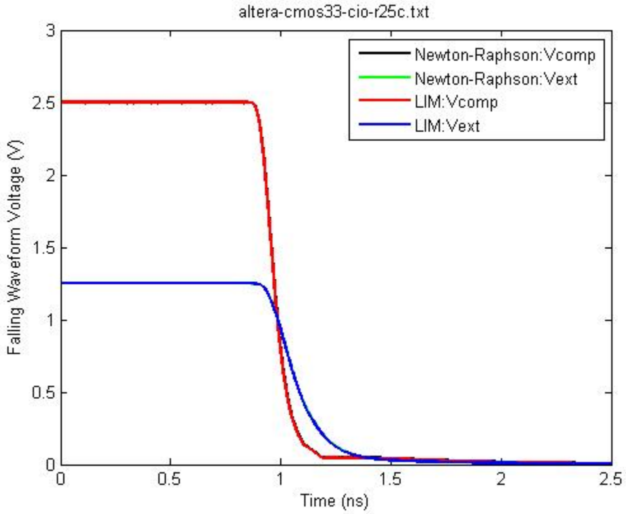
$$V_{comp}^{n+1/2}=\frac{-I_{out}^{n}-I_{dev}^{n}+\left(\frac{C_{comp}}{\Delta t}-\frac{G_{comp}}{2}\right)V_{comp}^{n-1/2}}{\left(\frac{C_{comp}}{\Delta t}+\frac{G_{comp}}{2}\right)}$$

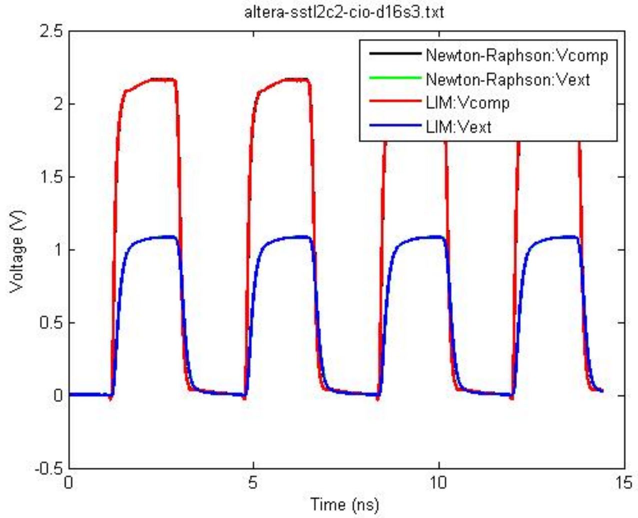
$$I_{dev}^{n}=K_{u}I_{pu}\left(V_{comp}\right)+K_{d}I_{pd}\left(V_{comp}\right)+I_{pc}\left(V_{comp}\right)+I_{gc}\left(V_{comp}\right)$$

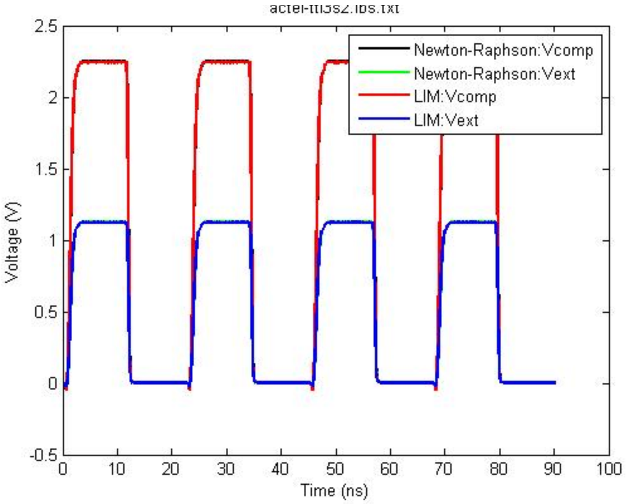
Transient Simulation Examples

NR and LIM give same results…





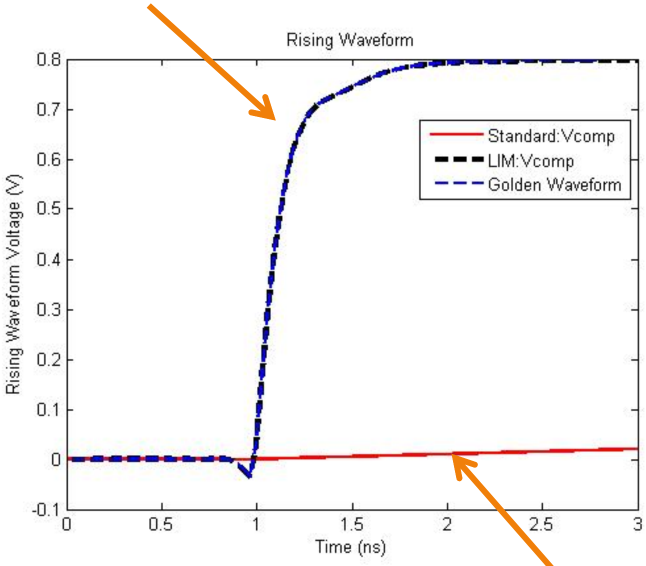




Transient Simulation Examples

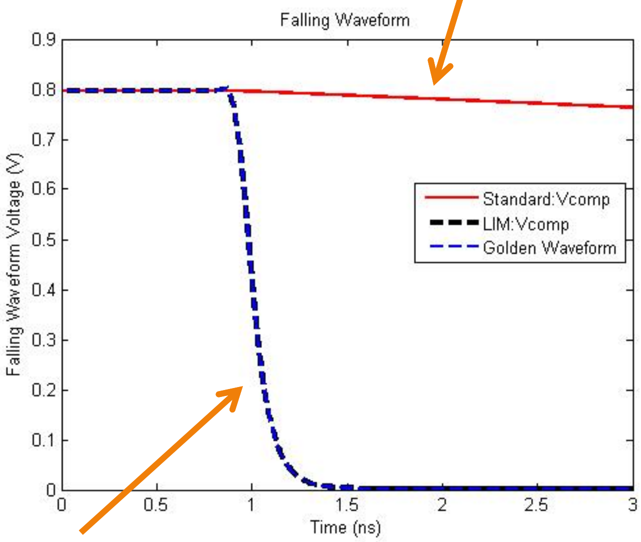
… in some cases Newton-Raphson fails to converge…

LIM



NR: failed convergence

NR: failed convergence

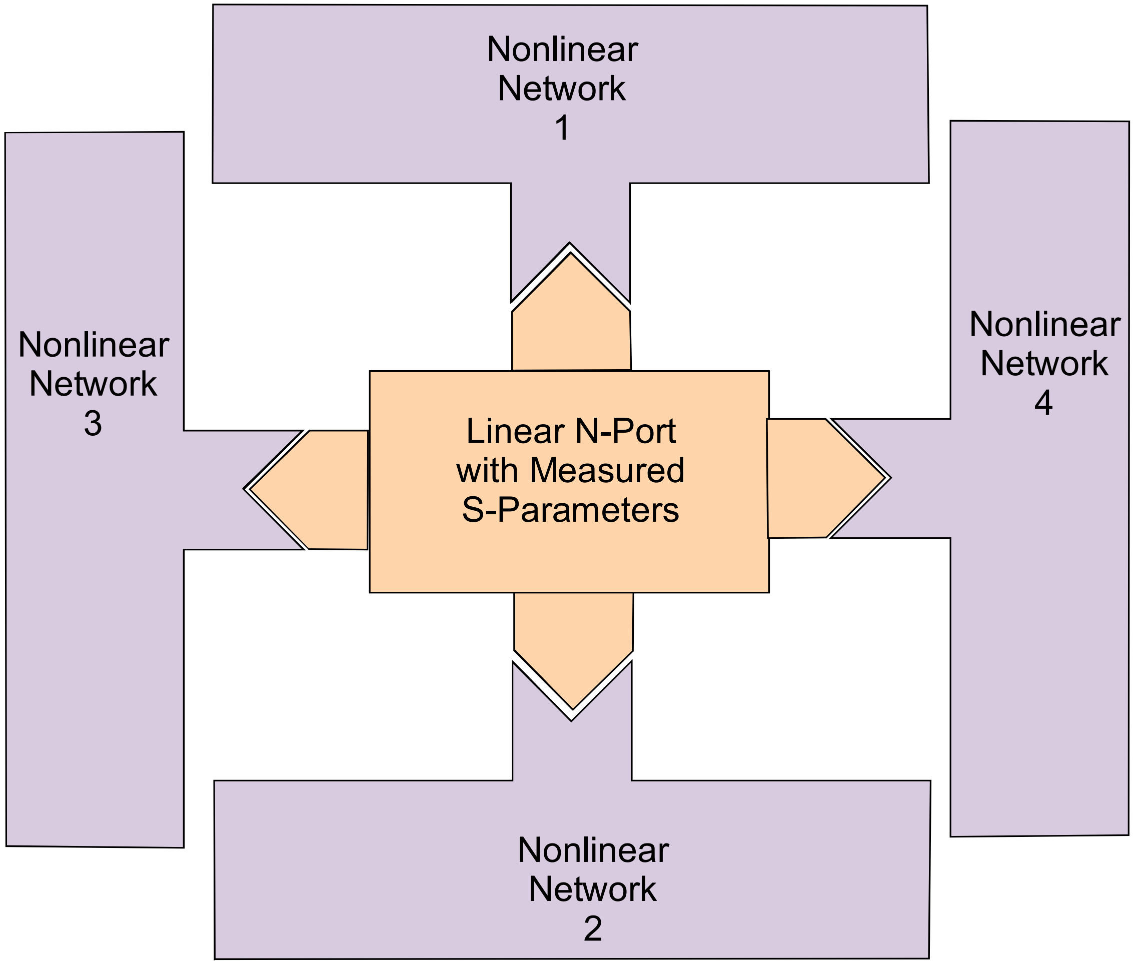


LIM

LIM and IBIS

- Demonstrated
- BIRDs 98 and 95 implemented
- Need to integrate latest improvements
- Need to integrate AMI capability

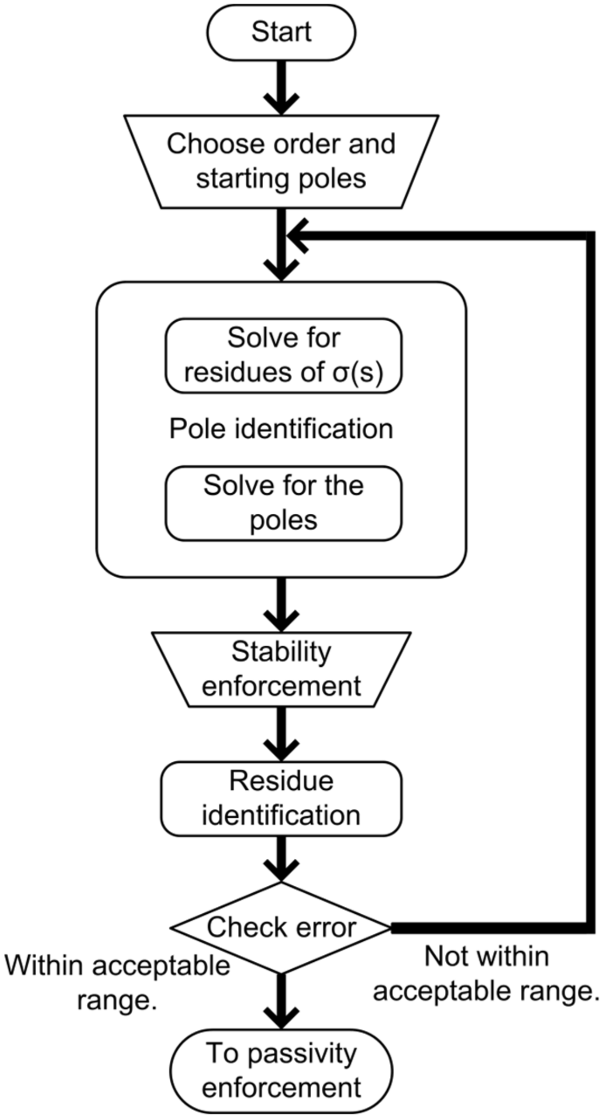
Blackbox Macromodeling



Objective:

Perform time-domain simulation of composite network to determine timing waveforms, noise response or eye diagrams

MOR via Vector Fitting



Start

Choose order and starting poles

Solve for residues of σ(s)

Pole identification

Solve for the poles

Stability enforcement

Residue identification

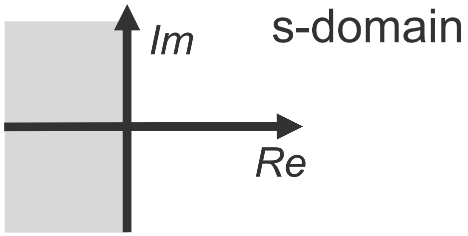
Check error

Within acceptable range.

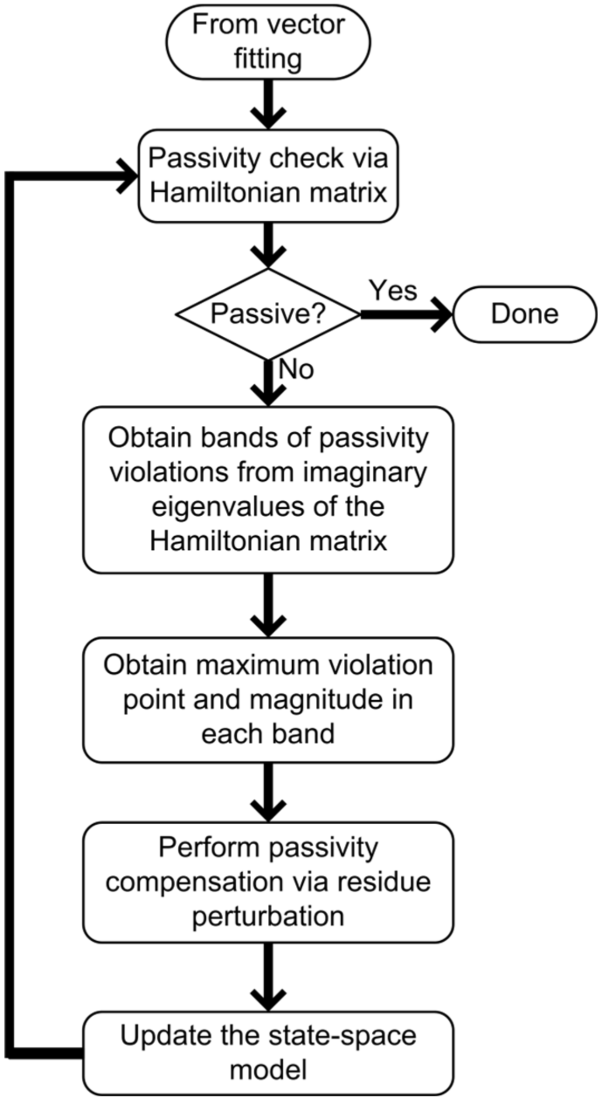
Not within acceptable range.

To passivity enforcement

- Rational function approximation: $$f(s) \approx \sum_{n=1}^{N} \frac{c_n}{s-a_n} + d + sh$$
- Introduce an unknown function *σ(s)* that satisfies: $$\begin{bmatrix} \sigma(s) f(s) \\ \sigma(s) \end{bmatrix} \approx \begin{bmatrix} \displaystyle\sum_{n=1}^{N} \frac{c_n}{s-\tilde{a}_n} + d + sh \\ \displaystyle\sum_{n=1}^{N} \frac{\tilde{c}_n}{s-\tilde{a}_n} + 1 \end{bmatrix}$$
- Poles of *f(s)* = zeros of *σ(s)*: $$f(s) \approx \frac{\displaystyle\sum_{n=1}^{N} \frac{c_n}{s-\tilde{a}_n} + d + sh}{\displaystyle\sum_{n=1}^{N} \frac{\tilde{c}_n}{s-\tilde{a}_n} + 1} = \frac{\displaystyle\prod_{n=1}^{N+1} (s - z_n)}{\displaystyle\prod_{n=1}^{N} (s - \tilde{z}_n)}$$
- Flip unstable poles into the left half plane.



Passivity Enforcement



From vector fitting

Passivity check via Hamiltonian matrix

Passive?

Yes

Done

No

Obtain bands of passivity violations from imaginary eigenvalues of the Hamiltonian matrix

Obtain maximum violation point and magnitude in each band

Perform passivity compensation via residue perturbation

Update the state-space model

- State-space form:

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

- Hamiltonian matrix:

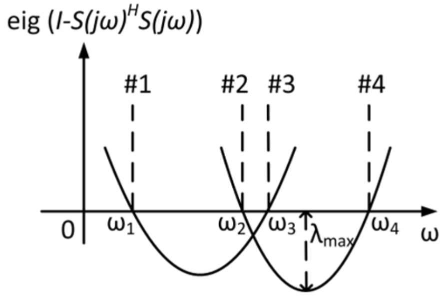
$$M = \begin{bmatrix} A + BKD^T C & BKB^T \\ -C^T LC & -A^T - C^T DKB^T \end{bmatrix}$$

$$K = \left(I - D^T D\right)^{-1} \quad L = \left(I - DD^T\right)^{-1}$$

- Passive if *M* has no imaginary eigenvalues.

- Sweep:

$$eig\left(I - S(j\omega)^H S(j\omega)\right)$$

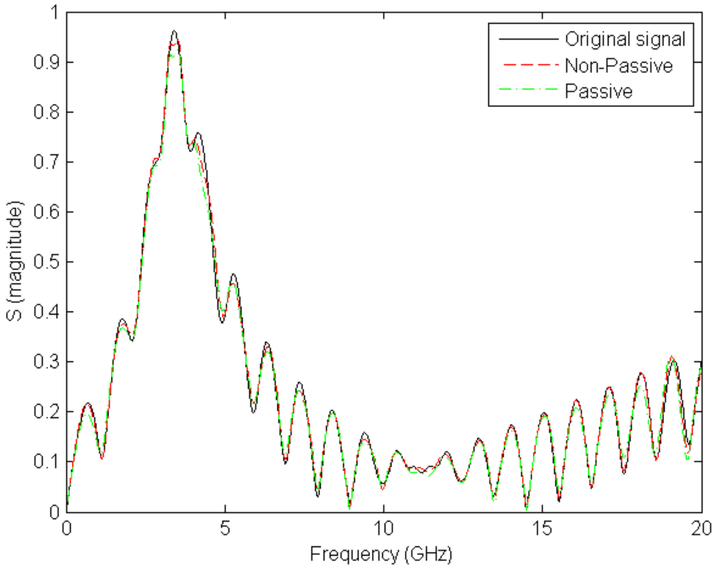


- Quadratic programming:
  - Minimize *(change in response)* subject to *(passivity compensation).*

$$\min\left(vec(\Delta C)^T H\, vec(\Delta C)\right) \quad \text{subject to} \quad \Delta\lambda = G \cdot vec(\Delta C).$$

Model-Order Reduction

- ➔ Start with S parameters from field solver
- ➔ Use vector fitting to get poles & residues
- ➔ Perform assessment via Hamiltonian
- ➔ Enforcement: Residue Perturbation Method
- ➔ Simulation: Recursive convolution➔Fast



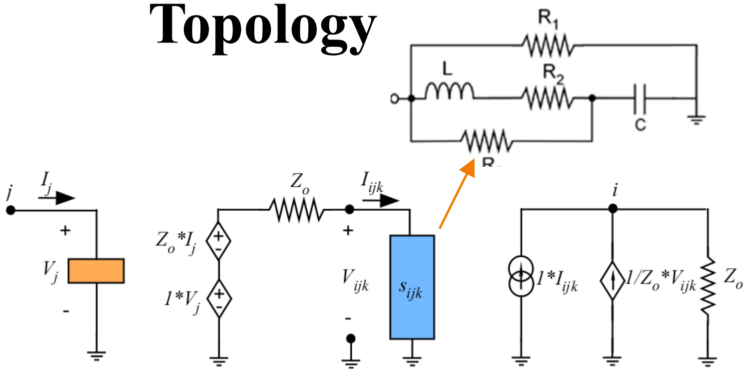
Number of Ports	Order	CPU-Time
4-Port	20	1.7 secs
6-port	32	3.69 secs
10-port	34	8.84 secs
20-port	34	33 secs
40	50	142 secs
80	12	255 secs

SPICE Netlist Synthesis

- Goal is to generate (using pole/residue information) a circuit netlist that will exhibit the same (frequency-dependent) behavior as that of the S-parameters of connector under study

Passive Poles/Residues from MOR

Circuit Cell Topology

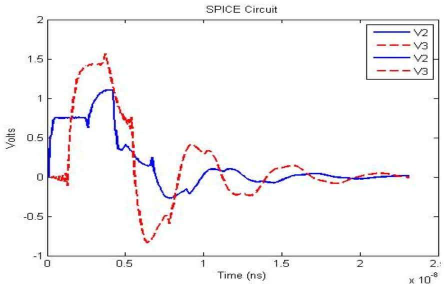


Synthesis Algorithm

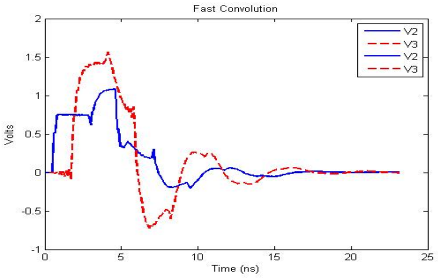
Netlist

Simulation

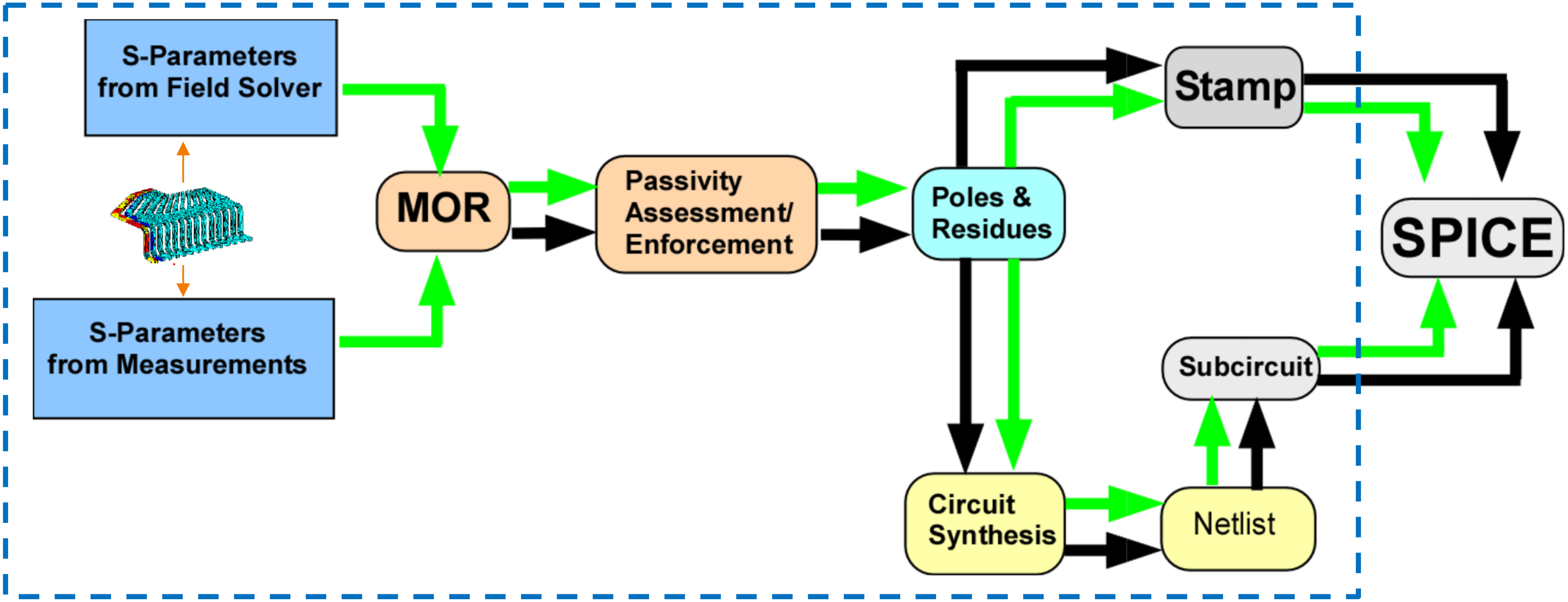
SPICE simulation from MOR generated Netlist



Direct Convolution



Model-Order Reduction


S-Parameters from Field Solver

S-Parameters from Measurements

MOR

Passivity Assessment/ Enforcement

Poles & Residues

Stamp

Circuit Synthesis

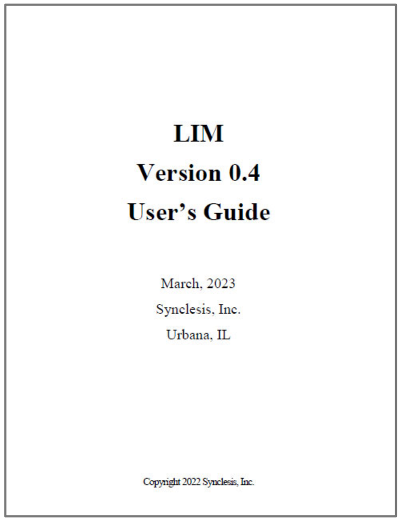
Netlist

Subcircuit

SPICE

- **Objective**: Incorporate frequency dependence into time-domain simulator
- **Approaches**: 1) Direct integration of code into SPICE
  2) Generation of SPICE-compatible netlist

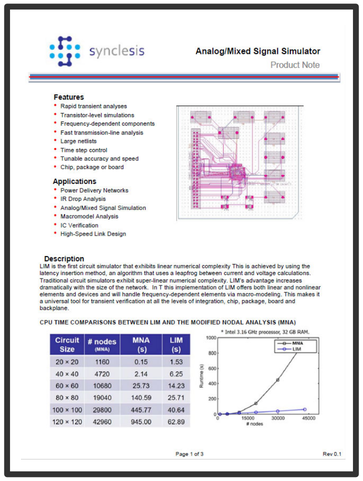
LIM Support



LIM
Version 0.4
User's Guide

March, 2023
Synclesis, Inc.
Urbana, IL

- ➤CSIM interface
- ➤Manual Version 0.4
- ➤Product Note
- ➤Application Notes



synclesis

Analog/Mixed Signal Simulator

Product Note

Features

- Rapid transient analyses
- Transistor-level simulations
- Frequency-dependent components
- Fast transmission-line analysis
- Large netlists
- Time step control
- Tunable accuracy and speed
- Chip, package or board

Applications

- Power Delivery Networks
- IR Drop Analysis
- Analog/Mixed Signal Simulation
- Macromodel Analysis
- IC Verification
- High-Speed Link Design

Description

LIM is the first circuit simulator that exhibits linear numerical complexity This is achieved by using the latency insertion method, an algorithm that uses a leapfrog between current and voltage calculations. Traditional circuit simulators exhibit super-linear numerical complexity. LIM's advantage increases dramatically with the size of the network. In T this implementation of LIM offers both linear and nonlinear elements and devices and will handle frequency-dependent elements via macro-modeling. This makes it a universal tool for transient verification at all the levels of integration, chip, package, board and backplane.

CPU TIME COMPARISONS BETWEEN LIM AND THE MODIFIED NODAL ANALYSIS (MNA)

Circuit Size	# nodes (MNA)	MNA (s)	LIM (s)
20 × 20	1160	0.15	1.53
40 × 40	4720	2.14	6.25
60 × 60	10680	25.73	14.23
80 × 80	19040	140.59	25.71
100 × 100	29800	445.77	40.64
120 × 120	42960	945.00	62.89

* Intel 3.16 GHz processor, 32 GB RAM.

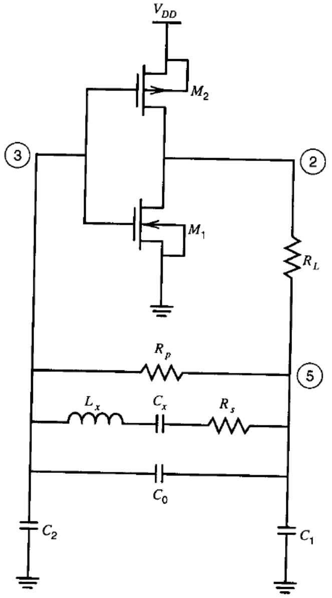
Page 1 of 3

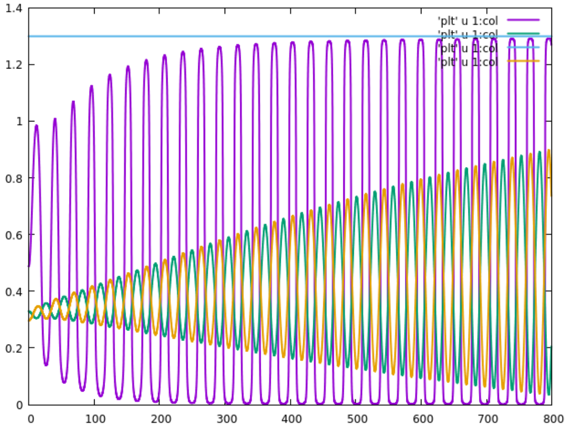
Rev 0.1

LIM Format

➢SPICE 3 Syntax

```
*PIERCE XTAL OSCILLATOR WITH CMOS
M1 2 3 0 0 NMOS W=40U L=10U
M2 2 3 1 1 PMOS W=80U L=10U
RL 2 5 10000
C1 5 0 22.0e-12
C2 3 0 22.0e -12
RP 3 5 22.0e+06
LEFF 3 5 0.18e -03
VDD 1 0 5
.MODEL NMOS NMOS LEVEL=1 VTO=1 KP=20U
LAMBDA=0.02
.MODEL PMOS  PMOS LEVEL=1 VTO= -1 KP=10U
LAMBDA=0.02
.tran 0.001e-09   6793.0e-09e-08
.LIM C=0.015e-12 L=0.1e-09 G=1.0e-20
.PRINT TRAN V(2) V(3) V(1) V(5)
.PLOT TRAN V(2) V(3) V(1) V(5)
.END
```

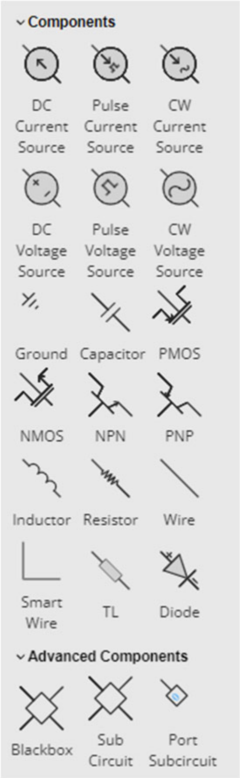


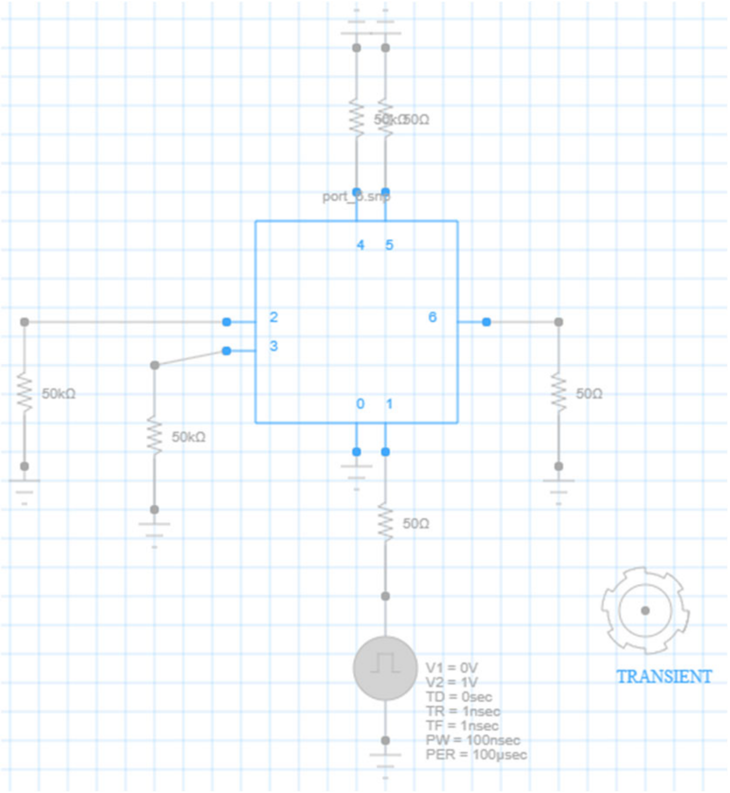


CSIM – User Interface

Front-end currently supports general SPICE elements:

- Passive elements: R, L, C
- Independent sources, diodes, transistors
- Sub-circuit (multi-stage: subckts inside subckts)
- Network parameter (S-parameter blackbox)





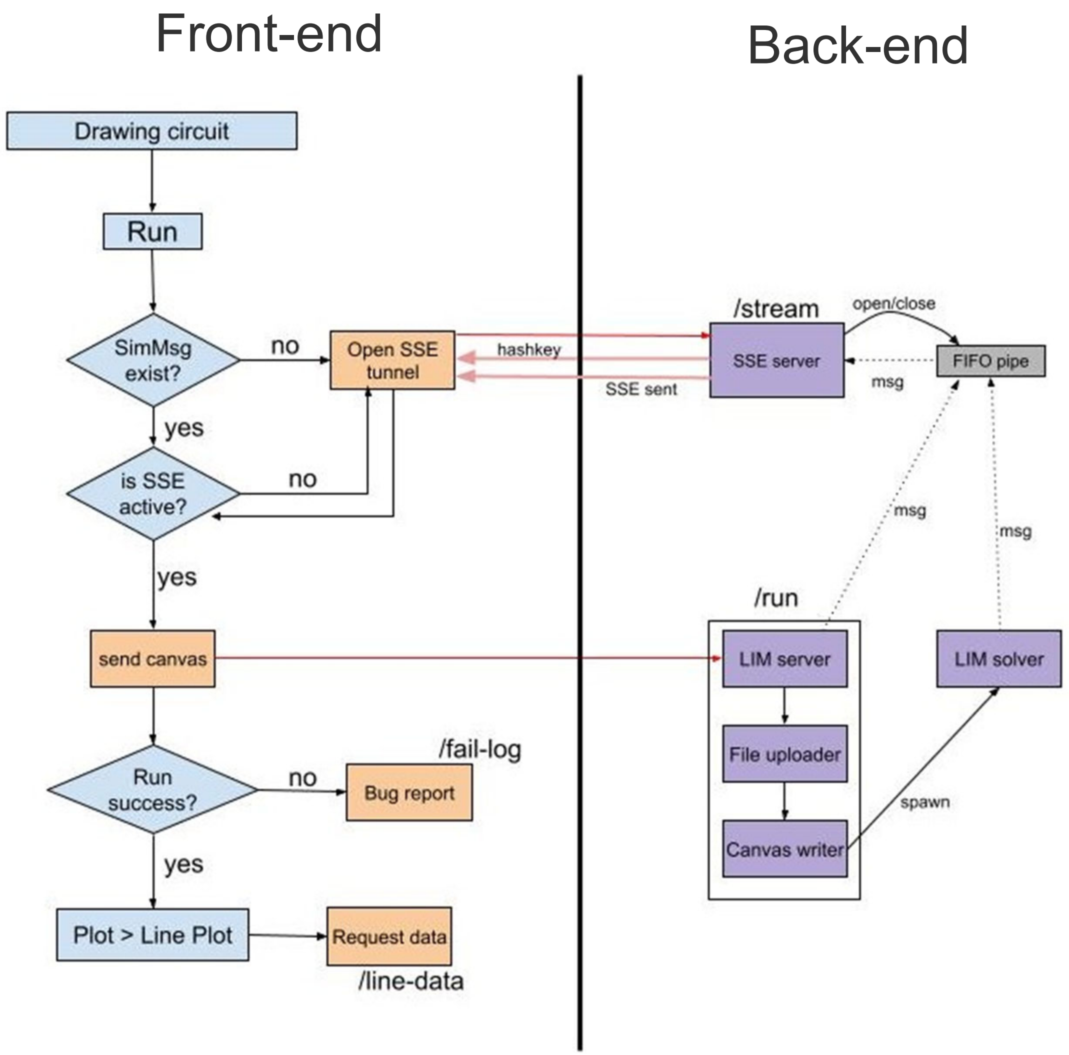
V1 = 0V
V2 = 1V
TD = 0sec
TR = 1nsec
TF = 1nsec
PW = 100nsec
PER = 100μsec

CSIM – User Interface

Full-stack overview

Overall flowchart shows on the left:

- **Front-end uses React for UI**
- **Back-end uses NodeJS interfacing LIM engine**



LIM/CSIM – Macromodel Example

Pole/Residue Formatting

```
SDATA_RX5.s4p_poles_and_residues   4-port S-parameter circuit model
185-pole approximation
 92 complex pole pairs
 0 real poles
================= POLES ===========================
 1 - complex:  -1.4132e+09 2.4987e+11
 2 - complex:  -1.4132e+09 -2.4987e+11
 3 - complex:  -1.1458e+09 2.4669e+11
 4 - complex:  -1.1458e+09 -2.4669e+11
:
---------residues for s[1][1]------------------------
 1 - complex:  5.2798e+08 -2.6935e+08
 2 - complex:  5.2798e+08 2.6935e+08
 3 - complex:  1.2583e+08 1.5229e+08
 4 - complex:  1.2583e+08 -1.5229e+08
 5 - complex:  -1.2293e+08 -9.7733e+07
 :
```

Circuit Formatting

```
*SDATA/RX5.s4p   4-port S-parameter circuit model
*185 -pole approximation

.subckt SUBCIRCUIT  925000  1110000  1295000  1480000
vsens925001 925000 925001 0.0
vsens1110001 1110000 1110001 0.0
vsens1295001 1295000 1295001 0.0
vsens1480001 1480000 1480001 0.0

*subcircuit for s[1][1]
*complex residue-pole pairs for S[1][1] at k=  1 -> 1st pole: -1.4132e+00 2.4987e+02 residue: 5.2798e-01 -2.6935e-01
*                                              -> 2nd pole: -1.4132e+00 -2.4987e+02 residue: 5.2798e-01 2.6935e-01
*circuit type =   9
elc1  1 0 925001 0 1.0
hc2  2  1 vsens925001 50.0
rtersc3  2  3 50.0
vp4  3  4 0.0
r1cd5  4  0 5.02185e+01
l1cd5  4  5 -1.86769e-08
r2cd6  5  6 -2.49907e+03
c1cd6  6  0 -6.69636e-16
r3cd6  4  6 1.14941e+04

*complex residue-pole pairs for S[1][1] at k=  2 -> 1st pole: -1.1458e+00 2.4669e+02 residue: 1.2583e-01 1.5229e-01
*                                              -> 2nd pole: -1.1458e+00 -2.4669e+02 residue: 1.2583e-01 -1.5229e-01
*circuit type =   9
elc7  7 0 925001 0 1.0
hc8  8  7 vsens925001 50.0
rtersc9  8  9 50.0
vp10  9 10 0.0
;
```

Conclusions

- LIM can be used to simulate IBIS-based circuits with optimum accuracy.
- LIM can handle macromodels with Vector fitting and produces a circuit netlist.
- CSIM is the LIM interface
- LIM and CSIM are under Development
- LIM is available for beta testing