Periodica Polytechnica
Electrical Engineering
and Computer Science

61(3), pp. 272-278, 2017
https://doi.org/10.3311/PPee.10934
Creative Commons Attribution ⓘ

RESEARCH ARTICLE

Design of HIL for Multilevel Inverter Using Zynq-7000 Platform – Part 2

Balázs Farkas[1*], Károly Veszprémi[1]

Received 24 April 2017; accepted after revision 25 May 2017

Abstract
Development of power electronic devices requires multi -disciplined engineering activities. These cover the thermal, electrical and software design. Due to this design complexity rapid prototyping methods and model-based design are becoming more and more important in the R&D projects in this field. This article is the second part of the series which introduces the development of Hardware-in-the-Loop (HIL) device for the simulation of Cellular H-Bridge inverter (CHB). Zynq-7000 platform is chosen as a hardware platform for HIL. This part focuses on the details of the model transformation, development of the hardware environment and the verification of the HIL. FPGA development is also demonstrated including interfaces, IPs and introduction of the resource utilization. Apart from them, operation of the system software in ARM core is also described including TCP/IP interface, IRQ handling and Matlab synchronisation mechanism. Finally, the Matlab interface and simulation results are introduced.

Keywords
Multi-level inverter, Zynq-7000, Hardware in the loop, CHB

[1] Department of Electric Power Engineering,
Faculty of Electrical Engineering and Informatics,
Budapest University of Technology and Economics
H-1111 Budapest, Egry J. utca 18, Hungary

[*] Corresponding author, e-mail: balazs.farkas86@gmail.com

1 Introduction

Nowadays researchers and developers face a challenge regarding the increasing product complexity and stricter time to market requirements. These trends imply that the rapid prototype solutions have been starting to be more popular [1]. One of the most promising rapid prototype solutions is the HIL based prototypes [2-4]. Due to the fact that the computational capacity is becoming cheaper and cheaper, there are currently many solutions to build HIL system with high performance. One of these devices is Zynq-7000 system on chip product. It is very attractive not only because of the high available resources but also because of the strong community behind it.

Development of power electronic devices requires multi-disciplined engineering competence and skill set as a result of the system complexity. Therefore, application of the HIL in the power electronic system design seems to be a promising opportunity to improve the efficiency of the design and development [5-7].

In the previous part of this article series, the development of the Simulink model is demonstrated. The aim of this article is to introduce the details of the Matlab Simulink model deployment to the aforementioned Zynq-7000 platform in case of CHB multi-level inverter. The deployment procedure covers the model transformation, FPGA design, development of the system software and Matlab interface as it can be seen in Fig. 1. Apart from technical details, the applied tool chain and verification environment are also shortly introduced.

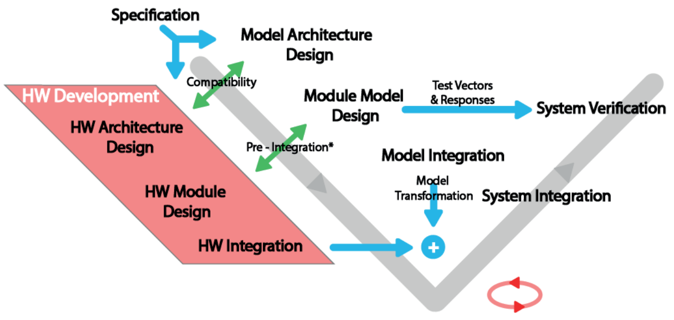

Fig. 1 Development process of the model and hardware in the V-model.

The article consists of four main sections. The first section introduces the applied tool chain. In the next section the details of the model transformation are described. The hardware environment is analyzed in the third section. Finally, the verification of the implemented system is demonstrated.

2 Applied Tool chain

This part of the article deals with the tool chain, which is used during the model transformation. The application of an efficient tool chain can extensively contribute to the success of the development. Due to the fact that an efficient tool chain makes the model transformation much easier, the development efforts can be mainly focused on models at high abstraction level. In this case, the observability of the system is much better than it would be at any lower abstraction layer. Another benefit of it is that the probability of the design error can be reduced thanks to the automated or semi-automated workflow. It is also beneficial that the tool chain which includes the highly supported tools can provide extra services like automatic driver generation, communication stack integration, inherent debug interfaces, data coherency management etc.

Fig. 2 introduces the details of the applied tool chain. At the top of the tool chain the Matlab can be found. It is responsible for the model development, handling of the number representation, code generation and verification. Simulink is used for model development. The transformation of the number representation is based on the Fixed Point Toolbox. The code generation can be executed by the SimulinkCoder or Embedded Coder. In case of the verification Simulink and Instrumentation Toolbox are extensively used.

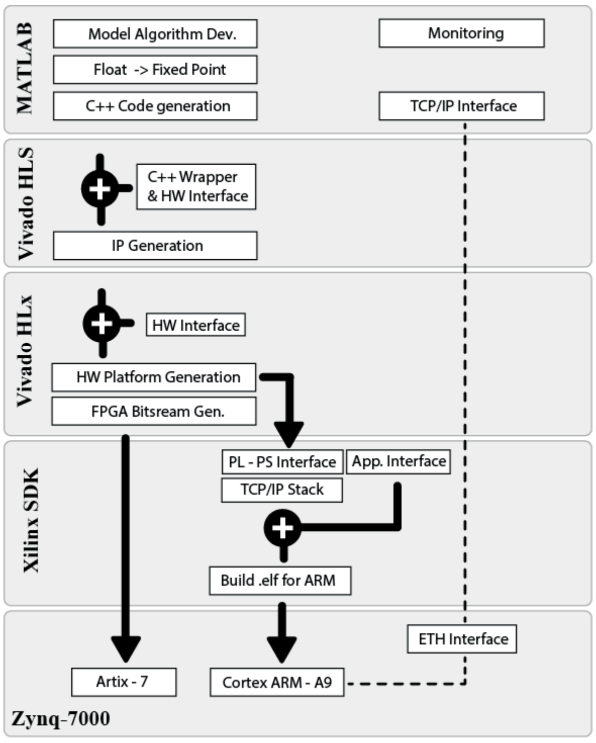

Fig. 2 The main stages of the Model based development in Matlab.

The generated C++ code is the input data of the Vivado High Level Synthesis (HLS) tool. This tool is developed by Xilinx. By means of this tool the developer can design FPGA hardware in C++ language. Another advantage of Vivado HLS is the easy integration of the hardware interface into the code of the model. These interfaces include standardized hand-shaking protocol, buses and stream ports as well. In our case the Advanced eXtensible Interface (AXI) was chosen as a general communication bus between the hardware units. The best practice for the interface integration is the wrapper concept. It means the general C++ wrapper is designed, which handle all hardware interface issues and the generated function of the model is integrated into this wrapper by a function calling. Apart from the communication interface implementation, the wrapper is responsible for the initialization, reset and signal mapping as well. In the last phase by means of the Vivado HLS the code is tested, based on the predesigned testbench. In addition the Vivado HLS can also analyze the designed system from the resource and timing point of view. Thanks to this fact the designer can have high amount of information of how the code is implemented. Moreover, the latency of the implemented design is also calculated. If everything is correct, Vivado HLS generates an IP for the next steps in the FPGA design procedure.

The next tool which is applied is Vivado HLx. This software is also the member of the Xilinx Vivado family. It focuses on the FPGA design. Based on the previously generated IP, the hardware system of the HIL can be easily implemented. The biggest benefits of the Vivado HLx are graphical design interface, huge available pre-implemented IP library and automated design data management. Apart from IP of the model, other hardware units are integrated into the design to implement the whole hardware environment. They are the interface to processing system of the Zynq-7000, hardware timer, block ram etc. The details of the hardware design is introduced in 4th section. Based on the hardware design Vivado HLx generates the bitstream file of FPGA and hardware definition data.

Xilinx Software Development Kit (SDK) is an Eclipse based tool for developing and debugging of the software components in the ARM cores. SDK can automatically generate board support package by using the Vivado HLx hardware descriptor files. It includes driver functions and API for the hardware units in FPGA and macros. Beyond them SDK has got inherent support for integration of TCP/IP communication stack into the designed software. Finally, the system build and debug can be executed in the SDK.

3 Model Transformation

The main mission of the introduced tool chain is to implement an efficient model transformation procedure. The process of the model transformation covers the conversion of the floating point Simulink model to executable codes for target. Currently the target is Zynq-7000. t includes two pieces of ARM cores and one piece of FPGA. As a consequence of this fact, at the end of the transformation we would need a bit stream file for FPGA and executable files (ELF) for the ARM cores. Finally, the SDK can download these executables into the targets.

The starting point of the model transformation is a verified floating point Simulink model. By means of the Fixed Point Toolbox the number representation can be converted from floating point into fixed point ones. This conversion is necessary, because DSP48 slice of Artix-7 FPGA in Zynq-7000 has not got inherent support for floating point calculation. As a result of this, the implementation of floating point arithmetic units would use too much resources. The tool uses the simulation based signal range data and user defined signal limits. In addition, it can derive the range information for the signals which have not got limit specification. To convert the number representation the tool still needs the word length and fraction length parameters. In our cases the word length is 32 bit and the fraction length is 16 bit. As far as the resource utilization is concerned, it is a worst-case scenario, since not all signal requires this resolution. Thereafter the fixed point model should be simulated again to check its behavior by comparing with the reference model.

The next step is C++ code generation based on the fixed point model by the tool of Simulink Coder or Embedded Coder. The settings of the model have got huge amount of influence over the code structure. The atomic subsystems and referenced models are implemented as a reusable function. It is worth applying this technique to structure the code and increase the understandability and traceability. In case of model interfaces, the non-virtual buses are implemented in the C++ function argument list. The parameters of the gains can be tunable. In this case they are also put into the function argument list. The outputs of the code generation are .cpp and .h files.

These files have to be imported into the Vivado HLS. In the Vivado HLS a top wrapper function is created around the model. The CHB model is referred in wrapper by the function calling. The arguments of the wrapper symbolize the implementable hardware interfaces. They are mainly based on the AXI bus. The reset, control, status, input and output signals of the model are accessible through AXI bus. Apart from the IP of the model some extra low level interfaces are additionally required which are automatically added by Vivado HLS, like irq, start and AXI low level signals. In addition, the micro architecture of the implemented design can be easily tuned by Vivado HLS as well. It means, the pipelining, clock constraints etc. are configurable by means of pragmas. Vivado HLS also supports the design analyses and optimization. Due to this facts, it can be investigated how much resources are used in specific units or how many clock cycles are required to execute any functions. Finally, the design can be automatically tested by the predefined testbench.

4 Hardware environment

This section deals with the introduction of the hardware environment. It has got four main groups: programmable logic, processing system, Zedboard and HostPC as it can be seen in Fig. 3.

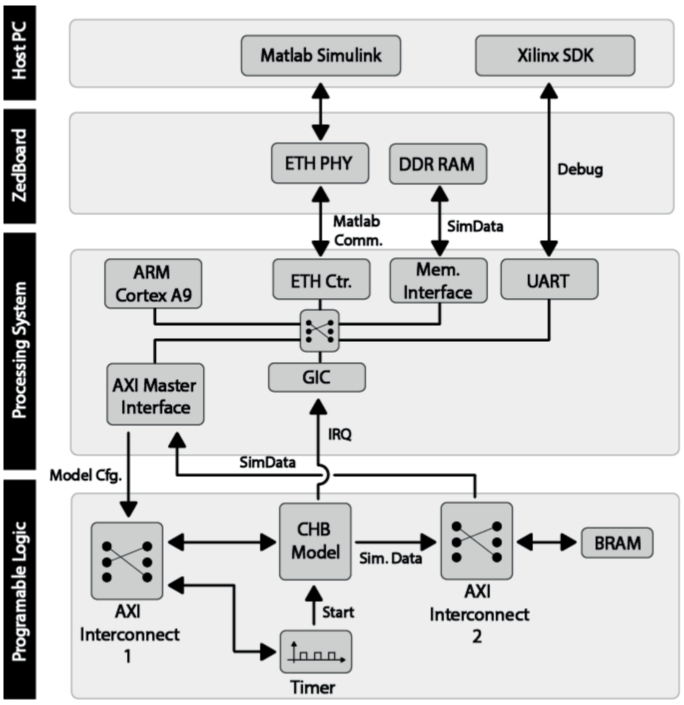

Fig. 3 Architecture of the hardware system.

4.1 System Architecture

Programmable logic practically covers the FPGA. Apart from the IP of the model, the communication interfaces and hardware timer are implemented here. The communication interfaces are based on the AXI protocol. To implement AXI interface between the processing system and the programmable logic, AXI interconnect, AXI master and slave interfaces are needed. AXI masters can initiate communication cycle in the direction of the AXI slaves. AXI masters have AXI slave interface and vice versa. Hardware timer, blockRAM and model have got AXI master interface. They are connected to the AXI slave interface of the processing system since the processing system is the AXI master in the communication. AXI communication channel is implemented not only between the programmable logic and processing system, but also between the CHB model and blockRAM. Thanks to this solution, blockRAM can be simultaneously accessible by CHB model and processing system. In case of the hardware timer and CHB model the AXI interface to the processing system is used for the configuration and signal transfer. The CHB model archives the simulation data of the last step into the blockRAM through the AXI interface.

The currently relevant parts of the processing system are ARM core, general interrupt controller (GIC), Ethernet controller, memory interface, UART interface and AXI

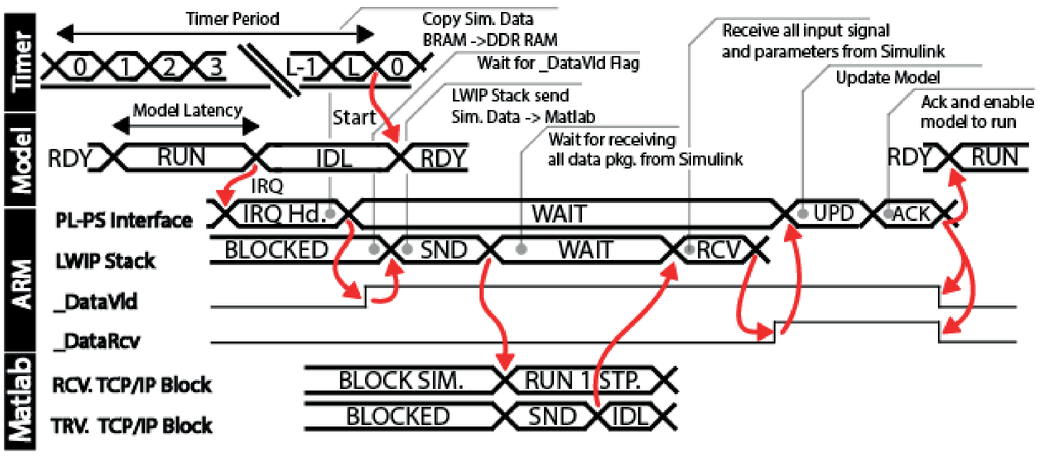

Fig. 4 Timing diagram of the HIL system.

masterinterface. The software developed in the SDK runs in the ARM core. GIC is responsible for receiving the IRQ from the CHB model and calls its handler function. AXI master interface is used to implement the communication between the processing system and programmable logic. Ethernet controller manages the low level of the TCP/IP based communication. DDR ram in the Zedboard is used for store simulation data and the memory interface controls DDR RAM. UART peripheral is extensively used for the purpose of debugging in SDK.

4.2 System Behavior

In this section the operation of the system is introduced. Fig. 4 shows the timing diagram of the system.

The starting point of the explanation is the hardware timer. The responsibility of this hardware is to periodically launch one step simulation cycle of CHB model. For correct operation, the period time has to be longer than the latency of the model. When the CHB model receives the start signal, it can get into the RUN or RDY state. In case of the RDY state, the next simulation cycle is not allowed to start because the IRQ of the previous cycle is not acknowledged by system software yet. If the acknowledge signal is already received and the hardware timer send the start signal, the model is allowed to run a new simulation cycle and getting into the state of RUN.

At the end of a simulation cycle, the model copies the simulation data into the blockRAM and initiates a IRQ signal to the GIC. Then GIC calls the IRQ handler function. It has to copy the data from the blockRAM into the DDR RAM. When it is done, IRQ handler sets the *_DataVld* flag and returns. This flag is continuously polled inside the infinite loop. After its value becomes true the write function of the LWIP stack in the ARM core is called and the simulation data are sent to Simulink. In the monitoring system in the Simulink, TCP/IP receive block from Instrumentation Toolbox blocks the simulation until it does not receive all required data. When they receive these data package, one Simulink simulation cycle is allowed. The sent data from the CHB model are acquired in Simulink by the monitoring system. Simultaneously, the values of the input signals like grid and motor voltages and parameters are firstly calculated and sent by TCP/IP transmit block.

When LWIP stack in the ARM core receives all input signal and parameters, it sets _DataRcv flag. This flag is also polled inside an infinite loop. These data are sent to the CHB model through AXI bus, if the flag value is true. This is practically the update process of the CHB model. The last step is to send the acknowledge signal to CHB model. It makes possible the next simulation cycle to start.

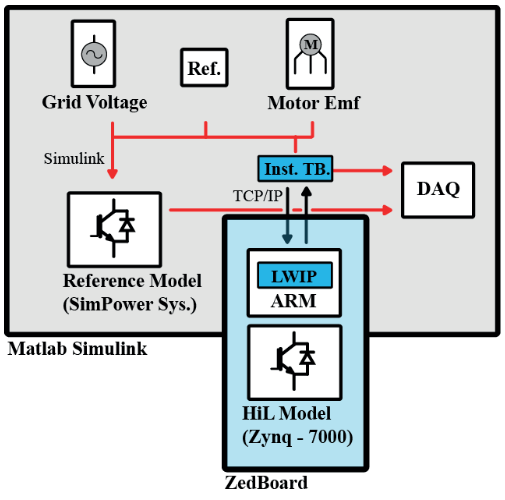

Fig. 5 Simulation environment of CHB model.

5 Implementation

In this chapter the characteristics of the implemented system and the implementation process are shortly described.

The architecture of the model has got a strong impact on code structure and the hierarchy of functions. The resource utilization is also influenced by the model architecture, because Vivado HLS can share the critical resources between the units which belong to the same function. Vivado HLS implements four instances of rectifier and six instances for the inverter functions. As a consequence of the fact that there are six instances of rectifier units in the model, pipelining at function level is required for the function calls.

During the development it is important for the developers to get information about the resource utilization. At the end of the model synthesis, Vivado HLS provides roughly resource estimation for the functional units. After the implementation of the model, Vivado HLS estimates the required resources again, but only for the total design. After the implementation of the hardware system, the Vivado HLx accurately calculates the used resources. The results of the aforementioned estimation can be seen in Fig. 6 and Fig. 7.

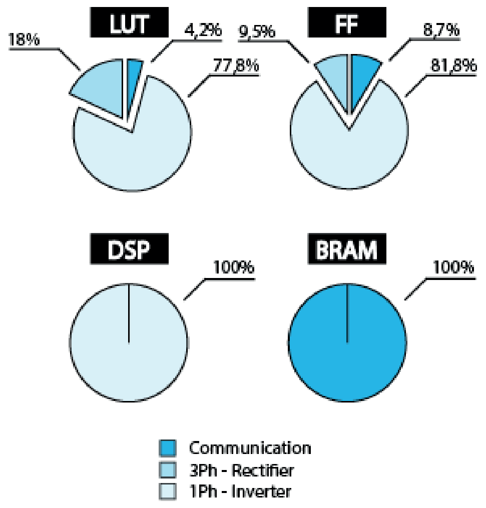

Fig. 6 Resource utilization of the main functional units.

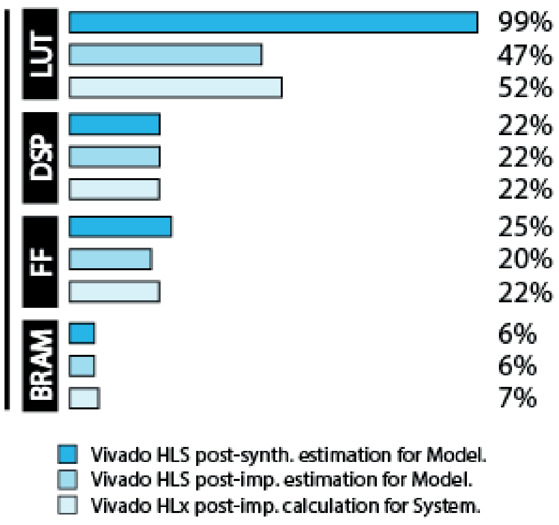

Fig. 7 Estimation and calculation for the utilization of the resources.

In terms of the implementation, the utilized LUT number is almost the half of the first estimation and in the other cases the results seem to be coherent. The auxiliary hardware system around the model does not require high amount of resources, see Fig. 6.

Tools can analyze not only the resource utilization but the timing aspects also. In this case, clock frequency and maximal latency of the model are critical. The clock of the FPGA was chosen to 10 ns. The latency is 243 clock cycles. This high value is implied by the low number of the utilized DSP slices and their pipelining. As a consequence of the clock frequency and model latency, the duration of a simulation cycle in HIL is around 2.5 – 3 us.

6 Simulation Results

The accuracy and behavior of the HIL are verified by means of the SimPower System reference model. The CHB control is the same as it was in the first part of the article. Therefore, an open-loop control system is implemented without current control loop. The cell reference signals are pre-calculated. HIL and reference model are simulated by the same parameter set and the SimPower System toolbox based reference model runs in the Simulink. The signals of the HIL are acquired by means of block from Instrumentation Toolbox. The acquired data are visualized by standard Simulink blocks, like Scope. The configuration data are introduced in Table 1. This parameter set is the same as it was in the previous part of this article.

Table 1 Simulation Parameters

Parameter	Value
Motor Nominal Data	2.3 kV / 140 A
Grid Nominal Data (Sec.)	1.1 kV / 50 A
Motor Leakage Inductance	300 uH (1%)
Transformer Leakage Induct.	300 uH (~0.8%)
DC Link Capacitor	3 mF
Switching Frequency	3 kHz
Frequency of Motor EMF	50 Hz
Grid Frequency	50 Hz

Firstly, the relevant signals of the HIL are introduced including input and motor currents and output voltages of the cells. Then, these signals are compared to the reference model. The input currents of the cell are shown in Fig. 8. The input current system is not symmetrical; it is mainly the consequence of the single phase load of the DC-link capacitors in the cells. This grid currents asymmetry is also the consequence of the fact that the frequency of the motor EMF is the same as the grid frequency.

Fig. 9. and Fig. 10. show the motor currents. The phase voltages of the CHB can be seen in Fig. 11. The number of the voltage levels meets the theoretical values. In case of N pieces of the two-level cell in each phase with symmetrical DC-link. It is 2N + 1, between line and neutral.

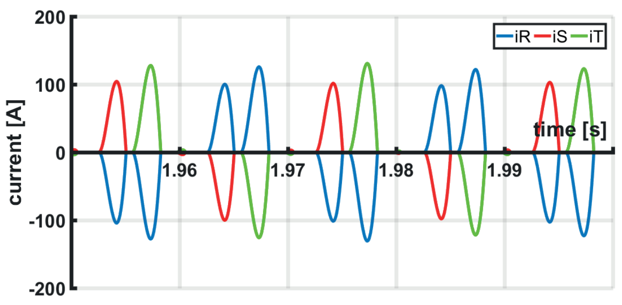

Fig. 8 Input currents of the upper cell in U phase.

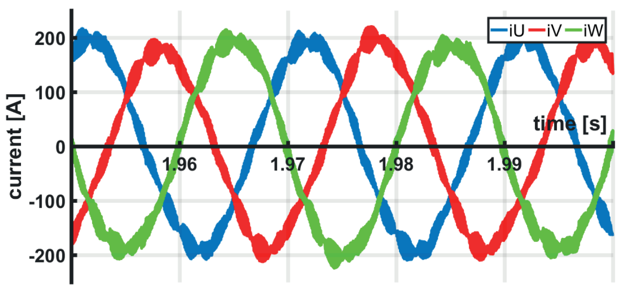

Fig. 9 Motor phase currents.

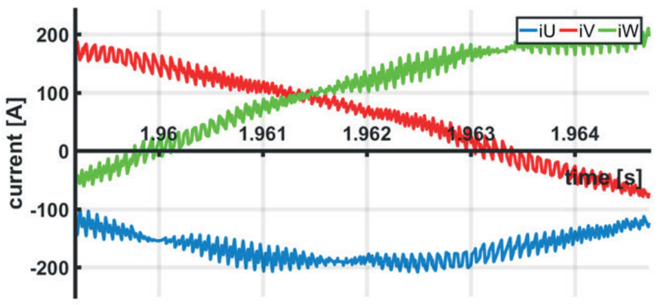

Fig. 10 Motor phase currents (enlarged).

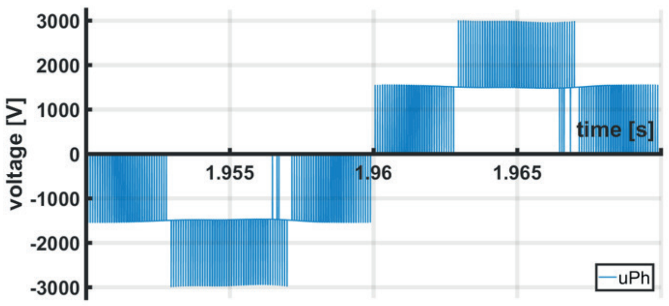

Fig. 11 U phase voltage at CHB output.

In Fig. 12 the line voltage of the inverter is introduced. Since there are 2 cells in each phase, the number of the voltage levels between line and line should be 4N + 1. It can also be observed in the aforementioned figures.

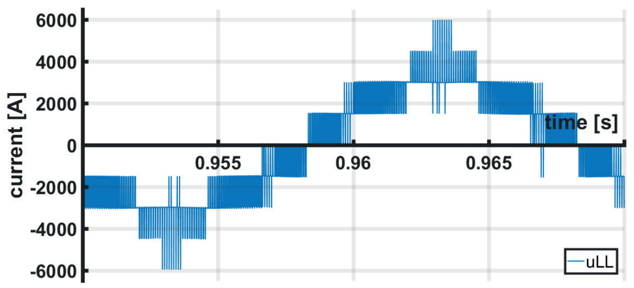

Fig. 12 U-V line voltage at CHB output.

Fig. 13 shows the DC link voltage of the cell in the U phase. The effects of the single phase load are the relative high voltage ripple and capacitor load in the DC link. The deviation between the signals of the HIL and the reference model is minimal.

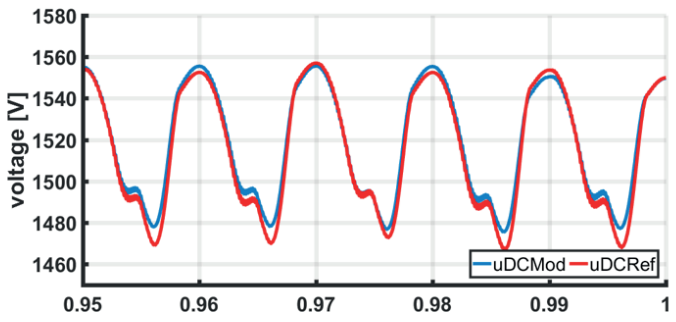

Fig. 13 DC link voltage of the upper cell in U phase.

Maximum deviation is around 10V if the average DC link voltage is roughly 1500V. It means around 0.7% error.

The deviation between the model in HIL and the reference model in case of the input currents and the motor currents is also shown in Fig. 14 and Fig. 15. In spite of the fact that the error of the DC-link voltage seems to be negligible, its effect on the motor current is significant. Its root causes are the small values of impedance.

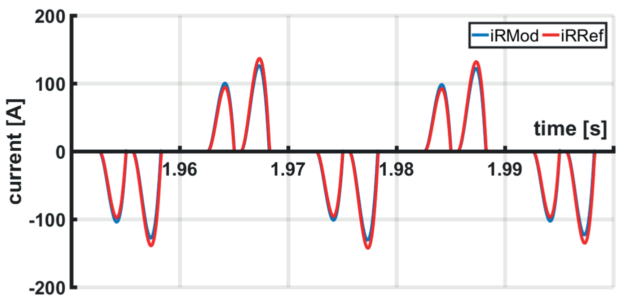

Fig. 14 Comparison of the implemented and the reference model in case of R phase Input current of the upper cell in U phase.

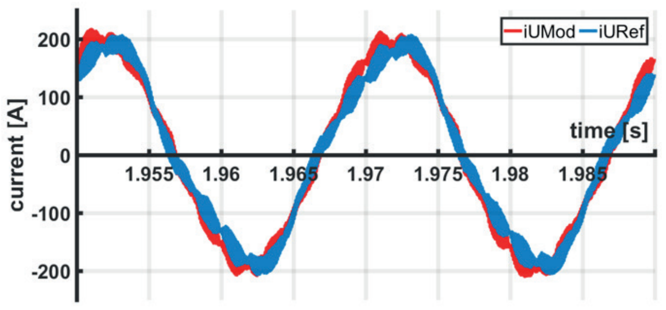

Fig. 15 Comparison of the implemented and the reference model in case of U phase motor current.

Taking everything into account the behavior of the model in the HIL and reference model is similar.

7 Conclusion

This two-part article series introduces power electronics model development and operation on the Zynq-7000 platform. The second part of the series (Part 2) focuses on the model transformation, hardware development and details of implementation in terms of the CHB HIL model.

In the first section, the applied tool chain is demonstrated. Apart from it, the hardware development workflow is also introduced with V-model based approach. It extends the model based development workflow, which was described in the previous part of the article series.

The next section focuses on the details of the model transformation. Not only the tools of the transformation are considered, but the main steps and practical issues also.

In the section of the hardware environment, the explanation of the system operation is described both from architectural and from behavior points of view. The structure diagram is divided into layers. It includes all relevant hardware units and their interfaces.

The core element of the behavior analyses is the timing diagram. The interaction between all functional units are demonstrated on this figure.

The fifth section deals with the characteristics of the implementation. Two critical aspects are considered here: the resource utilization and timing constraints. All in all, the implemented model needs less than half of the available resources. Its consequence is that the speed of the HIL model is moderate.

The verification is executed by the SimPower System based reference model. Its results are described in the last section. The errors of the HIL model are generally small. However, they require further investigation.

Future works will focus on increasing the speed and on further optimization of resource utilization in terms of the HIL model.

References

[1] Faruque, M. D. O., Strasser, T., Lauss, G. "Real-Time Simulation Technologies for Power Systems Design, Testing, and Analysis." *IEEE Power and Energy Technology Systems Journal*. 2(2), pp. 63-73. 2015. https://doi.org/10.1109/JPETS.2015.2427370

[2] Kökényesi, T., Varjasi, I. "Comparison of Real-Time Simulation Methods for Power Electronics Applications. In: 2013 4th International Youth Conference on Energy (IYCE), Siófok, Hungary. Jun. 6-8, 2013, pp. 1-5. https://doi.org/10.1109/IYCE.2013.6604137

[3] Matar, M. "An FPGA-Based Real-Time Simulator for the Analysis of Electromagnetic Transients in Electrical Power Systems." Phd Thesis, University of Toronto. 2009.

[4] Debreceni, T., Kökényesi, T., Sütő, Z., Varjasi, I. "FPGA-based real-time Hardware-In-the-Loop simulator of a mini solar power station." In: 2014 IEEE International Energy Conference (ENERGYCON), Cavtat, 2014, pp. 70-75. https://doi.org/10.1109/ENERGYCON.2014.6850408

[5] Li, W., Grégoire, L. A., Souvanlasy, S., Bélanger, J. "An FPGA-based real-time simulator for HIL testing of modular multilevel converter controller." In: 2014 IEEE Energy Conversion Congress and Exposition (ECCE), Pittsburgh, PA, 2014, pp. 2088-2094. https://doi.org/10.1109/ECCE.2014.6953678

[6] Matar, M., Iravani, R. "FPGA Implementation of the Power Electronic Converter Model for Real-Time Simulation of Electromagnetic Transients." *IEEE Transactions on Power Delivery*. 25(2), pp. 852–860. 2010. https://doi.org/10.1109/TPWRD.2009.2033603

[7] Ould-Bachir, T., Merdassi, A., Cense, S., Blanchette, H. F., Bélanger, J. "FPGA-based Real-Time Simulation of a PSIM Model: An Indirect Matrix Converter Case Study." In: IECON 2015 - 41st Annual Conference of the IEEE Industrial Electronics Society, Yokohama, 2015, pp. 003336-003340. https://doi.org/10.1109/IECON.2015.7392614