EasyChair Preprint

№ 10950

High-Performance Computing-Based Fast Virtual Prototyping of Power Electronics Converters for Ground Vehicle Powertrain Systems

Sushma Amara, Yi Li, Cayden Wagner, Shuangshuang Jin, Zheyu Zhang and Christopher Edrington

September 22, 2023

High-Performance Computing-based Fast Virtual Prototyping of Power Electronics Converters for Ground Vehicle Powertrain Systems

Sushma Amara
School of Computing
Clemson University
Clemson, USA

Yi Li
Department of Electrical and Computer Engineering
Clemson University
Clemson, USA

Cayden Wagner
School of Computing
Clemson University
Clemson, USA

Shuangshuang Jin
School of Computing
Clemson University
Clemson, USA

Zheyu Zhang
Department of Electrical and Computer Engineering
Clemson University
Clemson, USA

Christopher Edrington
Department of Electrical and Computer Engineering
Clemson University
Clemson, USA

Abstract — **Power electronics converters, as an enabler for future power and energy system, are heavily used in military ground vehicle powertrain systems to provide reliable and efficient power conversion. Virtual prototyping of power electronics converters using computer-aided modeling and simulation to create and test power electronics circuits and systems before they are physically constructed can help engineers to optimize designs, reduce development time, and lower costs. This paper demonstrates the effectiveness of developing a High-Performance Computing (HPC)-based power electronics modeling and simulation approach to speed up the entirety of simulation time to support high-speed power electronics-enabled power architecture for demanding ground vehicle powertrain applications. First, a multiple power electronics building block (PEBBs)-based model for the virtual prototyping of power electronics converters is developed in MATLAB. This is then followed by a parallel implementation of the same model in Julia utilizing its high-performance compiled programming language's fast computing capability and a variety of supported parallel computing programming interfaces. Later, more PEBBs are added to the multi-PEBB model to test the scalability of the parallel simulation. It is observed that the Message Passing Interface-based parallel multi-PEBBs simulation in Julia is both fast and scalable without a noticeable increase in execution times when more PEBBs are added to the system.**

Index Terms— Power Electronics Simulation, High-Performance Computing, Simulation Software, Faster-Than-Real-Time Simulation

I. Introduction

Power electronics converters, as an enabler for future power and energy system, are heavily used in military ground vehicle powertrain systems to provide reliable and efficient power conversion. An essential component in power electronics converters is power electronics building blocks (PEBBs) which are known for their high efficiency in controlling and converting electrical power [1], as illustrated in Fig. 1.

PEBBs are platform-based approaches where basic building blocks are consistent with one another. PEBBs share defined functionality, standardized hardware, and control interfaces that are widely adopted for multiple applications, which result in high-volume production with reduced efforts needed in engineering [2]. A PEBB typically includes diodes, transistors, capacitors, inductors, and resistors that can be used in different configurations through advanced control algorithms and switching techniques to create various types of power electronic circuits such as rectifiers, inverters, and DC-DC converters [3]. PEBB also offers scalability and modularity, making them easy to integrate into larger systems to create complex power and energy systems [1].

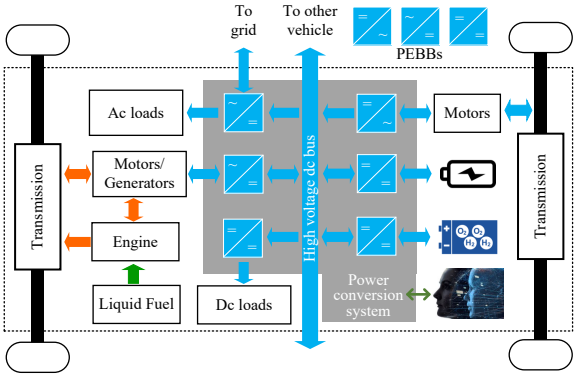

Fig. 1. Ground vehicle system with power and energy system emphasized, including energy conversion units enabled by PEBBs.

There are multitudes of research that have been published that is relevant to power electronics as enablers and on PEBBs approaches. However, none of the work on PEBB enablers has been translated to next-generation combat vehicle applications. In this work, we aim to enable fast virtual prototyping of PEBB-based power electronics converters for ground vehicle powertrain systems leveraging the latest high-performance computing techniques based on some prior work that we have performed relevant to this area [4-10]. This fast virtual prototyping of power electronics converters will allow us to create and test power electronics circuits and systems in a ground vehicle powertrain system before they are physically constructed and thus help engineers to optimize designs, reduce development time, and lower costs.

This paper is organized as follows: Section II gives the background about the platforms that can be used to simulate power electronics. Section III introduces the methodology and case study engaging the Multi-PEBB Modeling and Simulation using MATLAB, Julia Channels and Julia MPI. Section IV presents the results of various methodologies discussed in Section III. Lastly, Section V discusses the future work and summary.

II. Background

There are several simulation platforms that can be used for the virtual prototyping of power electronics components. MATLAB is one popular software tool that has many built-in functions and tools for simulating and analyzing power electronic circuits [11,12]. One of the advantages of using MATLAB for simulating PEBBs is its ability to handle complex mathematical equations and perform simulations for a wide range of power electronic circuits [13]. MATLAB/Simulink is a graphical simulation tool that allows engineers to model and simulate power electronics circuits and systems with a block diagram interface in a virtual environment, which can save time and reduce costs associated with physical prototyping [14]. They also offer various simulation models and analysis toolboxes that can help engineers to optimize designs and improve the overall performance of power electronics systems. Its advanced mathematical capabilities, graphical interface, and built-in toolboxes make it an ideal platform for simulating complex power electronic circuits.

However, simulating PEBBs using MATLAB can present some challenges in simulation speed, especially when dealing with large and complex circuits. First, simulating large and complex PEBBs can be computationally intensive and can take a long time to complete. This can be especially problematic when running numerous simulations to optimize circuit design parameters. Second, the choice of solver can also affect the stability of the simulation, especially when simulating circuits with high switching frequencies. Third, large and complex PEBBs can require a significant amount of memory to simulate, which can cause issues on computers with limited resources [15].

To address these issues, in addition to using efficient modeling techniques, choosing appropriate simulation settings, and optimizing the code, a more advanced simulation platform with parallel computing capability is also desired to overcome the computational constraints that come with MATLAB's interpreted language nature and memory-intensive bound for complex PEBBs simulation.

One solution to address these limitations would be to develop the simulation in Julia [16]. Julia is a high-performance, compiled programming language for technical computing, data science, and machine learning. It was first introduced in 2012 by a group of scientists, engineers, and programmers who sought to create a language that combined the ease of use of Python and the performance of traditional scientific computing languages like Fortran and C [16].

One of the primary advantages of Julia is its high performance. Julia is designed to execute code quickly, with a just-in-time (JIT) compiler that can optimize code on the fly, resulting in performance that can be comparable to that of traditional compiled languages like C and Fortran [16]. This makes it well-suited for computationally intensive tasks, such as simulations and numerical analysis. Julia's built-in support for distributed computing and parallelism allows it to take advantage of multicore processors or manycore processors for faster computations [16, 17]. As a result, we chose to develop our complex PEBB-based power electronics simulation for ground vehicle powertrain systems in Julia using parallel computing to achieve accelerated computational speed and scalability without sacrificing the simulation accuracy.

III. Methodology

A. Multi-PEBB Modeling and Simulation using MATLAB/Simulink

We first started the Power Electronics simulation on MATLAB as a benchmark based on the architecture in Fig. 2. The PEBB concept with a fixed topology results in a model with similar intrinsic variables, control variables, and disturbance variables for various energy conversion scenarios, therefore, generalizing a model in a universal format of state space equations becomes possible. In other words, PEBB enables not only a standardized power electronics hardware but can also be virtually represented by a generalized model. Hence, one universal PEBB model can be formed and programmed for later simulation with paralleled computing. This is beneficial for less complex and time-efficient modeling and scalable simulation for power electronics-intensive energy systems.

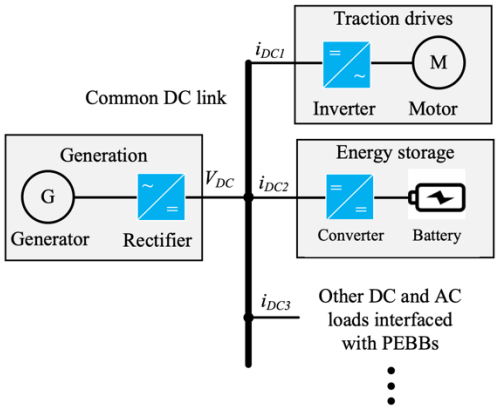

Fig. 2. Notional power and energy system enabled by PEBBs.

Specifically, based on the derivation of average model and *abc/dq* coordinate transformation [18], power electronics circuit can be defined by differential equations. For individual energy conversion scenarios with different control schemes (e.g., rectification, inversion, DC/DC conversion), different equations can be rearranged to the same format expressed by (1) where parameters $\vec{\boldsymbol{X}}$ is for intrinsic variables (e.g., (i_d, i_q, v_{DC}) as AC currents in the *dq* domain and DC voltage for rectifier), $\vec{\boldsymbol{u}}$ is for disturbance variables (e.g., (v_d, v_q, i_{DC}) as the AC input voltages in the *dq* domain and DC load current for rectifier). $\boldsymbol{A}$ and $\boldsymbol{B}$ are matrices with various coefficients to represent power stage parameters (e.g., L and C) and different control variables (e.g., d_d and d_q) [18].

$$\dot{\vec{X}} = \boldsymbol{A} \cdot \vec{X} + \boldsymbol{B} \cdot \vec{u} \quad (1)$$

Notably, the data through the information communication channel include DC link voltage from the generator rectifier PEBB to all load PEBBs (i.e., v_{DC} in Fig. 2), and DC load current i_{DC} from individual load PEBBs to rectifier PEBB (i.e., i_{DC1}, i_{DC2}, i_{DC3}, etc. in Fig. 2).

B. Multi-PEBB Modeling and Simulation using Julia -- Channels

To develop the parallel simulation of PEBBs in Julia, we implemented each PEBB component as an individual task running on a separate CPU thread/core. We started with the use of Julia's built-in advanced programming interface (API) -- Channel to enable the data exchange between each of these PEBB components. Channels are a type of data structure in Julia that allow for communication between parallel tasks. When creating a channel for each PEBB component and using these channels to communicate, the desired data could be exchanged between parallel tasks. Each PEBB component receives its perspective input values from its input channel, performs the necessary computations, and then sends the output values to its output channel for a receiving PEBB.

Using channels for PEBB simulation in Julia helps us simplify the implementation and organization of the simulation code and theoretically improve its performance through parallelism. However, it is important to carefully design the channel communication and task scheduling to avoid potential issues such as race conditions and deadlocks. To explore the best channel implementation, we developed three versions of communication mechanisms adaptively to study the performance of the parallel multi-PEBB simulation.

1) Channels Implementation with Sequential Communication

Our first attempt was a naïve implementation of parallel PEBB simulation with sequential communication. Fig. 3 illustrates this process with a (4 load PEBBs+ 1 rectifier PEBB) simulation using this implementation. Individual PEBBs run on separate CPU threads/cores and compute in parallel. The rectifier PEBB exchanges DC-link voltage values with all the load PEBBs in each time step of the entire simulation length through channels in Julia which are represented by the arrow lines in Fig. 3.

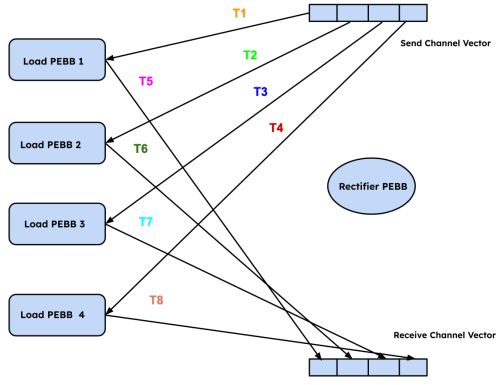

Fig. 3. Channel implementation with sequential communication.

Suppose the time taken for the rectifier PEBB to send data to load PEBB 1 through its respective channel and for load PEBB 1 to read from the channel is denoted as time T1, while the time taken for load PEBB 1 to send back data to the rectifier PEBB and read by the rectifier PEBB is denoted as T5. Similarly, the time taken for the rectifier PEBB to exchange data with other load PEBBs is denoted as T2 and T6 with load PEBB 2, T3 and T7 with load PEBB 3, and T4 and T8 with load PEBB 4, respectively. The total time required for this sequential communication would be the sum of all times, i.e., $\sum_{i=1}^{8} T_i$. This parallel implementation is straightforward but inefficient due to the accumulated communication costs. We regard it as a baseline implementation with improved performance and scalability compared to the traditional MATLAB implementation but still has a large room to improve the simulation performance with reduced communication cost.

2) Channels Implementation with One-Way Parallel Communication

As a second attempt, we first identified parallelizable communication channels on the rectifier PEBB side which do not necessarily require a specific order and updated the parallel PEBB simulation with one-way parallel communication. This allowed us to save significant time that would otherwise be wasted by processes waiting for preceding operations to complete in a sequential mode. In this modified model as shown in Fig. 4, the rectifier PEBB sends data to load PEBBs 1, 2, 3, and 4 simultaneously in time T1, while the returning communication still follows a serial implementation where the load PEBBs send data back to the rectifier PEBB individually in times T2, T3, T4, and T5, respectively. This modification was expected to significantly reduce the time required for the rectifier PEBB to send data to the load PEBBs, leading to a faster communication speed as $\sum_{i=1}^{5} T_i$.

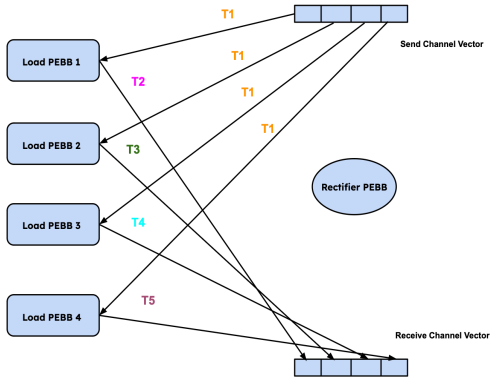

Fig. 4. Channel implementation with One-Way parallel communication.

3) Channel Implementation with Two-Way Parallel Communication

To further improve the communication efficiency, our third attempt was to parallelize the load PEBB side data passing to realize a two-way parallelization of the communication as shown in Fig. 5. The rectifier PEBB now sends data to all four load PEBBs 1, 2, 3, and 4, concurrently, which takes time T1.

Similarly, load PEBBs 1, 2, 3, and 4 return data to the rectifier PEBB simultaneously in time T2. The original idea was to further improve the communication efficiency to $\sum_{i=1}^{2} T_i$ as the total required communication time. However, due to the runtime differences in computation for each load PEBB, they may not necessarily reach the time point of data passing on their own CPU thread/core at the same time, which will result in less overlap in concurrent communication. The actual communication time would vary and falls anywhere between $\sum_{i=1}^{2} T_i$ to T1+4*T2, which is not necessarily faster than the channel implementation with one-way parallel communication.

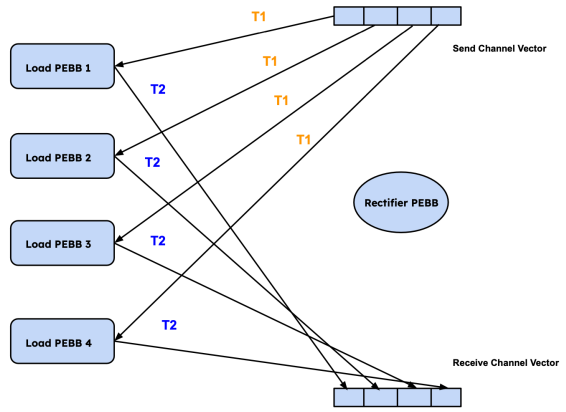

Fig. 5. Channel implementation with Two-Way parallel communication.

4) Limitation of Simulating PEBBs using Julia -- Channel

Although the parallel PEBBs implementation using the Julia channels can already improve the multi-PEBB simulation for its enabled parallelism in computation and reduced cost in communication, it still has some potential issues that we found out.

One challenge is that the channel method may not scale well to very large systems or systems with a high degree of complexity. This is because the channel method requires explicit communication between processes, which can result in a high overhead as the number of processes increases.

Another issue is that the channel method may be prone to deadlocks, where processes become stuck waiting for messages that will never arrive. To avoid deadlocks, careful design of the simulation code is necessary, which may require additional programming expertise and development effort to explicitly manage communication between processes.

Despite these challenges, the channel method remains a viable approach for simulating PEBBs in Julia, particularly for smaller systems or systems with moderate complexity. One should carefully consider the tradeoffs between performance, scalability, and development effort when choosing this simulation method.

C. Multi-PEBB Modeling and Simulation using Julia -- MPI

The constraints in Julia Channels communication led us to a natural consideration of an alternative parallel computing standard Message Passing Interface (MPI), which could enable efficient communication between a group of processes (collective communication) without creating specific communication channels between each pair of processes (point-to-point communication). This approach would eliminate the need for multiple channels between the rectifier PEBB and the load PEBBs, leading to a reduction in resource utilization and time consumption.

Julia does provide built-in support for MPI, making it a powerful tool for parallelizing computationally intensive tasks such as simulating PEBBs. To implement PEBB simulation using MPI in Julia, the PEBBs need to be split into several independent processes, each responsible for simulating one PEBB of the system. These processes can communicate with each other using common MPI communication paths (as illustrated in Fig. 6) through the MPI communication functions such as MPI Broadcast (one of the standard collective communication techniques being used when one process wants to send the same information to every other process in the communicator) and MPI Reduce (another standard collective communication technique being used which performs a reduction operation on data across all processes in a specified communicator and returns the result to a single process.) As shown in Fig. 7 and Fig. 8, MPI Broadcast was used by the rectifier PEBB to communicate with the load PEBBs 1, 2, 3, and 4, where the same message was transmitted from the source processor to all the target processors. On the other hand, MPI Reduce was used to combine the data coming from all the load PEBBs at the rectifier PEBB side. For example, the values 2, 3, 5, and 6 from the load PEBBs 1, 2, 3, and 4 on different CPU processes are aggregated to form a value of 16 on the process where the rectifier PEBB sits.

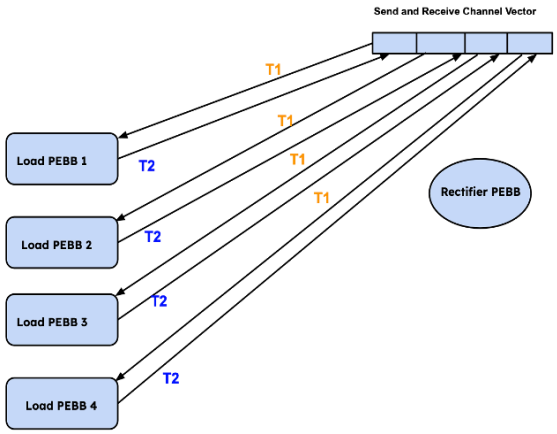

Fig. 6. Channel implementation with collective communication.

The MPI implementation of PEBB simulation in Julia can significantly improve the speed and efficiency of the simulation leading to an expected communication time of $\sum_{i=1}^{2} T_i$, as it allows for parallel processing on multiple CPU processors or nodes to enable faster execution times. Being a scalable parallel API to support parallel implementation on distributed memory architecture, it is also a favorable option for large-scale PEBB simulations of power electronics systems as demonstrated from our scalability tests.

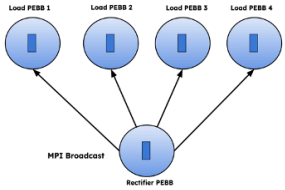

Fig. 7. MPI Broadcast.

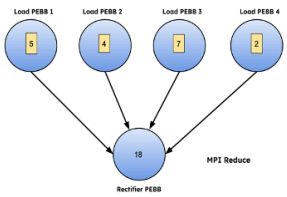

Fig. 8. MPI Reduce.

IV. Results

A. Simulation Speed and Scalability

Upon comparing the performance results of MPI, Channels, and MATLAB, the sequential MATLAB/Simulink was run as the baseline simulation without applying any parallelization techniques (e.g., the MATLAB Parallel Computing Toolbox). The results can be found in [10].

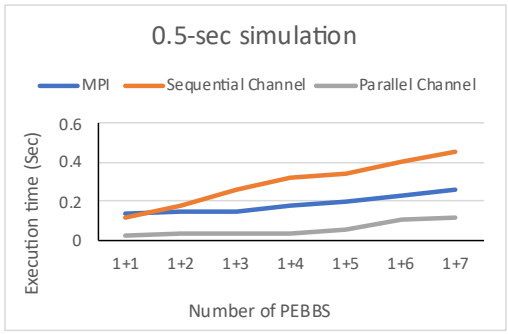

Fig. 9. Comparison between JULIA channel and MPI for 0.5s simulation.

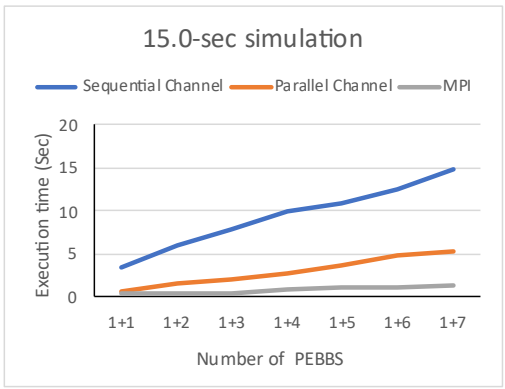

Fig. 10. Comparison between JULIA channel and MPI for 15s simulation.

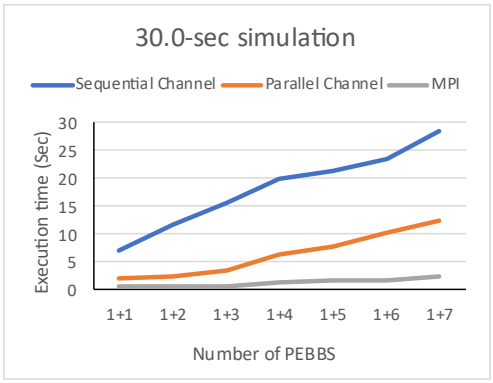

Fig. 11. Comparison between JULIA channel and MPI for 30s simulation.

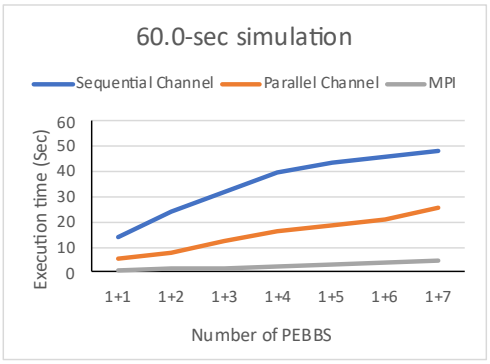

Fig. 12. Comparison between JULIA channel and MPI for 60s simulation.

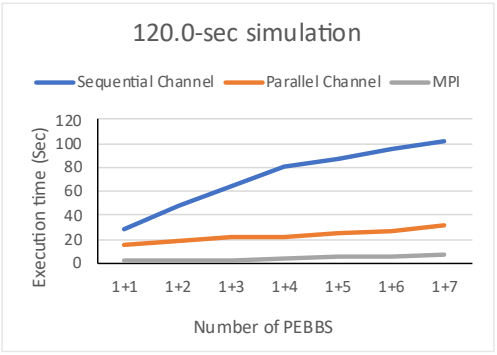

Fig. 13. Comparison between JULIA channel and MPI for 120s simulation.

Figs. 9-13 show a comparison of the execution times among three parallel implementations using Channels or MPI. A set of simulation lengths (e.g., 0.5 seconds, 15 seconds, 30 seconds, 60 seconds, and 120 seconds) have been applied to test the simulation performance. Other than the shortest 0.5s simulation case, MPI implementation stayed the fastest in overall simulation time.

Scalability-wise, when examining the execution times of all simulations, we can observe that the time cost with MPI implementation remained most constant from 1 load PEBB + 1 rectifier PEBB to 7 load PEBBs + 1 rectifier PEBB. This indicated a strong scalability of the simulation which introduced a neglectable overhead to simulation time when more PEBBs were to be added to the system for integrated simulation.

B. Simulation Accuracy

To validate the results of the Julia MPI version, a comparison of the accuracy between the Julia MPI version with the Julia sequential channel version was made. The R-square algorithm was employed, which measures the proportion of variation in the independent variable explained by the dependent variable in a regression analysis. The R-square values range from 0 to 1, with a value closer to 1 indicating a better fit of the model to the data. The R-square values for the outputs i_d, i_q, and v_{DC} of the MPI model were calculated and compared to the sequential channel model. The R-square values reported for the MPI model were v_{DC}: 1.0, i_d: 1.0, and i_q: 1.0, which indicate a perfect fit of the model to the data, explaining 100% of the variation in the dependent variable using the independent variable as shown in Figs. 14-16.

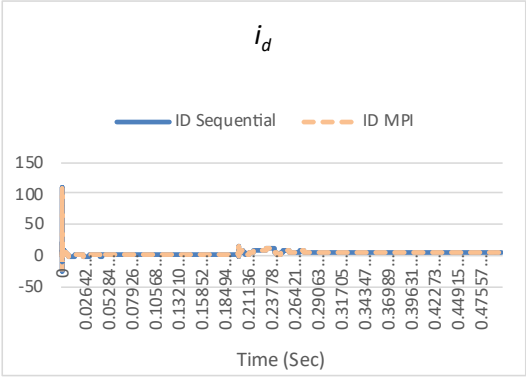

Fig. 14: i_d values for 1+2 combination on 0.5s simulation.

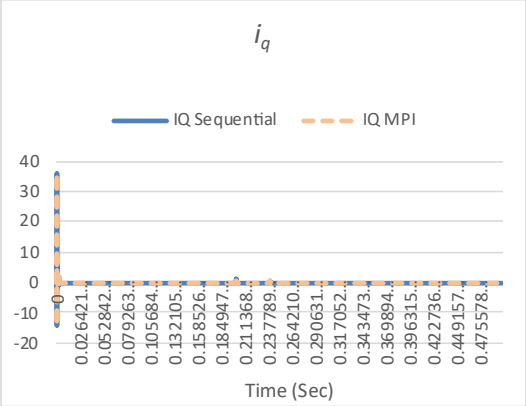

Fig. 15: i_q values for 1+2 combination on 0.5s simulation.

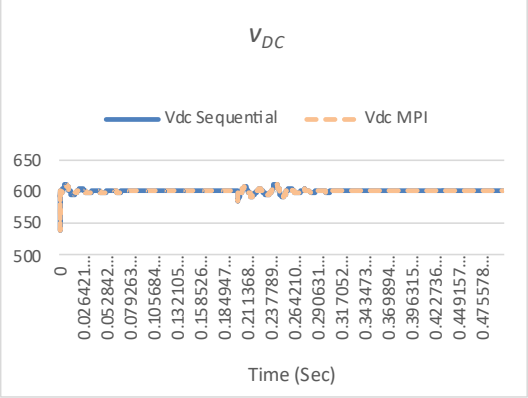

Fig. 16: v_{DC} values for 1+2 combination on 0.5s simulation.

V. Summary and Future Work

This paper introduces PEBBs and their applications in ground vehicle powertrain systems. We implemented a multi-PEBB model and simulation in MATLAB and Julia. The Julia Channels have several benefits but are not scaling well to very large systems, while the Julia MPI provides the best simulation speed as well as a constant scalability when the system is expanded to simulate more PEBBs. In the future work, we will continue to study the parallel implementation of diverse PEBB components in ground vehicle powertrain systems and investigate fast and scalable co-simulation between each component through high-performance computing techniques.

Acknowledgment

This work was supported by the Simulation Based Reliability and Safety (SimBRS) Program for modeling and simulation of military ground vehicle systems, under technical services contract W56HZV-17-C-0095 with the US Army DEVCOM Ground Vehicle Systems Center (GVSC). Distribution A. Approved for public release, distribution unlimited. (OPSEC 7522).

VI. References

[1] M. H. Rashid, Power Electronics Handbook: Devices, Circuits, and Applications, 3rd ed. Oxford: Elsevier, 2018.

[2] T. Ericsen, N. Hingorani and Y. Khersonsky, “PEBB - power electronics building blocks from concept to reality”, 2006 Record of Conference Papers - IEEE Industry Applications Society 53rd Annual Petroleum and Chemical Industry Conference, Philadelphia, PA, USA, 2006, pp. 1-7.

[3] R. W. Erickson and D. Maksimovic, Fundamentals of Power Electronics, 2nd ed. New York: Springer, 2001.

[4] Z. Zhang, H. Tu, X. She, T. Sadilek, R. Ramabhadran, H. Hu, and W. Earls, “High-efficiency silicon carbide (SiC) based buck-boost converter in energy storage system”, IEEE Industry Applications Magazine, Early Access.

[5] H. Gui, R. Rui, Z. Zhang, J. Niu, R. Ren, B. Liu, L. M. Tolbert, F. Wang, D. J. Costinett, B. J. Blalock, and Benjamin Choi, “Modeling and mitigation of multi-loops related device overvoltage in three level active neutral point clamped converter”, IEEE Transactions on Power Electronics, vol. 35, no. 8, Aug. 2020, pp. 7947-7959.

[6] F. Wang, Z. Zhang, T. Ericsen, R. Raju, R. Burgos and D. Boroyevich, “Advances in power conversion and drives for shipboard systems”, Proceedings of the IEEE, vol. 103, no. 12, Dec. 2015, pp. 2285-2311.

[7] G. Ozkan, B. Papari, P. Hoang, N. Deb and C. S. Edrington (2019). “An Active Thermal Control Method for AC-DC Power Converter with Sequence-based Control Approach”, 2019 IEEE Electric Ship Technology Symposium (ESTS), August 14-16, 2019.

[8] G. Ozkan, P. H. Hoang, P. R. Badr, C. S. Edrington and Papari, B. (2021). “Real-time thermal management for two-level active rectifier with finite control set model predictive control”, International Journal of Electrical Power & Energy Systems, vol. 131, 107057.

[9] P. H. Hoang, G. Ozkan, P. R. Badr, B. Papari, C. S. Edrington, “Integrating degradation forecasting into control and management system of DC microgrids”, 2021 IEEE Fourth International Conference on DC Microgrids (ICDCM), July 18-21, 2021.

[10] Y. Li, C Wagner, C Edrington, S. Jin, Z. Zhang, “Quantitative Analysis of Accelerated Power Electronics Simulation Using Advanced Computing Technology”, 2022 IEEE Applied Power Electronics Conference and Exposition (APEC), March 20-24, 2022.

[11] N. Mohan, T. M. Undeland, and W. P. Robbins, Power Electronics: Converters, Applications, and Design, 3rd ed. New York: Wiley, 2003.

[12] A. R. Bakhshai, “Power Electronics Simulation with MATLAB”, IEEE Ind. Electron. Mag., vol. 7, no. 1, pp. 41-48, March, 2013.

[13] J. L. Kirtley, “Simulation of Power Electronic Circuits”, IEEE Ind. Electron. Mag., vol. 3, no. 1, pp. 14-22, March, 2009.

[14] The MathWorks Inc., “Simulink - Simulation and Model-Based Design”, Available at: https://www.mathworks.com/products/simulink.html.

[15] R. W. Erickson and D. Maksimovic, Fundamentals of Power Electronics, 2nd ed. Boston, MA: Springer US, 2001.

[16] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, “Julia: A fresh approach to numerical computing”, SIAM review, 59(1), 65-98.

[17] J. Bezanson, S. Karpinski, V. B. Shah, and A. Edelman, “Parallel computing with Julia”, Proceedings of the 1st Julia User's Conference.

[18] S. Hiti, "Modeling and control of three-phase PWM converters," Dissertation, Virginia Tech., 1995.