Topology Identification for Smart Cells in Modular Batteries

Sebastian Steinhorst, Martin Lukasiewycz
TUM CREATE, Singapore, Email: sebastian.steinhorst@tum-create.edu.sg

***Abstract*—This paper proposes an approach to automatically identifying the topological order of smart cells in modular batteries. Emerging smart cell architectures enable battery management without centralized control by coordination of activities via communication. When connecting smart cells in series to form a battery pack, the topological order of the cells is not known and it cannot be automatically identified using the available communication bus. This order, however, is of particular importance for several battery management functions, including temperature control and active cell balancing which relate properties of the cells and their location. Therefore, this paper presents a methodology to automatically identify a topological order on the smart cells in a battery pack using a hybrid communication approach, involving both the communication and the balancing layer of the smart cell architecture. A prototypic implementation on a development platform shows the feasibility and scalability of the approach.**

I. Introduction and Related Work

Mobile and stationary Electrical Energy Storages (EESs) are an important component of sustainable technologies. Both in the area of emerging Electric Vehicles (EVs) as well as for smart-grid applications, batteries are the EES of choice. Here, Lithium-Ion (Li-Ion) batteries are superseding other cell chemistries due to their superior energy density, power density and cycle life.

Beside their significant benefits, Li-Ion batteries, however, require very sophisticated Battery Management Systems (BMSs) to operate safely and sustainably [1]. In state-of-the-art battery packs, centralized BMS architectures are dominating. Here, a central master controller specifically tailored to the pack acquires all sensor information such as voltage and temperature of individual cells as well as the pack current. Furthermore, it processes and creates control signals for cell balancing. Cell balancing is an important BMS function, maintaining the State-of-Charge (SoC) of all cells in their safe operating range such that upper and lower thresholds are not crossed in order not to damage the cells [2]. For larger packs, a hierarchical BMS architecture is commonly applied where slave boards control modules of cells, still fully supervised by the master controller [3].

Recently, approaches to decentralize the control of the battery pack have emerged, driven by the requirements of higher efficiency, modularity and easier integration of battery packs to cope with the perennial demand for shorter design cycles and time-to-market. This decentralization can either be achieved at the level of sensing and balancing hardware with conventional centralized control or by fully autonomous smart cells that individually control their parameters and coordinate their actions with other smart cells via communication [4], [5]. The concept of smart cells enables a plug-and-play integration of battery packs where individual cells or cell modules can be added or replaced any time and the decentralized system then automatically adapts to this new pack configuration.

Fig. 1 shows a battery pack consisting of four cells with a smart cell architecture. Each cell has its own Cell Management Unit (CMU) with a Sensor and Balancing Module (SBM) forming the balancing layer, a microcontroller forming the computation layer, and a communication interface forming the communication layer. Each smart cell manages its local properties itself, e.g., estimation of its SoC. Pack-level functions such as pack-SoC calculation or cell balancing are coordinated cooperatively via communication between the smart cells.

This work was financially supported by the Singapore National Research Foundation under its Campus for Research Excellence And Technological Enterprise (CREATE) program.

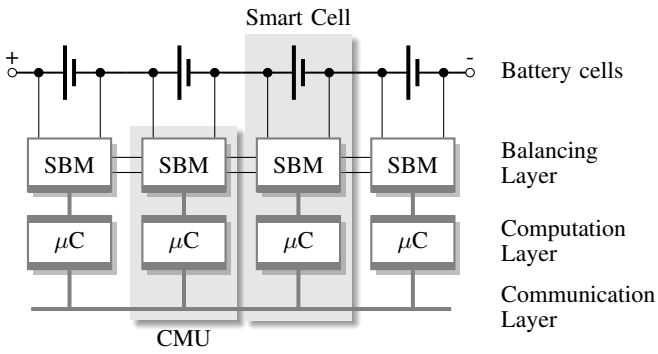

Fig. 1: *A battery pack formed by four series-connected smart cells. Each smart cell consists of a cell and a Cell Management Unit (CMU) which provides a Sensor and Balancing Module (SBM) forming the balancing layer, a microcontroller forming the computation layer and a communication interface forming the communication layer of the architecture.*

While the goal of the system architecture of smart cells is to make them self-organizing, state-of-the-art approaches have not considered to automatically identify the topological order of the series-connected smart cells in the battery pack. Topological information, however, is mandatory for many battery management functions such as temperature profiling and active cell balancing. Therefore, a manual configuration by mapping unique IDs of the cells to their linear order in the series-connection of battery cells is necessary during pack assembly or whenever the topology is changed by removing or adding cells.

Contributions of this paper. As a remedy to manually configuring smart cells to obtain a linear order, this paper introduces an automated methodology at runtime to identify the topological order of smart cells in a battery pack, providing three main contributions outlined in the following. (1) We present a hybrid communication approach for a smart cell architecture, enabling neighbor identification using both the balancing layer as well as the communication layer in Section II. (2) Utilizing the neighbor identification approach, we develop a topology identification algorithm for the battery pack to generate a linear order on the series-connected smart cells in Section III. (3) We discuss our implementation on a smart cell development platform, enabling experimental performance evaluation of the proposed methodology in Section IV. Concluding remarks are given in Section V.

II. Smart Cell Architecture enabling Neighbor Identification

In this section, the neighbor identification approach is introduced, utilizing both the communication layer as well as the balancing layer of the smart cell architecture. For this purpose, we discuss the architectural characteristics of smart cells and the integrated active cell balancing circuit which enable hybrid communication.

A. Smart Cell Architecture

In the context of the recent trend towards decentralization of battery pack architectures for higher modularity, aiming at plug-and-

play integration, it is considered to bring the BMS to the cell level in the form of smart cells [4], [5].

In contrast to a traditional hierarchical centralized BMS architecture where a master controller performs all control and monitoring tasks, smart battery cells manage themselves individually at cell level and coordinate cooperative actions via a communication interface to perform pack-level tasks such as cell balancing or pack-SoC determination. For this purpose, smart cells consist of a cell and a Cell Management Unit (CMU) as illustrated in Fig. 1.

The CMU enables the smart cell to perform computation, communication and control. Hence, it provides a SBM, microcontroller module for computation and a communication interface. The SBM senses the parameters of the cells such as its voltage and temperature. It furthermore provides cell balancing capabilities that are important for maintaining the usable capacity of the pack. The microcontroller module performs all computations such as control algorithms, involving SoC estimation for the cell and cell balancing.

Local properties of the cell are controlled autonomously by the CMU and all actions on the battery pack that require coordination involve distributed control, using messages exchanged on the communication layer.

Each CMU is directly powered by the cell it is attached to. Consequently, all BMS functions performed by the smart cells have to be highly energy-efficient in order to allow the battery pack to perform its main purpose of providing energy and using as little power for the actual management of the cells as possible.

The autonomous management at cell level and cooperative approach at pack level brings several advantages. While centralized BMSs have a single point of failure, the smart cell architecture is inherently resilient. Furthermore, the local CMU will stay with the cell even when battery packs are disassembled. This enables a second life application of cells, as all data such as the State-of-Health (SoH) of the cell are permanently available. Second life applications can, for instance, use cells from an EV battery pack in a less demanding environment such as for stationary storage of photovoltaic energy.

B. Active Cell Balancing

An important factor for both maintaining the health of the battery pack as well as operating in an energy-efficient fashion is cell balancing. We will introduce the architecture in the following and use it in the next subsection to enable the hybrid communication approach for neighbor identification.

Temperature and manufacturing variations cause an unequal SoC distribution to develop between cells during charging and discharging. Consequently, cell balancing has to be performed between the cells, as charging or discharging of the battery pack has to be stopped when the first cell reaches the upper or lower SoC threshold.

Widely applied state of the art of battery pack charge equalization is passive cell balancing by dissipating energy of cells with a higher SoC until it matches the charge of the weakest cell [6]. By contrast, active cell balancing transfers charge between cells and, therefore, can increase the SoC of cells that otherwise reach their lower threshold. This ultimately increases the usable capacity of the battery pack and conserves energy that would be dissipated in case of passive balancing. Inductor-based modular active balancing architectures such as [7] have been implemented in commercial products, see [8], and are well-suited to be integrated into the SBM of the CMU. With an active cell balancing module, smart cells can exchange charge with their neighbors, enabling an efficient form of charge equalization across the cells of a battery pack.

Fig. 2 shows SBMs containing a modular active cell balancing circuit architecture with two Metal-Oxide-Semiconductor Field-Effect Transistor (MOSFET) switches per cell which can transfer charge between neighboring cells. In order to perform the balancing, some MOSFETs have to be controlled using Pulse Width Modulation (PWM) signals, resulting in four transfer phases that are periodically repeated in the kHz-range.

C. Neighbor Identification using Hybrid Communication

With the active balancing circuit architecture and the communication interface both present in the CMU, a hybrid communication approach can be developed that enables neighbor identification. This neighbor identification will then be the basis for the topology identification methodology presented in the next section.

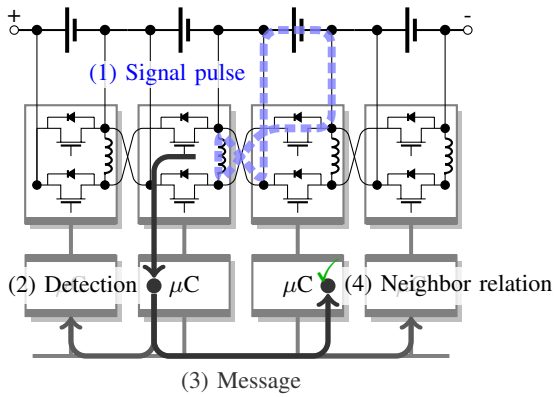

Fig. 2: *Neighbor identification by hybrid communication using balancing layer signaling with an active balancing architecture using two MOSFET switches and an inductor per SBM.*

While communication bus interfaces such as Controller Area Network (CAN) or wireless implementations such as ZigBee are capable of sending broadcast messages, there is no information on the position of the sender with respect to the network topology available in such messages. Hence, although physically determined by the electrical series-connection of cells in the battery pack, no topological order of smart cells can be derived from their messages on the bus alone, as both the relative as well as the absolute position of the sender are unknown to the receivers. Consequently, additional information has to be generated to identify the position of a sender and receiver. For this purpose, we propose a hybrid communication approach involving balancing layer signaling to augment the existing communication interface. With this hybrid communication, neighbor identification between smart cells is possible as illustrated in Fig. 2 and described in the following.

A smart cell configures its balancing layer to generate a charge pulse in the inductor of its higher (with regard to the positive terminal of the battery pack) neighboring smart cell (Fig. 2 (1)). This pulse, representing a signal in its inductor, is detected by the microcontroller of the neighboring smart cell (Fig. 2 (2)) which sends a message on the communication bus (Fig. 2 (3)). Consequently, the cell which triggered the signal is notified. As the message contains the node ID of the triggered neighbor, the neighbor relation between the two nodes is now known to the triggering cell (Fig. 2 (4)).

In the next section this neighbor identification will enable an algorithm to automatically identify a linear order on the series-connection of smart cells in a battery pack.

III. Topology Identification Methodology

In this section, the topology identification methodology is introduced. First, the approach is outlined and its goal defined in a formal terminology. Then, based on identification of the nodes in the smart cell network and their neighbor relation, a topology identification algorithm is developed and illustrated on an example.

A. Approach

The goal of the topology identification approach is to create a set of nodes represented by smart cells with a total order. While the cells have physically obtained a total order by being electrically series-connected in the battery pack, this order, as well as the amount of cells in the battery pack, is not known to the smart cells. Therefore, a methodology is required that enables the smart cells to identify the set of nodes, its cardinality and the total order of the set. This information is required for the network of smart cells to coordinate pack-level functions such as active cell balancing, where knowledge of the individual position of cells and neighbor relations in the battery pack are required.

Formally, we want to identify a set S of $|S| = n$ smart cells representing the nodes in the series-connected battery pack and a total order with relation $\leq$, resulting in a linearly ordered set. For a linearly ordered set, the properties of antisymmetry, transitivity and totality hold for all pairs of elements. Here, $(S, \leq)$ is representing the order of electrical connection of the nodes such that for two nodes $s, \tilde{s} \in S$ it holds $s \leq \tilde{s}$ if and only if the absolute voltage potential of the negative pole (anode) V_s^- of s has the same or a higher absolute potential than the voltage $V_{\tilde{s}}^+$ at the positive pole (cathode) of $\tilde{s}$. As the elements in S are unique, the totality of the order is given.

For obtaining S, n and $\leq$ with the given architecture of smart cells, the hybrid communication possibilities as discussed in Section II-C have to be utilized to develop an algorithm.

B. Identification of Set of Nodes

The communication layer is capable of sending broadcast messages that can be received by all nodes connected to the bus. In bus architectures such as CAN, messages contain node IDs (representing bus IDs) that are used to identify the sender of messages. The only requirement is that these IDs have to be unique for each node on the network. In the context of smart cell networks, the unique IDs have to be generated during initialization at runtime with a protocol that translates a hardware-implemented or randomly generated unique ID to a unique bus ID to avoid ambiguities and message collisions. Once a bus ID is assigned to each node, a broadcast message of each node with its ID enables identification of the set S. For this purpose, every node receives the messages and determines the number n of individual IDs.

C. Utilizing Neighbor Identification

$(S, \leq)$ can only be determined if, in addition to the bus-based communication layer that contains no topological information, the position of cells in the battery pack can be acquired. Therefore, the neighbor identification approach presented in Section II-C has to be involved to create information on the order of the cells.

Formally, we will generate directed local signals between neighboring smart cells such that a signal $\tau(s)$ sent from node s is detected as $\rho(\tilde{s})$ by a neighboring node $\tilde{s}$.

In the charge transfer network provided by the SBM, signals can be sent either to the *higher* neighbor, meaning that the neighboring node has a higher potential and is therefore located closer to the positive terminal of the battery pack, or to the *lower* neighbor, meaning that the neighboring node has a lower potential and is therefore located closer to the negative terminal of the pack. In the following, we will consider that a triggering signal created in a node always implies that the signal is directed towards the higher neighbor:

$$\tau(s) \wedge \rho(\tilde{s}) \Leftrightarrow V_s^+ = V_{\tilde{s}}^- \tag{1}$$

The detection of the signal is reported by the triggered node via a bus message containing its bus ID. With this approach, the triggering node now receives the information about the ID of its direct higher neighboring node. For the topological order, we now obtained the information that s and $\tilde{s}$ are directly adjacent:

$$\tau(s) \wedge \rho(\tilde{s}) \Rightarrow \tilde{s} \leq s \wedge \nexists s' \in S | \tilde{s} \leq s' \leq s \tag{2}$$

For iteratively generating a topological order on the series-connection of smart cells, this triggering is repeated in a controlled sequence until all nodes have identified the IDs of their higher neighboring node. The positive terminal node of the battery pack is reached if no response is received to a trigger signal, hence s is the minimal element in $(S, \leq)$:

$$\tau(s) \wedge \nexists \rho(\tilde{s}) \Rightarrow s = \min S \tag{3}$$

D. Topology Identification Algorithm

The topology identification is performed in several steps that are summarized in Algorithm 1 and described in the following.

In the initialization phase, generation and broadcasting of a random unique bus ID on each node is performed. With this information, the set S is available on each node. It contains an element for each received ID and a mapping $ID(s)$, storing the ID value for s (line 1).

Algorithm 1: Topology identification

```
1  obtain S with ∀s, s̃ ∈ S, s ≠ s̃ : ID(s) ≠ ID(s̃)
2  S' ← {}
3  while ∃s ∈ S \ S' do
4      p ← arg min_{s∈S\S'} ID(s)
5      I ← {p}
6      repeat
7          τ(p)
8          if ρ(p̃) ∧ p̃ ∉ S' then
9              I ← I ∪ {p̃}, p̃ ≤ p
10             p ← p̃
11         end
12     until p̃ ∈ S' ∨ ¬ρ(p̃);
13     (S', ≤) ← (S', ≤) ∪ (I, ≤), ∀s ∈ S', s̃ ∈ I : s ≤ s̃
14 end
15 return (S', ≤) representing linearly ordered S
```

The concept of the subsequently detailed algorithm is to select an initial pivot element and populate a linearly ordered set $(I, \leq)$ of identified neighbors between the pivot and the positive terminal of the battery pack. Further iterations will identify the remaining nodes between the negative terminal of the battery pack and the initial pivot element, iteratively merging $(I, \leq)$ into the finally returned $(S', \leq)$, representing the complete linearly ordered set S.

The topology identification algorithm is performed iteratively while not all elements from S have been added to S' (line 3). First, a pivot element p is selected which has the lowest ID in the network with respect to the linearly ordered set of received IDs (line 4). Consequently, p is the first element added to the linearly ordered set $(I, \leq)$ containing p and nodes towards the positive terminal of the battery pack (line 5). In the following, I is iteratively populated by applying $\tau(p)$ (line 7) which triggers $\rho(\tilde{p})$ (line 8) in its higher neighbor $\tilde{p}$, per definition broadcasting $ID(\tilde{p})$. $\tilde{p}$ is added to I with the ordering information $\tilde{p} \leq p$ (line 9). This process is iteratively repeated by setting $p \leftarrow \tilde{p}$ (line 10) for the next iteration until no response is received within a defined time interval, indicating that the positive terminal of the battery pack has been reached in the first iteration of the while-loop. In subsequent runs, the inner loop will iterate until it triggers an element already in S' (line 12). S' and I are merged such that the linear orders in S' and I are maintained and every element in S' is ordered in front of any element in I (line 13) when the inner loop terminates.

In subsequent iterations of the outer while-loop, the remaining nodes below the initial pivot element have to be handled as long as not all nodes in S have been added to S' (line 3). Finally, the algorithm returns $(S', \leq)$ which contains all linearly ordered elements from S (line 15).

Example. Fig. 3 illustrates an example run of Algorithm 1. We consider five smart cells that have physically obtained a topological order by their electrical connection in a battery pack, which is, however, unknown to the smart cells (line 1). Each smart cell has a unique ID and, hence, from the initial broadcasting, the unordered set S is known. With S' being empty in the beginning, we choose the set element s_2 with ID 2 to be the pivot node as it has the lowest ID in S (line 2). Filling I, the signaling is now started and identifies the element s_1 with ID 7 to be the higher neighbor of the pivot (line 3). s_1 becomes the new pivot, but I is not further extended as s_1 represents the positive terminal node in the battery pack and the signaling is hence not answered. Among the remaining elements $\{s_3, s_4, s_5\}$ in $S \setminus S'$, the one with ID 3 is chosen to be the pivot as it has the smallest ID in the remaining set (line 4). The signaling triggered from this node is answered by the node with ID 2, which is already in S' and therefore the signaling iteration stops (line 5). S' and I are merged into S', retaining the linear order (lines 6-7). In the final run, ID 6 is chosen as the smallest ID among the remaining

(1) electrical topology: 7 ≤ 2 ≤ 3 ≤ 9 ≤ 6
(2) $S = \{s_1, s_2, s_3, s_4, s_5\}$, $p = s_2$
(3) $I = S'$: 2 ≥ 7
(4) $S \setminus S' = \{s_3, s_4, s_5\}$, $p = s_3$
(5) I: 3
(6) merge
(7) S': 3 ≥ 2 ≥ 7
(8) $S \setminus S' = \{s_4, s_5\}$, $p = s_5$
(9) I: 6 ≥ 9
(10) merge
(11) S': 6 ≥ 9 ≥ 3 ≥ 2 ≥ 7
(12) $S \setminus S' = \{\}$, finished

Fig. 3: *Example run of the topology identification algorithm (Algorithm 1) for a battery pack with five smart cells and shown electrical topology. Each box represents a smart cell and is labeled with the respective bus ID.*

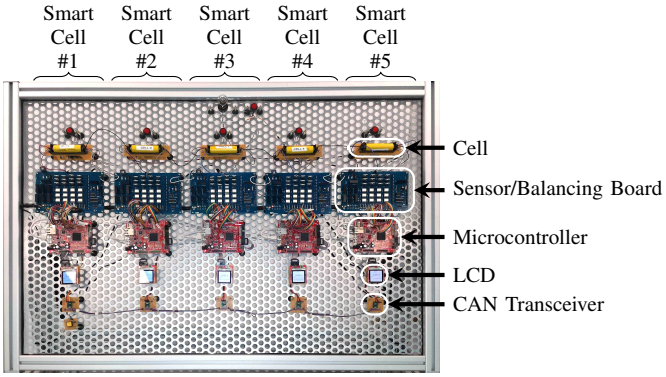

Fig. 4: *Development platform with five smart cells. Each smart cell has an individual Sensor/Balancing Board implementing the SBM, a microcontroller and a CAN bus communication interface.*

nodes s_4, s_5, triggering ID 9. The signaling of ID 9 is answered by ID 3, whose corresponding set element is already in S' (lines 8-9). Another merging is performed (lines 10-11). As there are no further elements in $S \setminus S'$, the algorithm terminates, returning $(S', \leq)$ as the linearly ordered set representing the topology of the electrical connection of the smart cells (line 12).

IV. Implementation and Scalability

In this section, we discuss the implementation details of the proposed topology identification methodology on a development platform with experimental measurements regarding the scalability of the approach.

A. Development Platform

The proposed topology identification methodology introduced in Section III is implemented on a smart cell development platform with five smart cells as shown in Fig. 4. Each smart cell is formed by a 18650 Li-Ion cell that is connected to a custom sensor and balancing Printed Circuit Board (PCB) implementing the SBM. The SBM is operated by a STM32F407 Cortex-M4 microcontroller board. This microcontroller is significantly overdimensioned in order to first develop the software architecture without restrictions and then map it to an appropriate smaller controller at a later stage. The communication between the smart cells is performed via a CAN bus with a data rate of 1Mbps, connecting the microcontroller boards. Our sensor and balancing board contains 12 Power-MOSFET switches and an inductor and can implement different active balancing architectures by setting some of the switches to be constantly on or off, respectively. Here, the board has been configured to the architecture presented in Section II-B. The boards for sensing and balancing, microcontroller and CAN communication logically form the CMU.

Our software architecture enables the smart cells to configure themselves in the initialization phase with the topology identification algorithm proposed in this paper. After this self-configuration, the smart cells perform active cell balancing whenever necessary.

B. Scalability

The overall initialization time for the self-configuration of the development platform with five smart cells is $t_{\text{init}}^{(5)} \approx 10\text{s}$ until it enters its normal operation state after performing the topology identification. Within t_{init}, the initial unique ID distribution with pivot node selection requires $t_{\text{UID}} \approx 4\text{s}$ on the platform. This time is not significantly influenced by the number of nodes to identify, as the chosen deadline introduces a high safety margin. Subsequently, each instance of neighbor signaling, detection and information broadcasting requires $t_{\text{neighborsignal}} \approx 1\text{s}$. The final order distribution is again performed with negligible influence by the number of network nodes $t_{\text{orderdist}} \approx 1\text{s}$. Consequently, only $t_{\text{neighborsignal}}$ scales with the number of smart cells installed in the battery pack. As exactly one signaling for each smart cell is required in the topology identification algorithm, we can obtain the following expression for a pack consisting of n smart cells:

$$\begin{aligned} t_{\text{init}}^{(n)} &= t_{\text{UID}} + n \cdot t_{\text{neighborsignal}} + t_{\text{orderdist}} \\ &\approx 4\text{s} + n \cdot 1\text{s} + 1\text{s} \\ &\approx (n+5)\text{s} \end{aligned} \tag{4}$$

The topology identification only has to be performed when a battery pack is either initially assembled or its topology is modified by replacing, adding or removing smart cells. Hence, the algorithm runtime is acceptable for realistic battery packs such as those in EVs where 50 to 100 layers of series-connected cells exist, requiring less than two minutes.

V. Concluding Remarks

This paper presented an approach to topology identification in smart cell architectures for modular batteries. To enable plug-and-play integration, smart cells have to identify their position within the battery pack automatically when connected. While the order is implicitly defined by the electrical connection, it is not known to the smart cells. Consequently, we present a hybrid communication approach for neighbor identification that uses both the cell balancing layer for local signaling to neighbors and the communication layer for message broadcasting. The neighbor identification is then utilized in a topology identification algorithm, providing the information of a linear topological order to the smart cells. The presented methodology contributes to making the smart cell architecture self-configuring, removing the requirement for manual programming of ordered IDs. The approach has been implemented for a smart cell development platform and experimental evaluation shows its feasibility and scalability.

References

[1] M. Brandl *et al.*, "Batteries and Battery Management Systems for Electric Vehicles," in *Proc. of DATE*, 2012.
[2] L. Lu, X. Han, J. Li, J. Hua, and M. Ouyang, "A review on the key issues for lithium-ion battery management in electric vehicles," *Journal of Power Sources*, vol. 226, no. 0, pp. 272 – 288, 2013.
[3] M. Lukasiewycz, S. Steinhorst, S. Andalam, F. Sagstetter, P. Waszecki, W. Chang, M. Kauer, P. Mundhenk, S. Shreejith, S. A. Fahmy, and S. Chakraborty, "System architecture and software design for electric vehicles," in *Proc. of DAC 2013*, 2013.
[4] A. Otto, S. Rzepka, T. Mager, B. Michel, C. Lanciotti, T. Günther, and O. Kanoun, "Battery management network for fully electrical vehicles featuring smart systems at cell and pack level," in *Advanced Microsystems for Automotive Applications 2012*. Springer, 2012.
[5] S. Steinhorst, M. Lukasiewycz, S. Narayanaswamy, M. Kauer, and S. Chakraborty, "Smart cells for embedded battery management," in *Proc. of CPSNA*, 2014.
[6] N. Kutkut and D. Divan, "Dynamic equalization techniques for series battery stacks," in *Proc. of INTELEC*, 1996.
[7] N. Kutkut, "A Modular Nondissipative Current Diverter for EV Battery Charge Equalization," in *Proc. of APEC*, 1998.
[8] S. Wen, "Cell balancing buys extra run time and battery life," *Texas Instruments Analog Applications Journal*, no. 1, 2009.