Periodica Polytechnica
Electrical Engineering
and Computer Science

60(3), pp. 163-170, 2016
DOI: 10.3311/PPee.9398

RESEARCH ARTICLE

FPGA-synthesizable Electrical Battery Cell Model for High Performance Real-time Algorithms

Tibor Debreceni[1*], Péter Szabó[1], Gergely György Balázs[2], István Varjasi[1]

Received 27 April 2016; accepted after revision 25 May 2016

Abstract

Modern battery management systems (BMS) for advanced battery energy storages are expected to provide sufficient and reliable State-of-Charge (SoC) and State-of-Health (SoH) information. Focusing also on mid- and long-term maintenance purposes, health monitoring can be realized only by using high performance real-time estimation algorithms involving online electrical battery cell model. Due to the nonlinear I-V characteristics of cells and multivariable nonlinear functions describing the model parameters, a real-time model synthesized to FPGA seems to be the best solution to fulfill also the strongest requirements of energy management and e-mobility applications in respect of scalability, modularity, accuracy and effectiveness. In this paper, an FPGA-synthesizable battery cell model is presented and proposed. The design approach is discussed from offline to online model design including the model considerations using MATLAB/Simulink®. The performance analysis and evaluation referenced to the offline model are presented and discussed.

Keywords

battery cell model, battery management, real-time simulation, SoC and SoH estimation, FPGA

[1] Department of Automation and Applied Informatics,
Faculty of Electrical and Informatics Engineering,
Budapest University of Technology and Economics,
H-1521 Budapest, P.O.B. 91, Hungary

[2] Siemens Zrt., H-1143 Budapest, Gizella út 51-57., Hungary

[*] Corresponding author, e-mail: tibor.debreceni@aut.bme.hu

1 Introduction

E-mobility and large-scale stationary energy storages specify even more specific and safety-critical requirements on advanced battery systems, hereby boosting the development need of special purpose electrical battery cell models. In opposite to this, applied research focuses more on the scalability and modularity.

In technical point of view, developing electrical battery cell models has two main driving forces. One is the offline (continuous-time with floating-point number representation) circuit simulation of battery systems providing I-V characteristics for electrical system design purpose [1]. In applications, where the battery pack is connected to nonlinear systems (especially high power motor drives, or grid-connected energy storages), it is crucial to verify the designed battery system behavior [2].

The other is the online simulation, where the goal is to create a real-time electrical battery cell model, which can be run in real-time simulations and in online, i.e. model-based SoC and SoH estimation algorithms, as well [3, 4]. Such online model is essential in advanced battery energy storages. Expected modularity allows extra functions to be attached easily, and the required scalability enables level-to-pack extensions within certain error tolerances and additional dependencies of parameters [5, 6]. All these complex requirements lead the developers to switch rather to FPGA from MCU in the BMS for running high performance, online SoC and SoH estimation algorithms. Next to this, if the battery cell model is available in synthesizable form, then one more great advantage is that it can be used also in modern FPGA-based Hardware-In-the-Loop (HIL) simulators. HIL is a test approach, where, in this case, the BMS is tested involving the original BMS electronics and the real-time simulator of the battery on the expected cell- or pack level solving the state equations numerically step by step [7]. By using battery HIL simulator, also the extreme failure cases are reproducible, which is not possible in the real system [8].

In this paper, an FPGA-synthesizable electrical battery cell model is introduced using fixed-point number representation, which is designed and verified in MATLAB/Simulink. This online Simulink model can be used in FPGA or in processor-based BMS and also in real-time HIL simulators.

The paper is organized as follows. At first, basic modeling considerations and assumptions are described in Section 2 while introducing the offline battery cell model, which is taken as reference for the online model and its verification process. Next, Section 3 deals with the offline-to-online model conversion process involving the fixed-point calculation method and discretization procedures. In Section 4, the verification process is explained as a preparation in order to get the best understanding of Section 5, where the online model performance will be shown and evaluated. In the final section, the conclusion discusses the impact of the designed online model.

2 Offline model design

Since the online battery cell model is expected to fit strict requirements, it is crucial to start from a corresponding model. The first step in model design was to get a good compromise between model complexity and implementability. This trade-off demand appears as a representation of the application-specific property, because here the priority of considering some parameters is boosted, others are decreased or totally neglected regarding which one is important in certain application.

In the model design, the aim was essentially to take into consideration only the most necessary parameters and dependencies that are needed to allow us to analyze and evaluate the online model performance in comparison to the offline model. Besides, according to the energy- and power density requirements coupled to the weight and volume limitations in advanced battery storages, lithium-ion technology-related considerations were principally taken. First, the chosen model topology is introduced.

Unite the advantages of Thevenin-based [1, 9], impedance-based [10, 11] and runtime-based [12, 13] electrical models, a combined battery cell model shown in Fig. 1 was designed and proposed [14]. It is widely used in practical applications due to its simplicity, scalability and effectiveness. This topology essentially fulfills the above mentioned claims with a certain degree of modifications needed.

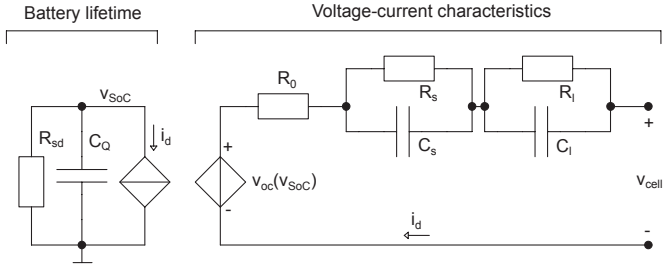

Fig. 1 Combined electrical battery cell model.

In "Battery lifetime" part, v_{SoC} is the per unit voltage representation of the SoC measured on C_Q capacitor, which represents the cell capacity in Ampere-seconds. The nonlinear open circuit voltage characteristic of the cell is stored in $v_{oc}(v_{SoC})$; the terminal or cell voltage is v_{cell}. The resistor R_{sd} is used to characterize the self-discharge energy loss when the battery is stored for a long time. These elements are inherited from runtime-based models. Internal resistance is represented by R_0. The RC network, composed of R_s, C_s, R_l and C_l, represents the transient response similarly to the Thevenin-based models. On one hand, using two RC time constants is practically the best trade-off between accuracy and computational complexity [4, 14]. On the other hand, when this equivalent circuit is considered as a reduced order model based on electrochemistry principles, the second-order RC network represents the effective double polarization such as electrochemical and concentration polarization separately [3, 4, 15].

Based on the combined model, the used model can be seen in Fig. 2, and its state equations can be followed in (1). The sign of the cell current i_d is positive when the cell is discharged, and negative when charged.

$$\begin{cases} \begin{bmatrix} \dot{v}_{DoD} \\ \dot{v}_s \\ \dot{v}_l \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1/\tau_s & 0 \\ 0 & 0 & -1/\tau_l \end{bmatrix} \begin{bmatrix} v_{DoD} \\ v_s \\ v_l \end{bmatrix} + \begin{bmatrix} 1/C_Q \\ 1/C_s \\ 1/C_l \end{bmatrix} i_d \\ v_{cell} = v_{oc} - i_d R_0 - v_s - v_l \end{cases} \tag{1}$$

where $\tau_s = R_s C_s$ denotes the short time-constant and $\tau_l = R_l C_l$ is the long time-constant, and $v_{DoD} = 1 - v_{SoC}$ means the Depth-of-Discharge (DoD). Hereinafter, v_{SoC} and v_{DoD} will simply be referred to as *SoC* and *DoD*. The self-discharge effect is ignored due to the very low per month self-discharge rate of advanced lithium-ion cells, and because daily or weekly usage of the batteries is assumed in this paper.

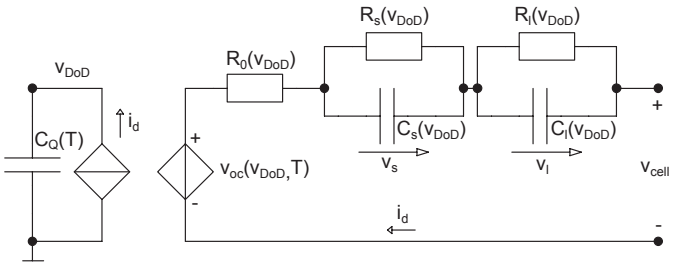

Fig. 2 Used electrical battery cell model.

The first order differential equations shown in (1) have to be modified to integral forms explained in the next section. The resulting state equations, therefore, got an additional constant component that has a physical meaning of the initial values. In case of v_s and v_l, these components are zero, but in case of *DoD*, the initial *SoC* value of the battery cell needs to be considered:

$$DoD = \frac{1}{C_Q}\int_0^t i_d dt + \underbrace{\left(1 - SoC_{Init}\right)}_{DoD_{Init}} \tag{2}$$

The cell temperature is calculated using a one-time constant equivalent thermal model of the battery cell shown in Fig. 3.

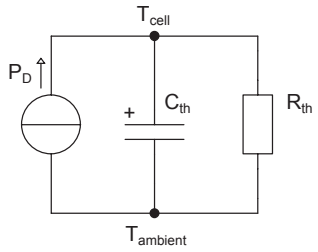

Fig. 3 Used equivalent thermal model of the battery cell.

In the thermal model, C_{th} capacitor represents the heat capacitance of the cell, and R_{th} resistor represents the resulting thermal junction between the cell internal heat generation and the ambient medium through the cell surface. Therefore, the thermal behavior is described in (3).

$$P_D = \left|\left(V_{oc} - V_{cell}\right) i_d\right|$$
$$T_{cell} = T_{ambient} + \frac{1}{C_{th}} \int_0^t \left(P_D - \frac{T_{cell} - T_{ambient}}{R_{th}} \right) dt \tag{3}$$

Theoretically, all the quantities/parameters in the electrical model vary with current, temperature, SoH, SoC and so on [6]. Battery model using such topology can be effectively identified to get realistic model parameters as multivariable nonlinear functions [3, 6, 14]. However, it is important to mention that regarding the performance analysis and verification process, the designed online model does not require realistic and valid model parameters, because it is verified directly to the offline model. Hereby, only the equivalence must be ensured between the offline and online model parameters, and the offline model parameters are not necessarily needed to be identified and validated based on real battery cell measurement data. In the online model, only the DoD-dependency of the internal resistance and the RC network elements are not identified; others were obtained as nonlinear functions derived from a real and widely-used lithium-ion battery cell. The detailed model dependency graph can be seen in Fig. 4, where the quantity/parameter at the end of the arrow is dependent from the quantity/parameter at the base of the arrow.

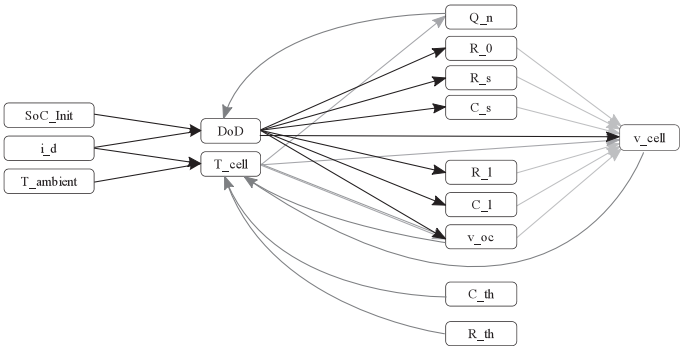

Fig. 4 Quantity/parameter-dependencies in the used model.

3 Online model design

In order to obtain the online model, certain modifications are needed in the offline model. By shortening cycle times, signals can be processed on higher frequencies according to the Nyquist-Shannon sampling theorem, which leads to a more realistic simulation. These methods require the implementation of digital integrators to store the values of the state variables. The iterative calculations of state variables are done by using forward-Euler numerical integration method described as follows:

$$y\left(t + \Delta t\right) = y\left(t\right) + \Delta t \cdot u\left(t\right) \tag{4}$$

where $u(t)$ is the input, $y(t)$ is the output quantity and Δt is the time-step. When keeping the time-step as low as possible compared in order of magnitudes to the system time constants, the numerical instability can be avoided and the truncation error of the numeric approximation can be kept at significantly low level [16, 17]. Therefore, this numerical method seems to provide the best trade-off between accuracy, effectiveness, complexity and offline-to-online design time. This is the main reason why it is worth modifying the state equations in (2) to integral forms. Since the online model is aimed to be synthesized to modern FPGAs that has integrated hardware multiplier units using minimum 18×25 bits, it was worth using fixed-point representation focusing on minimizing the non-generated coding while not losing significant precision.

The nonlinear functions for quantities/parameters are realized by look-up tables, where the output vectors belong to input vectors derived by the division of the domains into equidistant points. In case of v_{oc} being a two variable nonlinear function of *DoD* and T_{cell}, a 2-D look-up table would be the best choice. Nonetheless, the 2-D look-up table is part of a library containing Simulink blocks capable of HDL code generation, the realization considerations raised subject of great debate. Basically, v_{oc} measurement is available in 9 different temperature points (in every 10 °C steps from -20 to 60 °C). The *DoD* is divided into 4096 equidistant points. Hereby, the 4096×9 size 2-D table could be indexed by the calculated *DoD* and T_{cell} temperature. First problem is that in order to do this, these indexing quantities are expected to be represented exactly in the same fixed-point form. However, the ranges of the two quantities can be fitted, the precisions should be considered. The temperature resolution is 0.01 °C and the *DoD* resolution is 0.001 % in the used ranges. Since the *DoD* resolution is more critical, it is not modified to the weaker temperature precision. Otherwise, if the temperature would be represented the same as *DoD*, it would be a waste of resources.

The second problem was with the equidistance requirement of the table data. By changing the voltage and temperature vector elements to fixed-point representation, those became not equidistant, which generated block error in Simulink. Even when these indexing vectors are replaced by such auxiliary indexing vectors, a new external logic is also needed.

Due to all described above, instead of using 2-D look-up table, multiple 1-D look-up tables are applied (Fig. 5) and the synthesis to block RAMs is guaranteed, which will be detailed later.

Fixed-Point Designer is a built-in tool of MATLAB that has been used to determine the fixed-point representations featuring the *fixdt* form. The general form is *fixdt*(*s, s+n+m, m*), where *s* means the sign bit, *n* means the integer bit count and *m* means the fraction bit count. After specifying the resolution (q), the maximum (u_{max}) and minimum (u_{min}) values, the *fixdt* form can be achieved for all state- and auxiliary variables in the model:

$$
\begin{aligned}
s &= \begin{cases} 1, & u_{min} < 0 \\ 0, & \text{else} \end{cases} \\
n &= \begin{cases} \lceil A \rceil, & |u_{max}| \geq 1 \ \text{ and } \ |u_{min}| \geq 1 \\ \lfloor A \rfloor, & \text{else} \end{cases} \\
A &= \log_2\left(\max\left(|u_{max}|, |u_{min}|\right)\right) \\
m &= \begin{cases} \lceil -\log_2(q) \rceil, & q < 1 \\ \lfloor -\log_2(q) \rfloor, & \text{else} \end{cases}
\end{aligned}
\tag{5}
$$

Other equations are valid to get the representations of the multiplier factors belonging to the integrands. The absolute values of the multiplier factors are typically smaller than 1, thus the LSB of the product will be placed on a less significant bit than the LSB of the multiplicand. These bits might to be eliminated, but the error is accumulated by the integrator. This should be avoided, thus those bits have to be extended with bit count warranted by the multiplier factor. The unnecessary bits can already be left at the output of the integrator, since the error will not be accumulated, because the values are stored more accurately by the integrator. The bit depth is given by the hardware multipliers of the FPGAs, only the extension of the fraction are needed to be calculated, as shown in (6). The input signal is extended with the fraction bits of the multiplier constant (u). It is important to be stated that the effect of the extension is negative if the multiplier constant is bigger than 1.

$$
\begin{aligned}
m_{const} &= \begin{cases} \lfloor -\log_2(u) \rfloor, & u > 1 \\ \lceil -\log_2(u) \rceil, & \text{else} \end{cases} \\
m_{multiplier} &= m_{input} + m_{const}
\end{aligned}
\tag{6}
$$

For better understanding, the Simulink representation of the online model is split into parts shown from Fig. 5 to Fig. 8.

The presented online model is converted into HDL code with HDL Coder tool of MATLAB/Simulink. The tool generates the Verilog or VHDL files of the model, which can be directly imported into any synthesizer tool, such as Xilinx ISE Design Suite or Vivado Design Suite and so on. The HDL Coder generates registers from the look-up tables, which is wasteful of the resources. Placing a unit delay with one time-step after each 1-D look-up table in the Simulink model, the resulting delayed readout of the generated registers will be optimized to block RAMs by the synthesis tool. The detailed utilization is presented in Section 5 in comparison between 3 modern FPGAs.

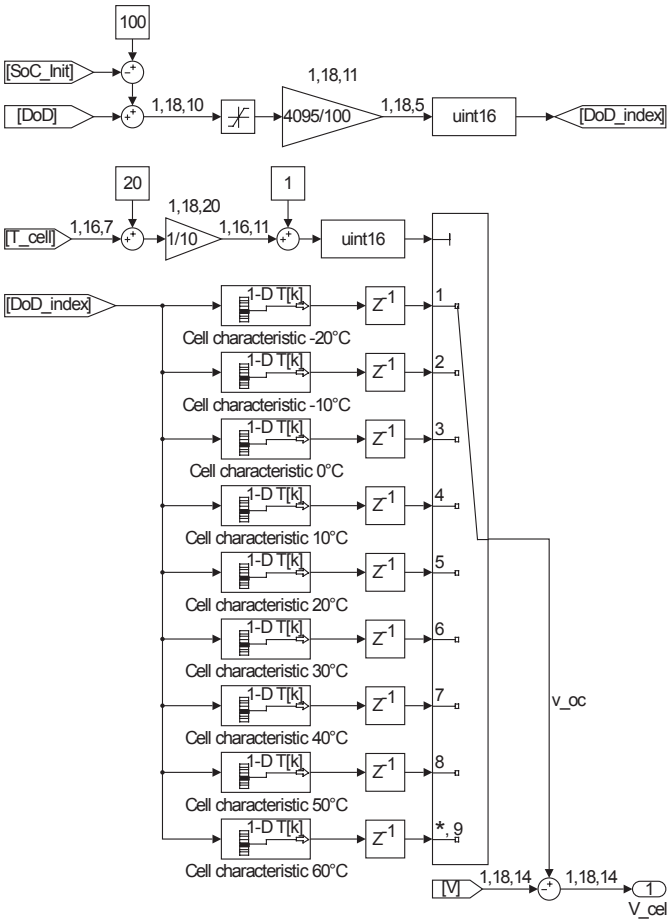

Fig. 5 Open circuit voltage part of the online model.

4 Verification process

The online model verification process was started with running the offline and online models in the same simulation cycle of MATLAB/Simulink. In order to get precious information also in terms of time-steps in different order of magnitudes, 1 ms, 100 ns and 10 ns version of online model is prepared for verification. The model performance is analyzed by calculating the errors of the quantities v_{cell}, SoC, T_{cell} and C_Q between the offline and online models observing the total DoD range. Additionally, the error analysis is taken also between 1 ms and 10 ns online models exploring the effects of 5 order of magnitudes difference in time-steps. The selection between using absolute- or relative error was considered based on which provides more information to conclude whether if it is accurate enough in certain applications for the BMS and HIL developers.

It has to be noted, that the *in-situ* validation of the FPGA-synthesized model requires a complete BMS or HIL simulator. Such developments are in progress, and the results are planned to be published later.

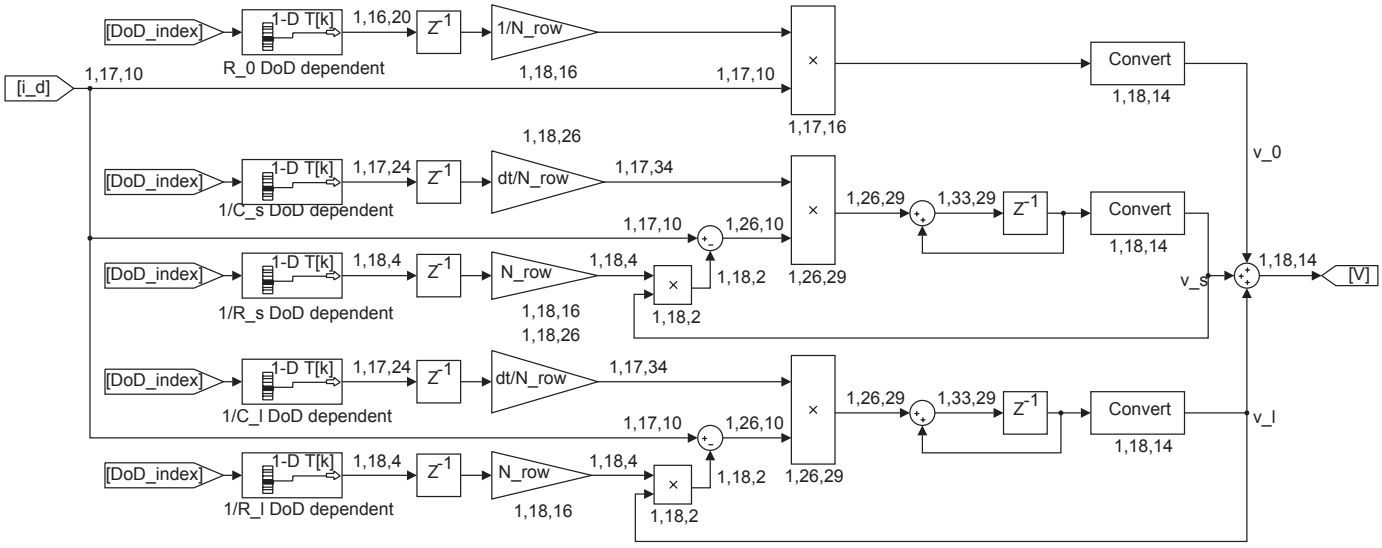

Fig. 6 Internal impedance part of the online model.

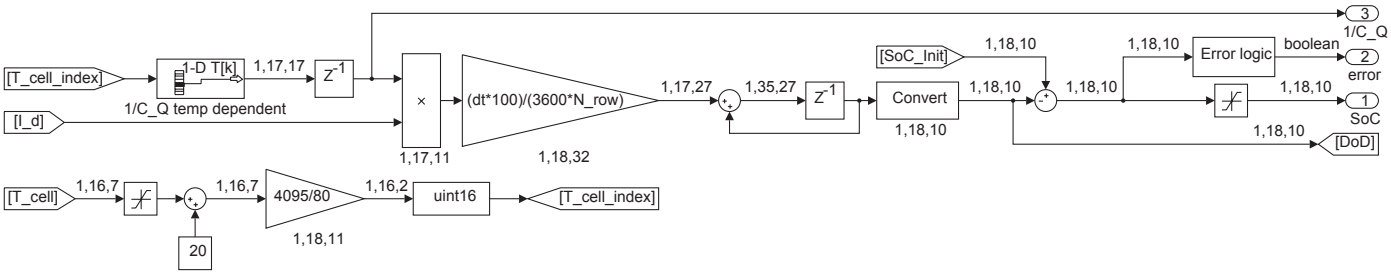

Fig. 7 Battery lifetime part of the online model.

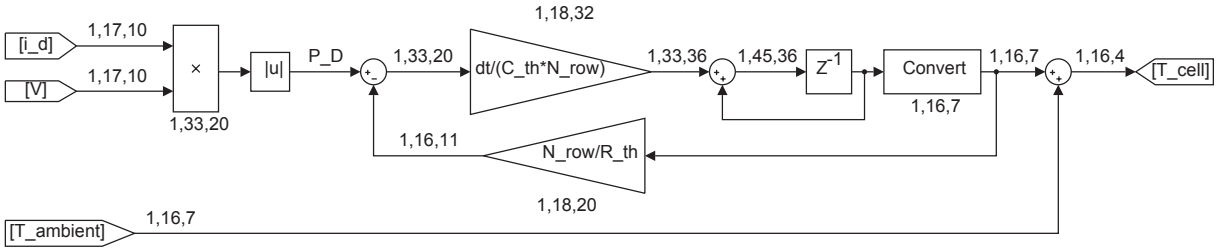

Fig. 8 Thermal part of the online model.

5 Results and evaluation

From Fig. 9, it clearly turns out how the $v_{oc} - T_{cell}$ storage and indexing considerations affect the discharge characteristics compared to the offline curves stored in 2-D look-up table with linear interpolation between the vectors elements represented as floating-point numbers.

Each of the three online models calculate with the open circuit voltage belonging to one actual temperature value until the cell heats up to the next temperature point, where the indexer changes to the other voltage table. During normal circumstances, the cell temperature heats up to 40-50 °C from about 20 °C when discharged by half of the maximum continuous discharge current considering no cooling as in this simulation. Therefore, cell voltage jumps can be found at 3 different *DoD* values. The maximum absolute error caused by this effect is 34.37 mV. However, this is in 2 order of magnitude greater than the cell voltage measurements could be done by a 12 bit A/D converter, but precise enough according to the cell temperature measurement accuracy done usually by PTC/NTC sensors.

Next analysis shows a more significant relative error on the cell voltage in Fig. 10. As it is known, at the end of the discharge characteristics from about 90 % *DoD*, the voltage curve slope highly decreases with normally increasing *DoD*. In theory, this slope approaches to infinity due to total usable capacity loss, hereby the simulated voltage error can be theoretically also infinite. In discrete time and using fixed-point numbers, the error will not be infinite but directly based on the sampling and precision properties of the stored data. One possible solution is that the *DoD* can be divided into two parts. One includes the values in range from 0 % to about 90 % *DoD*

considering the same sample elements. Other is for the rest 10 % storing more samples for the precision. The drawback of this trick is that an additional indexing logic is needed then.

In Fig. 10, the error peaks at near 100 % *DoD* with 8.6568 %. Using this model in SoC estimators, aside from the cell voltage measurement accuracy in the BMS, the most significant errors will occur at the 3 temperature switching points (30 °C, 40 °C and 50 °C) and at the very end of the DoD domain considering the same circumstances.

Since now, for the online model, this behavior is known, these errors can be compensated if it is necessarily needed. Figure 11 presents an SoC error, if the Ampere-integral used in the offline model would be used in online (discrete-time and fixed-point) form as an input for the online battery cell model. Most of the online estimation algorithms use the battery model with SoC or DoD input and cell voltage output to compensate the basic Ampere-integral (Coulomb-counting) based on the measured voltage and by using a controller. Therefore, Fig. 11 shows a worst case error for using the most basic Coulomb-counting estimation with a peak error of only 0.7398 % for the worst performance (1 ms) online model.

In dependency from the cell temperature accuracy, the usable capacity is important to know to get the most accurate cell voltages for the SoC estimator. For this, the relative cell capacity error of 0.0044 % (Fig. 12) and the relative cell temperature error of 0.0357 % (Fig. 13) are sufficient enough in almost all applications.

At this point, it is needed to mention that in case of realizing even more precise SoH estimation algorithms, the usable capacity part of this online model would be placed outside to the estimator, and the capacity would be a new input of the online model. It is mainly for when the cycle number is estimated based on the measured and calculated fractions of total charge/discharge cycles and internal resistance measurement-based correction of SoH is also used.

According to the BMS implementation in a certain application, it is also worth investigating whether an FPGA or high performance processor is necessarily needed or also a modern MCU is enough for running the estimation algorithms. In this respect, the errors of the online model using 1 ms time-step are examined referenced to when using 10 ns time-step.

It can be seen in Fig. 14 that the relative cell voltage error significantly peaks at the switching points of temperature. Certainly, it is because the fixed-point representation of state- and auxiliary variables is the same in the online models, therefore the errors can caused by the indexing refreshing at different instances. Since the time-steps are fixed in the used solver method, these errors will be systematic errors.

The SoC, usable capacity and temperature error analyses are similar to the described about the cell voltage errors. Table 1 summarizes the errors. It turns out, that the online model with 1 ms time-step is still accurate enough to be implemented in MCU compared to the 10 ns time-step model when considering the same fixed-point representation.

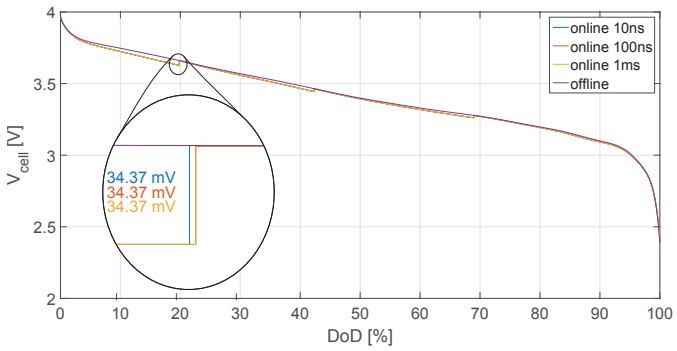

Fig. 9 Discharge characteristics of offline and online models at 1 ms, 100 ns and 10 ns time-step.

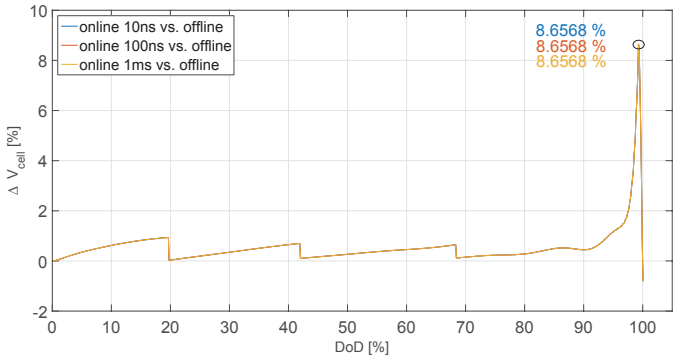

Fig. 10 Cell voltage error of online models at 1 ms, 100 ns and 10 ns time-step.

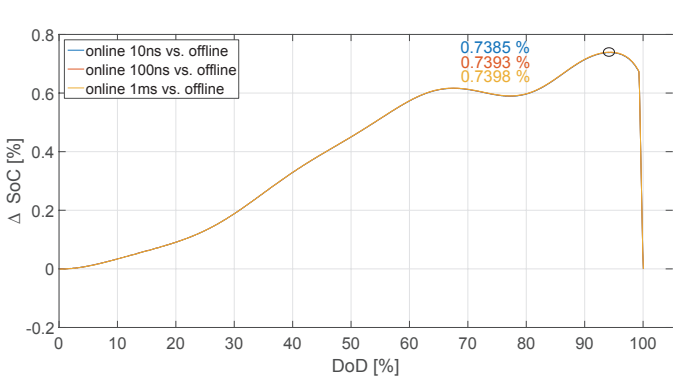

Fig. 11 SoC error of online models at 1 ms, 100 ns and 10 ns time-step.

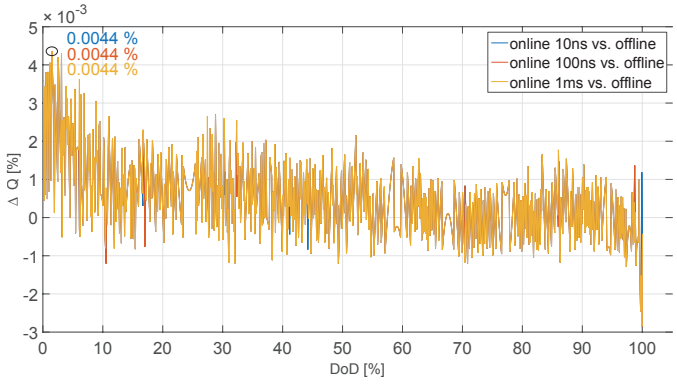

Fig. 12 Cell useable capacity error of online models at 1 ms, 100 ns and 10 ns time-step.

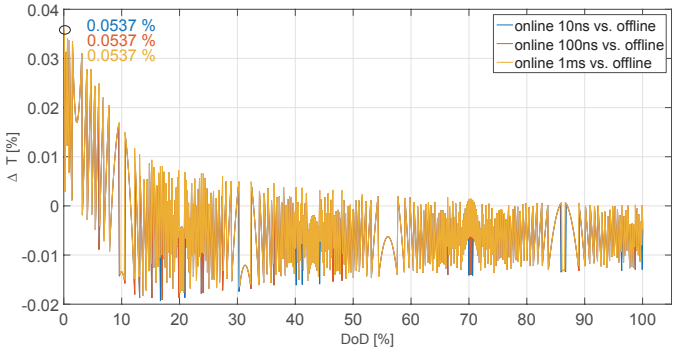

Fig. 13 Cell temperature error of online models at 1 ms, 100 ns and 10 ns time-step.

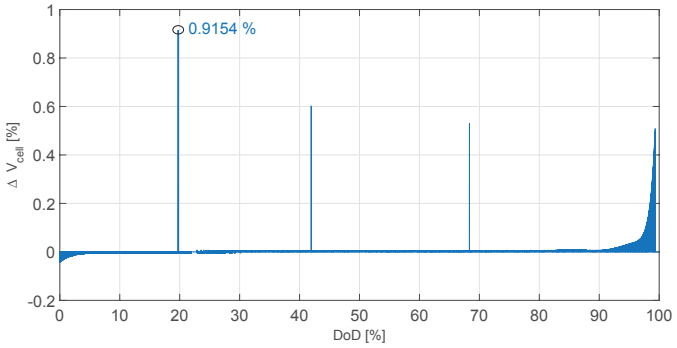

Fig. 14 Cell voltage error of online model at 1 ms compared to at 10 ns.

Table 1 Errors of the online model at 1 ms compared to 10 ns time-step

ΔV_{cell} [%]	ΔSoC [%]	ΔQ [%]	ΔT [%]
0.9154	0.002	0.004	0.039

However, based on the results in Table 1, the presented online model seems to be implementable also in modern processors, but either when the temperature is also considered in case of internal impedance parameters, or when more cell models have to be run real-time in the same BMS, or when the effects of active balancing with 10-100 kHz PWM are needed to consider, then it is crucial to use an FPGA in order to achieve the required efficiency of a model-based SoC or SoH estimation algorithm.

Table 2 summarizes the utilization of the introduced online model synthesized to 3 modern FPGAs highlighting necessary logic components and run frequency capabilities.

Table 2 Utilization and maximum possible frequency information

	Artix 7 XC7A100T	Kintex 7 XC7K70T	Virtex 7 XC7VX330T
Speed grade	-3	-3	-3
Slice Registers [%]	0.16	0.24	0.05
Slice LUTs [%]	2.27	3.50	0.70
BRAM [%]	16.30	16.30	2.93
DSP48E1s [%]	6.67	6.67	1.43
Max. possible frequency [MHz]	55	76	77

The maximum possible frequencies show that this online model could be synthesized allowing a minimum time-step of 12 ns. By increasing the input clock frequency from the actual 100 MHz using hardware phase-locked loop (PLL) units, the time-step could be decreased to 10 ns in order to achieve even 100 MHz run frequency. It is question of consideration whether if it is worth increasing the input clock frequency up to more hundred MHz while facing with hardware design and electromagnetic compatibility issues.

6 Conclusion

The aim of the present paper was, in one hand, to give an overview of the design approach of the battery cell model. The discussion has clearly showed the considerations taken into to get an easy-to-set MATLAB/Simulink implementation. This allows an effective design and rapid verification observing and analyzing the dynamic behaviour also for the offline and online representation. The introduced online model is designed for HDL code generation from MATLAB/Simulink that allows perfect traceability between the model behaviours in the different forms. The model performance is investigated and evaluated in respect of discussing the error analysis and regarding the utilization results in modern FPGAs.

In conclusion, it can be said that based on an existing electrical cell model topology, a verified, FPGA-synthesizable online electrical battery cell model is designed and evaluated. It can be used in high performance, model-based SoC and SoH estimation algorithms in BMS and real-time simulators, as well. The measurement-based validation of the FPGA-synthesized model is a future work, where a BMS or a HIL simulator is implemented, which will be published later on.

The published and proposed online model can be taken as basis for further cell- or pack-level models due to its ability to be effectively scaled, configured and extended.

References

[1] Gao, L., Liu, S., Dougal, R. A. "Dynamic lithium-ion battery model for system simulation." *IEEE Transactions on Components and Packaging Technologies*. 25(3), pp. 495-505. 2002. DOI: 10.1109/TCAPT.2002.803653

[2] Debreceni, T., Balázs, G. Gy., Varjasi, I. "Mission Profile-Oriented Design of Battery Systems for Electric Vehicles in MATLAB/Simulink®." In: International Conference on Renewable Energies and Power Quality (ICREPQ'16), Madrid, Spain, May 4-6, 2016, Renewable Energy and Power Quality Journal (RE&PQJ) Conference Proceedings.

[3] He, H., Xiong, R., Fan, J. "Evaluation of lithium-ion battery equivalent circuit models for state of charge estimation by an experimental approach." *Energies*. 4(4), pp. 582-598. 2011. DOI: 10.3390/en4040582

[4] Yuan, S., Wu, H., Yin, C. "State of Charge Estimation Using the Extended Kalman Filter for Battery Management Systems Based on the ARX Battery Model." *Energies*. 6(1), pp. 444-470. 2013. DOI: 10.3390/en6010444

[5] Zou, Z., Xu, J., Mi, Ch., Cao, B., Chen, Z. "Evaluation of Model Based State of Charge Estimation Methods for Lithium-Ion Batteries." *Energies*. 7(8), pp. 5065-5082. 2014. DOI: 10.3390/en7085065

[6] Lam, L. "A Practical Circuitbased Model for State of Health Estimation of Li-ion Battery Cells in Electric Vehicles." Master Thesis, University of Technology Delft, 2011.

[7] Sütő, Z., Debreceni, T., Kökényesi, T., Futó, A., Varjasi, I. "Matlab/Simulink Generated FPGA Based Real-time HIL Simulator and DSP Controller: A Case Study." In: International Conference on Renewable Energies and Power Quality (ICREPQ'14), Cordoba, Spain, Apr. 8-10, 2014. Renewable Energy and Power Quality Journal (RE&PQJ) Conference Proceedings

[8] Debreceni, T., Kökényesi, T., Sütő, Z., Varjasi, I. "FPGA-based Real-Time Hardware-In-the-Loop Simulator of a Mini Solar Power Station." IEEE International Energy Conference EnergyCon 2014, Dubrovnik, Croatia, May 13-16, 2014, pp. 70-75. DOI: 10.1109/ENERGYCON.2014.6850408

[9] Schweighofer, B., Raab, K. M., Brasseur, G. "Modeling of high power automotive batteries by the use of an automated test system." *IEEE Transactions Instrumentation and Measurement*. 52(4), pp. 1087-1091. 2003. DOI: 10.1109/TIM.2003.814827

[10] Buller, S., Thele, De Doncker, R. W., Karden, E. "Impedance-based simulation models of supercapacitors and Li-ion batteries for power electronic applications." In: Industry Applications Conference, 2003. 38th IAS Annual Meeting. Vol. 3, Oct. 12-16, 2003, pp. 1596-1600. DOI: 10.1109/IAS.2003.1257769

[11] Baudry, P., Neri, M., Gueguen, M., Lonchampt, G. "Electro-thermal modeling of polymer lithium batteries for starting period and pulse power." *Journal of Power Sources*. 54(2), pp. 393-396. 1995. DOI: 10.1016/0378-7753(94)02109-G

[12] Hageman, S. C. "*Simple pspice models let you simulate common battery types*." EDN, pp. 17-132, Oct. 1993.

[13] Gold, S. "A PSPICE macromodel for lithium-ion batteries." In: Twelfth Annual Battery Conference on Applications and Advances, Long Beach, CA, 1997, pp. 215-222. DOI: 10.1109/BCAA.1997.574106

[14] Chen, M., Rincón-Mora, G. A. "Accurate Electrical Battery Model Capable of Predicting Runtime and I–V performance." *IEEE Transactions on Energy Conversion*. 21(2), pp. 504-511. 2006. DOI: 10.1109/TEC.2006.874229

[15] Xiong, R., Gong, X., Mi, C. Ch., Sun, F. "A robust state-of-charge estimator for multiple types of lithium-ion batteries using adaptive extended Kalman filter." *Journal of Power Sources*. 243, pp. 805-816. 2013. DOI: 10.1016/j.jpowsour.2013.06.076

[16] Kökényesi, T., Varjasi, I. "FPGA-based real-time simulation of renewable energy source power converters." *Journal of Energy and Power Engineering*. 7(1), pp. 168-177. 2013.

[17] Cellier, F. E., Kofman, E. "*Continuous System Simulation*." Springer-Verlag US, New York, 2006. DOI: 10.1007/0-387-30260-3