Spring 2023

Improved Decap Model and Placement Optimization Algorithm for Power Distribution Network Design

Jack Juang
Missouri University of Science and Technology

Department:

IMPROVED DECAP MODEL AND PLACEMENT OPTIMIZATION ALGORITHM FOR POWER DISTRIBUTION NETWORK DESIGN

by

JACK JUANG

A THESIS

Presented to the Graduate Faculty of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

in

ELECTRICAL ENGINEERING

2023

Approved by:

Dr. Chulsoon Hwang, Advisor
Dr. Daryl Beetner
Dr. DongHyun Kim

PUBLICATION THESIS OPTION

This thesis consists of the following two articles, formatted in the style used by the Missouri University of Science and Technology:

Paper I, found on pages 4–24, has been published in 2021 IEEE International Joint EMC/SI/PI and EMC Europe Symposium

Paper II, found on pages 25–59, has been submitted to IEEE Transactions on Signal and Power Integrity

Paper III, found on pages 60 - 82, has been accepted to DesignCon 2023

ABSTRACT

Decoupling capacitors (decaps) are used in power distribution network (PDN) design to act as a low impedance return path for noise and act as a local source of charge when required by integrated circuits (IC). For placement of decaps, which can number even to the hundreds, many algorithms have been proposed including genetic algorithms (GA), iterative algorithm, and machine learning methods.

One limitation of iterative decap placement algorithms (adding one decap at a time) is how the construction of the algorithm affects the form of the final solution. For example, if the reward function (used in GA and machine learning methods) for evaluating a solution was based on maximizing the number of points below the target impedance, then larger package size decaps may be placed first, as low impedance points are easier to bring below the target using fewer number of decaps. These decaps may also be placed in locations near the IC for the benefit of low loop inductance. The expectation, however, should be for small package decaps to be placed near the IC as they contribute less inductance due to their smaller geometry. In this work, we propose two GAs for minimizing the required number of decaps in PDN design, with the goal of minimizing the number of assumptions about the structure of the solution within the genetic operators.

For the second topic, for capacitors mounted to a board, there is a mutual coupling between the capacitor and the return plane that is stack-up dependent and so cannot be captured in one measurement/model. We propose a physics-based curve fitting scheme to interpolate inductances over many stack-ups in just two measurements.

ACKNOWLEDGMENTS

I would like to thank my EMC lab mates for their support and friendship during my studies. I would also like to thank my advisor, Dr. Chulsoon Hwang, for the opportunity. Finally, I would like to thank my family for all their support and all they've done for me.

The proceeding is based upon work supported partially by the National Science Foundation under Grant No. IIP-1916535.

TABLE OF CONTENTS

Page

LIST OF ILLUSTRATIONS

PAPER I Page

PAPER II

LIST OF TABLES

1. INTRODUCTION

Decoupling capacitors (decaps) are used in printed circuit board (PCB) power distribution networks (PDN) to act as a low impedance return path for noise, as well as a local source of charge, on PCB power lines. For large designs, a very large number of decaps may be required to ensure the proper operation of all devices, which directly translates to higher costs. Many algorithms have been proposed to optimize the placement of decaps, which can number up to the hundreds in real design. Oftentimes algorithms make assumptions (though physics-based) on the form of the optimal solution, that is, the solution using the minimum number of decaps, that may not be universally true and results in missing critical solution characteristics and a narrowing of the search space. This may be done indirectly from the reward functions of a genetic algorithm (GA) or the training reward in a machine learning method, or directly such as when decaps are added one at a time and fixed in place.

Regardless of algorithm though, for simulation verification of the PDN design, accurate capacitor models are required for correlation of simulation to real product performance. For example, a capacitor mounted to a board has a mutual coupling between it and the return plane, resulting in a stack-up dependence of the amount of inductance contributed by the capacitor to the full loop inductance, which cannot be captured in a single measurement or model. For the designer, the result is possible over or under designs leading to increased monetary or time cost.

In the first paper, a GA for placing decaps with the goal of minimizing the total number of decaps required to meet the target impedance is proposed. GA optimization in

this work is performed by controlling only the number of decaps present in the population solutions, starting the optimization with the max number of decaps and slowly reducing the number to find the global minimum solution. By focusing only on limiting the decap number, it is hoped that the GA can find good solutions independently, without being biased by the reward function or any other externalities in the GA design. The proposed GA is compared against the results of other published algorithms, with the proposed GA finding comparable if not better solutions. Comparisons are also made between the decap types used and decap placement between algorithms to get a better idea of the effect of algorithm search method on the form of the solution found.

In the second paper, an improved GA is proposed that considers the design's target impedance and board parasitics as a factor in optimization. The effect of the target impedance, as well as board and decap parasitics, on the form of the optimal solution is studied to predict the what the optimal solution would look like. The form of the optimal solution for given inputs should be entirely independent of the algorithm used, and an accurate prediction of the solution form can be used to help direct the search of a GA, or any other optimization algorithm, in a way that would be globally correct. The initial population (solutions) of the GA will be generated with biased weights to immediately direct the GA search to promising search spaces. From the results, these weighted populations, along with new genetic operators, lead to solutions being found faster and improvements to occur faster.

In the third paper, a physics-based curve fitting scheme for capacitor ESL is proposed. Typically, vendor provided capacitor models lack exact detail of the characterization method, such as fixture design, stack-up, etc., and typically only provide

a single model for a given capacitor. As the coupling between capacitor body and return plane results in a stack-up dependent ESL, it is unknown what use cases vendor provided models are best used for. A basic curve fitting scheme is proposed to interpolate ESL over many stack-ups using two measurements, so that the mutual coupling can be accounted for.

PAPER

I. A MODIFIED GENETIC ALGORITHM FOR THE SELECTION OF DECOUPLING CAPACITORS IN PDN DESIGN

ABSTRACT

Decoupling capacitors are used to provide adequate and stable power for integrated circuits in printed circuit boards (PCB). For complicated and large designs, it is difficult to select capacitors to meet voltage ripple limits while also minimizing cost because the search space is too large. In this work, a new genetic algorithm (GA) is proposed for the selection and placement of capacitors to meet a target impedance using as few capacitors as possible. The GA is centered around controlling the number of unused port locations in the GA population solutions, with the result of smoothing out the GA convergence and speeding up the convergence rate. A result comparison is made of the proposed GA against other algorithms and found the GA competitive if not better for the select test cases.

1. INTRODUCTION

In power distribution networks (PDN) for printed circuit boards (PCBs), at higher frequencies, the inductances associated with the voltage regulator module (VRM) and current return paths becomes an increasing source of impedance. This presents significant power delivery issues on current switching events which greatly impacts the performance

of integrated circuits (ICs). A common method to ensure reliable power delivery is defining a target impedance, which is based on the maximum allowable voltage ripple that can be tolerated by devices on the power rail for continued functionality. With increasingly small, dense, and fast designs, meeting the ripple voltage tolerances becomes more difficult.

To reduce the power issues associated with high frequencies, decoupling capacitors (decaps) are used to provide a local source of charge while also providing a lower inductance/impedance return path. The problem is, designs with large numbers of decap ports contain too many placement possibilities. Of all decap patterns, there exists an application-based 'best solution(s).' This may be the pattern that satisfies a target impedance using the minimum number of capacitors or the pattern that minimizes a bill of material cost. Very large search spaces make brute force methods impractical for finding the best solution.

For this decap placement problem, different search methods have been proposed and implemented. Among those is a physics-based method for minimizing inductance [1], iterative methods [2] and machine learning methods to quickly determine the best solution for any input [3][4]. Different genetic algorithms (GA) have also been implemented. [5][6]. Nearly all search or iterative methods though, rely on specific objective functions and/or made assumptions about how the best solution is most easily found. As an example, adding decaps based on the distance to an IC would always use the same decap ports. While these assumptions are based in physics and do make the search efficient by narrowing the search space, the tradeoff is that there is no way to verify that the reduced search space includes the best solution.

In this work, we propose a new GA to find the decap placement that minimizes the number of capacitors required to meet a target impedance. To accomplish this, a new search method of limiting the number of capacitors in GA solutions is introduced. Contrary to traditional GAs, we also use very general fitness functions to avoid directly narrowing the search space. For disambiguation, the proposed GA will also be referred to as the gene suppressed GA.

2. PROPOSED GENETIC ALGORITHM OVERVIEW

First introduced by Holland, genetic algorithms are a class of optimization functions based on the principles of survival of the fittest [7]. Mimicking the process of natural selection, a GA population experiences the familiar pressures of survival fitness (selection), reproduction, and mutation. Iteratively, a GA population goes through cycles, called generations, where the most fit individuals will reproduce. The overall most fit individual, over all generations, is the best solution for the optimization problem. For the proposed algorithm, the code base for the GA is open-source, implemented in Python and freely available from [8]. Full documentation and the full code is available. The general structure for the proposed GA is described in Figure 1. The code for the genetic operators is unchanged.

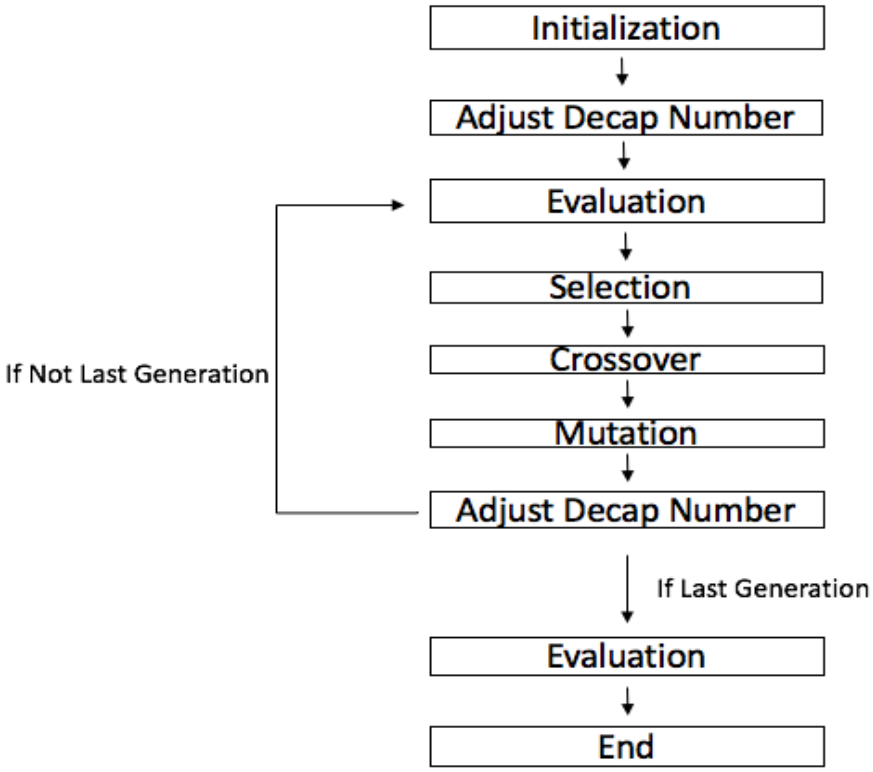

Figure 1. Genetic Algorithm Flowchart

2.1. GENETIC ALGORITHM STRUCTURE

The first stage of the GA is initialization, where the initial population of the GA is generated. Each member of the GA is referred to as a chromosome made up of genes [7]. To fit the GA scheme, each decap placement pattern is encoded as a vector of real, non-negative integers. A decap port location is represented by the vector index, with the length of the vector equal to the total number of ports. The specific capacitor placed at a port location is represented by the value at the corresponding index. An example solution is shown in Figure 2. The mapping of integer numbers to capacitor type is given in Table

1, with an integer value of ‘0’ representing no decap. For initialization, the population was filled by randomly generated solutions.

2	1	10	5	3	4	0	10

Figure 2. Example solution with 8 decaps. Decap number 5 is placed in port 4.

Table 1. Decoupling Capacitor Library

Type #	Decap Parameters		
	Capacitance (uF)	ESL (nH)	ESR (mΩ)
1	0.1	0.19	34.7
2	0.47	0.18	18.3
3	1	0.22	15.2
4	2.2	0.20	7.2
5	4.7	0.28	7.1
6	10	0.26	5.2
7	22	0.27	4.0
8	47	0.15	2.9
9	220	0.41	1.9
10	330	0.46	1.2

The next stage of the GA is evaluation, where the fitness of a solution is judged. In our case, the lower the fitness score given by a fitness function, the higher its fitness. Two different fitness functions are used for evaluating solutions; one for solutions satisfying the target impedance and one for those that don’t. For a solution satisfying the target impedance, the fitness function used is given by (1):

$$Score = -(Total\ \#of\ Ports - \#Used\ Ports + 1) \quad (1)$$

If the target impedance is not satisfied, the fitness score given is proportional to the largest difference between points of the target impedance, the target_z, and the solution_z, the impedance seen looking into an IC on the power rail. For the test cases in this paper, only the PDN AC impedance associated with the vertical vias and plane capacitance is considered and calculated using a BEM and node voltage method [9]; the horizontal routing was not considered. The effect of the thickness of the power and ground layers is assumed negligible on the vertical AC impedance. The algorithm still applies with inputs that consider DC resistance and horizontal routing. The frequency range targeted is 10 kHz to 20 MHz. The fitness score is given by the following fitness function (2):

$$Score = \max\left(\frac{solution\_z(f) - target\_z(f)}{target\_z(f)}\right) \quad (2)$$

To create the next generation, first, selection occurs to choose the parents. A percentage of the current generation, set at 30%, is selected through the roulette wheel method [10] to join the next generation. Equivalently this means the crossover rate, the number of solutions created by crossover, is 70%. These solutions are the potential parents for new solutions. An elitism component [7] is included where a percentage of the highest fitness solutions are guaranteed to join the next generation. The elitism percent is set at 1% with a minimum of 1 solution joining the next generation. The remainder of the population is generated through uniform crossover [11] of randomly selected pairs of the potential parents. Finally, mutation occurs where every gene has a chance of being changed. In our case, the decap placed at a particular port may have its decap type changed or be removed altogether. The mutation rate is set at 10%. After

mutation, one generation is completed. The entire process repeats again with fitness evaluation, generation after generation, until a defined number of generations have passed.

2.2. SOLUTION SIZE AND SIZE VARIATION

A change is proposed here to the traditional GA scheme. A distinction is made between gene value 0 and genes 1 – 10; by our fitness function, more expressions of gene 0 lead to a better score. The frequency of gene 0 appearing will be controlled by the GA to make the search more efficient.

The solution size is defined as the number of decaps used in the current best-known solution. The proposed change is to limit the number of decaps in all solutions around the solution size. With a solution size of 20, there is no need to consider solutions using > 20 decaps so solutions should be restricted to <= 20 decaps. Solutions with 20 decaps are still considered as they may provide alternate search paths for the GA. This parameter is dynamically updated and initially set as the total number of decap locations.

While the upper limit is defined by the solution size, a lower limit is defined by the size variation. Without a lower limit on the number of decaps, the search space may be too large for efficient search. If the size solution is 20 and the size variation is 5, then all solutions in the population are allowed only 20 – 15 decaps inclusively. It is more probable to find solutions nearer to the current solution size number than one with far fewer decaps. The size variation parameter was set at a rounded 10% of the total number of decap ports.

Let *S* be the solution size and *V* the size variation. Figure 3 describes the changes made to a solution with *N* decaps. For adding and removing decaps, the decap ports are randomly chosen from the solution. When adding, decaps are selected from the those already present in the solution. A solution that does not utilize decap number 8 will not have capacitor 8 as an option for adding. This is to avoid changing the overall behavior of the solution too much, such as by adding new resonances. Adjustments to the decap number occur after initialization and after mutation.

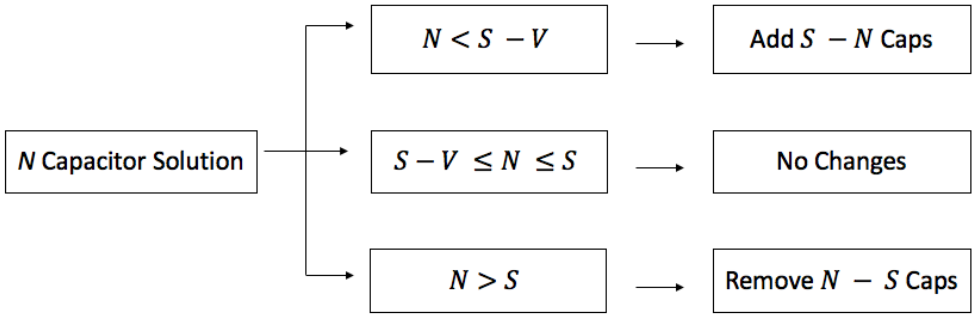

Figure 3. Modifications to be made to a solution of N capacitors, with solution size S and size variation V

3. GA VERIFICATION

To verify the performance of the proposed GA, 3 test boards were generated using code from [12]. The shapes and decap port layouts are given in Figure 4. The stack-ups are given in Tables 2 – 4. The dielectric relative permittivity is 4.4. The thickness of PWR and GND layers is again assumed to have negligible effects on the total via inductances and capacitances compared to dielectric layer thickness. As such, PWR and

GND layers are modeled as having 0mm thickness. Through-vias connect the appropriate layers and are used as decap ports.

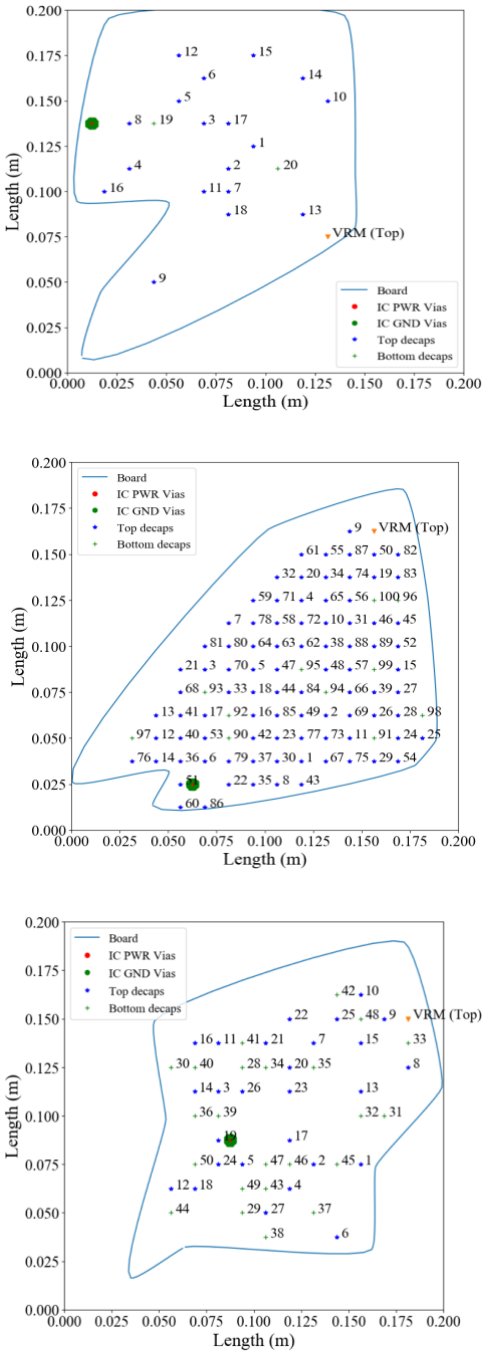

Figure 4. Board Shape and Decap Port Layout, for test cases 1, 2, and 3

Table 2. Stack-up for Case 1

Layer Type (Top Down)	Thickness (mm)
GND	0
Dielectric	0.3
PWR	0
Dielectric	1.7
GND	0
Dielectric	0.7
GND	0

Table 3. Stack-up for Case 2

Layer Type (Top Down)	Thickness (mm)
GND	0
Dielectric	0.4
PWR	0
Dielectric	0.7
GND	0
Dielectric	1.2
GND	0

Test board 1 (Case 1) has 20 capacitors ports. Test board 2 (Case 2) has 100 decap ports and test board 3 (Case 3) has 50. The decap port locations were randomly selected and placed on both top and bottom layers. The VRM was modelled as a series RL circuit with R = 3 mOhms and L = 2.5 nH and placed at the port farthest from the IC, but on the top layer. To set the target impedances for each case, the gene suppressed GA was run repeatedly, and the target impedance varied, until a reasonable number of capacitors that could satisfy the target impedance was found. The impedances are of RL type and are given in Figure 5. Results of the gene suppressed GA are compared against the open-source GA without any modifications (same fitness function), against the method described in [6] in the ideal case, and against the reinforcement learning algorithm described in [3].

Table 4. Stack-up for Case 3

Layer Type (Top Down)	**Thickness (mm)**
GND	0
Dielectric	0.2
GND	0
Dielectric	0.2
GND	0
Dielectric	0.3
PWR	0
Dielectric	0.3
GND	0
Dielectric	0.3
GND	0
Dielectric	0.3
GND	0
Dielectric	0.5
GND	0
Dielectric	0.2
GND	0

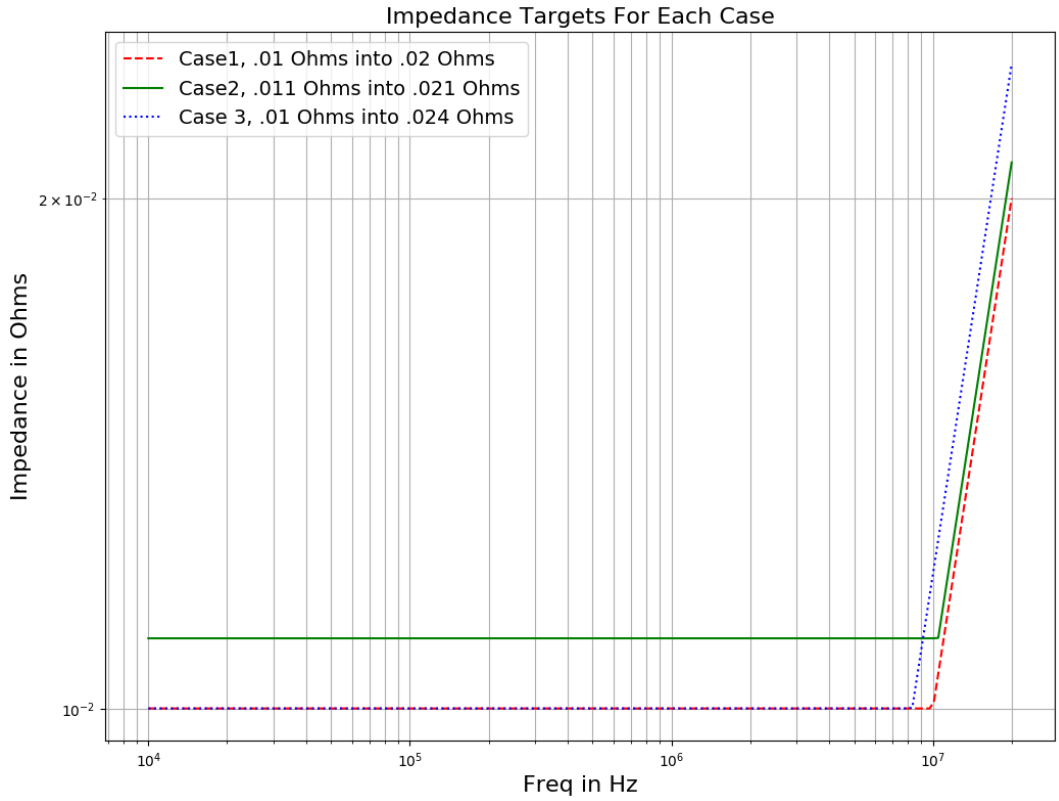

Figure 5. Impedance Targets in Each Case. The Slope is +20dB/decade. The frequency range is 10 kHz to 20 MHz

The unmodified version of the open-source GA is near identical to that of the gene suppressed GA, the only difference is that the number of capacitors allowed is not enforced. To check for search potential and consistency, both GAs' were ran 5 times, for each test case, with the population size and the number of generations = 50. They were rerun another 5 times, for each test case, with population size and the number of generations = 100.

The method proposed in [6] is an iterative GA, but rather than risk a poor recreation, a full search was performed. Decaps were selected one by one, by considering every decap type and location, and fixing the one that best minimizes the cost function.

Decaps are placed until the target is met or until all ports are filled. The solution found then, is the best possible solution based on the cost function of [6].

The reinforcement learning method described in [3] involves training a machine learning model to find the minimal number of decaps needed to satisfy the target impedance. A port sequence is first calculated by [1] to determine the order of ports to be used. The good convergence of the model and its ability to generalize was not considered, only the best solution the that could be found. The algorithm was run three times, for each test case, and the best solution found was recorded.

3.1. GENE SUPPRESSED AND OPEN SOURCE GA

The minimum number of decaps found by the GAs for each case is given in Table 5, along with the average time for each case. The GAs was run on a virtual Linux server with an Intel Xeon Gold 5118 processor. Better solutions are generally found with the gene suppressed GA, with large improvements as the number of decap ports increases. However, the time consumed increases considerably with the # of decap ports, population size, and generation number, making this method currently impractical for large industry design. It may take upwards of hours or more for large number of decap ports designs, depending also on GA parameters. One reason for improved results may be that genetic drift [13] is reduced with the introduction of the solution size and size variation parameters. Genetic drift describes the change in the frequency of genes in a population as the algorithm converges towards a local or global extrema. As better solutions are found, the frequency of gene 0 will increase in the population, especially with crossover considered. Solutions with too many empty ports are less likely to meet the target

impedance and the number of such solutions appearing in the population will only increase with crossover. By controlling the number of decaps that can exist in a solution, the effect of genetic drift is lessened and the convergence can be improved. The convergence curve for the number of decaps found for the gene suppressed GA and open-source is shown in Figure 6 and for Case 2. Due to the elitism implementation, the plot is non-increasing.

3.2. GENE SUPPRESSED GA AND OTHER METHODS

The minimum number of decaps found by the proposed GA, the ideal solution of [6], and the reinforcement learning method are given in Table 6. The performance of the proposed GA varies from finding a better or equivalent solution to at worst a competitive one. The inherent randomness in the search of any GA means that the best solution may not always be found despite its potential performance. The performance of the gene suppressed GA is as good if not better than the ideal solution of [6]. The method in [6] is a GA though, and the randomness of a GA search may still result in finding the global minimum solution. Compared to the reinforcement learning method, the proposed GA is competitive and sometimes better.

One possible reason for the sometimes-better results of the proposed algorithm is because no assumptions are made by the GA about how to find the best solution. In the method of [6], one decap is added at a time, as best minimizes the cost function. Per its cost function, bigger decaps should be added first because a bigger decap can quickly bring down the impedance in the low and medium frequency range. Progressively, smaller decaps will be added. In addition, the port locations are indirectly selected so that

with a decap connected, the inductance associated with that location is just right to maximize the effect of that decap by shifting its resonance point. As a result, searches by [6] should be somewhat consistent in behavior and result.

Table 5. Open Source vs Gene Suppressed GA Results

Genetic Algorithm	**Number of Decaps**		
	Case 1	***Case 2***	***Case 3***
Gene Suppressed GA Population and # Gen. = 50	**16 ~1 min**	**41 ~70 min**	**28 ~11 mins**
Open Source GA Population and # Gen. = 50	**16 ~1 min**	**71 ~75 mins**	**35 ~11 mins**
Gene Suppressed GA Population and # Gen. = 100	**15 ~5 mins**	**23 ~4.3 hrs**	**24 ~43 mins**
Open Source GA Population and # Gen. = 100	**15 ~5 min**	**66 ~5 hrs**	**32 ~45 mins**

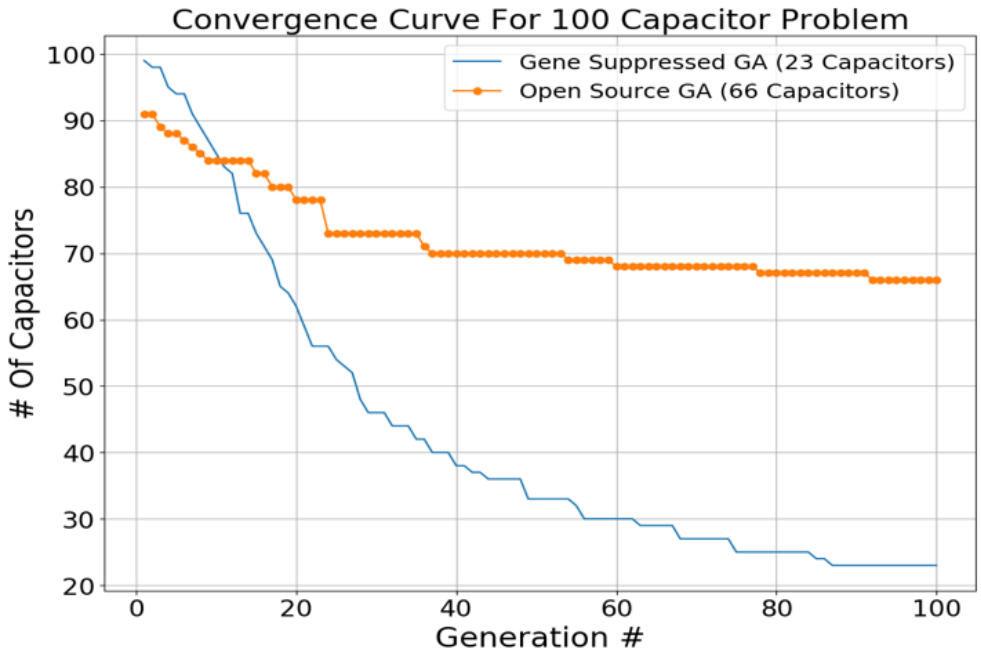

Figure 6. Convergence of Gene Suppressed GA vs Open Source GA, Case 2

Table 6. Gene Suppressed vs Other Methods Results

Algorithm	**Number of Decaps**		
	Case 1	***Case 2***	***Case 3***
Gene Suppressed GA Population and # Gen. = 50	16	41	28
Gene Suppressed GA Population and # Gen. = 100	15	23	24
Ideal Solution of [6]	No Solution	23	33
Reinforcement Learning Method	17	21	27

In the reinforcement learning case, the order in which the ports are filled was fixed. The capacitor selection was not fixed. Exploration was done in the search space so different placement patterns could be tested. This exploration offers the benefit of considering alternate search paths but all search paths are restricted to the calculated port fill order. For Case 2, using the calculated port sequence resulted in the best solution. In Case 3, the solution found by the proposed GA is better, but no full search was done to check if an equal or better solution exists using the port sequence.

Without a full search, there is no way of confirming that any specific objective function, or any assumptions made about the global minimum solution, is always true. No direct assumption is made by the gene-suppressed GA about the best solution. The fitness function for solutions that satisfy the target impedance is proportional only to the number of unused ports. Solutions using fewer decaps will be favored by the GA, but not directly port locations or decap types. Characteristics of the same solutions however, may still be

passed on repeatedly, pushing the GA towards a local minimum. Regardless, the possibility that better solutions can exist using decap ports and decaps that is unique from the currently known best solution is not closed off by the GA fitness functions. The downside though, is that there is no basis to judge two solutions using the same number of decaps, but using different locations and decap types, because they'd be rated the same fitness. From the results, limiting the expression of gene 0 help lead towards good solutions despite a less focused search space. But in some cases, like Case 2, it may not be sufficient.

The best decap placement pattern for Case 3, for each decap optimization method, is depicted in Figure 7, 8 and 9. Their impedance curves are given in Figure 10. Although there is no guarantee that the GA in [6] and the reinforcement learning method is unable to find an equal or better solution for Case 3, the port locations used by the proposed GA are ones that would not be considered by the ideal case of [6] and the port sequence of the reinforcement method. For instance, capacitor port 11 is unused in the ideal solution of [6] and for the reinforcement learning case, port 11 is not within the first 27 ports of the calculated port sequence (Best solution found by reinforcement learning method).

4. CONCLUSION

In this work, a new GA for the decoupling capacitor problem in power distribution networks of PCBs was presented. The gene suppressed GA can find the best known minimum capacitor number, or at least a comparable solution, to satisfy a user defined target impedance. This is verified by comparing against two other algorithms.

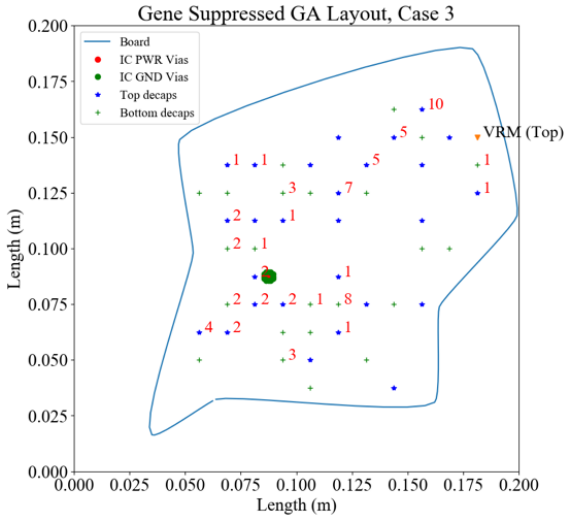

Figure 7. Gene Suppressed GA Capacitor Layout, Case 3
Ports Used: [4, 5, 7, 8, 10, **11**, 12, 14, 16, 17, 18, 19, 20, 24, 25, 26, 28, 29, 33, 36, 39, 46, 47, 50]

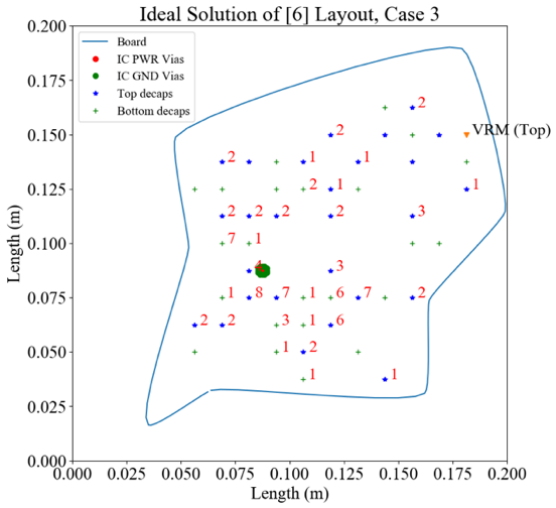

Figure 8. Ideal Solution of [6] Capacitor Layout, Case 3.
Ports Used: [1, 2, 3, 4, 5, 6, 7, 8, 10, 12, 13, 14, 16, 17, 18, 19, 20, 21, 22, 23, 24, 26, 27, 29, 34, 36, 38, 39, 43, 46, 47, 49, 50]

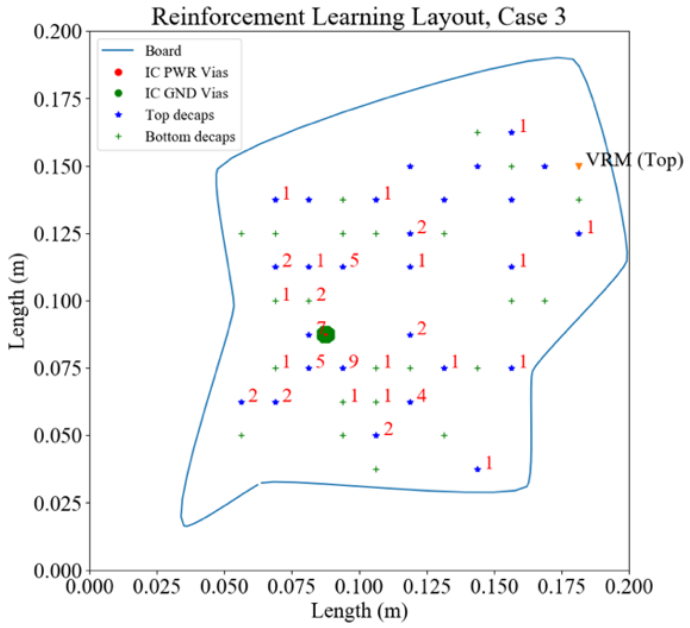

Figure 9. Reinforcement Learning Capacitor Layout, Case 3
Ports Used: [1, 2, 3, 4, 5, 6, 8, 10, 12, 13, 14, 16, 17, 18, 19, 20, 21, 23, 24, 26, 27, 36, 39, 43, 47, 49, 50]

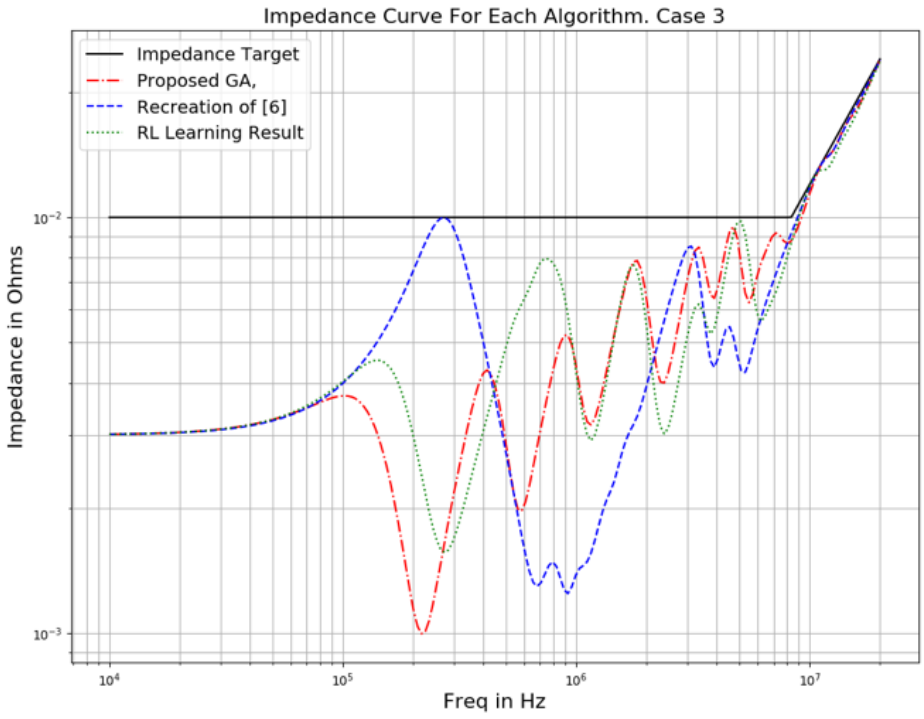

Figure 10. Resulting Impedance Curves of Best Solution for Each Algorithm

The convergence curve can also be smoothed out and the search made more efficient by limiting the number of capacitors in solutions.

From our experimental results, there is a relationship between the solution found and the GA parameters. Larger population sizes and longer iterations result in finding better solutions, especially for PDN with large numbers of capacitor ports, but at the expense of much longer algorithm time due to increase in the number of calculations.

REFERENCES

[1] K. Koo, G. R. Luevano, T. Wang, S. Özbayat, T. Michalka and J. L. Drewniak, "Fast Algorithm for Minimizing the Number of decap in Power Distribution Networks," in *IEEE Transactions on Electromagnetic Compatibility*, vol. 60, no. 3, pp. 725-732, June 2018, doi: 10.1109/TEMC.2017.2746677.

[2] S. Han and M. Swaminathan, "A Non-Random Exploration based Method for the optimization of Capacitors in Power Delivery Networks," *2020 IEEE 29th Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS)*, San Jose, CA, USA, 2020, pp. 1-3, doi: 10.1109/EPEPS48591.2020.9231448.

[3] L. Zhang, et al., "Decoupling Capacitor Selection Algorithm for PDN Based on Deep Reinforcement Learning," *2019 IEEE International Symposium on Electromagnetic Compatibility, Signal & Power Integrity (EMC+SIPI)*, New Orleans, LA, USA, 2019, pp. 616-620, doi: 10.1109/ISEMC.2019.8825249.

[4] H. Park *et al*., "Reinforcement Learning-Based Optimal on-Board Decoupling Capacitor Design Method," *2018 IEEE 27th Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS)*, San Jose, CA, 2018, pp. 213-215, doi: 10.1109/EPEPS.2018.8534195.

[5] J. Y. Choi and M. Swaminathan, "Decoupling Capacitor Placement in Power Delivery Networks Using MFEM," in *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 1, no. 10, pp. 1651-1661, Oct. 2011, doi: 10.1109/TCPMT.2011.2165954.

[6] de Paulis, Cecchetti, Olivieri, Piersanti, Orlandi, and Buecker, “Efficient Iterative Process based on an Improved Genetic Algorithm for Decoupling Capacitor Placement at Board Level,” *Electronics*, vol. 8, no. 11, p. 1219, Oct. 2019.

[7] K. F. Man, K. S. Tang and S. Kwong, "Genetic algorithms: concepts and applications [in engineering design]," in *IEEE Transactions on Industrial Electronics*, vol. 43, no. 5, pp. 519-534, Oct. 1996, doi: 10.1109/41.538609.

[8] R. Solgi, 2020. *Rmsolgi/Geneticalgorithm*. [online] GitHub. Available at: <https://github.com/rmsolgi/geneticalgorithm> [Accessed 15 January 2021].

[9] L. Zhang. J. Juang, Z. Kiguradze, et al. “Efficient DC and AC Impedance Calculation for Arbitrary-shape and Multi-layer PDN Using Boundary Integration,” *IEEE Transactions on Electromagnetic Compatibility*; to be submitted.

[10] A. Shukla, H. M. Pandey and D. Mehrotra, "Comparative review of selection techniques in genetic algorithm," *2015 International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE)*, Noida, 2015, pp. 515-519, doi: 10.1109/ABLAZE.2015.7154916

[11] A. Bala and A. K. Sharma, "A comparative study of modified crossover operators," *2015 Third International Conference on Image Information Processing (ICIIP)*, Waknaghat, 2015, pp. 281-284, doi: 10.1109/ICIIP.2015.7414781.

[12] L. Zhang, J. Juang, Z. Kiguradze, S. Jin, S. Wu, Z. Yang, J. Fan, and C. Hwang, “PCB-Level Decap Placement Using Deep Reinforcement Learning”, *IEEE Transactions on Microwave Theory and Techniques*, to be submitted

[13] A. Rogers and A. Prugel-Bennett, "Genetic drift in genetic algorithm selection schemes," in *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 298-303, Nov. 1999, doi: 10.1109/4235.797972.

II. AUGEMENTED GENETIC ALGORITHM FOR DECOUPLING CAPACITOR OPTIMIZATION IN PDN DESIGN THROUGH IMPROVED POPULATION GENERATION

Jack Juang, Ling Zhang, Francesco de Paulis, Chulsoon Hwang

Department of Electrical and Computer Engineering, Missouri University of Science and Technology, Rolla, MO 65409

ABSTRACT

In power distribution network (PDN) designs, a large number of decoupling capacitors (decaps) may be needed to satisfy target impedance limits. Many algorithms have been proposed and implemented for finding the optimal decap placement, including genetic algorithms (GA), and machine learning methods. In this work, an improved GA is proposed for finding the decap placement pattern that can satisfy a target impedance using the minimum number of decaps. The distribution of capacitors expected to appear in the global minimum solution is first predicted by determining how effective each decap type is towards satisfying certain critical impedance points. This estimation is used to inform the generation of initial solutions in order to put the initial search space nearer the global minimum and ensure certain solution characteristics appear. GA search using this improved population generation is found to be an improvement over a canonical GA implementation, by finding solutions where the latter could not, or finding a solution using fewer decaps.

1. INTRODUCTION

In power distribution networks (PDN), the major obstacles for consistent power delivery include, but are not limited to, inductance associated with intentionally designed current loop paths and non-ideal component behaviors such as the parasitics belonging to discrete components. These sources, coupled with faster rise times and larger power consumption, demand very careful designs to ensure the proper functionality of the active devices powered through the PDN. For integrated circuits (ICs) on a board, a common specification for their reliable operation is the target impedance. The target impedance is broadly defined using the maximum allowable voltage ripple divided by the total required current for all devices on a net, and is the maximum impedance limit that ICs must meet for consistent performance [1]-[3]. To reduce the PDN impedance to acceptable levels, decoupling capacitors (decaps) are often used. In this work, the objective is to find the decap placement pattern (the optimal pattern) that uses the minimum number of decaps in solving a given target impedance.

For large designs, finding the optimal pattern is difficult. The potential number of ways to place decaps scales exponentially with the number of possible decap locations and decap library size, and includes solutions that do not meet the target impedance at all, and solutions that meet the target using a variable number of decaps. For this optimization problem, many algorithms and tools have been developed to quickly find the optimal placement.

Of the many methods for decap optimization, there include machine learning methods [4]-[6], iterative selection [7]-[9], as well as numerous genetic algorithm (GA)

implementations [10]-[14]. For every algorithm however, the approach for finding the optimal solution can vary greatly, resulting in different outcomes as a result of the different search methodologies and judgment criteria. In the iterative algorithm of [7] for example, decaps are iteratively added by considering which decap type and required number of that type, when added, can keep anti-resonance points below the target and push the first impedance target violation point as high up in frequency as possible. The consequence of this is the general order in which decaps are added. Decaps added for a validation case in [7] shows that, in general, large capacitance decaps are added first, and iteratively smaller and smaller capacitance decaps are added. This trend makes sense as the goal is to push the target impedance violation as high up in frequency as possible. The work of [7] also proposes first filling up the ports nearest IC so as to reduce the loop inductance. While physically sound, the consequence should be that the larger decap types always end up being placed nearer the IC, inadvertently pushing smaller package decaps further away which may be contrary to expectations.

The GA implementation in [10] has a similar behavior over the course of its search, utilizing a cost function that rewards maximizing the number of points brought below the impedance target. The result is larger decaps being placed first, as many low frequency points can be immediately satisfied and done so with a fewer number of decaps. Higher frequency impedance points are harder to satisfy as the decaps effective in those ranges have a lower decrease in PDN impedance per decap, compared to decaps in the lower frequency region. Smaller package decaps as a result, are placed later in the optimization, after the low frequency points have already been met. An example progression curve of the input impedance in the work of [10] as decaps shows as such;

new self-resonance points appear in the direction of increasing frequency as decaps are iteratively added. In the works of [7] and [10], the form of the final solution, and also how that solution is constructed, is determined by how each algorithm seeks out the optimal solution.

Similarly in the reinforcement learning method of [4], for calculation of reward and Q values, the target impedance is first split into multiple regions over the frequency range, with rewards given based on how many of regions of the target impedance is satisfied during each step of training. While there is no analysis in [4] about what kinds of decaps are chosen and in what order the decaps are chosen over the individual training epochs, it should be logical to say that, with the specific input parameters of the example and the reward function, that larger decaps would be chosen first. A quick reward would be provided in [4] with the early addition of large decaps, and for the same reason as the work in [10]. A port priority method based on inductance was also adopted, meaning for the same input board, the order in which ports are filled will always be the same. The algorithm determines the structure of the solution, always forcing the same sequence of ports to be filled.

In the work of [11], the decap placement pattern found by [11] is compared with the placement of those found by [6] and [12] for a test case. The work of [6] is a reinforcement learning method that used the same port priority implementation as [4]. The work of [12] is a GA, but for the comparison in [11], decaps were placed, the type and location, by doing a full search, at each iterative step, to determine the solution that would most optimize the cost function of [12]. The best solutions found by each algorithm varied in the number of decaps needed, and generally chose similar, but not the

exact same port locations. The distribution of the types of decaps used and the port locations of the decap types relative to the IC location also differed between solutions, indicating that each algorithm is converging to similar, but different points in the search space. Ideally, the solution found by all algorithms should have similar, if not the exact same, port locations/decap types used as that of the most optimal solution. In reality, different algorithms may make different conclusions about the optimal solution as they converge. Irrespective of the optimization used however, the form of the optimal solution is entirely independent. If an estimation can be made at what this optimal solution looks like, optimization can be performed by steering for certain immutable solution characteristics, such as specific decap types, decap amounts, or port locations. By intentionally seeking and favoring these immutable characteristics, instead of letting the algorithm decide those characteristics over the course of its search, convergence could be made more consistent and more quickly by ensuring the necessary qualities of the optimal solution is captured.

In this work, we present an augmented genetic algorithm (GA) for finding the placement pattern that minimizes the required number of decaps to meet a target impedance. As an extension of our previous work in [11], this work focuses on identifying what the optimal solution should look like before beginning optimization, using that information to improve solution generation and by extension the GA search. An estimation is made of the decap types that are likely to be used in the optimal solution and initial populations will be generated using information, quickly directing the GA towards a better and more promising initial search space. In addition, the crossover operator is removed in favor of more targeted mutation operators that can make more

effective changes, especially as the total number of decaps in a solution begins to decrease. For validation, results are compared against those found by [11] and a canonical GA [15] for several test cases and target impedances.

2. TRENDS OF PREVIOUS WORK

The goal of the work in [11] is the selection and placement of decaps with the goal of minimizing the required number of decaps. The objective function used to evaluate solutions that meet the target impedance was a function of only the number of capacitors used. In addition, the operators of GA [11] did not directly impose any restrictions on decap placement. No decaps were intentionally fixed to any locations, nor were particular port locations forced to be used. Due to the underlying physics and the mechanics of a GA however, similar conclusions and repeated trends may be seen over many algorithm runs.

Solutions found by [11] tended to have ports closest to the IC of interest filled and filled with small capacitance/package size capacitors. One such solution is reprinted in Figure 1, with the numbers corresponding to specific decap types. The capacitor library used in [11] and this work is given in Table 1 and are in the form of publicly available measured S-Parameters [16]. From the same example, there are ports chosen and filled with capacitors that are further from the IC despite closer ports being available. The physical meaning of this could be that the effect of mutual inductance between ports becomes more significant as the number of filled ports decreases.

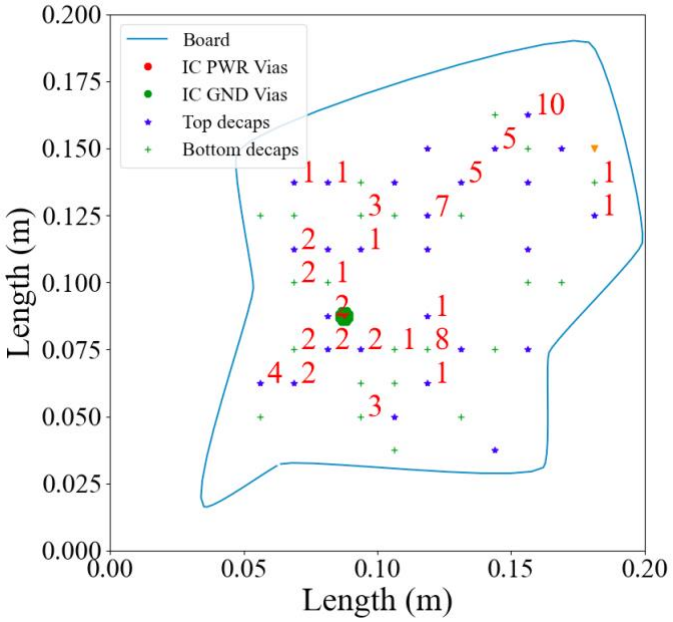

Figure 1. Example of an optimal solution, reprinted from [11].

Table 1. Capacitor Library

Decap Type	Capacitance (uF)	ESL (nH)	ESR (mΩ)
1	0.1	0.19	34.7
2	0.47	0.18	18.3
3	1	0.22	15.2
4	2.2	0.2	7.2
5	4.7	0.28	7.1
6	10	0.26	5.2
7	22	0.27	4.0
8	47	0.34	2.9
9	220	0.41	1.9
10	330	0.46	1.2

Another observation is on the distribution of the decap types used. For the solution in Figure 1, a disproportionate number of decaps used were of smaller package size/smaller capacitance value. For an input impedance profile increasing with frequency, as may be due to the parasitic inductance between power/ground plane pair and IC input port and/or the inductance introduced by a voltage regulator module (VRM) in the PDN design, it is reasonable to assume that higher frequency points are harder to satisfy. An example of such an increasing input impedance with an R-type target impedance is given in Figure 2. The impedance at higher frequency points will be higher due to inductive parasitics. The decaps effective in the higher frequency range will also tend to have higher ESR due to smaller package sizes. It is reasonable to assume then, for any increasing input impedance, a non-trivial R-type (constant impedance) or RL-type (constant impedance, transitioning to a +20 dB/decade slope) target impedance, that the distribution of capacitors in the global minimum solution will lean towards smaller decap types. In the work of [10], an uneven distribution in the types of decaps used can also be seen in the optimized solutions, and also for an example with increasing input impedance as seen by an IC.

For verification, the algorithm of [11] was run for many different R and RL type target impedances, and for the same board example as in Figure 1. Figure 3 shows a heat map for the percentage of decaps in the best solution found that are of type #1, #2, and #3, and as a function of Z_{max}. Z_{max} is defined as the value of the target impedance at the final frequency point of interest. In this work, the frequency range of interest is 10 kHz to 20 MHz, making Z_{max} the impedance of the target at 20 MHz. The darker the color (which follows increasing Z_{max}), the lower the percentage of decap type #1, #2, and #3 in

the best solution found. Decap #1, #2, and #3 were chosen as they are the 'high' frequency capacitors in the frequency range of interest, having self-resonance points in the same decade as 20 MHz.

From Figure 3, the percentage of decaps in the best solution that is of type #1, #2, and #3 is inversely proportional to Z_{max}. This result should be reasonable because as the target impedance becomes more difficult to satisfy (low impedance targets), more decaps are required to meet the target and the relative difficulty of meeting high frequency points increases. Conversely, for generous (high impedance) targets, fewer high frequency decaps are needed, if needed at all. In general, if the input impedance is increasing, and the target impedance is of R or RL type and non-trivial to meet, then the distribution of decaps used is likely to skew towards smaller package sizes.

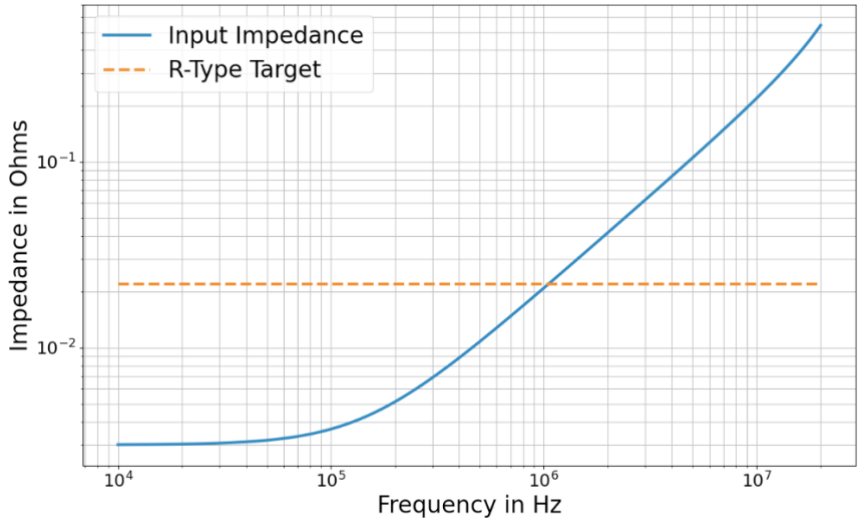

Figure 2. Example of increasing input impedance and R-Type target.

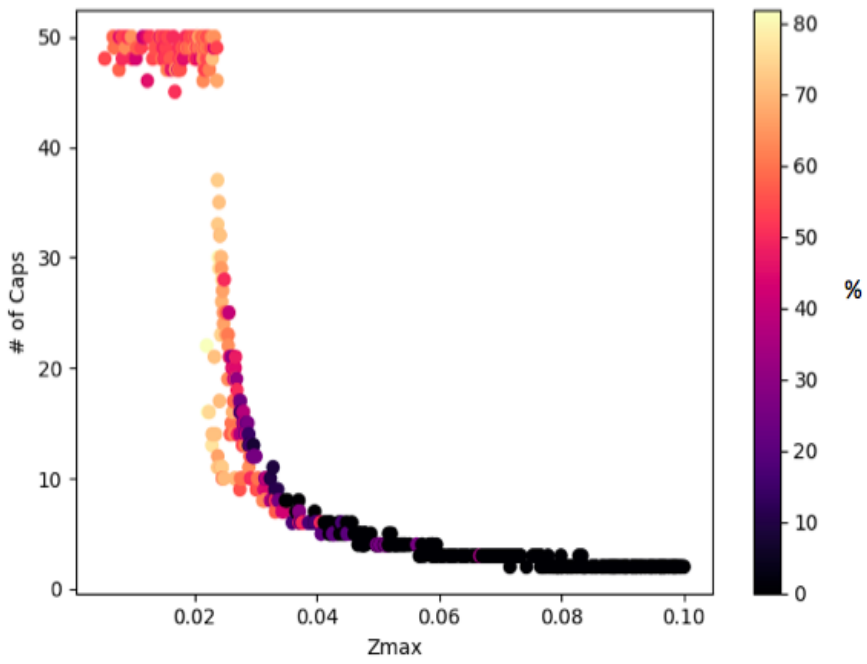

Figure 3. Heat map for the % of decaps in best solution that are of decap type #1, #2, and #3.

This assumption should typically be true of the optimal solution, for boards and impedance targets behaving under the given conditions, and be independent from the method used to find the optimal solution. In this work, the population initialization step of the GA will be augmented by generating solutions with decap distributions that are more consistent with the distribution of decaps expected in the optimal solution.

In [11], comparisons of algorithm results were also made, comparing against a canonical GA [15] implementing the same fitness functions as [11], a reinforcement learning method [6], and the best solution found through a full search optimization of the fitness function of [12]. For the remainder of this work, the proposed GA will be referred to as ‘Augmented GA’, the work from [11] as ‘GA [11]’, and the canonical GA implementing the fitness functions of [11] as ‘Canonical GA’. While GA [11] could find

competitive or better results, the quality of the final solution heavily depended on the search time and search radius. Figure 4 and 5 shows convergence curves for the results of the different algorithms, performed on the same board as Figure 1. By optimizing the generation of the initial set of solutions, the aim is to find better solutions while cutting down on the needed search time.

3. AUGMENTED GENETIC ALGORITHM

3.1. BASIC STRUCTURE OF GENETIC ALGORITHM

The GA is an optimization algorithm [17] revolving around the principles of natural selection. In nature, individuals who can survive and reproduce will pass down their genes to the next generation; genes that in theory, are conducive to future survival and future reproduction. The analog to optimization is the idea that the best solution can be found by repeatedly mixing good solutions to find better ones. The canonical flowchart of the genetic algorithm is given in Figure 6.

In general, a GA follows the same basic steps: population initialization, evaluation, selection, crossover, and mutation. In population initialization, a set of solutions, called a population, is first generated. A solution to the optimization problem is first encoded into a form fit for manipulation.

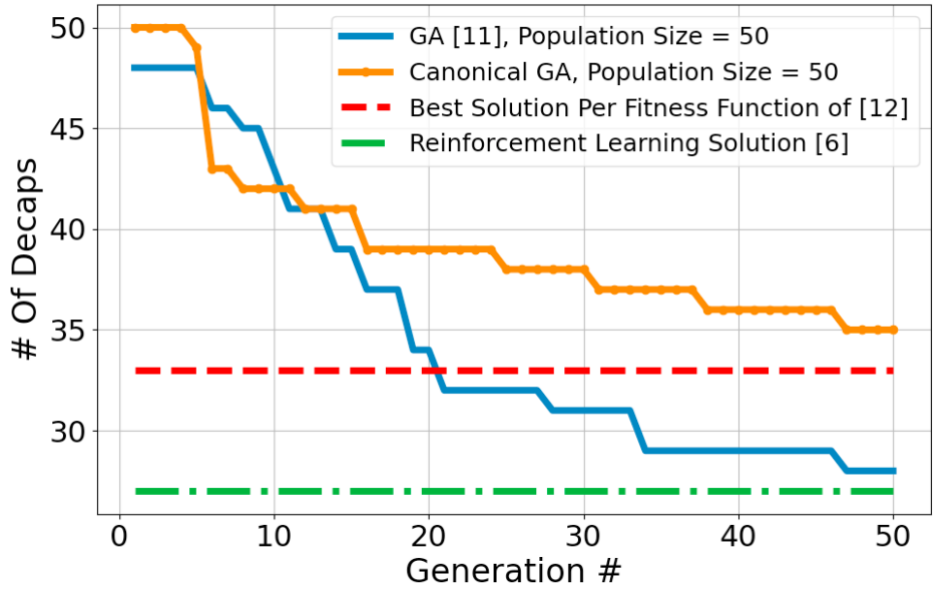

Figure 4. Convergence curves for different algorithms, using searches with population sizes of 50 and 50 iterations of search.

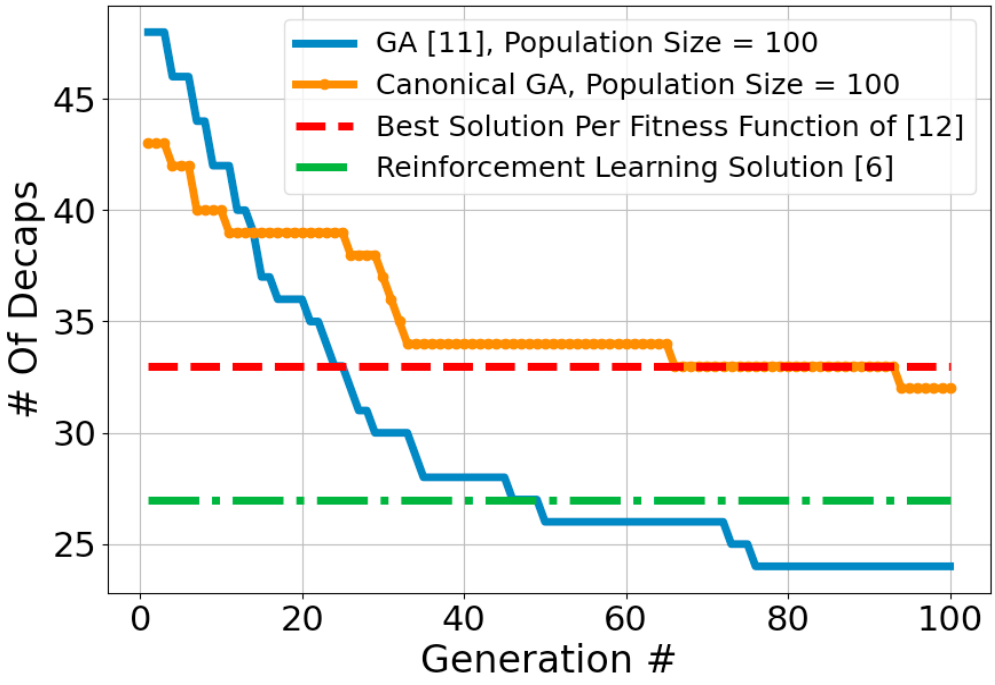

Figure 5. Convergence curves for different algorithms, using searches with population sizes of 100 and 100 iterations of search.

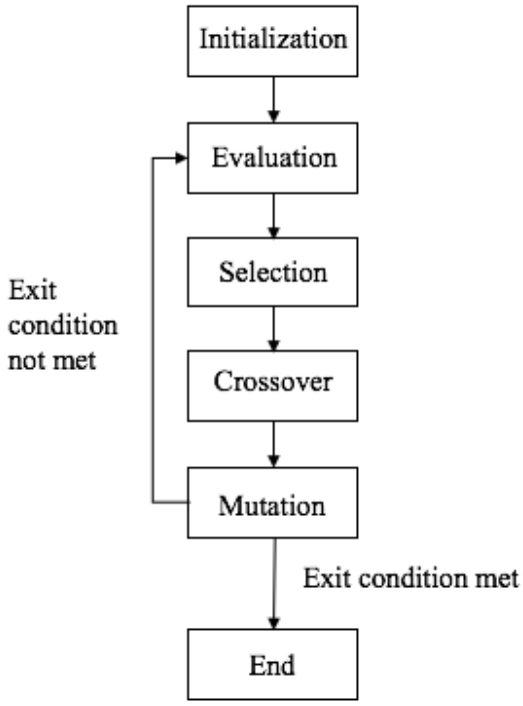

Figure 6. Canonical GA Flowchart.

In this work, a solution is encoded as a vector of discrete numbers, where an index of the vector corresponds to a decap port location, and the value at that index corresponds to the type of decap placed there. An example of a solution using this encoding is given in Figure. 7. A decap type of ‘0’ indicates no decap is placed in the corresponding port location. For the initialization of the population, the classic approach is to generate a set of uniformly random solutions. Equivalently for this work, it would be a set of numerical vectors with uniformly random decaps generated at each port.

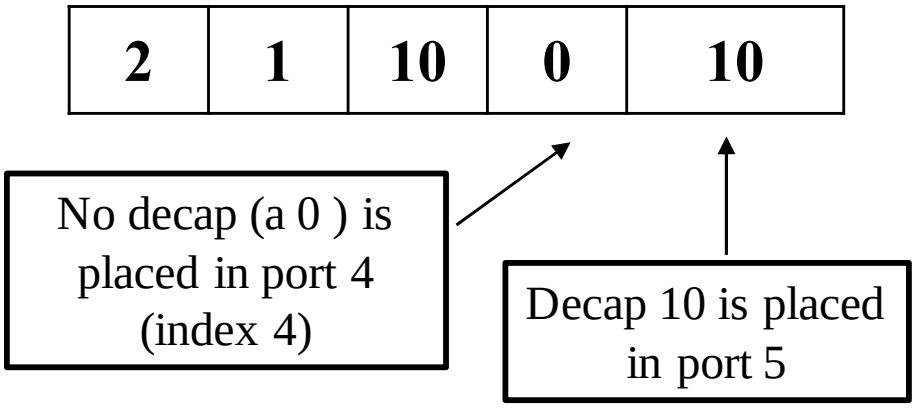

Figure 7. Example of an encoded solution for the decap placement problem.

In the evaluation step, a solution is evaluated for its' fitness; a measure of how likely an individual is to successfully reproduce. In this work, solutions satisfying the target impedance have fitness given by (1). For solutions that fail to satisfy the target impedance, they are evaluated by (2) [11]. For the variables, *solution_z(f)* is the frequency dependent input impedance as seen from the IC, for the evaluated decap placement pattern. *target_z(f)* is the frequency-dependent target impedance. The goal of (1) and (2) is to minimize the fitness functions.

$$\text{Fitness} = -(\text{Total \# of Ports} - \text{\# of Ports Used}) + 1 \quad (1)$$

$$\text{Fitness} = \left(\frac{\text{solution\_z(f)} - \text{target\_z(f)}}{\text{target\_z(f)}}\right) \quad (2)$$

In the selection stage, solutions are selected as parents for reproduction based on their fitness. One standard technique is the roulette wheel method [17], used in GA [11], where the selection of potential parents is proportional to their fitness. Once a list of potential parents is selected, they are paired off to produce new solutions by mixing characteristics of the parents in a process known as crossover. Within context of the

decap solutions, new solutions are generated by mixing parts of the parent solution vectors.

The final step is mutation, in which individual parts of a solution is changed. For this work, the decap placed at a port location is changed to a different decap type (or emptied) by changing the value at that index of the solution vector. The GA will loop from mutation back to evaluation, with each loop being termed a generation, until some exit condition for the algorithm is reached. The exit condition for this work is when the preselected number of generations is reached.

3.2. MODIFIED IMPLEMENTATION OF GENETIC ALGORITHM

In this work, several changes were made to the canonical GA implementation, with the modified flowchart given in Figure 8. Changes include controlling the number of decaps in each solution, the removal of the crossover operation, changes to how the initial population is generated, and finally changes to the mutation operators.

For controlling the number of decaps in each solution, the number of decaps in all solutions of the GA is restricted to $N - 1$, where N is the currently known minimum number of decaps required to meet the target. If no solution that can satisfy the target is currently found, N is set equal to the total # of port locations. This means that (1) is only used to detect solutions where the total number of needed decaps decreases and (2) is used to incrementally improve and find the next solution that meets the target using fewer decaps.

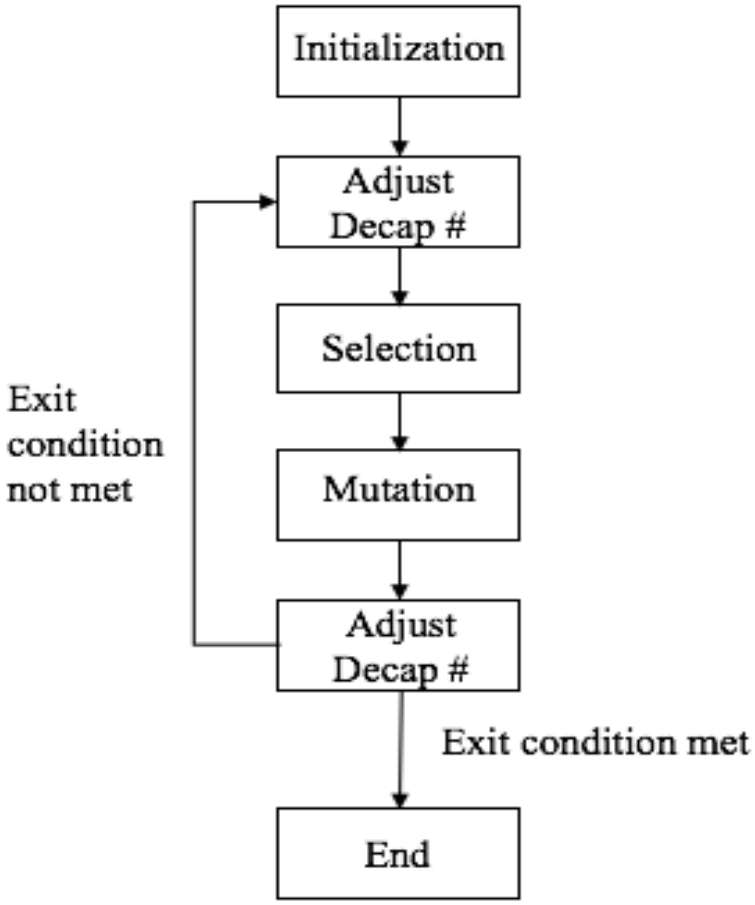

Figure 8. Augmented GA Flowchart.

In this modified implementation, the crossover operation has been removed. Based on observations of the results in [11], as the GA progressed, newer solutions seemed to be found mostly through mutation or through [11]'s version of the decap control step. In general, the closer a population reaches the optimum (solution using the fewest decaps), the effect of any single change to a solution is more significant. A change in the value of a single decap or of the location of a decap may significantly worsen a solution. Crossover operations, acting over entire solutions, may make too many changes

at once, and even incremental improvements may be hard to achieve. This is exacerbated with our encoding scheme, where a large of number of ‘0s’ will begin appearing in the solution vectors as the optimum is approached. For this reason, the crossover operator was removed in favor of more controlled mutation operators aimed at making smaller but hopefully more effective changes. Only the best solution in the current population is chosen for mutation; all new solutions in the next generation will come from the mutation of a single-parent solution.

3.3. AUGMENTED POPULATION GENERATION

For this decap placement problem, generating an uniformly random initial population may not be the best approach. If there is a specific distribution of decap types for the global minimum solution, then the GA algorithm, and all search algorithms in theory, should slowly converge to that same distribution over the course of their search. Premature convergence can occur if, during the search, the algorithm gets stuck in a local minimum while it is still searching for the distribution of the optimal solution. If the GA search can begin in a search area that includes or is close to the global minimum, then both the convergence and time required to find any valid solution should be made faster.

To that end, the initial population should be generated in a way that considers the constraints imposed by the target impedance on the decap types used. Focusing on R and RL-type target impedances, and increasing input impedances as seen from an IC, this work identifies certain ‘critical’ frequency points. Decap weights for each decap type is generated based on how much ‘effort’ is required to satisfy each critical point, and these weights are used to inform the initial population generation. For RL and R-type target

impedances, the impedance of the target at the last frequency point is chosen as one such critical point. This is done with the assumption that the last impedance point will be the hardest to satisfy and would require more effort. For RL-type target impedances, another critical point is identified as the impedance of the target at the frequency in which the target transitions to +20 dB/decade. This additional point is chosen to consider the optimization of the constant and non-constant target impedance region separately. An example of an RL-type target is shown in Figure 9, with identification of the critical points.

For generating the decap weights, each port in the board is filled with one specific decap type, one at a time, until the minimum number of that type required to satisfy the critical point is found. This is done for every decap in the library and for every critical point. If a decap type cannot a satisfy a given critical point, the required number of decaps for that point is set to *M*, where *M* is the total number of ports. As a simplification, the impedance of the critical point, after filling all ports with the same decap type, is calculated first, with the assumption that this would result in the lowest impedance for that point. If all ports are filled but the critical point is not satisfied, the required number is immediately set to *M*. This is a valid assumption to make only if all decaps in the library are acting wholly inductively around the frequency of the critical point, which may not necessarily be true. The impedance is calculated by connecting the decap Z-parameters to the board Z-parameters (board S-Parameters taken as input) through a segmentation method [4].

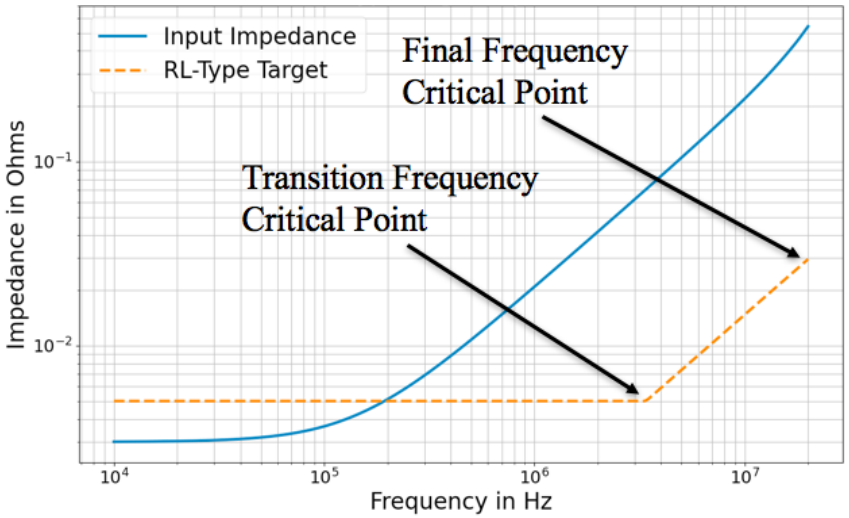

Figure 9. RL type target with two critical points.

For a 50-port board example with RL-type target impedance, Table 2 gives the number of decaps required for each critical point, using the target of Figure 9. The order in which the ports are filled is calculated based on a ‘port priority’ that will maximally reduce the equivalent inductance seen by an IC [8], calculated using the inductance seen from each port at 20 MHz, as extracted from Z-Parameters of the board.

After computing the required number of each decap, the decap weights are determined. The weight for each decap type is directly proportional to how useful they are towards satisfying the critical points; if a fewer number of them is required to meet the critical points, their weight is higher. If for all critical points, no decap can be used to satisfy those points, then it is equivalent to every decap having the same weight, and leading to a uniformly generated initial solutions.

Table 2. Number of Decaps To Satisfy Critical Points Of Figure 8

Decap Type	# Required for last frequency	# Required for transition frequency
1	4	48
2	21	12
3	26	10
4	27	2
5	35	17
6	33	29
7	32	28
8	33	31
9	43	40
10	43	36

Equation (3) describes the process for generating weights for the R-Type target impedances. N_i is the number of required decaps of type *i* for the single critical point. N_{max} is the largest number of required decaps among all types used. *K* is the total number of decaps in the library. N_j is the required number of the *j*th decap in the library. Finally W_i is the calculated weight for the *i*th decap.

$$W_i = \frac{|N_i - N_{max}| + 1}{\Sigma_{j=1}^{K}(|N_i - N_{max}| + 1)} \tag{3}$$

Only a percentage of a solution is devoted toward solving the critical point for R-Type targets; the remainder of the solution is uniformly generated to give some variety. The percentage of a solution dedicated to solving the R-Type critical point is given by

(4), where N_{total} is the total sum of the number of decaps, of all types, required to satisfy the critical point.

$$\%\ \text{of Solution to Solve R-Type Critical Point} = 100 * \frac{N_{total}}{\#\ \text{of Ports} * \#\ \text{of Decaps in Library}} \quad (4)$$

For RL-Type target impedances, the solution is split into two groups, each for satisfying a different critical point. Weights are generated for each group separately, using (3) for each critical point. Equations (5) and (6) are then used to determine the percentage of the solution dedicated to solving each critical point. f_1 indicates the final frequency point and f_2 the transition frequency point. $N_{total,f1}$ is the total number of all decaps required to satisfy critical point f_1, and $N_{total,f2}$, the same for f_2.

$$\%\ \text{of Solution to Solve } f_1 = \frac{N_{total,f1}}{N_{total,f1} + N_{total,f2}} * 100 \quad (5)$$

$$\%\ \text{of Solution to Solve } f_2 = 100 - \%\ \text{to Solve } f_1 \quad (6)$$

For R-Type target impedances, for the percentage of the solutions dedicated to solving the critical frequency point, the decap port locations will be filled one at a time, based on the port priority. The chance of each decap being chosen at each port is the calculated weights. The remainder of the solution is uniformly generated. For RL-Type target impedances, the percentage of the solution devoted towards solving the critical frequency point f_1 is filled first. Using the port priority, decaps are added based on the weights calculated from (3) for the critical point f_1. The remainder of the solution will be filled using the weights calculated from (3), for critical point f_2, again using the remainder of the port priority.

If for some critical point, 1 and only 1 decap type can be used to satisfy it, then in the initial solutions, only the required number of those decaps is forced into the initial solutions. The order of ports to fill will still be the port priority. The remainder of the solution is generated with uniform weights for all decaps.

3.4. NEW MUTATION OPERATORS

With the crossover operator removed, the algorithm will rely entirely on the mutation operators to generate new solutions. Careful changes need to be made to avoid drastically worsening a solution and at the same time, changes made must work towards finding better solutions. To that end, three new mutation operators have been created. The new mutation operators are the target mutation, interchange mutation, and shift mutation. The goal of these mutations is to mimic what a designer may do in practice. Two new solutions are produced for each set of mutation operator calls.

The target mutation aims to mutate decaps to a type more suited to correcting a target violation. Before starting the GA search, the first self-resonance frequency point for each decap type is determined by connecting each decap to the first port of the port priority. The generated list of self-resonance frequencies states which decap is best suited for a given target impedance violation, based on the shortest distance from the offending point to a self-resonance point. When computing the PDN impedance during GA search, the first impedance violation point of the best solution in the population is recorded. For target mutation, decap types in a solution will mutate in the ‘direction’ of the best decap. To do this, the value at an index of the numerical solution vector will be incremented or decremented towards the best decap for the violating point. An example is given in

Figure 10. Every decap has a 10% chance of mutating in this way, with a fixed increment/decrement of 1. As an optional argument, if an empty port (a ‘0’) in the solution vector is selected for mutation, that port will automatically mutate to the best decap for the target violation. If the decap in a location is already the most suitable decap type, then its value will not change. Two solutions are generated from target mutation. The first solution will allow the empty ports to be mutated but the second solution will not.

The second mutation operator, the interchange operation, is applied to the first solution generated by target mutation. This function will swap the locations of two different decap types in the solution vector. Each decap has a 10% chance of this mutation and is not applied to empty ports. Figure 11 gives an example of this mutation.

The third mutation operator is a shift operation. It is applied to the second solution generated by target mutation. This operation will exchange the locations of two decaps based on a maximal ‘distance,’ with the ‘distance’ based on the port priority list. An example is given in Figure 12, with a max shift distance of 1 port. For this work, each decap has a 10% chance of shifting, with 5 ports maximums shift.

With the new mutation operators, the solutions may all have different number of decaps. The number of decaps for each solution will be set to *N-1*, as described in Section III.B, and *N* representing the currently known minimum number of decaps required to meet the target impedance.

Figure 10. Example of target mutation. Decap #8 mutates to #7, decap #3 mutates into #4. The solution is mutating 'inwards' towards the best decap for a violating point.

Figure 11. Example of shift mutation. The decaps in indices 2 and 4 swapped.

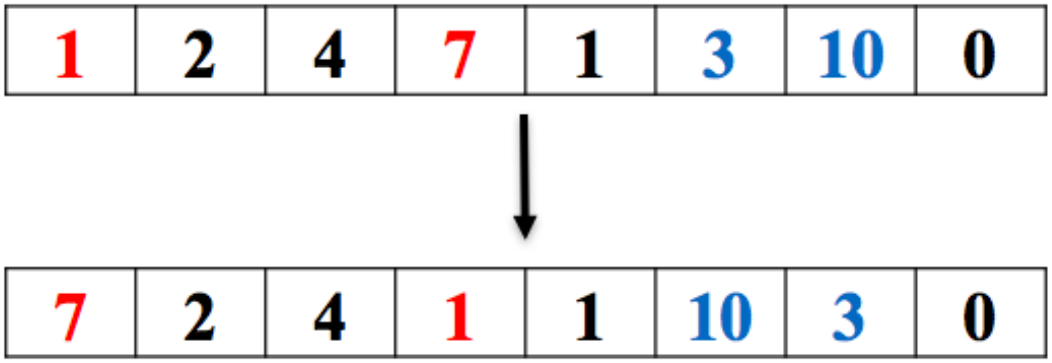

Figure 12. Example of shift mutation. A port priority is given by the port sequence [4,1,2,5,3,6,7,8]. With a move max of 1, the decap in port 4 can only exchange places with port 1. The decap in port 6, can exchange places with the decap in port 3 or port 7.

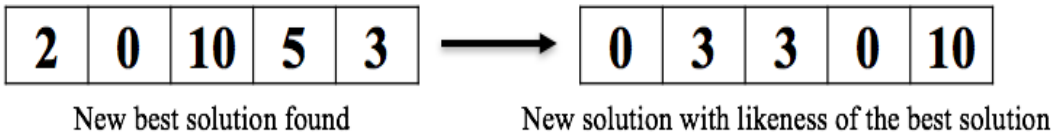

Figure 13. Example solution for a newly generated population. Solutions are created in the likeness of the best-known solution.

3.5. NEWLY IMPLEMENTED FUNCTIONS

Two new functions were also added to help with GA convergence. The first function is a brute force check. When a solution that can meet the target impedance is found, decaps will be removed from the solution, one at a time, using the reverse of the port priority. One decap is removed first, and if the target impedance is still met after removing, that port will be kept empty and the procedure is repeated with the next port. If, after removing the next decap the target is still met, that port will be kept empty. If the target is not met, that port keeps its decap and the next port is checked.

The second function added will generate a new population whenever a solution that meets the target is found, with the new population generated in the likeness of the improved solution. The new solutions will use similar decap types and similar decap distributions, with the goal of pointing towards a search space more likely to yield better solutions. The total number of decaps in the new solutions will then be decremented by 1. An example of a newly generated solution in this vein is given in Figure 13. In this work, the two functions can call each other.

4. VALIDATION

To validate the effectiveness of the GA modifications, comparisons are made between this work, the algorithm of [11], and a canonical GA [15] implementing (1) and (2) as the fitness functions. 9 test boards are generated using the method proposed in [19,20], with 3 boards of 25, 50, and 75 decap ports each. For each test board, 3 target impedances of R and RL types are generated. Each GA was run 5 times for each target

impedance. For all test cases, the frequency range of interest is 10 kHz – 20 MHz. All test cases were run with population sizes of 50 solutions and 50 generations of search.

Two examples will be looked at in more detail; 1 example with 50 ports and R-Type target and 1 example with 75 ports and RL-Type target. The results for the Augmented GA, GA [11] and the canonical GA, over all generated test cases, impedance targets, and algorithm runs, are given in Figure 14. Vertical lines in Figure 14 separate boards with 25, 50, and 75 decap port locations. The current implementation of the GA leads to better end results, finding better solutions or finding solutions where GA [11] and a canonical implementation could not.

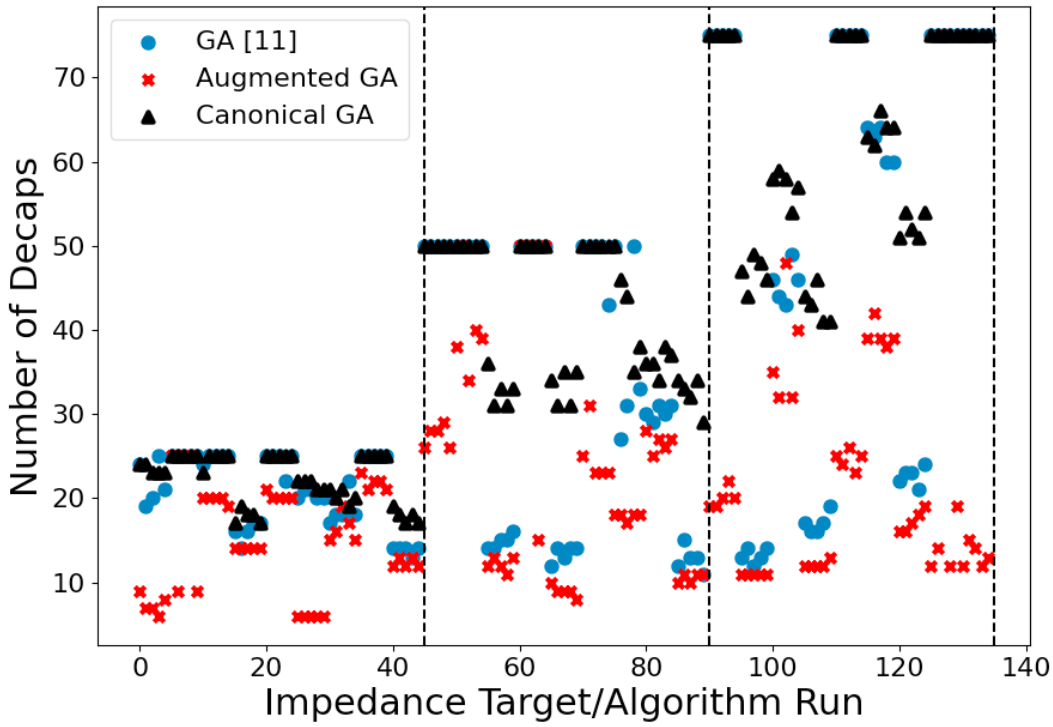

Figure 14. Overall result comparison of multiple GAs.

4.1. 50 PORT EXAMPLE WITH R-TYPE TARGET IMPEDANCE

The first example to highlight is a 50-port case, where the final GA results are given in Table 3. 'No Sol' in Table 3 indicates that no solution satisfying the target impedance was found. The Augmented GA could find solutions where the other GAs could not, using only about half the total number of ports. Figure 15 gives the convergence curve for each of the runs of the Augmented GA. Rapid improvement in the convergence curve comes from the implemented functions of Section III.E, especially due to the brute force check.

Table 3. Number of Decaps Found For 50 Port Case

Decap Type	Canonical GA [11]	GA [10]	Augmented GA (this work)
1	No Sol	No Sol	26
2	No Sol	No Sol	28
3	No Sol	No Sol	28
4	No Sol	No Sol	29
5	No Sol	No Sol	26

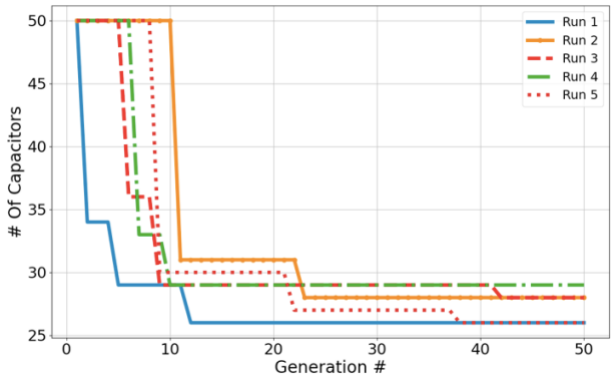

Figure 15. Convergence curves for augmented GA, 50 port case.

The distribution of the decaps in one of the 26 decap solutions is given in Table 4. The dominant portion of decaps used is of type #1. Anecdotally, of the 5 solutions found by the Augmented GA, all 5 solutions had a minimum of 21 type #1 decaps in the solution, indicating that there may be some minimum number of a specific decap type needed to even meet the target impedance. In initial population generation, the Augmented GA had forced every solution to have at least 20 type #1 decaps in every initial solution; no other decap type could be used to satisfy the critical point of 20 MHz. The Augmented GA identified a more promising search space and quickly found better solutions.

Table 4. Number of Decaps Found For 50 Port Case, Augmented GA

Decap Type	# of Each Decap in Solution
1	21
3	l
4	2
6	1
8	1

4.2. 75 PORT CASE WITH RL-TYPE TARGET IMPEDANCE

The second example to highlight is the 75-port case, where the convergence curves for GA [11] and the Augmented GA are given in Figure 16 and Figure 17 respectively. In general, the Augmented GA performs better, but there is more variance in this case compared to the 50-port case. Part of this inconsistency may be due to the difficulty of optimizing in the +20 dB/decade region of the target impedance. Around self and anti-resonances, the impedance might have slopes larger/smaller than +/-20 dB/decade, causing the impedance to cross the target at multiple points. Small changes to a solution could push some points over and some below the target.

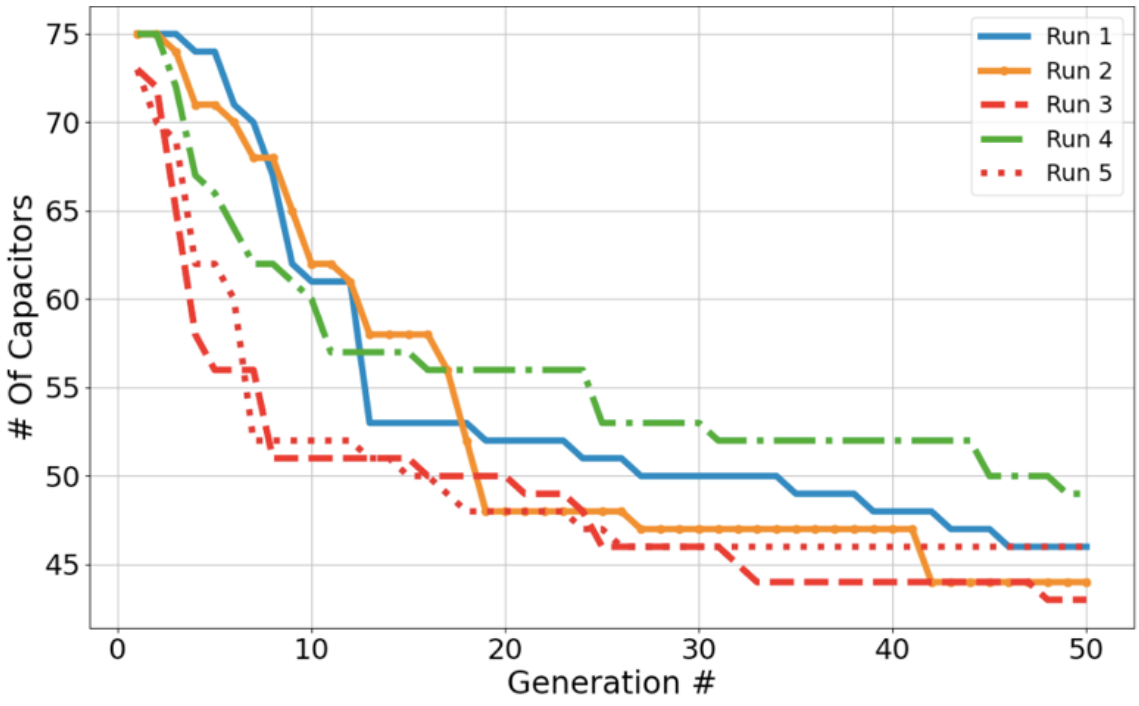

Figure 16. Convergence curves for 75 port case, using GA [10].

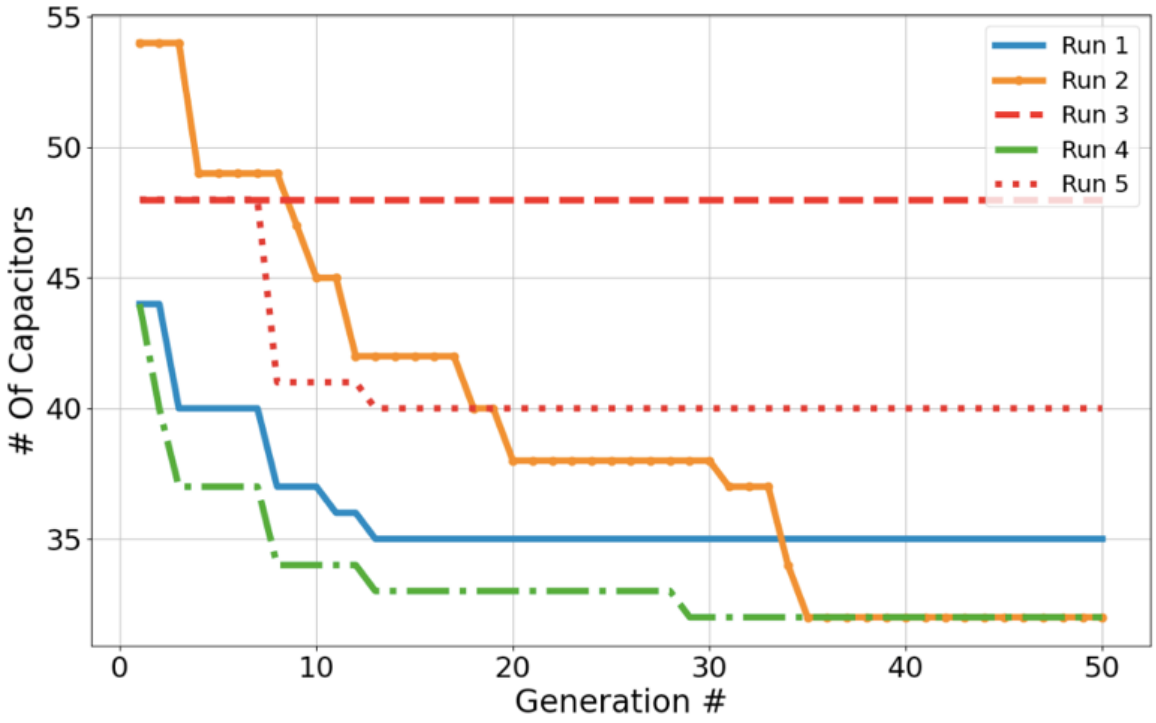

Figure 17. Convergence curves for 75 port case, using proposed Augmented GA

Table 5 gives the predicted distribution of decaps in the global minimum solution made by the Augmented GA vs. the actual distribution of decaps in one of the best solutions found (32 decaps). The '% of Expected Distribution' is the expected percentage, E_i, of each decap type appearing in the global minimum solution, and is calculated, for the *i*'th decap, using (7). The variables in (7) are the same as in (3), (5), and (6). $N_{i,f1}$ is the number of required decaps of type *i* to satisfy critical point f_1, and $N_{i,f2}$ for critical point f_2.

$$E_i = \frac{|N_{i,f1} - N_{max,f1}| + 1 + |N_{i,f2} - N_{max,f2}| + 1}{\Sigma_{j=1}^{K}(|N_{i,f1} - N_{max,f1}| + 1 + |N_{i,f2} - N_{max,f2}| + 1)} \quad (7)$$

Table 5. Presence of Decaps In Best Solution Of Augmented GA vs Augmented GA Predicted Distribution

Decap Type	% of Each Decap Type in Best Solution	% Predicted Distribution
1	9.375	17.1
2	53.125	18.3
3	12.5	17.3
4	9.375	17.1
5	3.125	9.5
6	3.125	6.4
7	3.125	7.6
8	3.125	5.8
9	3.125	0.33
10	0	0.33

While a full search was not performed to confirm that the global minimum solution is one using 32 decaps, the predicted distribution also dsoes not match well with the distribution of a best-known solution. For further validation of biasing the initial GA population, the decap weights are adjusted to give a '% Predicted Distribution' matching that of the solution in Table 5. Figure 18 gives the convergence curve of the Augmented GA with the adjusted weights. Due to the code implementation, no decap type has a weight of zero. The same percentage of the solution is devoted to solving each critical point as in the original algorithm runs. With the adjusted weights biasing a population's decap types towards a known solution, better solutions are found sooner with faster improvement and more consistency.

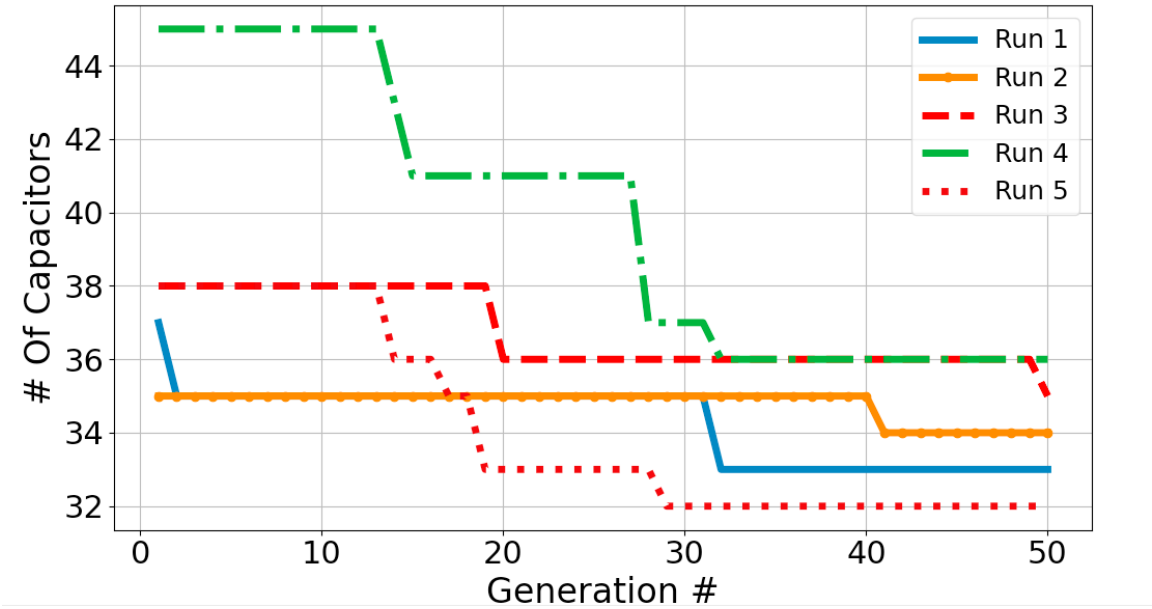

Figure 18. Convergence Curve for Augmented GA, with adjusted decap weights so that the '% of Expected Distribution' matches '% in Best Solution' for the case in Table 5.

5. CONCLUSION

By considering board parasitics and the shape of the target impedance, predictions are made of what immutable characteristics the optimal solution should have, such as the types of decaps used, the amounts used, and the locations used. For R or RL type target impedances, with increasing input impedance such as the case of a power net with VRM, it is expected for there to be a disproportionate number of smaller package size decoupling capacitors in the minimum solution, with smaller decaps placed nearer to ICs.

By considering these immutable characteristics in the initial population generation, it is assumed that the initial search space will be placed much closer to the space containing the optimal solution, than if the population was uniformly generated.

The result of the proposed augmented GA is compared with the algorithm of [11] as well as a canonical GA [15] implementation. It was found that the proposed GA could find better solutions or a solution that could meet the target impedance at all as compared to [11] and [15].

The search space containing the optimal soluton can be estimated from the physics of the problem and from designer experience. By generating the initial population of solutions with characteristics of the optimal solution, the initial search space could be placed much nearer the optimum resulting in better and faster convergence. While the methodology proposed in this work is applied specifically to increasing input impedances as a function of frequency, and R or RL-type target impedances, the proposed strategy can be extended to different input impedances and arbitrary target impedance profiles.

REFERENCES

[1] E. Bogatin, "The Power Distribution Network (PDN)," in *Signal Integrity Simplified*, 3rd ed, Page 922-929, Boston: Prentice Hall, 2018, pp 922-929.

[2] M. Swaminathan, J. Kim, I. Novak and J. P. Libous, "Power distribution networks for system-on-package: status and challenges," in *IEEE Trans. Compon. Packag. Manuf. Technol.*, vol. 27, no. 2, pp. 286-300, May 2004.

[3] L. D. Smith, R. E. Anderson, D. W. Forehand, T. J. Pelc and T. Roy, "Power distribution system design methodology and capacitor selection for modern CMOS technology," in *IEEE Trans. Adv. Packag.*, vol. 22, no. 3, pp. 284-291, Aug. 1999.

[4] L. Zhang, W. Huang, J. Juang, H. Lin, B. -C. Tseng and C. Hwang, "An Enhanced Deep Reinforcement Learning Algorithm for Decoupling Capacitor Selection in Power Distribution Network Design," *2020 IEEE Intl. Symp. Electromag. Compat. & Signal/Power Integrity (EMCSI)*, 2020, pp. 245-250.

[5] H. Park et al., “Deep Reinforcement Learning-Based Optimal Decoupling Capacitor Design Method for Silicon Interposer-Based 2.5-D/3-D ICs,” *IEEE Trans. Compon. Packa Manuf. Technol.*, vol. 10, no. 3, pp. 467-478, March 2020.

[6] L. Zhang et al., “Decoupling Capacitor Selection Algorithm for PDN Based on Deep Reinforcement Learning,” *2019 IEEE Intl. Symp. Electromag. Compat. & Signal/Power Integrity (EMCSI)*, 2019, pp. 616-620.

[7] Seunghyup Han, Osama Waqar Bhatti, Madhavan Swaminathan, "A Knowledge Based Method for Optimization of Decoupling Capacitors in Power Delivery Networks," *IEEE Trans. Compon. Packag. Manuf. Technol.*, vol.12, no.5, pp.828-838, 2022.

[8] K. Koo, G. R. Luevano, T. Wang, S. Özbayat, T. Michalka and J. L. Drewniak, “Fast Algorithm for Minimizing the Number of decap in Power Distribution Networks,” in *IEEE Trans. Electromagn. Compat.*, vol. 60, no. 3, pp. 725-732, June 2018.

[9] J. Wang, Z. Xu, X. Chu, J. Lu, B. Ravelo and J. Fan, "Multiport PDN Optimization With the Newton–Hessian Minimization Method," in *IEEE Trans. Microw Theory Tech.*, vol. 69, no. 4, pp. 2098-2109, April 2021.

[10] F. De Paulis *et al.*, "A Methodical Approach for PCB PDN Decoupling Minimizing Overdesign with Genetic Algorithm Optimization," *2022 IEEE Intl. Symp. Electromag. Compat. & Signal/Power Integrity (EMCSI)*, 2022, pp. 238-243.

[11] J. Juang, L. Zhang, Z. Kiguradze, B. Pu, S. Jin and C. Hwang, A Modified Genetic Algorithm for the Selection of Decoupling Capacitors in PDN Design,” *2021 IEEE International Joint EMC/SI/PI and EMC Europe Symposium*, 2021, pp. 712-717.

[12] F. de Paulis, R. Cecchetti, C. Olivieri and M. Buecker, “Genetic Algorithm PDN Optimization based on Minimum Number of Decoupling Capacitors Applied to Arbitrary Target Impedance,” *2020 IEEE Intl. Symp. Electromag. Compat. & Signal/Power Integrity (EMCSI)*, 2020, pp. 428-433.

[13] J.R. Keuseman. Et. Al, “Capacitor Optimization in Power Distribution Networks Using Numerical Computation Techniques,” DesignCon, 2021.

[14] Z. Xu, Z. Wang, Y. Sun, C. Hwang, H. Delingette and J. Fan, "Jitter-Aware Economic PDN Optimization With a Genetic Algorithm," in *IEEE Trans. Microw. Theory Tech.*, vol. 69, no. 8, pp. 3715-3725, Aug. 2021.

[15] R. Solgi, “Genetic Algorithm”, pypi.org, https://pypi.org/project/geneticalgorithm/ (accessed Sept. 1, 2020).

[16] Murata, “Multilayer Ceramic Capacitor Measurement Data”, ds.murata.co.jp, https://ds.murata.co.jp/simsurfing/mlcc.html?lcid=en-us (accessed Sep. 28, 2022).

[17] K. F. Man, K. S. Tang and S. Kwong, “Genetic algorithms: concepts and applications [in engineering design],” in *IEEE Trans. Ind. Electron.*, vol. 43, no. 5, pp. 519-534, Oct. 1996.

[18] L. Zhang, “PDN modeling for high-speed multilayer PCB boards and decap optimization using machine learning techniques,” Ph.D. dissertation, Missouri Univ. Sci. Technol., Rolla, MO, USA, 2021.

[19] L. Zhang *et al*., “Efficient DC and AC Impedance Calculation for Arbitrary-Shape and Multilayer PDN Using Boundary Integration,” in *IEEE Trans. Signal and Power Integrity*, vol. 1, pp. 1-11, 2022.

III. SCALABLE CAPACITOR ESL CURVE FITTING FOR VARIOUS STACK-UPS

Jack Juang, Hanfeng Wang, Kelvin Qiu, Chulsoon Hwang

Department of Electrical and Computer Engineering, Missouri University of Science and Technology, Rolla, MO 65409

ABSTRACT

Decoupling capacitors are used in PCB power distribution networks to act as a low impedance return path for current and as a local source of charge. Achieving accurate board level simulations and measurement correlation require accurate capacitor models.

For a capacitor mounted to a board, there is a mutual coupling between the capacitor body and the return current on the return plane. The result is a stack-up dependent variation in the inductance added to the loop by the capacitor, which cannot be captured in a single measurement. In this work, a curve fitting scheme is proposed to interpolate the inductance of a capacitor, while accounting for this mutual coupling, in two measurements. The curve fitting method, as well as the variation in extracted inductance due to stack-up, is verified through simulation and measurement validation.

1. INTRODUCTION

As designs increase in complexity and data speeds, the amount of noise generated, in both amplitude and spectra, is an ever-growing concern for power and signal integrity engineers. Noise generated by the switching inside integrated circuits (IC) can propagate

'down the line,' whether through conduction or radiation, resulting in the best case, an acceptable degree in the degradation of signals, or in the worst case, a total system collapse. Required IC voltages are also continuing to drop, resulting in any generated noise on power rails and traces becoming proportionally more significant. To better manage noise and ensure performance, improvements can be made by optimizing board layout, defining limits for what is an acceptable amount of noise for continued operation, and many other methods. One way to deal with the presence of noise, especially for power lines, is the careful design and implementation of the power distribution network (PDN).

Broadly speaking, PDN design and power integrity are concerned with ensuring adequate power delivery to all onboard components, and that this power delivery is consistent over every component's range of frequency operation. The flowchart for a generic PDN is given in Figure 1.

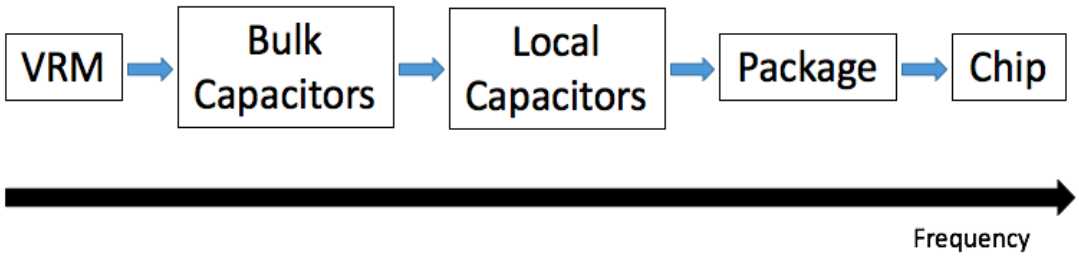

Figure 1. Generic PDN Flowchart

In the low frequency region, board power components including voltage regulator modules (VRM) and low dropout regulators (LDO) down convert and maintain the DC

voltages for components on the board. Bulk decoupling capacitors are used to smooth this supplied DC voltage by shunting noise through a low impedance return path. These capacitors also act as a source of charge when needed and reduce the impedance seen by ICs by counteracting VRM inductance. Smaller size capacitors are also placed locally on IC power lines for the same purpose. The effectiveness of a decoupling capacitor depends on the inductive parasitics that exist both internal to the capacitor and in the complete loop path. Careful choice of capacitor capacitance and consideration of capacitor parasitics is required for effective PDN design. For correct simulation of the PDN design, however, accurate capacitor models are required. If the capacitor models used for simulation and measurement are missing information, are not fit for the use case, or fit for use only in certain solvers (SPICE vs EM solvers), this may lead to over or under designs and increased costs.

Part of the inductance associated with a capacitor comes from a mutual coupling between the capacitor body and the current on the return plane, with this coupling a function of their separation. As this variation is stack-up dependent, this mutual coupling cannot be captured in a single model or measurement. From our measurement results and de-embedding method, we found the inductance of an 0201-package size capacitor to vary by about ~30 pH (~20%), when comparing extracted inductances at 0.2 mm, 2-layer board dielectric thickness to 1 mm, 2-layer board dielectric thickness. This difference, extracted at 200 MHz, translates also to a measured impedance difference of about 35 mOhms at 200 MHz.

In this work, we propose a simple method for inductance curve fitting to account for the variation in extracted inductance due to stack-up. The curve fit aims to interpolate

for the inductance of the points in between two characterized thicknesses, providing an estimation of capacitor inductance over a range of stack-ups.

A natural logarithmic function and a simpler square root function will be used for curve fitting and will be verified through simulation and measurement. For simulation verification, the largest error between the curve fit interpolated inductances and a discrete simulated point is 3.1 pH (2.44%). For measurement validation, the largest error between the curve fit inductance and a single measured validation case was 3.3 pH (2.53%).

2. CAPACITOR AND BOARD PARASITICS

The parasitics of a capacitor can be characterized as an equivalent series resistance (ESR) and equivalent series inductance (ESL), which when combined with the capacitance, results in a series RLC description of the behavior of a capacitor. For larger capacitor package sizes, the path length through the capacitor is longer, resulting in an ESL that generally increases with package size.

When mounting a capacitor to a board, the parasitic inductance of any via and trace connections will increase the total loop inductance. The additional inductance provided by a via or trace will in general be in a series connection with the RLC model of the capacitor, resulting in a decrease of the capacitor self-resonant frequency. The standard practice for minimizing this interconnect inductance is to keep the loop path as short as possible in order to reduce the overall loop inductance.

2.1. CAPACITOR BODY TO RETURN PLANE COUPLING

For a capacitor mounted to a board, the total inductance of the current loop path can be decomposed into smaller segments, consisting of vias, plane, and trace inductances, calculated analytically through partial inductances [1]. The lowest impedance return path for such a loop would be to return directly below the capacitor body and traces but on the return plane below. Two very basic current loops are given in Figure 2, with one loop consisting of a rectangular loop of wire, and one loop path containing the same loop of wire, with the replacement of a capacitor on the top segment of the loop. Each segment of the loop can be decomposed into partial self-inductances and the interaction between segments as partial mutual inductances.

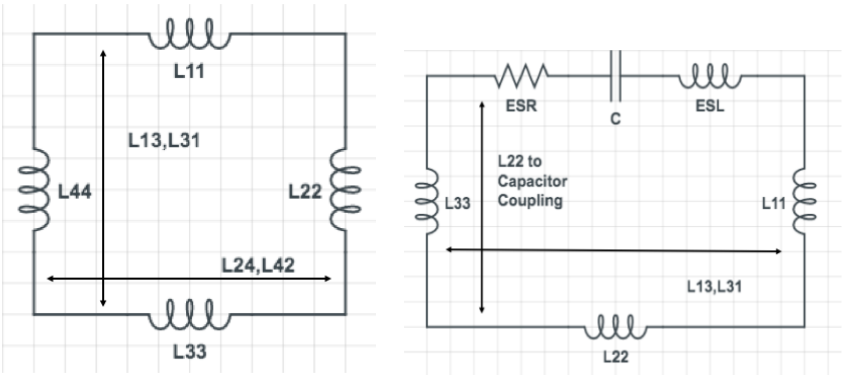

Figure 2. Basic Current Rectangular Current Loops. In the second case, the top segment is replaced with an RLC model of the capacitor

Replacing the top segment with a capacitor, there is still a complete loop path and still a mutual coupling between the capacitor and the bottom conductor segment. The partial self-inductance of the segment containing the capacitor would be different than the inductance of just a wire segment. As a result, the value of the capacitor body to the

bottom segment coupling is different compared to the coupling from wire to wire as the value of the mutual coupling is bounded by the self-inductances. The inductance added to the full loop is more than just the inductance associated with the physical structure of the capacitor, but also includes the mutual inductance. Replacing the bottom segment with a return plane and the vertical segments as vias, the result is a stack-up dependent variation in the inductance contributed by the addition of the capacitor to the full loop. There are several works describing this variation in inductance due to stack up with both simulation and measurement result [2,3]. The ESL in this case, could be extended to include the mutual coupling and be also stack-up dependent.

As the strength of this coupling varies with separation, the next question would be how much the inductance varies as a function of separation. Taking the case of a rectangular loop of wire, the mutual inductance between two segments can be calculated through integration of the total magnetic flux density, for the total flux produced by one segment that links another segment, divided by the current through the flux generating segment [1]. It can also be determined, and sometimes more easily, through substitution of the magnetic potential vector into the integration. Fundamentally, however, flux is produced from the magnetic fields generated by current-carrying segments, with the strength of this magnetic field inversely proportional to separation ‘r’. Increasing the separation between two segments, the field strength decreases with 1/r, the number of linking magnetic field lines decreases, and the total flux linked decreases. By considering the separation as a variable, integration of a 1/r dependent magnetic flux density results in the natural logarithm ln(r). We expect the variation in extracted inductance, to vary in some way, with the natural logarithm, and as a function of separation/stack-up.

The expression for the partial mutual inductance between two parallel wires in space is given by Equation 1 [1], showing a natural logarithmic variation. The lengths of the parallel wires are given by *l*, the wire center-to-center separation by *d*, and the radius of the wires given by r_w.

$$M = \frac{\mu_0}{2\pi} l * \left[\ln\left(\frac{l}{d+r_w} + \sqrt{\left(\frac{l}{d+r_w}\right)^2 + 1} \right) - \sqrt{\left(\frac{d+r_w}{l}\right)^2 + 1} + \frac{d+r_w}{l} \right] \quad (1)$$

Extending the case to either a single wire over a return plane or that of a three-dimensional conductor over a return plane, if the dimensions of the conductors are held constant and only the separation of the conductors varies, there should be a natural logarithmic variation in the inductance as a function of the conductor separation.

3. CAPACITOR MODELS FOR SIMULATION

For design verification, ideally measurements could be performed for the exact design stack-up, layout, and geometry. While it may be ideal to always perform measurement validation, it is not time or cost practical. Simulations are performed ahead of time using commercial tools such as PowerSI, Ansys, ADS, or 3D solvers before validation. For simulation, capacitor models can be in the form of measured S-parameters, SPICE models, or simple RLC circuits. Different models may be provided by vendors, though it may be possible that the measurement conditions are unknown, unfit for the use case, or even unknown if the models are fit for the use case [4].

Oftentimes vendor provided data is typically for only a single measurement, and while some information about the characterization method may be given, it may be that

not all measurement specifics are known. Specifically for this capacitor body-to-plane coupling, the stack-up information would need to be known in what cases the capacitor model is applicable for. If the designer does not have this information, they may run into issues of over or under design. In summary, knowing only the inductance associated with the physical structure of the capacitor is not enough to know how much inductance is contributed. Measured models including the capacitor body-to-plane coupling give more information, but as it is stack-up dependent, a single measurement/model is not enough.

While it is not practical for the designer to characterize a prototype board for every design iteration, it would also be impractical for the vendor to measure and provide capacitor models for all potential stack-ups. However, for the test fixtures used for characterization, if only the stack-up is varied, then the variation in the de-embedded capacitor inductance should be predictable as it is related to the capacitor to return plane separation. In this work, a curve fitting scheme is proposed to capture the variation in the mutual coupling between the capacitor body and the return plane. The curve fit will be performed by characterizing capacitors at two different stack-ups, one point at a smaller/shorter stack-up, and another at a taller stack-up.

4. CHARACTERIZATION METHOD IN SIMULATION AND MEASUREMENT

For capacitor characterization, many methods have been published, including PCB waveguides with micro probing, PCB characterization fixtures, and others. For this work, the capacitor characterization method used is the same as in [5,6]. A 2-port measurement method using microprobes was employed for this work, extracting the

parasitics of the capacitor and parasitics of the test fixture from the S21 instead of S11 for more measurement accuracy [7]. Boards in which to mount capacitors and boards for de-embedding are designed for validation of an ESL curve fitting scheme that accounts for the mutual coupling.

4.1. BOARD DESIGN IN SIMULATION AND MEASUREMENT

The designed boards are of 2.8 mm × 3.3 mm dimensions, and consist of two layers, top and bottom copper at 1oz thickness, and FR-4 dielectric in between. Two pads are located at the top with which to land microprobes for the 2-port measurement. Through-hole vias on the upper pad connect down to the bottom layer, where the current travels on the bottom plane, up the bottom via, through the capacitor, and returns to the ground point of the microprobe. For the de-embedding fixture, the via for the capacitor is instead shorted to the top plane.

We can calculate the shunt impedance associated with the mounted capacitor and the fixture contribution, the Z21, using the measured S21 and Equation 2 [7].

$$Z_{21} = 25 * \frac{S_{21}}{1-S_{21}} \tag{2}$$

The extracted impedance of the capacitor mounted to the test fixture, using Equation 2, is given as $Z_{cap+fixture}$ and the extracted impedance associated with the de-embedding fixture, is given as Z_{short} [5,6]. The impedance of the de-embedding fixture is subtracted from the impedance with a capacitor mounted using Equation 3. This removes the inductance contributed by the fixture such as the via inductance and the plane spreading inductance.

$$Z_{cap} = Z_{cap+fixture} - Z_{short} \tag{3}$$

To capture the inductance of the capacitor in simulation, a conductive block is created as in [3]. In the work of [3], it was found that substituting a full 3D model of a capacitor with all its internal plates with a solid conductive block gives an acceptable inductance approximation, provided the external and internal dimensions of the capacitor are accurate. For capturing the variation in inductance due to stack-up, the difference in inductance from using a solid block versus discrete plates should not significantly matter. In this work, the capacitor used for characterization is of nominal 2.2uF capacitance and 0201 package size. For the capacitor block in simulation, some dimensions were taken from the datasheet, and some dimensions were obtained by cross sectioning the capacitor and taking measurements under an optical microscope. Figure 3 gives the 3D models for the test fixture and de-embedding fixture used, with the same dimensions as the fixtures to be manufactured.

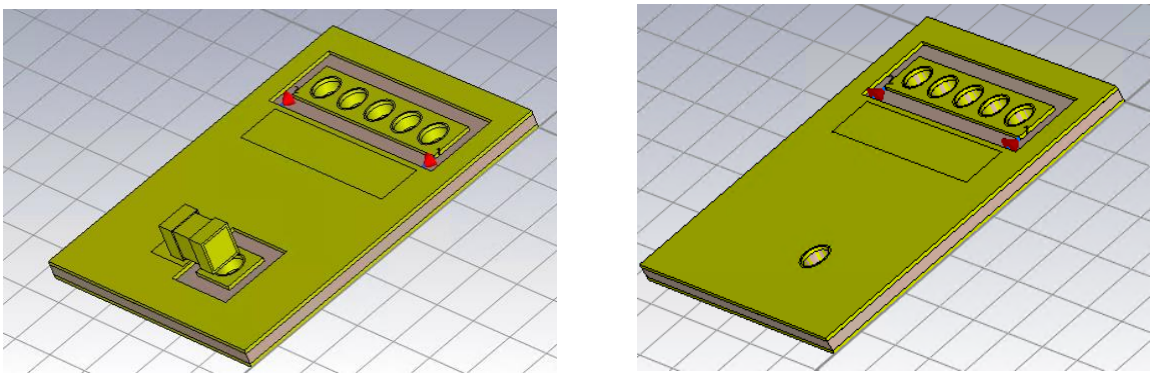

Figure 3. Simulated and Manufactured Board Designs. On the left is test fixture with mounted capacitor. On the right is the fixture for de-embedding

5. SIMULATION VALIDATION AND CURVE FITTING SCHEME

For verification of both a variation in extracted inductance as well as the feasibility of a curve fitting scheme, simulations were first performed on test fixtures simulated in CST. In the simulation, discrete ports are placed across the upper two rectangular pads, mirroring the placement of microprobes in measurement. The frequency range of interest used for the extraction is 100 kHz to 200 MHz, with the inductance extracted at 200 MHz. The inductance per vendor provided data, for their measurement setup, fixture design, and de-embedding method, is about 150.54 pH at 200 MHz . The simulation setup used in this work is given in Figure 4, for both the capacitor mounted board and the de-embedding test fixture.

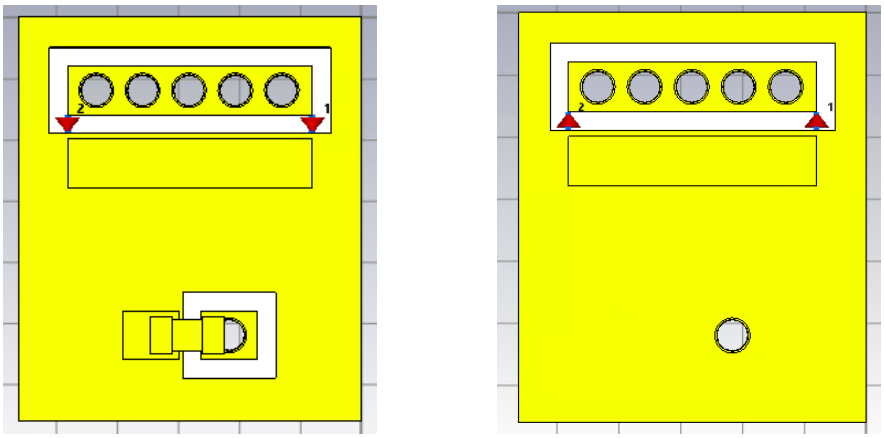

Figure 4. Simulation Setup with Capacitor mounted fixture and de-embedding fixture

5.1. SIMULATION VERIFICATION OF EXTRACTED ESL

Two port simulations are performed for both the capacitor mounted fixture and the de-embedding fixture, for various dielectric thicknesses, to capture the variation in extracted capacitor inductance. The inductance of the capacitor was extracted by subtracting the contribution of the fixture from the fixture with capacitor mounted and is calculated using the imaginary part of the simulated Z21s. The results of the extracted inductances are given in Table 1, for various dielectric thicknesses. From 0.1mm to 1 mm dielectric thickness, for the de-embedding method and board design in this work, the inductance varies by about 40pH, or 28% variation.

Table 1. Simulated Extracted Results for Various Dielectric Thicknesses

Dielectric Thickness (mm)	Fixture Mounted with Capacitor Inductance (pH)	De-embedding Fixture Inductance (pH)	Extracted Capacitor ESL (pH)
0.1	171.24	69.88	101.36
0.2	268.83	150.21	118.62
0.3	366.53	239.47	127.06
0.4	466.84	334.81	132.03
0.5	569.41	434.43	134.98
0.6	673.67	536.33	137.34
0.7	779.14	640.29	138.85
0.8	885.46	745.36	140.10
0.9	992.38	851.26	141.12
1.0	1099.80	957.92	141.88

5.2. CURVE FITTING SCHEME

From simulated results, it is found that the extracted inductance of the capacitor increases with plane separation. The rate of increase of the extracted inductance also

decreases as the separation increases meaning that the coupling weakens as the separation between the planes increases.

Taking the thickness of the dielectric to be infinity, the extracted inductance then is the inductance of the capacitor associated only with the physical structure of the capacitor, any horizontal coupling or coupling related only to the capacitor body, and any part of the characterization fixture that was not de-embedded. Any coupling from the capacitor body to the bottom plane should be decayed to zero. Taking the dielectric thickness to be zero, the extracted inductance for the capacitor should be zero as the planes are shorted together. These two points together form limits, starting at 0 H inductance for the capacitor at a dielectric thickness of 0 m, to some limit value as the dielectric thickness approaches infinity. These two limits alone are not enough for curve fitting as no information can be extracted regarding the exact rate of decay of the mutual inductance. At a minimum, information from two finite data points between thicknesses of 0 m and infinity to describe the rate of mutual inductance change.

For verification of curve fitting, the simulated, de-embedded inductances at 0.1 mm and 1 mm will be used. Dielectric thicknesses in between will then be interpolated from the curve fit and checked against the discrete simulated points of Table 1. The critical point for curve fitting is the identification of appropriate fitting functions, which was to some extent determined through trial and error. Two functions are proposed and used for curve fit, one using the natural logarithmic function, and one using the square root. The curve fitting was performed using MATLAB’s inbuilt Curve Fitter app.

5.3. NATURAL LOG CURVE FITTING WITH SIMULATED RESULTS

A proposed curve fitting function using the natural log is given in Equation 4, where *x* is the dielectric thickness:

$$L = a + b * \frac{\ln(x+c)}{x+1} \tag{4}$$

To relate the function to physical properties, the *a* term would describe the limit value that is approached by the de-embedded inductance as the dielectric thickness increases, with units of inductance, and *b* and *c* to some extent, describing the variation in mutual coupling as related to the physical dimensions of the capacitor and the fixture. At this point, it is unknown what physical properties they are related to, or how to determine them analytically. The natural log in the second term describes the weakening of the coupling between the capacitor and the return plane. The additional x+1 term was added as it yielded a slightly better fit, though the fit is still adequate without. With 3 unknown constants in *a*, *b*, and *c*, a minimum of 3 points is required for curve fitting. In this case, the points used will be the simulated, extracted inductances at a thickness of 0.1 mm and 1 mm, and the assumption that, for a dielectric thickness of 0 mm, the inductance extracted is 0 H. One caveat to mention is the curve fit value of *c* for this case. The value fit for *c* is much smaller than 1, so the sign of the logarithmic term is negative and increasing for dielectric thicknesses between 0 mm and 1 mm.

The result of the curve using the three points is given in Figure 5, with a maximum error between a discrete simulated point and curve fit point of 3.1 pH (2.44%) at 0.3 mm dielectric thickness. For this fit, the value of *a* was 141.9, *b*, 19.41, and *c*, 0.0006685.

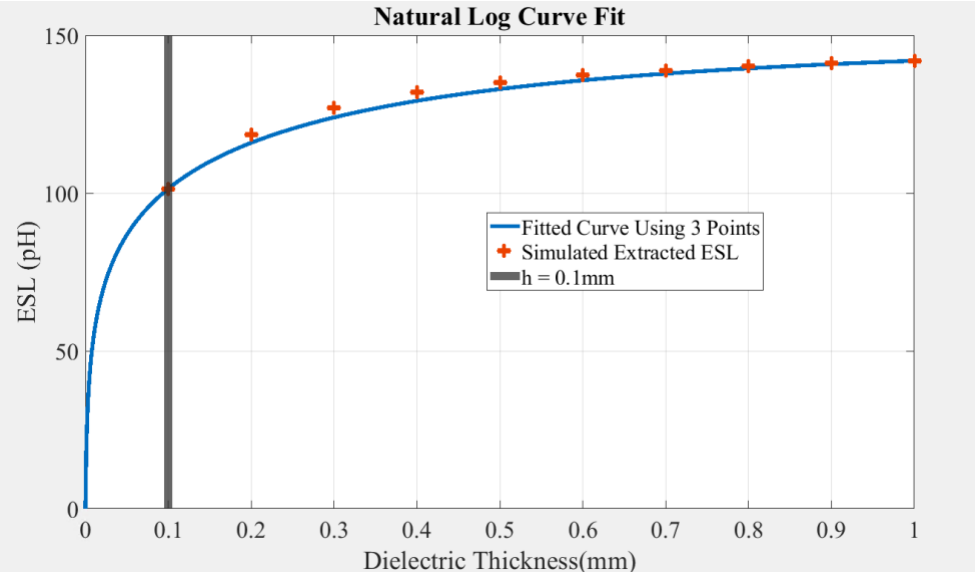

Figure 5. Simulation Curve Fitting Using Natural Log Function

While the fit is good, there is an issue with using this fitting function. The natural logarithmic function ln(x) approaches negative infinity as x approaches 0 and changes sign at x = 1. The curve fit value of *c* is very small, but allows the curve fit function to be defined at x = 0 and allows the logarithmic term to be negative for x < 1. For x > 1, the value of the curve fit function will initially increase, hit a peak value, then decrease, approaching the value of *a*. In the range of 0 mm to 1 mm this behavior is no issue as the extracted inductance is monotonic. The curve fit function, however, indicates that the mutual inductance is not continually decreasing with increased separation which does not align with the physics.

5.4. SQUARE ROOT CURVE FITTING WITH SIMULATION RESULTS

A separate curve fitting function using the square root is proposed, and is given in Equation 5, where *x* is the dielectric thickness.

$$L = a * \sqrt{\frac{x}{x+b}} \quad (5)$$

In Equation 5, *a* again describes the limit that is being approached in the de-embedded inductance as the thickness approaches infinity. The square root term describes the decay in the mutual coupling, which in this case is monotonically decreasing for $b > 0$. Only a single constant *b* is left to describe the decay in coupling due to the geometries of the capacitor and fixture. The function is also defined for an extracted inductance of 0 H for a dielectric thickness of 0 mm.

The curve fit using the square root function is given in Figure 6, with absolute errors of less than 1 pH (0.223%), between the simulated points and the curve fit points. The value of *a* is 150.1, and *b*, is 0.1193. An interesting note is that the limit value *a* approached by the two fitting functions are different, though it may be a mathematical consequence related to the rate of change of the natural logarithm and square root.

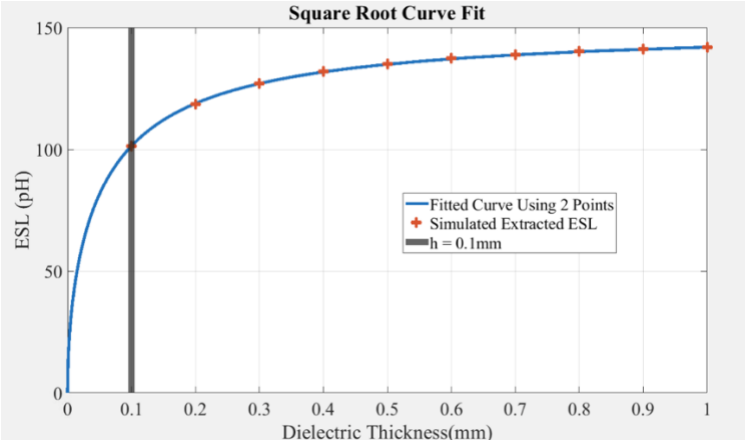

Figure 6. Simulation Curve Fitting Using Square Root Function

6. MEASUREMENT VALIDATION OF CURVE FITTING SCHEME

For measurement validation, three sets of the characterization fixtures were made, one at a dielectric thickness of 0.2 mm, one at 0.4 mm, and one at 1 mm. The curve fitting will be performed using the de-embedded capacitor inductance of the 0.2 mm and 1mm dielectric thickness boards. The curve fit expression will then be used to predict the de-embedded inductance of the 0.4 mm board, which will be verified through measurement. A 0.4 mm board was chosen for verification as the mutual coupling weakens with separation; it would be clearer to see the variation in extracted inductance if a thinner board is chosen for validation. Real designs may have stack-ups thinner than 0.2 mm or even 0.1 mm, which may result in a more significant variation in the inductance contributed by the capacitor to the loop than is demonstrated in this work. The manufactured PCBs are pictured in Figure 7.

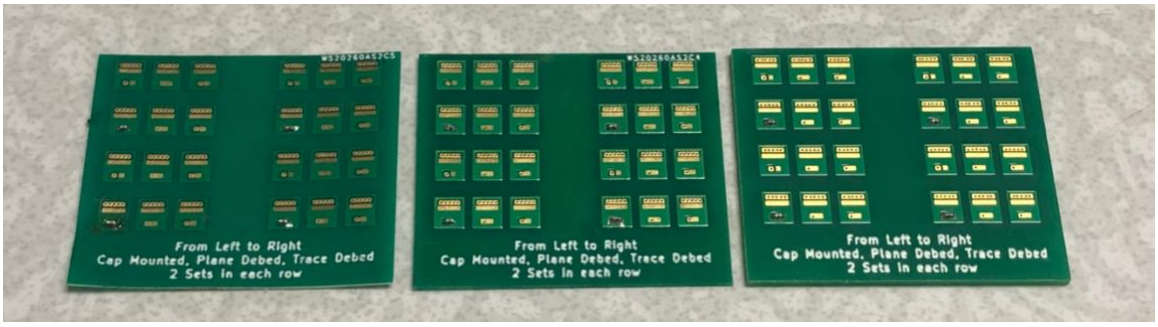

Figure 7. Manufactured PCBs for Characterization. From left to right, 0.2 mm, 0.4 mm, and 1 mm dielectric thicknesses

6.1. MEASUREMENT SETUP

The measurement setup is given in Figure 8. The VNA is the Agilent E5071C 100 kHz – 8.5 GHz VNA. Cables are connected to each of the two available ports, and connected at the other end to PacketMicro's RP-GR-121505 microprobes. Calibration of the setup is performed up to the tips of the microprobes using PacketMicro's TCS50 calibration substrate. Two port measurements were performed by landing the probes on the rectangular pads across the top of the manufactured PCBs. The distance between the probes when probing the fixtures, however, is different from the distance between probes when performing the thru-calibration and could lead to differences in probe-to-probe coupling. When probing, we tried to keep the probe separation the same between all fixtures, but it was based only on an eye test. Inductance from soldering was not calculated or de-embedded, though we tried to push the capacitors as flat as possible to the board when soldering. Current chokes were not used to account for the internal VNA ground loop. The measured S-parameters were exported for post-processing in MATLAB. Measurement was performed from 100 kHz to 200 MHz, with the inductance extracted at 200 MHz.

6.2. MEASUREMENT RESULTS AND CURVE FITTING

On each fixture, 4 capacitors were soldered and their parasitics extracted. Figure 9 plots the extracted impedance for one sample at each fixture thickness. When the inductance dominates the behavior of the capacitor, the extracted impedance at 200 MHz is seen to vary. The extracted capacitor inductance for every sample and for each of stack-up is given in Table 2.

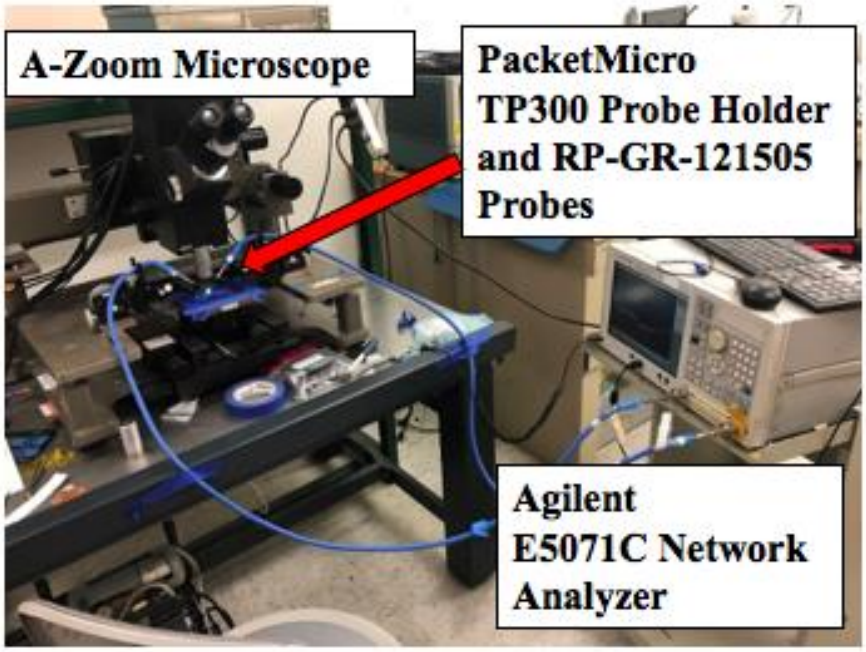

Figure 8. Measurement Setup

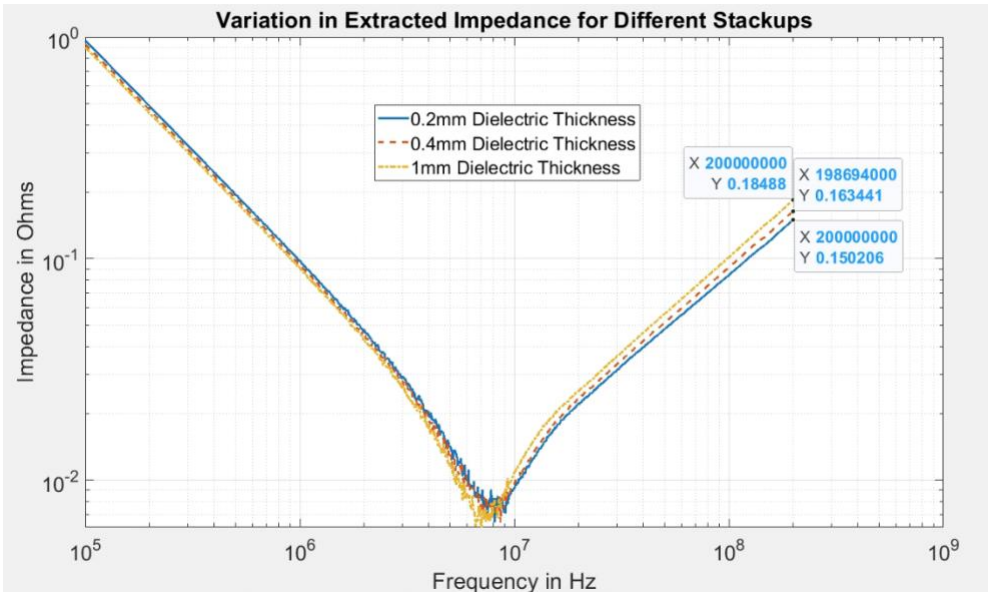

Figure 9. Extracted Impedance at 200 MHz for Various Dielectric Thicknesses

Table 2. Measured, Extracted Inductances for Various Dielectric Thicknesses

Inductance at 0.2 mm thickness (pH)	Inductance at 0.4 mm thickness (pH)	Inductance at 1 mm thickness (pH)
115.37	127.02	145.53
127.64	127.14	**120.74**
113.44	127.83	143.35
121.22	126.59	143.22

From results, the extracted inductance can have variation in the absolute value, for the same stack-up, which may be related to the orientation of the capacitor plates relative to the PCB plane [9,10]. For curve fitting, extracted inductances that appear to be from the same plate orientation (progressive increase in extracted inductance as thickness increases) is used and their values averaged. The samples not used in the averaging are bolded in Table 2. The averaged values, used for curve fit, are given in Table 3, along with the simulated extracted inductances at their respective dielectric thickness.

Table 3. Averaged Measured Extracted Inductances

Dielectric Thickness (mm)	Averaged ESL for Curve Fitting (pH)	Simulated Extracted ESL (pH)
0.2	114.40	118.62
0.4	127.15	132.03
1	144.04	141.88

For the average measured ESL, compared to simulated results, the difference for the validation thicknesses was about 4.88 pH, with a percent difference of 3.7%. Though there is a close correlation between measured and simulated extracted inductance, it may be a coincidence due to measurement and calibration setups. For both measured and simulated data though, the extracted inductances is seen to vary with stack-up. Curve

fitting is performed using both the natural log and square root functions and is pictured in Figure 10 and Figure 11 respectively.

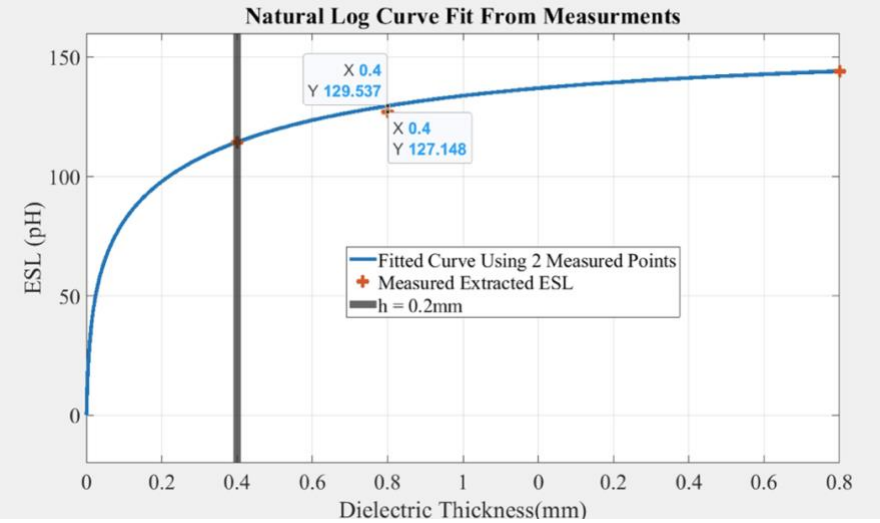

Figure 10. Measurement Curve Fitting Using Natural Log Function. Max error between measured and curve fit is about 2.4 pH

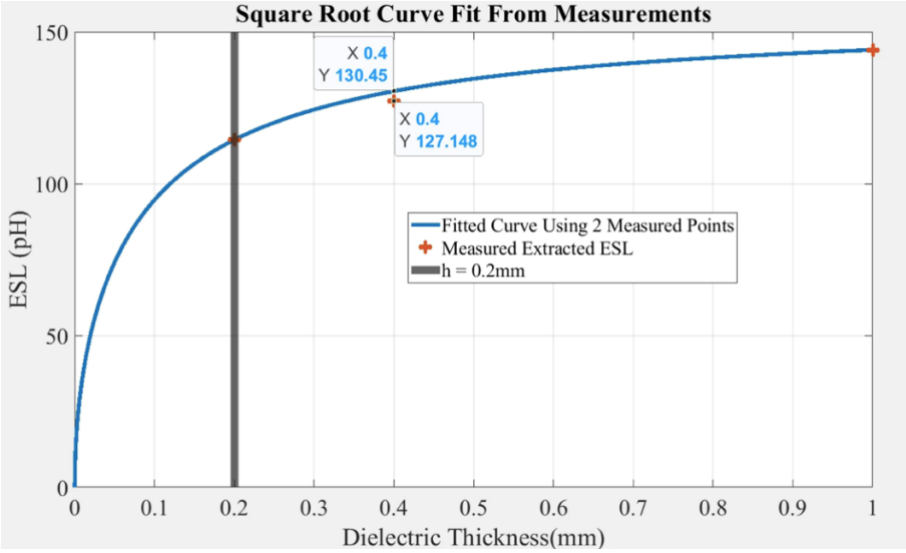

Figure 11. Measurement Curve Fitting Using Square Root. Function. Max error between measured and curve fit is about 3.3 pH

For the validation case, the largest error between the curve fit inductances and the measured inductances is 3.3 pH, or about 2.53% error.

7. CONCLUSIONS

In this work, we propose a scheme for curve fitting capacitor inductances over a range of dielectric thicknesses (plane separations). The extracted capacitor inductance will vary as a function of dielectric thickness because of the change in mutual coupling between the capacitor body and the return plane. Provided the same characterization fixture, with only a change in stack-up thicknesses, the variation in extracted inductance should be predictable as only this coupling is varying. With a minimum of one characterization measurement made at a thinner dielectric thickness (0.2 mm in this work) and one at a thick dielectric thickness (1 mm in this work), inductances in between can be interpolated.

Curve fitting is proposed using the natural log and square root functions. Simulation of capacitor inductances at discrete thicknesses has been performed and curve fitting has been applied. An absolute error of at most 3 pH, or 2.54%, occurred between discrete simulated inductances and simulated curve fit inductances. Measurements were performed for validation, using test fixtures of 0.2 mm dielectric thickness and 1 mm dielectric thickness for curve fitting, and one of 0.4 mm thickness for validation. Curve fitting for the measured values, there is a max absolute error of about 3.3 pH, or 2.53%, between the curve fit inductance value and the measured validation case.

REFERENCES

[1] Clayton R. Paul, “Inductance Loop and Partial”, John Wiley & Sons, 2010

[2] M. Li, S. Bai, T. Makharashvili, A. E. Ruehli, J. L. Drewniak and D. Beetner, "Impact of Accuracy of Capacitor ESL Values in HighSpeed Power Delivery Network Design," *2021 IEEE International Joint EMC/SI/PI and EMC Europe Symposium*, 2021, pp. 549-553, doi: 10.1109/EMC/SI/PI/EMCEurope52599.2021.9559279.

[3] T. Makharashvili *et al.*, "Accurate Inductance Models of Mounted Two-Terminal Decoupling Capacitors," in *IEEE Transactions on Electromagnetic Compatibility*, vol. 63, no. 1, pp. 237-245, Feb. 2021, doi: 10.1109/TEMC.2020.2987995.

[4] Benjamin Dannan, Steve Sandler, “Partial Inductance – The Secret to Correlating Simulation and Measurement”, EDICON Online, August 2021

[5] Wojewoda, Leigh & Hill, Michael & Radhakrishnan, Kaladhar & Goyal, Nitin. (2009). Use Condition Characterization of MLCCs. Advanced Packaging, IEEE Transactions on. 32. 109 - 115. 10.1109/TADVP.2008.2004811.

[6] Leigh Wojewoda *et al.*, “Measurements and Modeling of Microprocessor Decoupling Capacitors”, TecForum TF7, DesignCon 2005

[7] Keysight Technologies “Ultra-Low Impedance Measurements Using 2-Port Measurements” Application Note

[8] Masayuki Shimizu, *et al.*, “Measurement ESL/ESR of the Passive Components and Compensation of the Fixtures”, TecForum TF7, DesignCon 2005

[9] H. Kwak, H. Ke, B. H. Lee and T. Hubing, "Plate Orientation Effect on the Inductance of Multi-Layer Ceramic Capacitors," *2007 IEEE Electrical Performance of Electronic Packaging*, 2007, pp. 95-98, doi: 10.1109/EPEP.2007.4387133

SECTION

2. CONCLUSIONS

Two genetic algorithms were proposed for the selection and placement of decoupling capacitors in PDN design. Comparing to the results of other published algorithms, the proposed algorithms could find competitive or better solutions. Further analysis was performed looking at the structure of the best decap placement solution by considering input target impedance and board parasitics as inputs. It was found that the distribution of the best solution could be reasonably predicted from those inputs, a distribution that is in theory globally true and entirely independent of the method of optimization. By biasing the initial search space with such a distribution, solutions can be found quickly, and improvements can be made quickly.

A physics-based curve fitting method has also been proposed for interpolation of capacitor ESL. When mounted to a board, the inductance that a capacitor contributes to the full loop is dependent also on the coupling between the capacitor body and the return plane, with this coupling being a function of separation between the two. As such, it is not possible to capture this variation in just a single measurement or model. We propose curve fitting models based on the natural logarithm and the square root, and through measurement validation, we find an error of about ~3.3 pH, or 2.54%, between the interpolated curve fit inductance and the measurement validation case.

VITA

Jack Juang received his Bachelor of Science in Electrical Engineering from Missouri University of Science and Technology in May 2020.

Jack Juang received his Master of Science in Electrical Engineering from Missouri University of Science and Technology in May 2023

His research interests included PDN optimization, PDN modelling, and capacitor characterization.