Fast PDN Impedance Prediction Using Deep Learning

Ling Zhang\textsuperscript{1}, Jack Juang\textsuperscript{1}, Zurab Kiguradze\textsuperscript{1}, Bo Pu\textsuperscript{1}, Shuai Jin\textsuperscript{2}, Songping Wu\textsuperscript{2}, Zhiping Yang\textsuperscript{2}, Jun Fan\textsuperscript{1}, Chulsoon Hwang\textsuperscript{1}

\textsuperscript{1}Missouri University of Science and Technology
\textsuperscript{2}Google Inc., Mountain View, CA

Aug 12, 2021
Speaker Introduction

Ling Zhang
Postdoc at Zhejiang University, Hangzhou, China
2015~2021: PhD student at Missouri S&T

Co-authors:
Jack Juang, Missouri S&T
Zurab Kiguradze, Missouri S&T
Bo Pu, Missouri S&T
Shuai Jin, Google Inc.
Songping Wu, Google Inc.
Zhiping Yang, Google Inc.
Jun Fan, Missouri S&T
Chulsoon Hwang, Missouri S&T
Power Distribution Network (PDN)

- PDN design is important to reduce voltage supply noise caused by IC switching current and ensure power integrity for IC.
- Decoupling capacitors (decaps) are utilized to reduce PDN impedance so as to reduce voltage ripples.
PDN Impedance Modeling

Cavity model [Kim et al. 2010]

\[ L_y = \frac{\mu d}{ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{E_m^2 E_n^2}{k_{mn}^2} f(x_i, y_i, x_j, y_j) \]

- Cavity model: can only handle rectangle power plane shapes

Plane-Pair PEEC (PPP) Model [Wei et al. 2016]

- PPP: Relatively time-consuming

A fast calculation approach for multi-layer PDN with irregular board shapes is desired!!!
Train a deep learning model that can predict the PDN impedance given any:

- Board shapes
- Stackup
- IC location
- Decap placement
A boundary integration method is adopted to calculate PDN impedance for arbitrary shape and stackup from DC to AC.

The boundary integration method is much faster than full-wave simulations.

Generate Random Shape & Stackup

Generate random 2D shape

Generate random stackup

- https://stackoverflow.com/questions/50731785/create-random-shape-contour-using-matplotlib
# Decap Library

<table>
<thead>
<tr>
<th>Type #</th>
<th>Capacitance (uF)</th>
<th>ESL (nH)</th>
<th>ESR (mΩ)</th>
<th>Serial number (Murata)</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.1</td>
<td>0.19</td>
<td>34.7</td>
<td>GRM033C80J104KE84</td>
<td>0201</td>
</tr>
<tr>
<td>2</td>
<td>0.47</td>
<td>0.18</td>
<td>18.3</td>
<td>GRM033R60J474KE90</td>
<td>0201</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0.22</td>
<td>15.2</td>
<td>GRM155B31C105KA12</td>
<td>0402</td>
</tr>
<tr>
<td>4</td>
<td>2.2</td>
<td>0.20</td>
<td>7.2</td>
<td>GRM155C70J225KE11</td>
<td>0402</td>
</tr>
<tr>
<td>5</td>
<td>4.7</td>
<td>0.28</td>
<td>7.1</td>
<td>GRM185C81A475KE11</td>
<td>0603</td>
</tr>
<tr>
<td>6</td>
<td>10</td>
<td>0.26</td>
<td>5.2</td>
<td>GRM188R61A106KAAL</td>
<td>0603</td>
</tr>
<tr>
<td>7</td>
<td>22</td>
<td>0.27</td>
<td>4.0</td>
<td>GRM188B30J226MEA0</td>
<td>0603</td>
</tr>
<tr>
<td>8</td>
<td>47</td>
<td>0.15</td>
<td>2.9</td>
<td>GRM219D80E476ME44</td>
<td>0805</td>
</tr>
<tr>
<td>9</td>
<td>220</td>
<td>0.41</td>
<td>1.9</td>
<td>GRM31CR60J227ME11</td>
<td>1206</td>
</tr>
<tr>
<td>10</td>
<td>330</td>
<td>0.46</td>
<td>1.2</td>
<td>GRM32EC80E337ME05</td>
<td>1210</td>
</tr>
</tbody>
</table>
Matrix Representation

- **Board contour**
- **Board shape + IC location**
- **Stackup**
  - **1D matrix**

- **Top decaps**
- **Bottom decaps**

- Board information: 16×16 matrices
- Stackup (4~9 layers): 1×17 matrix
- Maximum size: 200mm×200mm; number of decap locations: 20
- IC and decap locations are generated randomly
- One unit contains one horizontal decap

CNN Training

Convolutional neural network (CNN) structure:

• 1.3 million board data are generated, 10,000 used for testing
• Output: dB value is used
• Loss function: root mean square error (RMSE)
• Learning rate: 0.0001; Adam optimizer; batch size 128
• Training time: 80 hours (1 NVIDIA Tesla K80 GPU)

Test Trained Model

- RMSE for the testing data is just around 1dB
- Two testing cases are randomly picked and plotted here

<table>
<thead>
<tr>
<th>Methods</th>
<th>Case #1</th>
<th>Case #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full-wave</td>
<td>35 min</td>
<td>40 min</td>
</tr>
<tr>
<td>BEM</td>
<td>10 s</td>
<td>30 s</td>
</tr>
<tr>
<td>DNN</td>
<td><strong>0.1 s</strong></td>
<td><strong>0.1 s</strong></td>
</tr>
</tbody>
</table>

Summary

• A deep learning model is developed to fast predict the PDN impedance for arbitrarily-shaped power plane and arbitrary stackup
• The trained model can predict PDN impedance within 0.1s with a tolerable accuracy
• Code link on GitHub: https://github.com/lingzhang0319/PDN-Impedance-Prediction-Using-Deep-Learning/tree/master
Ling Zhang
Zhejiang University; Missouri S&T
lingzhang_zju@zju.edu.cn; lzd76@mst.edu

Chulsoo Hwang
Missouri S&T
hwangc@mst.edu