International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 7 (2017) pp. 1329-1333

MCU-based Battery Management System for Fast Charging of IoT-based Large-Scale Battery-Cells

Meng Di Yin, Jiae Youn, Jeonghun Cho, and Daejin Park*

School of Electronics Engineering, Kyungpook National University
Daehak-ro 80, Buk-gu, Daegu, Republic of Korea.

Abstract

There is a growing demand for various applications requiring the large-scale battery cells, such as electrical vehicles and industrial appliances. The Traditional constant current, constant voltage (CC-CV) charging method cannot satisfy the demand of fast charging. Pulse based battery-charging approaches have been proved as fast charging method for the battery cells. Our work is focusing on build a testing framework to evaluate the performance of pulse based charging methods. In order to facilitate the progress of evaluation, the charging control program is developed in MATLAB environment and then the actual charging tests are done on the customized hardware circuit board. As a case study, a newly-designed dynamic frequency and duty cycle charging algorithm is evaluated by using the proposed testing framework.

Keywords: battery cells; electrical vehicle; frequency; duty cycle; testing framework; customized hardware

INTRODUCTION AND MOTIVATION

Reducing gas emissions from cars is critical to management of air pollution. Electrical vehicle (EV) has the advantage of zero tailpipe emission but it has not been widely used because there are many problems such as long charging time. There are a lot of studies [1][2] on how to increase the speed of battery charging such as variants of CC-CV charging, polarization curve charging and pulse charging methods [3][4]. The pulse charging method for battery cells has been recognized as a fast and efficient way to overcome the shortcoming of slow charging time [5]. The optimal frequency is determined in order to minimize battery impedance. The adaptation of the controlled pulse duty cycle decreases the concentration of the polarization on battery cells.

Model-Based design (MBD) has the advantages of improving the product quality and short development cycle [6]. It allows developers to create equivalent circuit of lithium battery and charging controller model in Simscape [7] and Stateflow [8], respectively. Using the Simulink toolbox, the simulation of equivalent circuit and charging control models can be virtually run on computer to quickly assess the charging controller's performance. In this way, designer-specific algorithm can be interactively modified and evaluated at starting point of design flow rather than the final phase so that improve software reliability and reduce development time. The schematic diagram, model implementation, finite state machine and experiment environment are introduced in section II. Finally, the implementation results and conclusion are discussed in section III and IV, respectively.

PROPOSED ARCHITECTURE

A. Schematic diagram for charging test

Using benefit from C code-generation features of the Simulink and state flow model from MATLAB, the corresponding C code is obtained through automatic code generation from state flow model. All of the low-level driver is developed in the Arduino IDE (integrated development environment), such as current sensor acquisition, temperature sensor acquisition, SPI communication control, PWM control and timer control code. The low level driver code is integrated with MATLAB generated code and then they are download to the Arduino UNO board [9].

By using a LTC6802-2 battery stack monitoring chip [10], this chip can monitor battery terminal voltage not more than 12 cells connected in serial. One other thing to note is that this chip must be powered at least 4 cells. In this project, 4 cells are used and the registers of this chip need to be controlled by a microcontroller via SPI. In the experiment, the model INR18650-25R 2500mAh lithium battery as an object of experimental research, which has the main parameters: maximum charge voltage and minimum discharge voltage are 4.2V and 2.5V, respectively. Standard charge current and fast charge current are 1250mA and 4000mA provided by the manufacturer. Considering the complexity and ease to development, this project adopted the Arduino UNO board as the control board shown in Figure 1. Microcontroller can communicate with LTC6802-2 monitoring chip via SPI interface. ACS712 is a hall effect-based linear current sensor, has bidirectional current measurement ability, convert the sensing current into a 1.5V to 3.5V voltage as output. Output voltage responds directly according to the current magnitude. Charging current should be measured as frequently as possible throughout the charging process to calculate the SOC of the battery [11].

To calculate the SOC, sampling interval of charge current is set as 10 milliseconds and the interval of cell voltage and temperature acquisition set as 2 seconds. After request to LTC6802-2 from the SPI master, measurement of 12 channels takes 13 milliseconds. Modification of frequency and duty cycle is shown at the bottom in Figure 1.

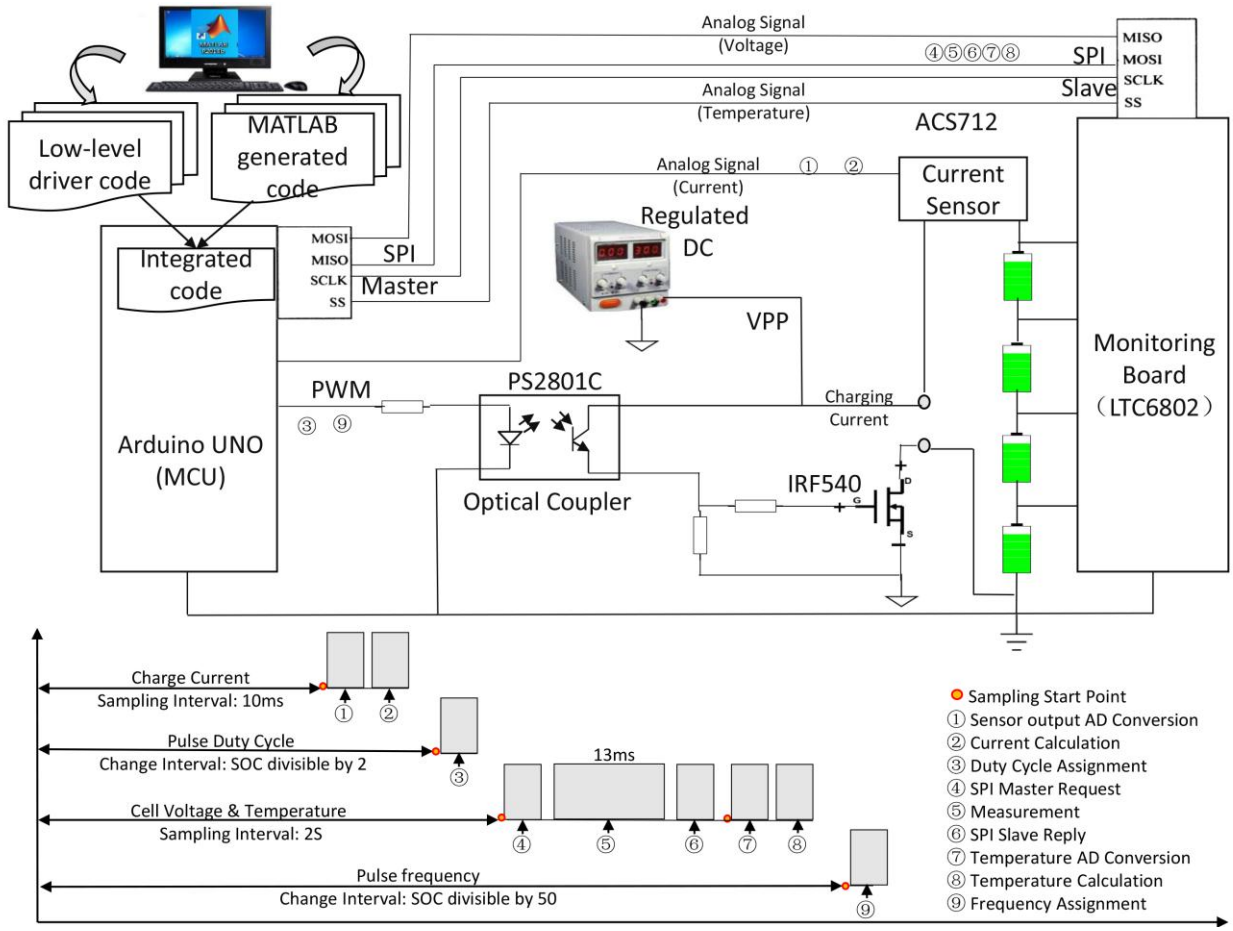

Figure 1. Schematic diagram for charging test

B. Matlab/Simulink-based model implementation

The evaluation framework is implemented using the MATLAB/Simulink environment to design the proposed battery charger system including the proposed searching algorithm of the pulse frequency and the duty cycle, which is shown in Figure 2. The physical model of the battery cells is developed using the Simscape™ package, which requires a solver reconfiguration.

The dotted box on the left in Figure 2 indicates variable declaration for the controlled frequency and duty cycle value. The controlled voltage source block is selected from Simscape™ to model pulse voltage for the charge process of the battery pack. Duty cycle and frequency, as the two input signals, can be adjusted by changing the gain coefficient of the gain blocks. The convective heat transfer block offers a way to exchange convective heat effect between the battery and the ambient environment. The second dotted box from the left shows that ambient temperature is set at 20◦C, which can be adjusted as an input variable.

We designed a charge controller, described by a dashed line on the right. The charge controller decides whether to stop charging based on two input signals: the SOC and the temperature of the battery. The *Cur_Inte* signal is an accumulated value provided by the integrator block at the upper right corner to calculate the average current within seconds, so as to obtain the optimal frequency and duty cycle. The charge controller is in charge of resetting the integrator block with the charge switch signal every 2 seconds based on design requirements. Through observation and analysis of battery state parameters, such as charging current, terminal voltage, SOC, and temperature, achieved the most efficient frequency and duty cycle for a fast pulse-based charging method. The charging time and temperature rise due to two important parameters in the charging algorithm. The rise of temperature indicates the heating on internal resistance of the battery cells during the charge process.

The third dashed line from the left shows the high-fidelity battery model provided by Simscape™. As a high-fidelity model design, also consider SOC and thermal effect on the modeled component. Each component of the equivalent circuit model will change with SOC and temperature of the battery over the process of the battery charging. The amount of heat, which is generated on R0 and R1 as an output of the thermal model, is connected by the convective heat transfer. The equivalent circuit model [12] consists of an ideal voltage source (Em), an RC block (R1, C1) and an internal resistance (R0), can describe the characteristics of the lithium battery as capacitance and resistance in Figure 3. The capacitance characteristic of the battery cell causes different responses at different frequency pulses.

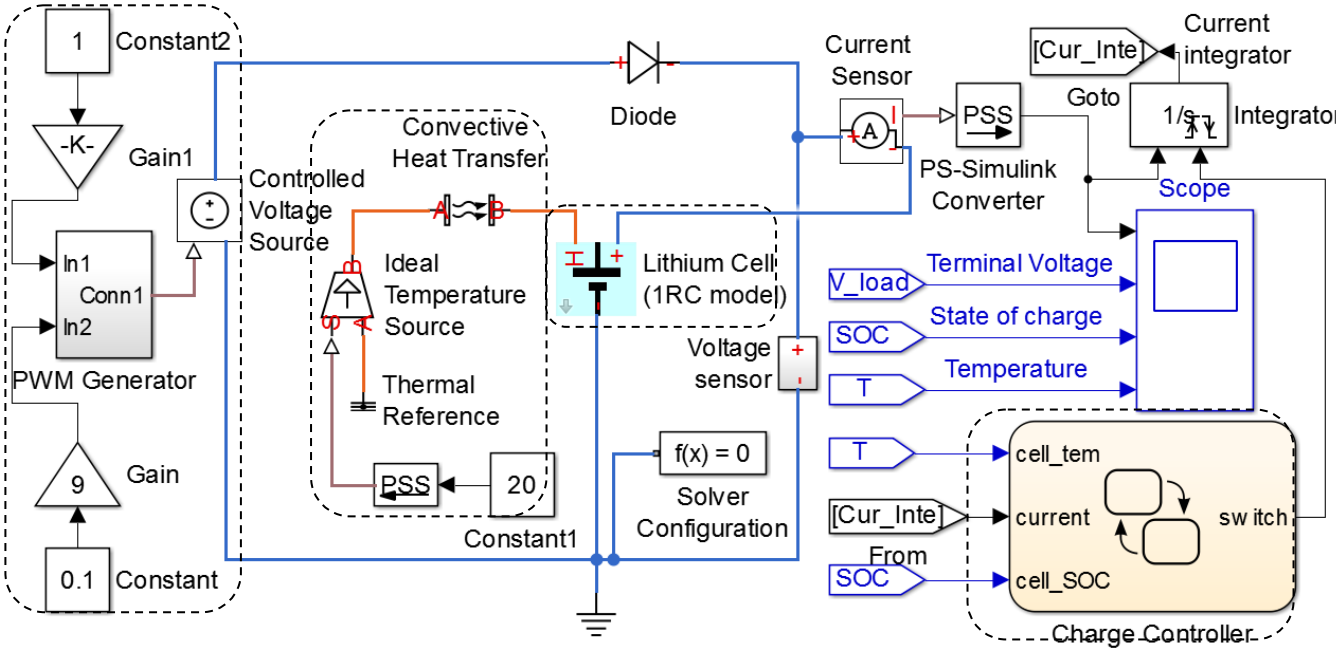

Figure 2. Matlab/Simulink-based model implementation for pulse-based charging

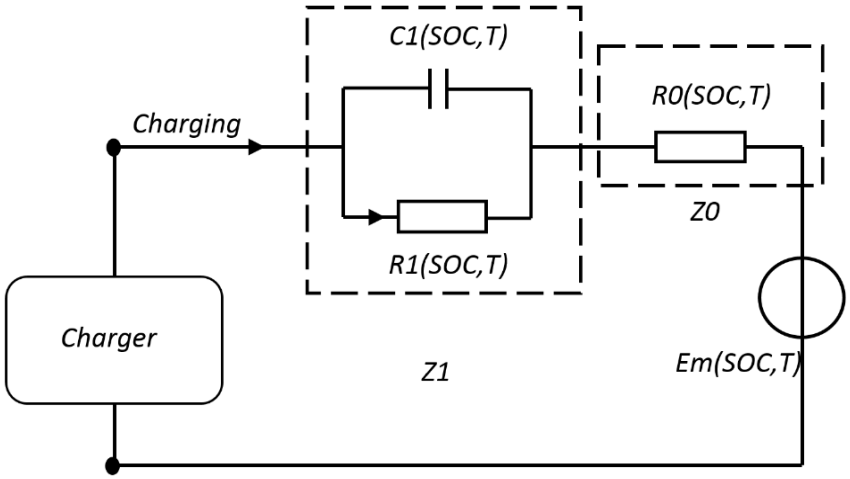

Figure 3. Equivalent circuit model for lithium battery

C. Finite state machine of the proposed charger system

The implemented state flow of the proposed charger system is shown in Figure 4. We proposed five states including *Charge_Start*, *Search_Fre*, *Search_Duty*, *Charge* and *Charge_Complete* state. The start state is a default state, initializing all parameters and checking the battery status. The charger has to verify that the temperature and SOC of the battery are under the respective thresholds of 45 ◦C and 80% by using the function of *status_check* at the lower left corner. Throughout the charge process, the battery status is checked at regular intervals.

The optimal frequency is obtained in *Search_Fre* state by calling the *cur_check* function. In *Search_Duty* state, the duty cycle is adjusted to ensure that the charging current abides by the polarization curve. Through the LUT, we can check the polarization curve by calling *acc_cur* check function. Adopt the efficient pulse in the Charge state until the SOC is divisible by 0.2% or 5%, then cycle back to the *Search* state once again. Eventually, entering the *Charge_Complete* state means the end of the charge process. The user-defined functions update_param_fre and update_param_duty in MATLAB/Simulink models are shown at the top left corner. In the function, get_param and set_param are used to get and change the gain coefficient of the gain blocks shown in Figure 3 during the simulation.

Generated code by Simulink Coder support for Stateflow which is integrated with low level code will be used in the hardware-in-the-loop testing in the following section. Once all the above work is done, the new charging algorithm is ready for the hardware test.

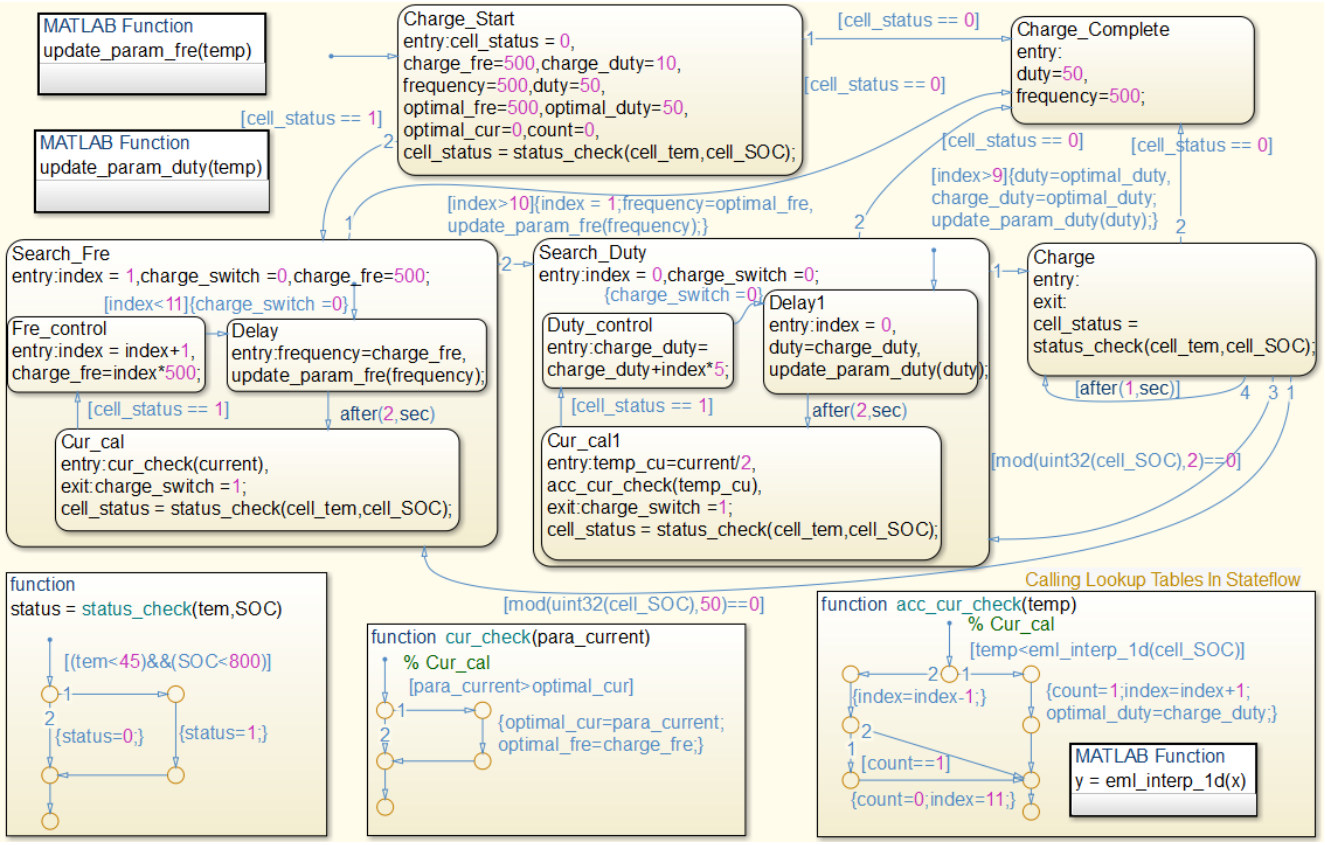

Figure 4. Finite state machine of the proposed charger system

D. Experiment environment for charging test

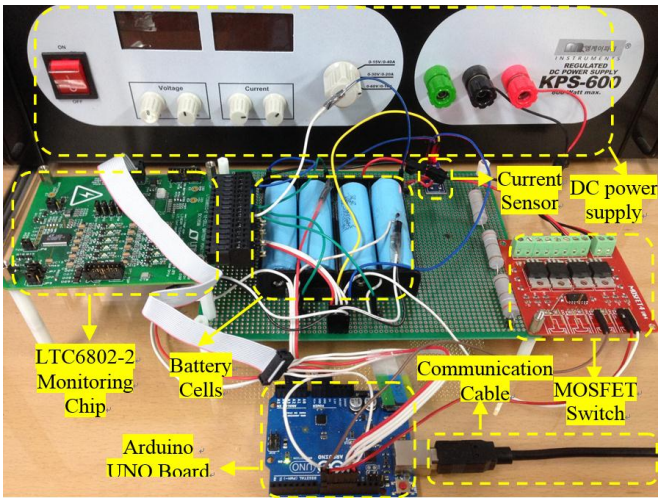

Figure 5. Experiment environment for charging test

The model KPS-600 regulated power supply as the charging source, using MOSFET switch module to control the frequency and duty cycle. But the micro controller IO port cannot control IRF540 MOSFET, indirectly control it via PS2801C optical coupler. PWM is generated by Arduino UNO. The experiment environment setup here is shown in Figure 5. Through the communication cable, Arduino UNO board can report battery status to host computer, such as charging current, cell terminal voltage, cell temperature, pulse frequency and duty cycle.

IMPLEMENTATION RESULTS

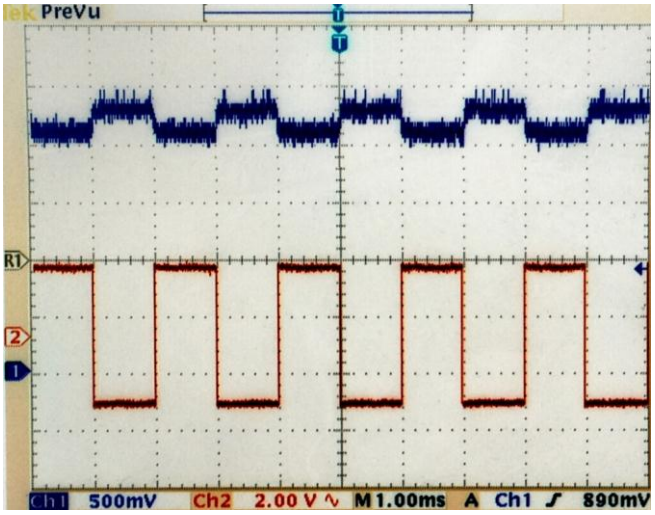

Figure 6. Wave form captured by oscilloscope

Take 500Hz frequency and 50% duty cycle as an example, obtained the output voltage signal from current sensor and the corresponding PWM control signal via oscilloscope is shown in Figure 6. One column means 1ms, pulse period is 2ms. There are two channels in this figure. One horizontal line of the first channel (blue line) means 500mV and the second channel (red line) means 2V. The first channel on behalf the magnitude of charging current, can be infer that charging current is approximately 1.35A (185mV/A)

because current sensor output value is 2.5V when current is zero. The second channel on behalf the PWM control signal, which has the same period and the amplitude is 5V.

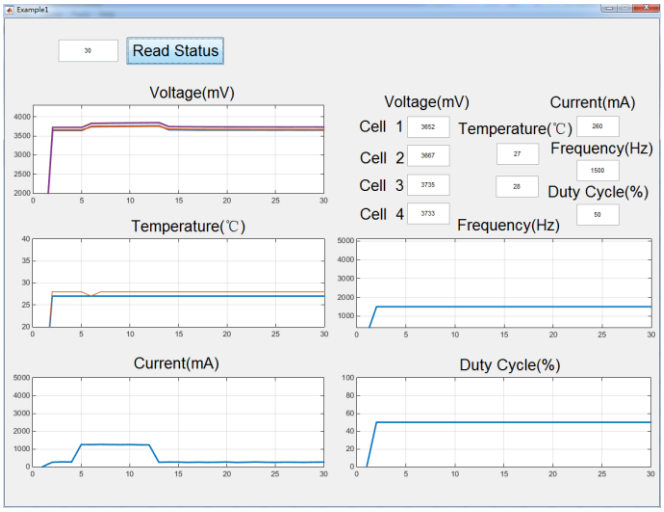

Figure 7. GUI interface for battery status observation

The graphical user interface (GUI) is developed in MATLAB [13]. Through the observation in the charging process, the status of the battery can be displayed in real time is shown in Figure 7. With MATLAB support package for Arduino, Arduino board can communicate with MATLAB by a USB cable. Through the defined communication protocols, MATLAB can acquire real time data such as charging current, pulse frequency, duty cycle, cell voltage and temperature from the Arduino board. In this way, we can take further measures such as stop charging to prevent cell damage in case of the abnormal status.

CONLUSION

In order to speed up the evaluation of the implemented charger system, improve the evaluation quality and make the results of evaluation more comparable, a development methodology consisting of three steps in terms of development procedure was proposed. We developed a workflow to access the performance of pulse charging algorithm for multiple cells. Equivalent circuit model is adopted to simulate the lithium battery and charging strategy is built in Stateflow, which is a control logic tool developed by Mathworks^{TM}. A quick verification of the charging control logic can be performed and then integrated the generated code from Statefow with the low level hand-code. Testing is done on the customized hardware for pulse charging. Sumsung 18650 type Li-ion battery as the test case to evaluate the newly designed charging algorithm. A design flow of developing fast pulse charging algorithm and a hardware platform linked with MATLAB are introduced in this paper. Developers are able to reuse our experience in rapid design and validation as a reference throughout the development process. As a case study, we showed fast development capabilities of designer-specific battery-charger algorithm and its evaluation experience using the proposed hardware-software emulation framework. Through the above work, we successfully developed the pulse charging strategy and evaluated the charging performance in three month ahead of schedule.

ACKNOWLEDGMENT

This research was supported by the Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(2014R1A6A3A04059410).

REFERENCES

[1] J. Li, “Design of a li-ion battery charger with cc-cv-ct regualtion loop,” Electric Information and Control Engineering, April 2011, pp. 4088–4091.
[2] M. D. Yin, J. Youn, D. Park, and J. Cho, “Efficient frequency and dutycycle control method for fast pulse-charging of distributed battery packs by sharing cell status,” International Workshop on Ubiquitous Wireless Sensor, August 2015, pp. 40–45.
[3] L.-R. Chen, “A design of an optimal battery pulse charge system by frequency-varied technique,” Industrial Electronics, IEEE Transactions, February 2007, pp. 398–405.
[4] L.-R. Chen, “Design of duty-varied voltage pulse charger for improving li-ion battery-charging response,” Industrial Electronics, IEEE Transactions, February 2009, pp. 480–487.
[5] Purushothaman B K, Landau U, “Rapid charging of lithium-ion batteries using pulsed currents,” Journal of The Electrochemical Society, 2006, pp. 533–542.
[6] Broy, Manfred, Sascha Kirstan, Helmut Krcmar and Bernhard Schätz. "What is the Benefit of a Model-Based Design of Embedded Software Systems in the Car Industry?." Emerging Technologies for the Evolution and Maintenance of Software Models, 2012, 343-369
[7] Miller S, Wendlandt J, “Real-Time simulation of physical systems using Simscape,” MATLAB News and Notes, 2010, pp. 1–13.
[8] Arduino UNO and Genuino UNO, https://www.arduino.cc/en/Main/ArduinoBoardUno
[9] Kumar, Ashish, “Schaem: A method to extract statechart representation of FSMs”, Advance Computing Conference, 2009, pp. 1556-1561.
[10] Liao Y G, Fu D T, “Design and development of a teaching tool for lithium-ion battery management system,” Conference for Industry and Education Collaboration, 2015, pp. 1–12.
[11] Dai Haifeng Z X, “A smart current and voltage acquisition system with high accuracy for EV applications,” International Journal on Smart Sensing and Intelligent Systems, December 2012, pp. 1–19.
[12] T. Huria, M. Ceraolo, J. Gazzarri, and R. Jackey, “High fidelity electrical model with thermal dependence for characterization and simulation of high power lithium battery cells,” Electric Vehicle Conference, Mar 2012, pp. 1–8.
Creating Graphical User Interfaces, https:// www.mathworks.com/help/pdf_doc/matlab/buildgui .pdf