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

<table>
<thead>
<tr>
<th></th>
<th>Vendor_0</th>
<th>Vendor_1</th>
<th>Vendor_2</th>
<th>Vendor_3</th>
<th>Vendor_4</th>
<th>Vendor_5</th>
<th>LIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency Dependence</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>Uses MNA</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>BSIM-CMG</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
<td>NOT YET</td>
</tr>
</tbody>
</table>

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

\[ I_{ij}^n \rightarrow V_{i}^{n+1/2} \rightarrow I_{ij}^{n+1} \rightarrow V_{i}^{n+3/2} \rightarrow I_{ij}^{n+2} \rightarrow V_{i}^{n+5/2} \]

\[ V_{i}^{n+1/2} = \frac{C_{i}V_{i}^{n-1/2}}{\Delta t} + H_{i}^{n} - \sum_{k=1}^{N_{a}} I_{ik}^{n} \]

\[ I_{ij}^{n+1} = I_{ij}^{n} + \frac{\Delta t}{L_{ij}} \left( V_{i}^{n+1/2} - V_{j}^{n+1/2} - R_{ij}I_{ij}^{n} \right) \]

Leapfrog method achieves second-order accuracy, i.e., error is proportional to \( \Delta t^2 \)
**VinC – Voltage in Current**

VinC formulation achieves higher accuracy and stability

\[ V_{i}^{n+1} = \frac{C_{i}V_{i}^{n-1} + H_{i}^{n+1/2} - \sum_{k=1}^{N_{a}} I_{ik}^{n+1/2}}{\Delta t} = \Gamma V_{i}^{n} + Z_{ni}H_{i}^{n+1/2} - Z_{ni}\sum_{k=1}^{N_{a}} I_{ik}^{n+1/2} \]

\[ I_{ij}^{n+1/2} = I_{ij}^{n-1/2} + \frac{\Delta t}{L_{ij}} \left( V_{i}^{n+1} - V_{j}^{n+1} - R_{ij}I_{ij}^{n-1/2} \right) \]

\[ I_{ij}^{n+1/2} = I_{ij}^{n-1/2} + \frac{\Delta t}{L_{ij}} \left( \Gamma V_{i}^{n} + Z_{ni}H_{i}^{n+1/2} - Z_{ni}\sum_{k=1, k \neq j}^{N_{a}} I_{ik}^{n+1/2} - \Gamma_{j}V_{j}^{n} - Z_{nj}H_{j}^{n+1/2} + Z_{nj}\sum_{k=1, k \neq j}^{N_{a}} I_{jk}^{n+1/2} - R_{ij}I_{ij}^{n-1/2} \right) \]

\[ \left[ 1 + \frac{Z_{ni}}{L_{ij}} + \frac{Z_{nj}}{L_{ij}} \right] \]

## LIM vs SPICE

### Simulation Times

<table>
<thead>
<tr>
<th>No of Nodes</th>
<th>20,000</th>
<th>30,000</th>
<th>40,000</th>
<th>50,000</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SPICE</strong> (sec)</td>
<td>1224</td>
<td>2935</td>
<td>4741</td>
<td>7358</td>
</tr>
<tr>
<td><strong>LIM</strong> (sec)</td>
<td>9</td>
<td>13</td>
<td>17</td>
<td>21</td>
</tr>
<tr>
<td><strong>Speedup</strong></td>
<td>136</td>
<td>225</td>
<td>278</td>
<td>350</td>
</tr>
</tbody>
</table>

![Graph showing LIM vs SPICE speedup](image-url)
Example – RLG C Grid

• Comparison of runtime for LIM and Vendor_3.

<table>
<thead>
<tr>
<th>Circuit Size</th>
<th># nodes</th>
<th>VENDOR_3 (s)</th>
<th>LIM (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20 × 20</td>
<td>400</td>
<td>0.15</td>
<td>1.53</td>
</tr>
<tr>
<td>40 × 40</td>
<td>1600</td>
<td>2.14</td>
<td>6.25</td>
</tr>
<tr>
<td>60 × 60</td>
<td>3600</td>
<td>25.73</td>
<td>14.23</td>
</tr>
<tr>
<td>80 × 80</td>
<td>6400</td>
<td>140.59</td>
<td>25.71</td>
</tr>
<tr>
<td>100 × 100</td>
<td>10000</td>
<td>445.77</td>
<td>40.64</td>
</tr>
<tr>
<td>120 × 120</td>
<td>14400</td>
<td>945.00</td>
<td>62.89</td>
</tr>
</tbody>
</table>

• LIM exhibits linear numerical complexity!
  – Outperforms conventional SPICE-like simulators.
LIM – Simulation of Large Circuits

- 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
Large Circuit Results*

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

Simulator Flow

NETLIST ELEMENTS

Sources ➔ Branches ➔ Pseudo Branches ➔ Devices ➔ Node

DC Analysis

Transient Analysis

Calculation ➔ Output
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

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

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

**MACROMODELS**
- Model-Order Reduction
- Convolution Model

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

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

- **Done**
- In Development
IBIS-LIM Formulation

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

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

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

\[ I_{\text{out}}^{n+1} = \frac{\left( V_{\text{comp}}^{n+1/2} - V_{\text{ext}}^{n+1/2} \right) + I_{\text{out}}^n \left( \frac{L_{\text{pkg}}}{\Delta t} - \frac{R_{\text{pkg}}}{2} \right)}{\left( \frac{L_{\text{pkg}}}{\Delta t} + \frac{R_{\text{pkg}}}{2} \right)} \]
IBIS-LIM Solution

Explicit equations

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

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

\[
I_{dev}^n = K_u I_{pu} \left( V_{comp} \right) + K_d I_{pd} \left( V_{comp} \right) + I_{pc} \left( V_{comp} \right) + I_{gc} \left( V_{comp} \right)
\]
Transient Simulation Examples

NR and LIM give same results...
Transient Simulation Examples

... in some cases Newton-Raphson fails to converge...

NR: failed convergence

LIM

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**

- **Rational function approximation:**

  \[ f(s) \approx \sum_{n=1}^{N} \frac{c_n}{s - a_n} + d + sh \]

- **Introduce an unknown function** \( \sigma(s) \) **that satisfies:**

  \[ \begin{bmatrix} \sigma(s)f(s) \\ \sigma(s) \end{bmatrix} \approx \begin{bmatrix} \sum_{n=1}^{N} \frac{c_n}{s - \tilde{a}_n} + d + sh \\ \sum_{n=1}^{N} \frac{\tilde{c}_n}{s - \tilde{a}_n} + 1 \end{bmatrix} \]

- **Poles of** \( f(s) \) **= zeros of** \( \sigma(s) \):

  \[ f(s) \approx \sum_{n=1}^{N} \frac{c_n}{s - \tilde{a}_n} + d + sh = \prod_{n=1}^{N+1} \frac{s - z_n}{s - \tilde{z}_n} \]

- **Flip unstable poles into the left half plane.**
Passivity Enforcement

- State-space form:
  \[ \dot{x} = Ax + Bu \]
  \[ y = Cx + Du \]

- Hamiltonian matrix:
  \[ M = \begin{bmatrix}
  A + BKD^T C & BKB^T \\
  -C^T LC & -A^T - C^T DKB^T
\end{bmatrix} \]

- Passive if \( M \) has no imaginary eigenvalues.

- Sweep:
  \[ \text{eig} \left( I - S(j\omega)^H S(j\omega) \right) \]

- Quadratic programming:
  - Minimize \( \text{(change in response)} \) subject to \( \text{(passivity compensation)} \).

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

<table>
<thead>
<tr>
<th>Number of Ports</th>
<th>Order</th>
<th>CPU-Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-Port</td>
<td>20</td>
<td>1.7 secs</td>
</tr>
<tr>
<td>6-port</td>
<td>32</td>
<td>3.69 secs</td>
</tr>
<tr>
<td>10-port</td>
<td>34</td>
<td>8.84 secs</td>
</tr>
<tr>
<td>20-port</td>
<td>34</td>
<td>33 secs</td>
</tr>
<tr>
<td>40</td>
<td>50</td>
<td>142 secs</td>
</tr>
<tr>
<td>80</td>
<td>12</td>
<td>255 secs</td>
</tr>
</tbody>
</table>
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
```

```
SPICE simulation from MOR generated Netlist
Direct Convolution
```

```
Simulation
Netlist
```

IBIS Summit, Aveiro, 2023
Model-Order Reduction

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

- CSIM interface
- Manual Version 0.4
- Product Note
- Application Notes
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)
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
e1c1 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
e1c7 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