

# Migration of automotive powertrain control strategies to multi-core computing platforms – lessons learnt on smart BMS

Eric Armengaud, Ismar Mustedanagic, Markus Dohr, Can Kurtulus, Marco Novaro, Christoph Gollrad, Georg Macher

AVL List GmbH {eric.armengaud, ismar.mustedanagic, markus.dohr, georg.macher}@avl.com

AVL Research and Engineering {can.kurtulus}@avl.com

AVL Software and Functions GmbH {christoph.gollrad}@avl.com

Ideas and Motion {marco.novaro}@ideasandmotion.com

**Abstract**— An important acceptance criteria for electric mobility is the capability to efficiently use the energy stored in the cells of a battery over the vehicle lifetime. The BMS (Battery Management System) plays a central role by estimating the state of charge (current energy available) and state of health (degradation due to ageing effects) of the cells. Improvement of the estimation quality has a direct impact on the battery and thus vehicle range. It is the target of the INCOBAT project to improve the BMS system by means of new electronic components, new control strategies and new development methods in order to achieve cost reduction and performance (driving range) increase. In this context, the introduction of multi-core computing platforms aim at providing more computing resources and additional interfaces to answer the needs of new automotive control strategies with respect to computing performances and connectivity (e.g., connected vehicle, hybrid powertrains). At the same time, the parallel execution, resulting resources and timing conflicts require a paradigm change for the embedded software. Consequently, efficient migration of legacy software on multi-core platform, while guaranteeing at least the same level of integrity and performance as for single cores, is challenging. In this paper, the lessons learnt during the migration of the BMS control strategies to the INCOBAT BMS computing platform will be presented.

**Index Terms**—Multi Core, Electric Vehicle, Battery Management Systems

## I. INTRODUCTION

IN recent years, electric mobility has been promoted as the clean and cost-efficient alternative to combustion engines. Although there are already solutions on the market, mass take-up has not yet taken place. There are different challenges that hinder this process from an end user point of view such as costs of the vehicle, driving range, or infrastructure support. Several of these challenges are directly connected to the battery, the central element of the full electric vehicle (FEV). The costs of the battery sum up to 40% of the total costs of a FEV, and the driving range of a FEV is strongly reduced in comparison to the combustion engine.

The aim of INCOBAT<sup>1</sup> (INnovative COst efficient management system for next generation high voltage BATteries, started in October 2013) is to provide innovative and cost efficient battery management systems for next generation HV-batteries. To that end, INCOBAT proposes a platform concept in order to achieve cost reduction, reduced complexity, increased reliability as well as flexibility and higher energy efficiency. Moore's law [1], stating the doubling of the computer capacity every 2 years, is still a strong enabler for this fast function increase and at the same time cost-per function decrease. The current development trend for computing platforms has moved from increasing the frequency of single cores to increasing the parallelism (increasing the number of cores on the same die) to limit the power dissipation while improving the performance. Multi-core and many-core technologies have strong potential to further support the different technology domains, but at the same time present new challenges.

Hence, the automotive industry is facing a growing gap between the technologies and required level of expertise to make best use of them. The computing platforms are becoming more and more high-performance with concurrent computing capabilities, larger embedded memories as well as increasing number of integrated peripherals. Low-level mechanisms (e.g., memory protection, diagnostics) typically provided by the basic software or operating system are now being moved into the microcontroller. The complexity of these computing platforms is very high, the related user guides is made of several of thousands of pages. Regarding automotive operating systems and low-level basic software (BSW), the AUTOSAR approach is following a similar trend by standardizing several tens of BSW modules in several tens of thousands pages of specification. Similarly for the application software (ASW, e.g., control strategy for hybrid powertrains), the complexity is already very high and still growing by the introduction of new applications such as advanced driver assistance systems (ADAS) or predictive energy management strategies.

<sup>1</sup> <http://www.incobat-project.eu/>

Additionally, the functional integration of the control strategies (e.g., transmission with combustion engine and e-drive) further raises the complexity of the resulting application.

The automotive industry is confronted to the central question how to migrate, optimize, and validate a given application (or set of applications) on a given computing platform with a given operating system. A knowledge transfer is required to take over the role of control system integrator and identify the application requirements (both functional and non-functional) and to perform a mapping to the SW and HW architecture. The quality of this mapping has a direct impact on the performance of the control system, and thus of the entire mechatronic system.

Main contribution of this paper is to summarize the lessons learnt during the migration of the existing BMS control strategy to the INCOBAT BMS platform based on multi core technology. The paper is organized as follow: Section 2 introduces the INCOBAT project as well as the BMS platform based on Infineon AURIX<sup>TM</sup> CPU. Section 3 will present the AVL BMS core functions (state of charge SoC, state of health SoH and State of Function SoF estimation) and the adaptation at the functional level that were performed to take advantage of the multi-core platform as well as to enable migration. Section 4 discusses the migration at software integration level and summarizes performance increase achieved. Finally, Section 5 concludes this work.

## II. THE INCOBAT PROJECT – AN OVERVIEW

The aim of INCOBAT is to provide innovative and cost efficient battery management systems for next generation HV-batteries. To that end, INCOBAT proposes a platform concept in order to achieve cost reduction, reduced complexity, increased reliability as well as flexibility and higher energy efficiency [2].

The targeted outcomes of the project are:

- Very tight control of the cell function leading to an increase of the driving range
- Radical cost reduction of battery management system
- Development of modular concepts for system architecture and partitioning, safety, security, reliability as well as verification and validation, thus enabling efficient integration into different vehicle platforms.

To achieve these ambitious targets, the technical approach chosen in INCOBAT primarily relies on the following 12 technical innovations (TI) regrouped into four innovation groups (see Fig. 1):

- *Customer needs and integration aspects*: ensures a correct identification of customer needs and enables an efficient integration into different platforms. This is supported by the use of mission profiles (TI-01) – in order to take into account the different driving styles of the customers, the different traffic conditions in the same scenarios and the different tracks – and by the integration into a demonstrator vehicle (TI-12)

- *Transversal innovation: consistent concept and specification*. This second group targets the optimization of the system architecture and its consistent description over the technologies and over the system hierarchies. This aspect aims at providing a consolidated basis in order to simplify later industrialization of the proposed technologies. This includes the TI-02 “Model-based systems engineering” to improve correctness / completeness / consistency of system specification, the TI-03 “System architecture - efficient partitioning of the functionalities” for system optimization at BMS or even vehicle level and the TI-04 “Integration of multiple functionalities” to reduce the number of electronic control units (and thus related costs) in the vehicle.
- *Technology innovation: E/E control system*: This third group aims at improving the components of the E/E control system. Regarding the electronic parts, it consists of TI-05 “Multicore computing platform for additional computing resources” and the TI-06 “Smart and integrated module management unit”. From the software part, this is achieved by the TI-07 “Modular SW platform” and by TI-08 “Improved BMS control algorithms”
- *Transversal innovation: improving system maturity*: This last group targets the evidences related to the trust on the technical solutions with respect to correct operation (TI-10 “Design and validation plan including reliability consideration”), functional safety and security (TI-09 “Definition and integration of safety and security concept”) as well as reliability (TI-11 “Reliability and robustness validation”). This group of technical innovations is an indicator for the maturity of the proposed technology and further provides information on the efforts required for proper integration and validation of the system.

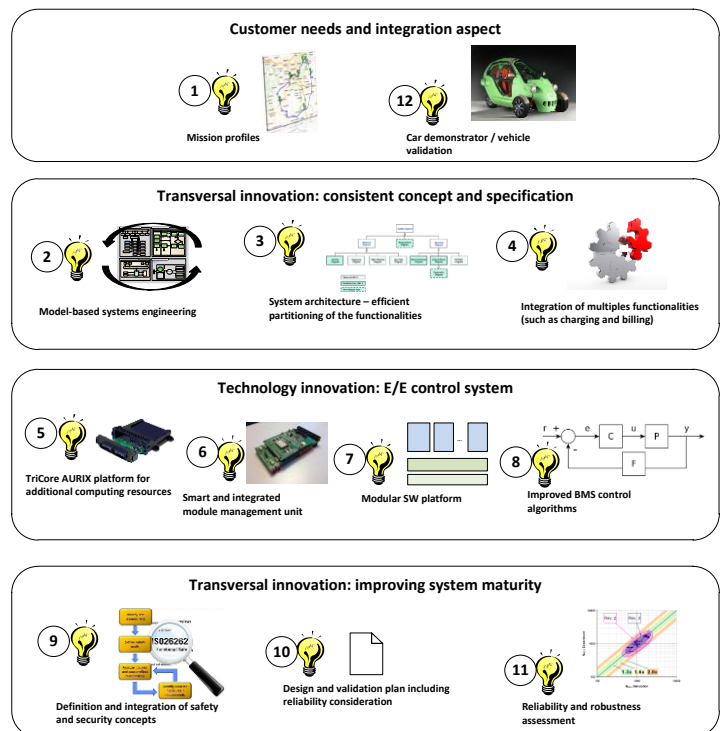


Figure 1: Technical innovations within INCOBAT

The INCOBAT BMS CCU (see Figure 2) is based on the Infineon multicore processor AURIX TC275 with an innovative multicore architecture [3]. This device supports the concurrent execution of mixed ASIL functions up to ASIL-D [4]. It offers a rich set of peripherals such as A/D converters and timers for data capturing and it has a reasonable number of IOs to support BMS applications. In conjunction with the specific power-supply ASIC TLF35584 it is possible to supply the CCU and support ISO26262 requirements with a minimum number of components.

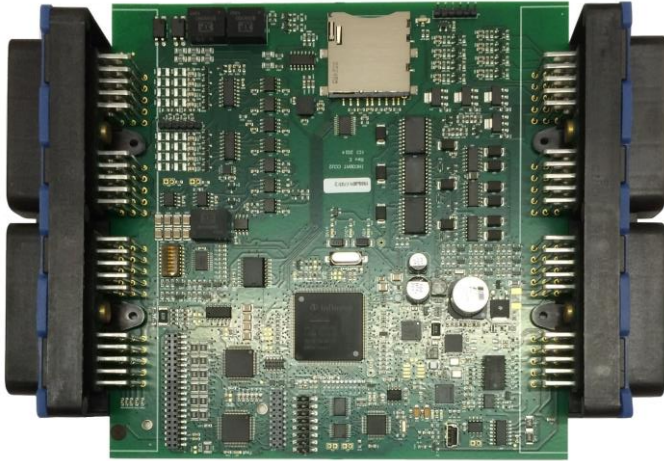


Figure 2: INCOBAT BMS CCU Prototype Hardware

Regarding the SW developments in INCOBAT, a modular development platform is required. Hence, the control strategy and the application software in general are expected to come from different providers and to require different levels of criticality. The activities of SW architect – to define the SW blocks as well as their interfaces – and the activities of SW integrator – integrating the different SW modules and ensuring correct operation of the entire control system – are especially challenging in the context of automotive supply chain with constraints related to functional safety (ISO 26262 [5]). A modular platform is required to enable the distributed development and flexible deployment of different control strategies and applications in an efficient way.

For the SW developments in INCOBAT, the proposed common, modular software development platform consist of:

- a layered SW architecture, consisting of several layers and components as well as their interfaces, providing access for the applications to the underlying HW capabilities,
- a suitable SW development tool chain, which supports the application SW developers by means of an effective and consistent development process to seamlessly integrate their particular applications to an overall BMS.

### III. FUNCTIONAL MIGRATION OF THE BMS CONTROL STRATEGY

#### A. Model-based battery state estimation

##### 1) Introduction

Accurate estimation of battery parameters such as SoC, SoF and SoH requires model-based estimation methods in which a representative model of the battery is utilized as part of the algorithm. Incorporation of a battery model enables many possibilities including:

- Capturing expected behavior, to be compared with actual measurements for inference of parameters causing the deviation,
- Ability of algorithms to handle different operating conditions and usage scenarios,
- Generation of a “prediction error” signal that is necessary for modern estimation methods such as Kalman Filters, or other similar types of Observers.
- Ability of the algorithms to be adapted to different cell types, and cell chemistries via only adapting the incorporated battery model
- Possibility of making predictions of various battery behaviors as well as battery condition

Model based approach to battery state and parameter estimation, therefore has many advantages.

An overview of how a battery model may be used as part of the algorithms is given in Figure 3. This architecture uses the output of the battery model for the prediction step of the estimation method, and compares it with the actual measurement to generate an “error” signal. This error signal is then utilized as part of the algorithm to calculate important signals such as SoC and eventually to update the model and adapt it to the actual battery and operation conditions. This architecture is what enables adaptation to various operating conditions and handles situations where the model may not represent battery behavior accurately.

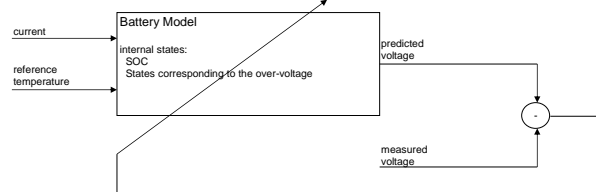


Figure 3: Overview of Model Based Estimation for Battery States

The battery model has to be constructed and parameterized in accordance with the type of cell used in the target application since there are large differences of behavior between various types. The model parameterization requires test data that captures the behavior of the cell (e.g. terminal voltage response, surface temperature) under various operation conditions in terms of ambient temperature and usage (i.e. load current).

However, there are certain drawbacks included with usage of model based estimation techniques, such as relatively high computation power demand, necessity of a high quality parameterization to obtain satisfactory performance and the necessity to ensure stable behavior over all possible operating conditions.

## 2) SoC Estimation

The State-of-Charge is defined as the percentage of the maximum possible charge that is present in the battery. The SoC can't be measured directly, but an accurate SoC on pack and on cell level is mandatory for the energy management control system (State of Function Calculation). Several methods have been developed in the past:

- Coulomb counting (Ampere-Hour Counting)
- Open circuit voltage
- Neural Networks
- Heuristic interpretation of measurements, Fuzzy Logic
- Model-based estimation methods
  - Kalman Filter for battery SoC determination
  - Luenberger Observer
- Sliding-Mode Observer
- LPV Observer

A detailed description of available methods for SoC estimation is given in [6]. Furthermore, common algorithms for State-of-Health (SoH), State-of-Function (SoF), Remaining useful Life (RUL) are introduced.

Nowadays, the most widely used SoC-estimation algorithms are Kalman-filtering techniques. They are based on an equivalent circuit model (ECM) of the battery.

The ordinary KF can be used for linear models. Since the battery is a highly nonlinear system, a Kalman filter is necessary, which permits the use of nonlinear models. The Extended Kalman Filter has the ability to handle such kinds of models. A big advantage of a KF is that it considers measurement and process noise due to voltage sensor inaccuracy, temperature fluctuations etc. to estimate the states of the battery.

One of the first use of an EKF (Extended Kalman Filter) for SoC estimation of lithium-batteries is published in [7], [8] and [9]. These papers describe the mathematical background, the modeling of the battery with its identification requirements and the final implementation of the EKF for SoC estimation. Additionally, an algorithm for SoH estimation is presented.

The publications [10] and [11] describes a modified KF. The Sigma-Point Kalman filter is a more accurate estimation approach, although the computational demand is of the same order as EKF. A comparison between SPKF and EKF show an improvement of the estimation. Moreover, an advanced algorithm is presented, where state and parameter estimation is done simultaneously. A summary of the mentioned algorithms is presented in [12].

A slightly modified Kalman filter with lower computational demand is used for SoC estimation as part of INCOBAT. The algorithm runs on module level.

The SOC-function estimates:

- Module States: States of the battery model (model has 6 states, including SoC of the battery)
- Module State of Charge
- Module OCV: Open circuit voltage of the module

The relatively high computational demand makes it very challenging to run the KF on cell-level with state-of-the-art battery management systems (BMS), especially for battery packs intended for high voltage applications. That is why SOC-estimation is done on module level.

## 3) SoH Estimation

The State-of-Health is a measure of the condition of the battery compared to the fresh battery. It is characterized in the loss in capacity and the increase in resistance.

To assure correct SoF and SoH estimation of a battery pack, it is necessary to have information of the cell-SoC and the cell-resistance as well. An algorithm to estimate these values is described in [13]. The basic mathematical background of the estimation approach is described in the next sections.

The cell observer is used to determine the deviation of each cell compared to the mean module state already estimated as part of the SoC function and the module resistance observer. The aim is to utilize a much simpler linear algorithm for computational efficiency.

The used model is a linearized model that describes the deviation of each cell from the module mean rather than full cell dynamics. The algorithm is a modified version of the Recursive Least Squares, a well-established algorithm commonly used in state and parameter estimation problems.

## 4) SoF Estimation

The State-of-Function consists of measures for the ability to fulfill the application specific function of the battery. The BMS has to calculate current and voltage limitations such that the battery is operated in a safe operating mode and the performance and lifetime targets are met.

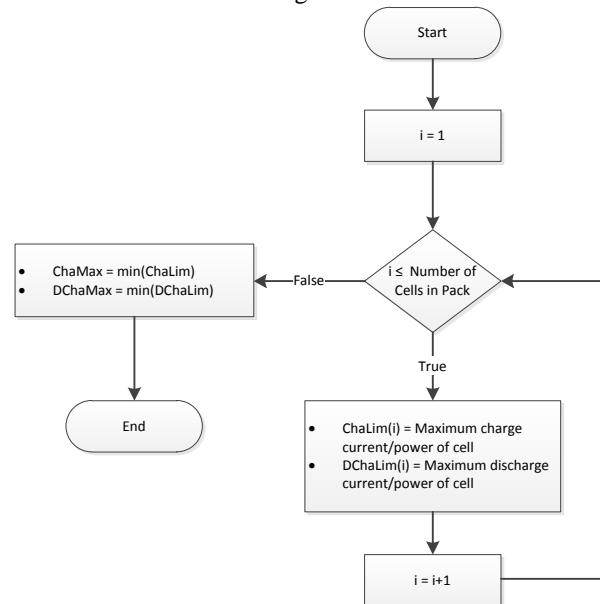


Figure 4: State of Function Flow Chart

State of Function calculation is responsible for determining available functionality of the battery, which would be either current or power that can be supplied to the powertrain, considering the maximum allowable cell voltage and the maximum allowable operating current. The SoF calculation is based on a prediction of future cell voltages for a calibrated prediction time with an electrical model.

Basically, the limits are calculated for each cell, as it is shown in Figure 4. The flowchart shows an iteration over the maximum number of cells in the battery pack. For SoF limits,

the results of the ‘worst’ cell (e.g. highest inner resistance) are taken into account. In more detail, the calculation estimates a current limit, based on the average cell voltage in a module. Afterwards, the limits are corrected by the worst cell in a module. The worst cell is identified by the calculated dSoC and dR values.

The algorithm provides the maximum charge and discharge current limits, as well as the maximum charge and discharge power. The power limitations can be calculated by a multiplication of the current and voltage limits.

### 5) *Function inter-dependencies*

Figure 5 provides an overview how the algorithms for SoC, SoH and SoF estimations are implemented. SoC and SoH estimations run on module-level due to the high computational demand, whereas the SOF are calculated per cell. The battery pack SoC is simply the average of the estimated Cell SoC values. To get a plausible SoH of the module resistances of the battery pack, delta SoC and delta R should be considered.

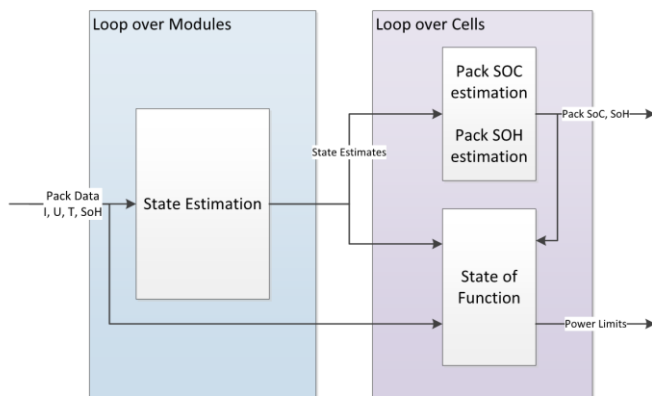


Figure 5: Information flow of estimation algorithms

### 6) *Functional migration to parallel computing scheme*

From the point of view of the model-based battery state estimation, the main focus is on the analysis of the existing algorithms (currently running on single core computing platform) in order to identify possible improvements with respect to SoX estimation accuracy while making use of the additional computing power of a multicore processor. The main target is to improve the accuracy of the estimations from a group of cells (e.g. a module) down to single cell level. In our case, the sensor platform already provides the information required to measure the cells. The challenging factor here is to run a dedicated instance of the existing model estimation algorithm for each single cell instead of one instance for a group of cells. The required computing power (number of algorithm instances running in parallel) is therefore directly dependent on the modelling accuracy (reduction of the number of cells taken into consideration for one algorithm instance). A higher modelling accuracy provide more accurate information on the status of the cells, thus moving the limits for the use of each single cell (and therefore of the entire battery at the end) from a conservative boundary to a more real limit. Consequently, it can be assumed that the range of the vehicle

and cycle life of the battery can be increased due to the precise estimation approach.

## B. *Electrochemical Impedance Spectroscopy (EIS)*

### 1) *Introduction to EIS*

This methodology measures dielectric properties of a medium as a function of frequency. In particular, as applied to the battery cells, the goal of the EIS is to determine the impedance parameters, and the state of health (SoH) of the cells as a function of the impedance. In order to successfully determine the EIS spectrum it is necessary to take into account certain inherent problems in the method and the component under test. The EIS analysis is based on the following prerequisites:

- the system must be linear
- the system parameters should not vary over time
- the system must be single input, single output (SISO)

The lithium battery is not generally satisfying these requirements: therefore, additional assumptions have to be made.

First, the characteristic of a battery is not linear: To calculate the impedance it is therefore necessary to proceed to a linearization. The used technique is to identify a working point on the electrical characteristic and to generate a small perturbation of it. Analyzing a small enough portion of a cell's current versus the voltage curve, it is considered to be linear. Therefore, in normal EIS practice, a small (1 to 10 mV) AC signal is applied to the cell: this is small enough to confine the test into a pseudo-linear segment of the cell's current versus voltage curve.

Second, the battery parameters are not constant: in general, even with open battery-terminals (i.e. zero current), the battery voltage varies over time depending on the previous history. To allow the stabilization of the battery voltage it is necessary to wait for the conditions of electrochemical balance in the battery. The required time, also referred to as “settling time” or “relaxation time” depends on the temperature (ion mobility) and is estimated in the order of a few hours. A measurement made before reaching the equilibrium condition produces data with variations especially in the lower part of the spectrum. These variations are more or less evident depending on the imbalance inside the cell.

Lastly, the voltage in a battery does not depend exclusively on the current flowing through it, but also on other parameters, in particular temperature and SoC. During each EIS measurement, these parameters must remain constant, in order not to influence the output voltage. In general, it must be ensured that the battery open circuit voltage does not vary within the range of the test, or this change will be computed in the spectrum of impedance.

Based on the assumptions above, the stimulus signal needed for the EIS test shall have the following characteristics:

- The spectrum of the stimulus shall adequately cover the whole frequency range that has to be analyzed (typically from 0.01 Hz to 1 kHz)
- The signal amplitude shall be “small enough” to avoid triggering any nonlinear response in the battery

As a drawback, the smaller is the signal amplitude, the worse is the signal to noise ratio. For this reason, a dedicated HW solution is needed to obtain a good resolution in the acquired signal. In particular, the proposed solution is:

- Amplify and adequately filter the signal, due to the small signal amplitude
- Remove the OCV voltage, which is not useful for EIS measurements
- Read each voltage value through a differential amplifier

All of the above-mentioned features are implemented in a dedicated EIS daughterboard, working together with the CCU-BMS board. Figure 6 below shows a functional block diagram of the daughterboard, integrating:

- The EIS Command Generator: a voltage DAC needed to generate the stimulus signal
- 12x EIS cell voltage measurement circuitry (differential amplifier + OCV cancelling + 4th order Bessel anti-aliasing filter)
- 2x EIS current measurement circuitry (4th order Bessel anti-aliasing filter)

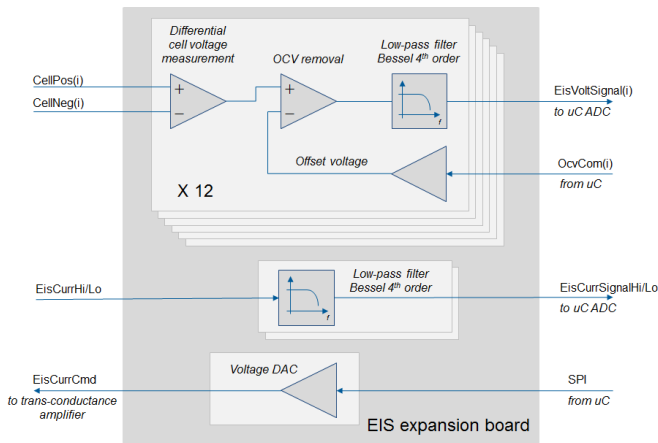


Figure 6: EIS daughterboard

In particular, the OCV has to be measured before applying the stimulus signal, and then removed through the dedicated DAC; this is part of the features implemented in the EIS software.

Finally, the stimulus signal generated by the daughterboard is a voltage signal; an external amplifier (in particular, a transconductance amplifier) is needed to drive the current that is injected into the cells. The chosen signal, used as a current stimulus, is a sum of sinusoidal current waveforms with predefined frequencies, in the range 0.01 Hz to 1 kHz, with a selectable current amplitude.

## 2) The resulting EIS algorithm

The EIS algorithm (Figure 7) injects a known current stimulus into the battery cell, reading the resulting voltage. Due to the assumptions described previously, the EIS algorithm will run after a relaxation time, needed for the battery parameters to reach a steady condition.

The measured signals from the battery shall then be analyzed, for each frequency, to determine the spectrum of the signal at that frequency. The idea is to correlate the measured signal to

the input waveform, to obtain magnitude and phase information about the analyzed signal.

The response waveform from the battery typically has a DC offset, harmonic distortion components, and noise components generated by the cell. Nevertheless, the element of the measured signal, which needs to be analyzed, is the one at the same frequency as the generator waveform. All of the spurious components of the measured signal need to be rejected so that accurate measurements of the fundamental signal at the generator frequency can be made.

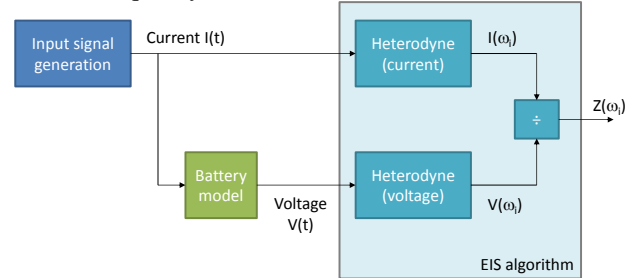


Figure 7: EIS algorithm overview

The measured system output is multiplied by both the sine and cosine of the test frequency  $\omega$ . The results of the multiplications are then fed to two identical integrators, where they are averaged over  $T$  seconds. As the averaging time increases, the contribution of all unwanted frequency components go to zero and the integrator outputs become constant values which depend only on the gain and phase of the system transfer function at the test frequency.

Harmonics are rejected by the correlation process, and noise is rejected by averaging the signal over a number of cycles; the averaging associated with the correlating frequency response analyzer acts as a band pass filter with center frequency  $\omega$ . As the average time  $T$  increases the bandwidth of the filter becomes narrower, thus the corrupting influence of wide band noise is increasingly filtered out as the correlation time is increased.

Averaging over a complete cycle avoids certain measurement errors associated with offsets on the system output; the performed simulations demonstrates that acquiring on a time window of three complete periods, we obtain an effective rejection of all frequencies above 0.1 Hz. Since the minimum frequency is 0.01 Hz, three complete periods corresponds to 300 seconds.

The result of the correlation process is made up of two components one of which is referred to as the *Real* (or in phase) component, the other is the *Imaginary* (or quadrature) component. By performing simple mathematical operations on these raw measurement results, it is possible to obtain the magnitude and phase of the impedance.

## 3) EIS Software implementation

The EIS Software consists of several SW components, using resources either directly from the Aurix microcontroller or through the EIS daughterboard

In particular, the EIS consists of:

- Complex Device Drivers (CDD):

- OCV removal: generates the signals needed to remove the OCV from the measured cell voltage; OCV shall be measured before applying the stimulus signal, and then canceled through a dedicated DAC signal.
- EIS Command Generator: its purpose is to generate the EIS Command signal (voltage reference) representing the current stimulus to be forced into the battery pack
- **iLLD** (Low-Level-Drivers from Infineon): mainly this will be used for the analog input signals acquisition and for the coherent measurement of:
  - Cell voltages - both DC and AC (useful EIS signal) - for each battery cell
  - EIS current flowing in the battery module/pack
- **Application**
  - Data Processing System for the calculation of the EIS spectrum

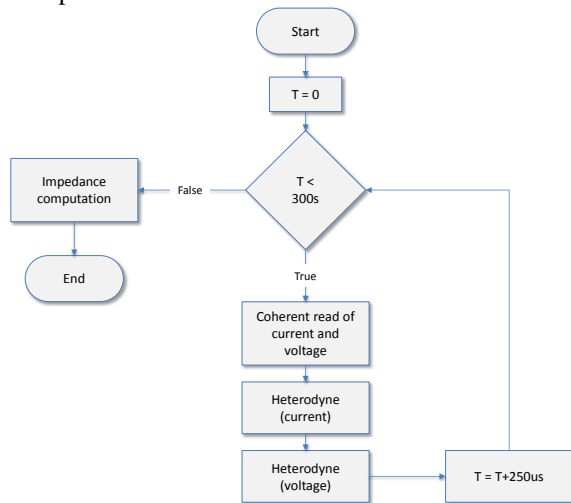


Figure 8: EIS application flowchart

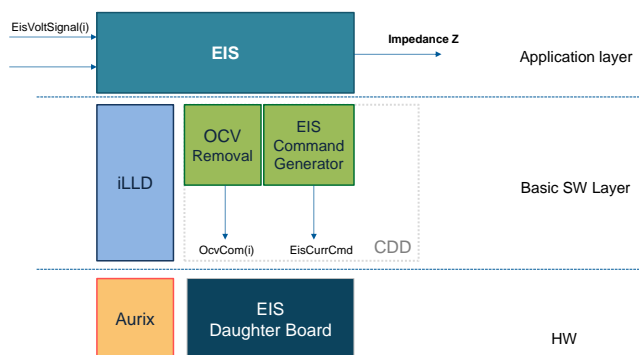


Figure 9: EIS software architecture

In particular, the application layer implements the EIS algorithm, as described above; the flowchart in Figure 8 shows an overview of the algorithm, that is executed for the whole time of the test (300 seconds), subsequently using then the last output from the integrators to compute the impedance. The resulting SW architecture is depicted in Figure 9.

#### 4) Functional migration to parallel computing scheme

Since the EIS algorithm has been developed from scratch within the INCOBAT project, no specific migration from

single core (sequential) to multi-core (parallel) computing scheme was required. The challenges are focused on the proper SW integration and resource managements. This will be discussed further in the next section.

## IV. SOFTWARE INTEGRATION ON MULTICORE PLATFORM

### A. Software development environment

The large variety of use cases as well as business organization is leading to different requirements on the development framework and build environment:

- Flexible configuration of source files, include files and directories for building code for each core. This targets increase in build efficiency as well as constructive integration [14] by the capability of updating a core independently from each other
- To have a sufficient intellectual property (IP) protection linking of external pre-compiled objects / libraries to the main binary of each core shall be possible. This is especially required in case of distributed development – means different teams / company integrating their IP into a common computing platform
- Adaptations to other compilers shall be possible with less effort.
- Integration of additional tools shall possible with less effort. These two last items are related to the distributed development of the entire SW by different teams relying on different development processes and consequently different tools

In the context of INCOBAT, the development environment shall be a low cost solution with capabilities to be deployed by each INCOBAT partner while minimizing the licensing costs. To meet all these requirements for the INCOBAT project a set of tools was used to establish the SW development framework. The basic configuration of the SW development environment consists of a standard set of make files and target rules, a common memory mapping and compiler associated make and linker files.

To allow utilization of the ERIKA<sup>2</sup> operating system from Evidence into the integration and build process, the command line interface (CLI) from RT Druid was integrated in the build environment in form of make target rules. The OS configuration, including the definition and core allocation of counters, alarms, task, spinlocks and resources is done via OSEK implementation language (OIL) file, which is feed in RT Druid for the generation of the OS Erika related code and header configuration files for each core.

The build process is setup in such a manner that for each core a separate binary image is generated. This allows SW updates on one core without the need of rebuilding the other cores. Of course, this mechanism is only applicable if the applied changes do not affect the other's core SW and if the SW of the different cores can access peripherals of the microcontroller only via one dedicated interface. The SW code allocation to the different cores is done statically via one manual configurable make file, in which for each core application

<sup>2</sup> <http://www.evidence.eu.com/>

source and include files or directories, pre-compiled objects and libraries can be setup.

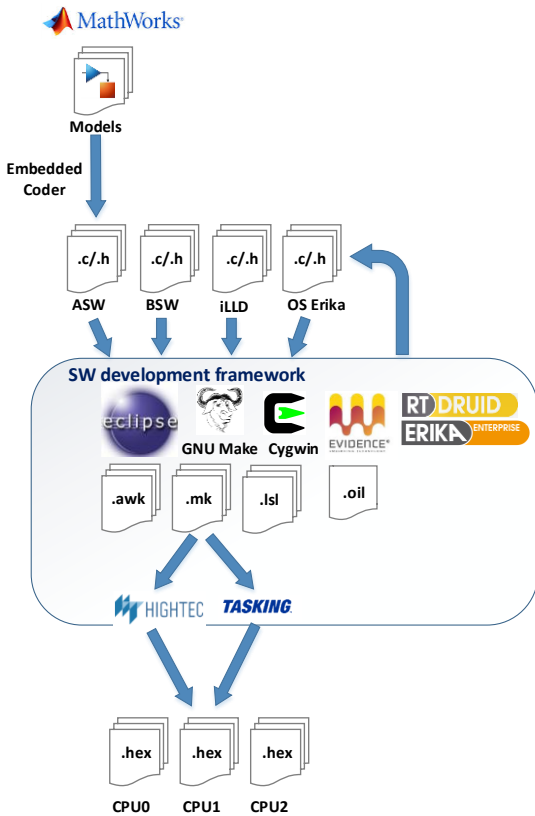


Figure 10: INCOBAT SW development framework

The memory allocation is done aligned to the AUTOSAR memory mapping approach and configured memory sections in the linker script. Depending on the currently identified CPUx in generation, the linker performs allocation of code and data to predefined flash and core local data scratchpad RAM memory sections.

The shared data of the cores is defined and allocated by the master core and placed into the local data scratchpad memory of that core which is the producer of the data element. The exchange of the memory information to the slave cores is done by dumping of master core’s binary shared memory sections and export to a separate shared sub linker file which is generated during the build of the master core. The sub-linker script is used in the later build phases of slave cores for address resolutions of the shared data elements.

Another important aspect for the SW development environment was tool integration – in our case the mapping of the system information with the SW development framework. During the scope of the project, different tool interfaces for generating AUTOSAR aligned SW information were generated. The proposed tool interfaces mainly relies on four level of exchanges – all aligned with the AUTOSAR or OSEK standard. The first level (AUTOSAR tool-bridge) aims at describing the SW components (SW-C) and their interfaces – and serves proper integration of control strategies and application SW. The second level (RTE configuration) targets the description of the real-time environment for according

configuration. The third level focuses on basic SW (BSW) configuration, while the fourth level aims at describing the operating system (tasks available in the system and their related options). More information is available in [15], [16] and [17].

*B. Software architecture*

To ensure modularity and reusability the SW architecture was split in several layers aligned to AUTOSAR:

- Infineon iLLD – similar to AUTOSAR MCAL, providing abstraction to HW I/O’s, other peripheral modules, and startup code for the cores
- OS Erika – OSEK/VDX certified asymmetric operating system, where each core has its own copy of the OS instance
- BSW including complex device drivers and other coded BSW services
- ASW and ASWIL, to reduce the complexity of the system each ASW component is accessing its data via separate interfaces from the BSW or IOC module. During the execution of the function group, each core is using its local buffered data wherever possible to minimize execution time caused by inter-core accesses and remote blocking.

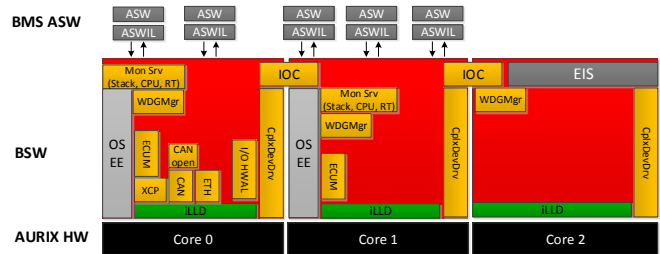


Figure 11: INCOBAT SW architecture

For the multicore capabilities, several SW functionalities were used similar to the currently defined and supported AUTOSAR concepts:

- Synchronized master slave startup and shutdown approach of the cores. During startup of master core 0 the other two slave cores are in idle. They startup with a synchronization barrier during startup of the OS. During shutdown, the reversed order is used and master core waits until synchronized shutdown of slave cores.
- Functional based inter-core data exchange of single signals or groups similar to the AUTOSAR Inter-OS-Applicator communicator (IOC).
- Usage of spinlocks to guarantee data consistency for core-to-core data exchange. The spinlock mechanism was combined with immediate suspension of interrupts in order to reduce the time of remote blocking. Additionally, to prevent from deadlocks nested acquisition of spinlocks was avoided.

*C. Verification environment – mini-HiL*

An important target and basis for the proper SW development, verification and integration is the deployment of appropriate test environment. Hence, the test environment shall be flexible



enough to enable different kind of stimulation for the functions developed and realistic enough to accurately model the physics related to the system to control. In the context of INCOBAT, the simulation environment is playing an important role in different work-packages and tasks:

- White box testing: verification of single SW function such as control strategy (e.g., battery state estimation), safety function (e.g., control of battery’s main relays) or basic SW (e.g., low-level drivers). Target is to provide the direct environment for these functions, therefore sometimes shortening the SW system by investigating only one function
- Grey box testing: validation of SW system and especially correct integration of the functions into the control system as well as correctness of the interfaces
- Black box testing: validation of the safety mechanisms – especially ensure correct reaction of the control system in case of hazardous situations

In the context of INCOBAT, different approaches are used:

- Model or SW in the loop (MiL / SiL): direct verification of single SW function
- Hardware in the loop (HiL): verification and validation of set of functions up to SW system in a real control system
- Vehicle demonstrator: prototyping validation in vehicle

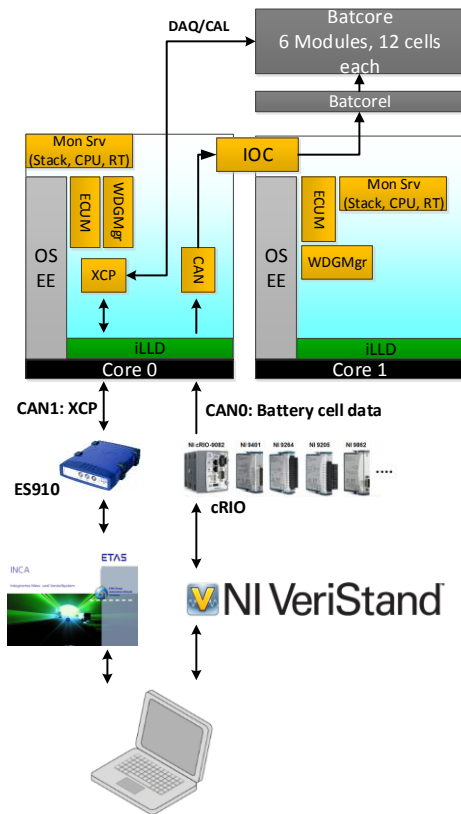


Figure 12: HiL test environment

While MiL and SiL are state of practice and will not be discussed any further, a dedicated mini-HIL platform for the efficient validation of the BMS (E/E system including satellite units and central controller) has been deployed. This platform

is conjointly used for white, grey and black box testing – with different focuses as described previously, For a validation of the SW multicore integration approach and the battery SoX estimation algorithms following HiL test environment was setup, see Figure 12.

In a first phase, the battery estimation algorithms were stimulated with battery cell data via CAN wrapper. For the stimulation of the battery load profiles the test and automation environment from NI Veristand was used in combination CompactRIO HW. With support of the XCP protocol, measurement and calibration access to each core’s local memory was established.

*D. Performances achieved and lessons learnt*

A first comparison of the battery state estimation, while moving the computation accuracy from module to cell level, is shown in the following. The red curves represent the results of the modified estimation approach. Figure 13 illustrates the minimum and maximum cell SoC, while the discharge current limits during the cycle is illustrated in. Figure 14. Especially at the end of the cycle (at 6000s), a higher difference between the estimation on cell-level (red curve) and the estimation on module-level (blue curve) can be recognized. It is important to note that the results achieved in SiL and MiL environment are highly coherent (SoC and SoH difference below 0.05%) therefore confirming the correct integration into the multi-core computing platform.

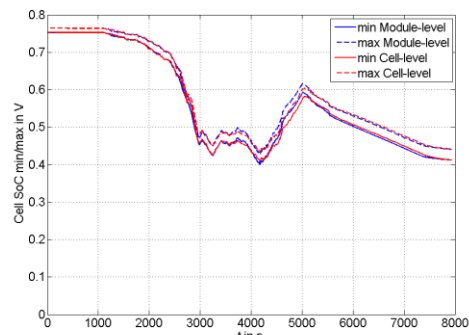


Figure 13: SoC computation at module and cell level

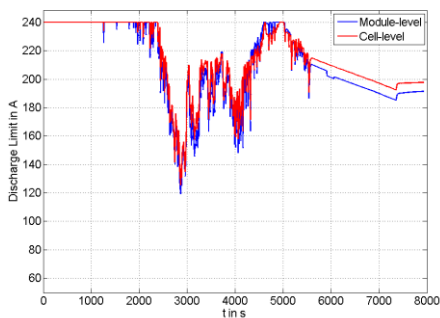


Figure 14: SoF computation at module and cell level

Figure 15 is summarizing the computing resource usage (task execution time) for the current implementation. As stated in Section 4-B, an instance of the operating system ERIKA is

running on each core. Core 0 is managing the BSW and drivers, the battery state estimation algorithm is run on Core 1, and Core 2 is reserved for EIS. It can be noticed that the operating system task is consuming slightly more than 10% core time for this configuration. The computation of the SoX function at pack level requires less than 5% core time. By improving the computation accuracy to cell level, then a computation time of 15% of the core time is required. This illustrates the computing requirement (factor 3) in comparison to computation at module level. At the same time, it illustrates that integration of other functionalities into the AURIX platform (thus reducing the number of electronic control unit) is easily possible.

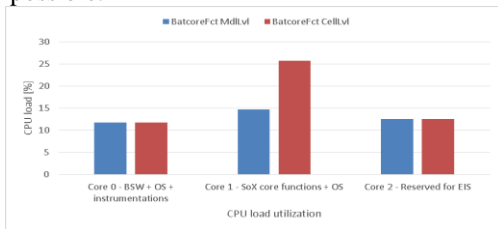


Figure 15: Computing resource usage for the three cores

It must be noted that at this stage of the project no particular approach was deployed for the systematic exploration of timing behavior and resource management. The decisions related to startup and shutdown sequences and handling, possible deadlocks, delays caused by remote blocking, memory allocation, as well as ASW functionality scheduling and partitioning were made based on expert knowledge. This step (scheduling analysis) will become essential to ensure an efficient and balanced functionality partitioning and scheduling and is part on ongoing work.

## V. CONCLUSION

Accurate understanding and modeling of the physical behaviors of the cells' chemistry is pre-requisite for proper and optimized control of the HV battery, thus moving the limits for use of each single cell (and therefore of the entire battery at the end) from a conservative boundary to a more real limit. Consequently, it can be assumed that the range of the vehicle and cycle life of the battery can be increased due to the precise estimation approach.

The continuous advances in chip design (multi-core computing platforms for automotive applications) and embedded SW engineering (AUTOSAR) are providing important basis to deploy complex and accurate control strategies. At the same time, the competences and skills gap is growing apart between the different technologies. The seamless migration of an existing control strategy to a multi-core platform, while considering functional and non-functional requirements (e.g., performances, timeliness, safety), is not an easy task. During this paper, the migration of battery estimation functions (SoX) to an AURIX platform was presented. The migration has led to more accurate battery state estimation and illustrated that the proposed CPU provides enough performances for integration of further functionalities, thus providing the potential for reduction of number of discrete electronic control units within

the vehicle. At the same time, an important lesson learnt was the need to properly analyze and manage startup and shutdown sequences, possible deadlocks, delays caused by remote blocking, memory allocation, as well as ASW functionality scheduling and partitioning. These aspects are already part of ongoing work.

## ACKNOWLEDGMENT

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 608988

## REFERENCES

- [1] G. E. Moore. Craming more components onto integrated circuits. *Electronics*, 38(8):114 – 117, April 1965.
- [2] Eric Armengaud, Georg Macher, Can Kurtulus, Riccardo Groppo, Sven Haase, Günter Hofer, Claudio Lanciotti, Alexander Otto, Holger Schmidt, Slawomir Stankiewicz, Improving HV battery efficiency by smart control systems, *Smart Systems Integration conference 2015 (SSI 2015)*
- [3] Brewerton, Schneider, & Grosshauser. (2009). Practical use of AutoSAR in Safety Critical Automotive Systems. SAE #2009-01-0748.
- [4] Schneider, Eberhard, & Brewerton. (2010). Multicore vs. Safety. SAE #2010-01-0207.
- [5] ISO 26262: "Road vehicles – Functional safety", 2011.
- [6] W. Waag, C. Fleischer and D. U. Sauer, "Critical review of the methods for monitoring of lithium-ion batteries in electric and hybrid vehicles," in *Journal of Power Sources* 258, 2014.
- [7] G. L. Plett, "Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs Part1. Background," *Journal of Power Sources*, pp. 252-261, 2004.
- [8] G. L. Plett, "Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs Part 2. Modeling and identification," *Journal of Power Sources*, pp. 262-276, 2004.
- [9] G. L. Plett, "Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs Part 3. State and parameter estimation," *Journal of Power Sources*, pp. 277-292, 2004.
- [10] G. L. Plett, "Sigma-point Kalman filtering for battery management systems of LiPB-based HEV battery packs Part 1. Introduction and state estimation," *Journal of Power Sources*, pp. 1356-1368, 2006.
- [11] G. L. Plett, "Sigma-point Kalman filtering for battery management systems of LiPB-based HEV battery packs Part 2: Simultaneous state and parameter estimation," *Journal of Power Sources*, pp. 1369-1384, 2006.
- [12] G. L. Plett, "Battery management system algorithms for HEV battery state-of-charge and state-of-health estimation," in *Advanced Materials and Methods for Lithium-Ion Batteries*, 2007.
- [13] B. Kortschak, C. Kurtulus, M. Dohr, U. Wiedemann and V. Hennige, "Detection Method of Battery Cell Degradation," in *Vehicle Power and Propulsion Conference*, Chicago, IL, 2011.
- [14] Hermann Kopetz, "Real-Time Systems: Design Principles for Distributed Embedded Applications". Kluwer Academic Publishers., Apr 2011, ISBN 978-1441982360
- [15] G. Macher, E. Armengaud, and C. Kreiner. Integration of Heterogeneous Tools to a Seamless Automotive Toolchain. In R. O'Connor and R. Messnarz, editors, *EuroSPI 2015*, 2015.
- [16] G. Macher, M. Atas, E. Armengaud, and C. Kreiner. A Model- Based Configuration Approach for Automotive Real-Time Operating Systems. *SAE International Journal on Passenger Cars – Electronics and Electrical Systems*, 8(2), 2015.
- [17] G. Macher, R. Obendrauf, E. Armengaud, and C. Kreiner. Automated Generation of Basic Software Configuration of Embedded Systems. In *ACM RACS Conference Proceedings*, 2015.