Improving Battery Management System Performance and Cost with Altera FPGAs

by Mark Bingeman, Principal Design Engineer, Nuvation and Ben Jeppesen, Manager, Industrial, Automotive and Broadcast System Solution Engineering, Altera Europe

WP-01247-1.0 White Paper

The purpose of this white paper is to evaluate improvements to Battery Management System (BMS) performance and cost with Altera® FPGAs. In many high-voltage battery systems, including electric vehicles, grid attached storage and industrial applications, the battery is a significant portion of the system cost, and needs to be carefully managed by a BMS to maximize battery life and to optimize charging and discharging performance. This white paper presents the BMS functional requirements for these applications and outlines existing BMS architectures. Key BMS architectural challenges are discussed and opportunities for Altera devices are identified. For each of these opportunities, the performance and cost of the existing solution are compared with Altera FPGA solutions. Altera devices provide architectural flexibility, scalability, customization, performance improvements, and system cost savings in BMS applications.

Introduction

The rechargeable battery industry is experiencing significant growth and is projected to continue to expand into the future. This growth is driven by an increase in portable battery powered devices, electric vehicles, energy storage, and industrial applications. These applications use various battery chemistries including lead-acid, nickel-cadmium, nickel-metal-hydride, lithium ion, and other chemistries currently in development. All of these applications have need of a BMS. The main objective of a BMS is to maintain the health of all the cells in the battery pack to deliver the power needed by the application, while protecting the cells from damage and maintaining all the cells within the manufacturer-recommended operating conditions in order to prolong the life of the battery pack.

BMS Functional Requirements

While each application and battery chemistry has specific requirements, most applications have the following common set of BMS functional requirements:

- *Cell Protection*—Safety is the primary requirement of a BMS. This includes protecting the cells from operating outside of manufacturer-recommended conditions, as well as protecting the rest of the system from the battery in the event of cell failures.

101 Innovation Drive
San Jose, CA 95134
www.altera.com

Cell protection encompasses the following BMS elements:

- *Data acquisition* of cell voltages, temperatures, and current
- *Data analysis* to determine state of charge and state of health of the battery pack
- *Communication* of warnings and error conditions detected in the battery pack
- *Control* of external components to maintain cells within manufacturer-recommended conditions (e.g. fans, heaters), control of charger/inverter to delivery energy to the pack during charging or to the load during discharging, and control of components to isolate the battery pack in the event of a cell failure (contactors)

- *Charge and discharge control*
  - Inappropriate charging is the leading source of battery damage
  - The BMS must manage the rate of charging to maintain cells within manufacturer recommendations. For multi-cell battery chains, this includes cell balancing.
    - *Cell balancing* provides a way to compensate for weaker cells by equalizing the charge on all cells in the battery pack to extend battery life. Without cell balancing, weaker cells can be over-stressed on each charge/discharge cycle causing premature failure of the battery.
  - The BMS needs to determine the *state of charge* (SOC) of the battery to allow it to control the rate of charging and discharging.
- *SOC and state of health*—Determination of the *SOC and state of health* (SOH) of the battery and/or the individual cells in the battery pack
  - The BMS needs to determine the SOC in order to report the capacity left in the battery – typically called the 'gas gauge'. SOC is also needed to control battery charging and discharging
  - SOC determination can be obtained using the following methods:
    - *Measure cell voltage to calculate SOC*. This can work well for battery chemistries such as Lead-Acid, for which voltage and capacity are fairly linearly related, but ineffective for other battery chemistries, such as Lithium-Ion, for which voltage is mostly flat over the battery capacity, except at the extremes, where it is very non-linear.
    - *Coulomb counting*. Integral over time of the current being delivered (or received) by the battery.
  - The available capacity of a battery depends on cell chemistry, charging times and discharge rates, as well as cell age and temperature. All of these factors need to be considered in the calculation of the SOC.
  - SOH is a figure of merit that represents the health of the battery as compared with a new battery. It includes factors such as voltage, charge capacitance, internal resistance, and self-discharge rate.

There are no standards for calculating SOH. Measurements and calculations to determine SOH are battery chemistry and application specific.

- *History*
  - The BMS needs to keep track of the age of the battery (the number of charge/discharge cycles), and must log when batteries are out of range of manufacturer-recommended operating conditions.
    - Minimum/maximum voltage, temperature, and maximum charge and discharge currents are important parameters to track
    - History can include cell serial numbers and manufacturing dates for traceability
- *Communication*
  - Most applications require the BMS to communicate with external components including a power inverter/charger, environmental control units (e.g. fans, heaters), and system controllers.
  - Communication interfaces provides user access to BMS state and diagnostics information, as well as supports modifications of control parameters

BMS Architectures

There are a number of different BMS architectures depending on the number of cells in the system and the needs of the application. These architectures can be divided into three main categories.

Centralized BMS Architecture

In the centralized BMS architecture, a single controller board is responsible for monitoring and controlling all the cells in the battery pack. Voltage levels for all cells in the battery along with temperature sensors distributed throughout the battery are all routed to the controller board, usually via a large number of wiring harnesses. The centralized BMS architecture is typically the most economical, but does not scale well, and requires extensive wiring from all the cells to the controller board. The centralized BMS architecture is well suited to battery packs with a small number of cells.

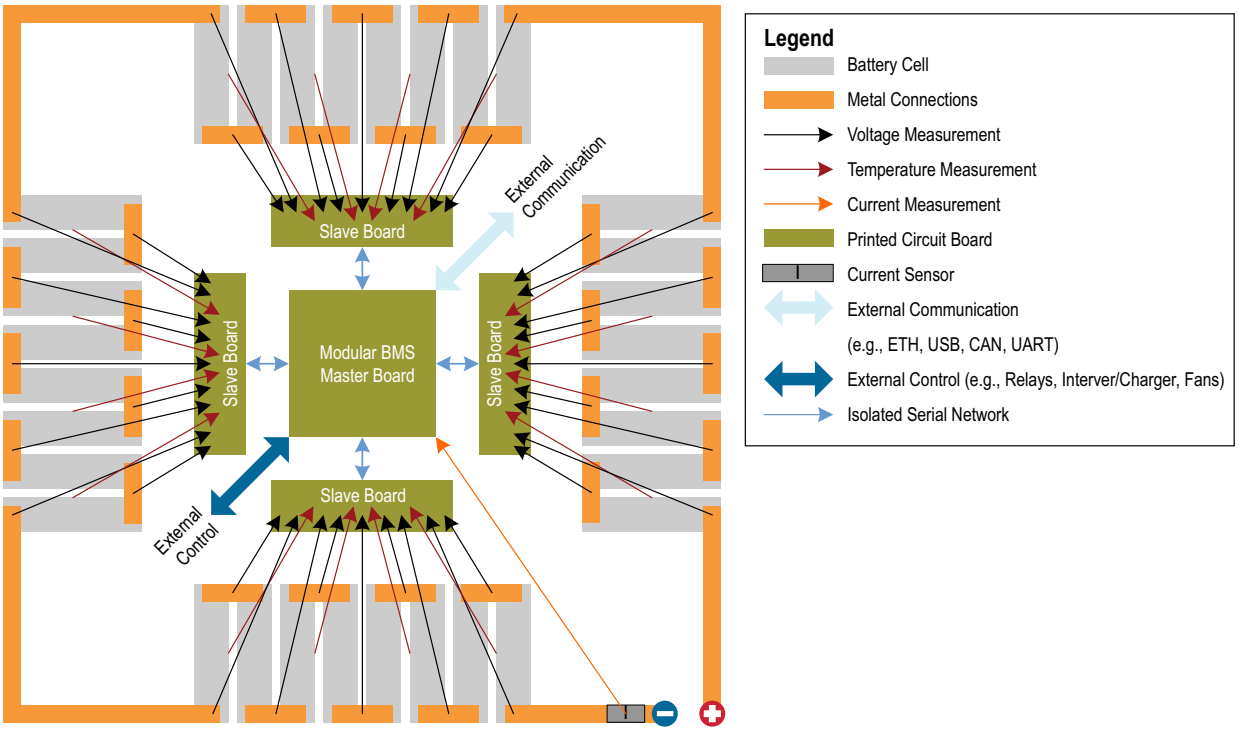
Distributed BMS Architecture

In the Distributed BMS architecture, each cell in the battery has a slave board that measures and controls an individual cell. Slave boards are interconnected via a serial network and controlled by a master control board. A distributed BMS architecture is typically much easier to install than a centralized one. Each cell only has a pair of voltage signals and a temperature sensor to connect, and serial interface cables in and out. For battery packs with a large number of cells, the cost of a distributed system can be significantly higher than a centralized system.

Modular BMS Architecture

In the modular BMS architecture, each slave board monitors and controls a group of cells. A master control board interfaces with the slave boards via serial interfaces to control the functionality of the system. The modular BMS architecture provides a trade-off between the benefits and short-comings of centralized and distributed architectures. Figure 1 shows the topology of the modular BMS architecture.

Figure 1. Master-Slave BMS Architecture

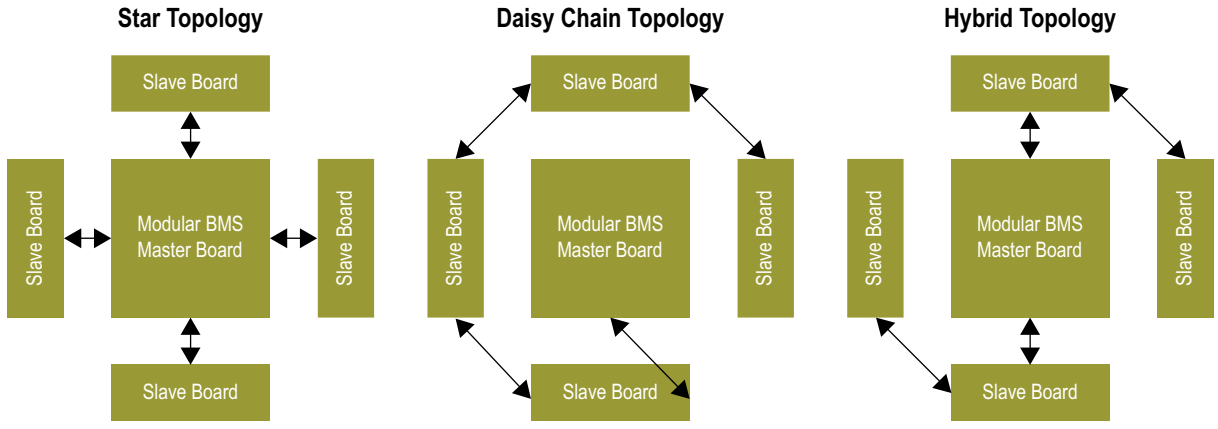


In addition to the master-slave modular BMS architecture, there is also a peer-to-peer modular BMS architecture. In the Peer-to-Peer modular BMS architecture, there is no master board, and each peer board monitors and controls its own set of battery cells and communicates with other peer boards in the system. This architecture divides the software tasks amongst the peer boards, but does require additional software for communication and coordination among peers.

In addition to master-slave or peer-to-peer architectures, there are a number of serial network topologies that fall under the modular architecture category.

Figure 2 shows three different serial interface networks that may be used to interconnect the master control board and the slave boards.

Figure 2. Modular BMS Network Topologies

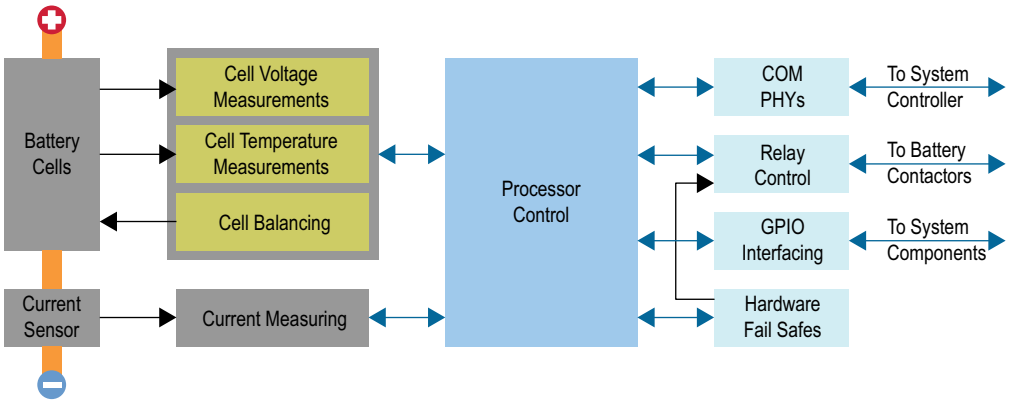


The serial networks can be a star topology or a daisy chain topology or a hybrid of both network types, in which the master control board supports a number of serial daisy chain networks. The topology of the serial networks in the BMS depends on the application. Daisy chain networks provide scalability, but also increase the required serial network bandwidth for systems with a high number of slave boards.

BMS Functional Block Diagram

Figure 3 shows the functional block diagram of a typical BMS.

Figure 3. BMS Functional Block Diagram



On the left side of Figure 3 are the battery cells, current sensor and data acquisition portion of the BMS. The cell voltage and temperature measurements, and cell balancing are shown with a dotted box around these functions because there are various multicell battery monitor ICs that perform these functions with the addition of some external components. Similarly, there are ICs available that perform current measurement and coulomb counting.

On the right side of Figure 3 are the communication and control components of the BMS. Typically, a BMS communicates with a system controller, a power inverter/charger, and other components in a larger system. Communication interfaces, such as Ethernet, USB, CAN, and UART are typically used for these purposes.

- A BMS often includes external relays/contactors to connect the battery pack to the external load or to the charger. Relays or contactors are also used to disconnect the battery pack from the rest of the system in the case of cell failures or other safety issues.
- A BMS can control other system components, such as fans, heaters, alarms, and so on via user controllable general purpose input/output (GPIO) signals.
- A BMS also requires hardware fail-safe circuitry to protect the hardware and the battery in the case of a failure of the software components, or hardware components internal or external to the BMS.

The processor or control functional block is responsible for communicating with and controlling the data acquisition functional blocks and the external communication or control functional blocks.

The functional blocks in Figure 3 need to be implemented in each of the previously discussed BMS architectures. In the case of a centralized BMS architecture, all of the functional blocks are implemented on a single master control board. For the other BMS architectures, the functionality is divided among the various boards in the BMS. The division of this functionality and the communication interfaces between functional blocks for each of the BMS architectures provide opportunities for Altera's programmable devices.

BMS Design Challenges

This section describes common challenges encountered when designing a BMS.

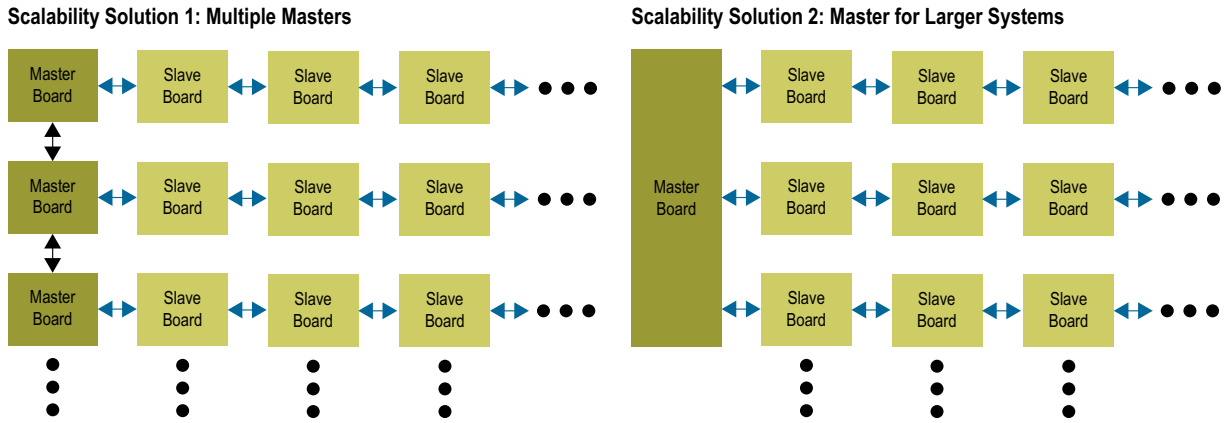
Scalability

For markets such as energy storage and industrial applications, the number and types of battery cells may be significantly different from application to application. BMS products targeting these markets require an architecture that scales well with the number of battery cells in the system. There are three main areas to consider for scalability.

- *The number and types of boards, and interconnects between boards.* The distributed and modular architectures are well suited to scaling since smaller systems require only a few slave boards, and larger systems are supported with additional slave boards.
- *The serial networks interconnecting the slave boards.* Each serial network can only support a certain number of cells, due to signal integrity and bandwidth limitations. At the serial network bandwidth limit, additional cells can only be supported by decreasing the rate at which cell voltages and temperatures are received and control settings are calculated and transmitted. If this rate becomes too slow, the system can become difficult to control. Typically, this issue is addressed by adding more serial interfaces to the master control board. For very large BMS configurations, this can require 4, 8 or even more serial networks. This number of serial masters is not typically found in midrange microprocessor ICs.
- *The amount of software processing.* As the number of cells in the system increases, so does the amount of processing. For BMS architectures with a single main processing unit, the processor must be selected to support the largest system, or the sophistication of the data processing algorithms needs to be dialed back in larger systems to remain within the capabilities of the processor. In a peer-to-peer BMS architecture, the processing can be shared amongst the peers. However, even in this architecture, there is additional processing overhead for coordination between the peers that increases with each additional peer added to the system.

Figure 4 shows two possible solutions for scalability in the master-slave BMS architecture.

Figure 4. BMS Scalability



When the number of slave boards required is greater than the capacity of the serial interface and/or the processing capacity of one master board, then in Solution 1, additional master boards can be added to support additional slave board networks. This requires communication and coordination between master boards. Typically, one master board will be the overall system master, and the remaining master boards slave to the overall system master. This solution scales well, but does require additional processing overhead for master board to master board communication, and can result in unused functionally on the additional master boards.

In Solution 2, a custom master board is used to support larger systems. This master board requires additional serial interfaces and more processing capacity. FPGA devices have a distinct advantage in this application, since they can easily support several serial interfaces and can instantiate more soft-core processors as needed to support the processing needs of the system. A common master board can scale with the needs of the system with FPGA device size migration in the same physical footprint. A common master board platform reduces design and development effort and provides a more cost-effective solution as compared with adding multiple master boards.

Isolation

In many BMS applications, the number of cells connected in series can be quite high, resulting in significant differences in voltage among components in the system. These large voltage differences necessitate component-to-component isolation. In particular, the serial communication links between boards need to be isolated, typically via transformer coupling circuits. Even within each board, there are portions of the circuitry requiring isolation, such as current measuring, relay/contactor drivers, external communication and control interfaces, hardware failsafe circuitry, and power supplies. The division of functionality due to isolation requirements provides opportunities for low-cost FPGA and CPLD devices to replace a number of function-specific ICs with a single programmable device.

Safety

Safety is the primary requirement of a BMS system, especially in automotive and industrial applications where the battery capacity and the transfer of energy into and out of the battery can be very significant and the protection of users is paramount. The BMS needs to be designed and validated with full knowledge of the applicable safety standards for each application. Altera was the first FPGA supplier to achieve IEC 61508 functional safety certification for the industrial market and is in a leading position towards ISO 26262 functional safety certification for automotive applications. The Cyclone® V SoC safety manual and device level FMEDA are available now, and the full qualification to ISO 26262 is expected by mid-2015. The MAX® 10 FPGA will also be supported in a future release of the functional safety pack, TUV Certified to IEC 61508 and ISO 26262. Altera is an active member of the ISO 26262 working group and co-chairs the ISO 19451 PLD sub-group, which helps to drive PLD requirements for the 2nd edition of the ISO 26262 standard. Altera's functional safety approach is a holistic one, which provides guidance in methodology, backed up with qualified tools and safety- ready suite of diagnostics intellectual property (IP) and expert technical support along with certified PLDs. Altera's functional safety documentation, reliability reports and TUV certificate will assist in system certification.

Altera Solutions for BMS

Altera programmable devices provide the following benefits for BMS applications:

For electric vehicles

- *Safety*—Hardware fail-safe logic, dual image configuration (redundant boot image) and ISO26262 support
- *Reliability*—Serial protocol customizations for increased robustness and reduced latency
- *Energy efficiency*—Hardware acceleration of digital filtering for better SOC/SOH estimations to improve battery performance and to off-load the processor
- *Flexibility*—Programmable I/O provides more flexibility in system architecture design

For grid attached storage and industrial applications

- *Reduced system cost and power*—FPGA device migration enables a common hardware platform to scale according to the processing and serial interface requirements of each system.
- *Safety*—Hardware fail-safe logic, dual image configuration (redundant boot image) and IEC61508 support
- *Energy efficiency*—Hardware acceleration of digital filtering for better SOC/SOH calculations to improve battery performance and to off-load the processor
- *Flexibility*—Programmable I/O provides more flexibility in system architecture design. Can support many communication interfaces and various communication protocols in a single device.

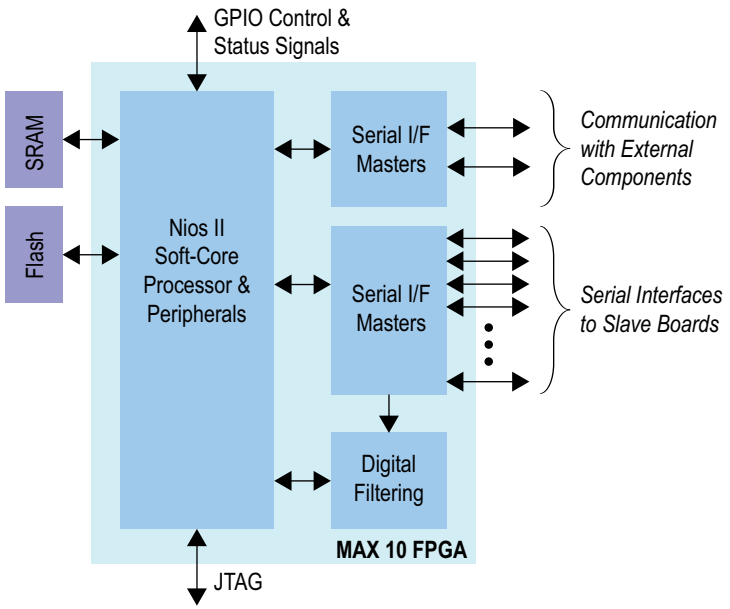
Three Altera solutions for BMS are examined in this white paper:

- The MAX 10 FPGA with Nios® II soft-core processor and digital filtering for SOC/SOH calculations provides a flexible and cost effective solution to address the serial I/O and processing needs for scalable systems.
- The Cyclone V FPGA with hard processing system provides a high-performance, peripheral-rich single device solution for higher system integration, scalability, and more advanced monitoring and control algorithms.
- The MAX V CPLD proves a low cost, small footprint, lower power solution for slave boards in the distributed BMS architecture.

MAX 10 FPGA-Based Master Board

Figure 5 shows the block diagram of the MAX 10 master board FPGA design.

Figure 5. MAX 10 Master Board FPGA Design Block Diagram



The design is centered on the Nios II processor and associated peripherals to support an external flash memory and SRAM for storing and executing software. Serial master blocks are used to communicate with external components as well as with the slave board serial networks. The number of serial masters can scale according to the needs of the system. The Digital Filtering block performs matrix and vector multiplications for Kalman filtering to determine SOC/SOH of the battery cells. The digital filtering block can receive data directly from the slave board serial networks or from the Nios II processor.

Table 1 lists the FPGA resource estimate for this design targeting a 10M08DF device.

Table 1. MAX 10 Master Board FPGA Design Resource Usage Estimate (10M08DF)

Block	Logic (LEs)	Memory Blocks	DSP Blocks
Nios and Peripherals	4,700	17	3
Serial I/F Masters	1,000	10	0
Digital Filtering	800	4	20
Total	6,500	31	23
Available	8,064	42	24
% Usage	80%	74%	96%

Nios II /f variant with two PIOs, two timers, one dual boot master, one tri-state bridge (flash memory, SRAM), 4 kilobytes (KB) instruction cache, 2 KB data cache, 4 KB of on-board memory. Ten serial masters (serial peripheral interface (SPI), I^2C or UART), eight for slave board networks and two for external interfaces.

The design requires approximately 80% of the logic, 74% of the embedded memory blocks, and 96% of the digital signal processing (DSP) blocks. DSP usage is driven by the complexity of the Kalman filtering calculations. MAX 10 FPGA family supports 16 - 144 DSP blocks depending on device size. Each DSP block supports an 18 x 18 bit multiplication. Applications requiring more multipliers will require a different device family, such as the Cyclone V FPGAs.

Table 2 summarizes the advantages of the MAX 10 FPGA master board solution.

Table 2. MAX 10 FPGA-Based Master Board Advantages

Metric	Typical Cortex-M4 Class Microprocessor	MAX 10 FPGA With Nios II
Clock frequency (MHz)	80-125	80-135
MIPS/MHz	1.25	1.13
MIPS performance	100-156	90-153
Number of processors	1	1 or more
Custom instructions	0	Up to 256
Power (mW)	150-200	170-240
Serial peripherals	2-4 (typical)	10+
Dedicated multipliers	0	16-144

The Altera MAX 10 FPGA with Nios II soft-core processor provides an alternative solution to the ARM® Cortex®-M4-based microprocessors typically used in BMS applications. The MAX 10 FPGA solution provides comparable processor performance and the flexibility to add custom instructions and hardware accelerators, or to instantiate multiple soft-core processors to meet the processing needs of the system. The MAX 10 FPGA solution also provides serial peripheral scalability for large systems, and dedicated multipliers for digital filtering for SOC/SOH calculations. The MAX 10 FPGA solution is more cost effective and requires less power in large scale battery packs that are beyond the serial interfacing capabilities and/or processing capacity of a single microprocessor-based solution. Processor comparison and digital filtering are examined in more detail in the following sections.

Microprocessor Replacement

In a modular, master-slave BMS architecture, the processor on the master board is responsible for managing all the serial interfaces to the slave boards to obtain voltage and temperature measurements, controlling cell balancing, calculating SOC/SOH, managing the battery pack, and interfacing with external devices. The ARM Cortex-M4 low-power RISC processor core is intended for incorporation into microcontroller systems that include a moderate mix of external interfaces, making it a good match with BMS requirements. This core is available in many microcontroller products from a number of vendors, and for BMS applications, typically operates at 80 - 125 MHz with a performance of 1.25 MIPS/MHz (Dhrystones 2.1 benchmark). There are some ARM Cortex-M4 based microprocessors that operate at higher frequencies (200 MHz+), but these devices are at a higher price point and have a richer mix of peripherals than is commonly used in BMS applications. In comparison, the Altera Nios II soft-core processor in a MAX 10 FPGA can operate at 80 - 135 MHz with a performance of 1.13 MIPS/MHz. Therefore, the Nios II processor is comparable to the ARM Cortex-M4 based microprocessors typically used in BMS applications in both clock frequency and performance. The Nios II processor has the distinct advantage of being able to use the FPGA fabric to implement custom instructions or hardware accelerators to increase the effective performance of the processor significantly higher than the values shown in Table 2. Multiple Nios II processors can also be instantiated in a single FPGA device to support the processing needs of the system.

For power consumption it is important to make the comparison at the device level, taking into account the processor core and the peripherals. The microprocessors presented above have power consumption in the range of 150 - 200 mW. In comparison, the MAX 10 FPGA (10M08DF) power consumption for the equivalent MIPS performance is 170 - 240 mW. This estimate is based on Altera's PowerPlay Early Power Estimator with device utilization at 75% for all resource types (logic, memory, and multipliers). The Nios II /f variant with peripherals requires approximately 60% of the logic, 40% of the memory, and 15% of the multipliers in the MAX 10 FPGA. This leaves room to add additional peripherals or hardware accelerators as needed.

Therefore, at a device level, the MAX 10 FPGA power dissipation is comparable to the microprocessor. In the scalability scenario in Figure 4, where the microprocessor does not have sufficient serial interfaces, one MAX 10 FPGA has lower overall system power consumption as compared with two or more master boards.

A direct cost comparison of the ARM Cortex-M4 based microprocessors and the Altera 10M08DF device is difficult due to the various mixes of peripherals in the microprocessor devices. Typically, ARM Cortex-M4 based microprocessors are more cost effective than a comparable MAX 10 device. However, at the system level, when the number of serial networks in the BMS exceed the capabilities of the microprocessor devices or the SOC/SOH calculations exceed the capacity of the microprocessor, the MAX 10 FPGA solution is more cost effective than a higher performance microprocessor or a multi-microprocessor, or multi-master board solution. This scenario is most common in grid attached storage and industrial applications where the number of cells in the system is very high. Typical master board cost savings and power savings is approximately one half, using a single FPGA-based master board, where two (or more) microprocessor-based master boards would otherwise be required.

Math Processor

The most computationally intense operation in a BMS is the calculation of the battery pack SOC and SOH. These calculations require numerous data inputs including cell voltages, temperatures, rate of charge and discharge, number of charge/discharge cycles, age of cells, and so on. SOC and SOH algorithms are an on-going subject of research at universities and in industry. Common methods for calculating SOC and SOH use Kalman filtering. Kalman filters calculate an estimated state of a system and the variance (or uncertainty) of the estimate. In a two-step process, the Kalman filter calculates a prediction of the next state of the system, then uses external measurements to update (or correct) the prediction and the variance of the estimate. The predictor-corrector equations are updated at each time step. The Kalman filter calculations require large matrix and vector multiplications that can be a significant drain on the processing capacity of the main processor in the BMS.

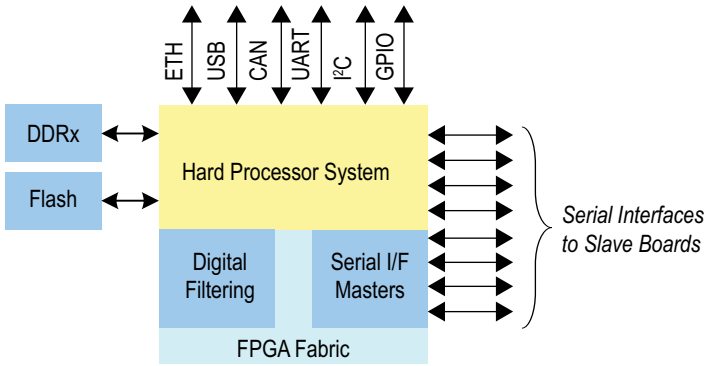
In a Kalman filter, the state of the system is expressed as a column vector with one entry for each state controlled by the Kalman filter. Similarly, the filter uses a column vector of system control inputs, and a column vector of system measurements. Matrices are used to represent the state estimate variance, state transition model, control input model and state observation model. In a BMS, the size of the vectors and matrices in the Kalman filter can be quite large and require a significant number of calculations for the main board processor. FPGA devices have dedicated hardware for multipliers that can be used for matrix and vector multiplication to reduce the load on the processor. FPGA device fabric can also be used to augment the dedicated hardware multiplier blocks.

Master Board Cyclone V with Hard Processor System

In very large BMS applications, there is a need for an integrated single-chip solution with Ethernet, USB, and CAN interfaces, additional processing capability for complex SOC/SOH calculations, and the flexibility of FPGA fabric for serial interface scalability and hardware accelerator blocks. The Altera Cyclone V FPGA with hard processor system can meet these requirements.

The block diagram of the FPGA design is shown in Figure 6.

Figure 6. Cyclone V FPGA with Hard Processor System Block Diagram



The Cyclone V FPGA is divided into two portions: the hard processor system and the FPGA fabric. The hard processor system includes one or two ARM Cortex-A9 processors as well as hard-blocks for interfacing with flash memory and SDRAM and peripheral blocks for interfacing with Ethernet, USB, CAN, UART, I2C, SPI, and GPIO interfaces. The FPGA fabric is used to add additional soft-core serial I/F masters for scalability. The digital filtering block takes advantage of the DSP blocks in the FPGA fabric for hardware acceleration of SOC/SOH calculations. Altera has a reference design that uses both the ARM Cortex-A9 processor and the FPGA fabric in the Cyclone V FPGA to implement an extended Kalman filter.

The rich mix of peripherals in the Cyclone V hard processor system opens up opportunities for remote monitoring and control of the BMS over the Internet, communication with other peripherals or hosts over the USB interfaces, and communication with other industrial devices over the CAN interfaces.

Admittedly, the Cyclone V FPGA is a higher performance solution than the typical BMS processor solution. However, in large systems, where the alternative is two or more master boards to support the number of serial networks, and the computational requirements of SOC/SOH calculations, the Cyclone V FPGA becomes a more compelling single chip solution. The Cyclone V FPGA solution is especially appealing in large grid attached storage applications.

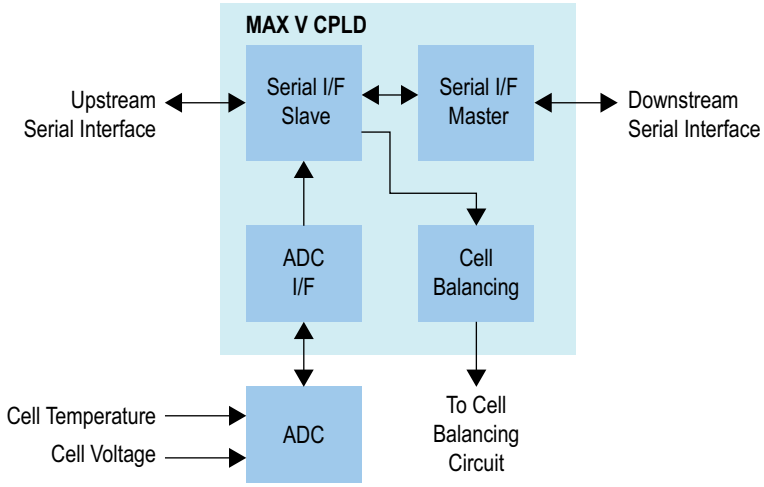
MAX V CPLD Slave Board – Distributed BMS Architecture

In distributed BMS architectures, each slave board performs cell voltage and temperature measurements, cell balancing, and interfacing with other slave boards and the master board, typically via daisy-chained serial networks. Very small board size, low power, and low cost are essential for the slave boards, since there is one board for each cell in the battery pack. The Altera MAX V CPLD is well suited for this application.

The MAX V CPLD is a low-cost, low-power, non-volatile CPLD requiring a single external voltage supply, and is available in package sizes as small as 4.5 x 4.5 mm.

Figure 7 shows the block diagram of the MAX V CPLD design

Figure 7. MAX V CPLD Slave Board Distributed BMS Architecture (CPLD Block Diagram)



The serial I/F slave and serial I/F master blocks interface with the daisy-chained serial network. The analog-to-digital (ADC) I/F block controls an external ADC to measure cell voltage and temperature. The cell balancing block receives instructions from the master board processor to control external cell balancing circuitry.

ADCs for cell voltage monitoring must support differential inputs with an input range of up to 5 V (for lithium-ion cells). 16 bit delta-sigma ADCs with frequency programmable noise filters are typically used in this application. These features are not usually found in ADCs included in some microprocessor products. Therefore a separate ADC is required.

Table 3 shows the CPLD resource estimate for this design.

Table 3. Slave Board, Distributed BMS Architecture, Resource Usage Estimate *(1)*

Block	Logic (LEs)	I/O Pins
Serial I/F Slave	50	4
Serial I/F Masters	50	4
ADC I/F	50	4
Cell Balancing	25	4
Total	175	16
Available	240	52
% Usage	73%	31%

Note:

(1) Assumes SPI serial interfaces and serial interface with ADC

The design requires approximately 73% of the logic and 31% of the external pins of a 5M240Z device in the 68-pin MBGA package. The MAX V devices also have 8,192 bits of user flash which can be used for storing information about the battery cell such as serial number and manufacturing date.

In comparison, this functionality is typically implemented using an 8 bit MCU, 28-pin QFN package (6 mm x 6 mm), with 850 µW maximum power. The MAX V CPLD 5M240Z in the 68 pin MBGA package is 5 mm x 5 mm and consumes 820 µW maximum power. The MAX V CPLD solution is a smaller area and lower power solution than the 8 bit MCU.

BMS Serial Networks

In both distributed and modular BMS architectures, the serial interfaces between boards are critical for system performance. These serial interfaces must be highly reliable communication channels that can operate even in very electromagnetically noisy environments and in some applications, over long distances and/or wide temperature ranges. The selection of components for the physical level of these serial interfaces and the selection of communication protocol are essential to obtain highly reliable and robust communication between boards.

In larger daisy chain networks, the latency introduced by each slave board can have a significant impact on the network capacity. A low-latency serial network can increase the number of slave boards that a serial network can support, or conversely, can reduce the operating frequency of the serial network, thus increasing network reliability. In a microprocessor-based solution, the processor must control the upstream and downstream serial peripherals separately, and transfer data between

serial peripherals. This can add significant delay between the upstream and downstream networks. By comparison, programmable logic devices can implement the serial peripherals in logic fabric and transfer data between networks with minimal delay, possibly as low as a few clock cycles. A programmable logic implementation can increase the number of slaves on a serial network for a given bandwidth and system scan period by two times or more, as compared to an implementation in which both upstream and downstream channels are managed by a single microprocessor.

Serial network performance and robustness can also be improved by adding error checking and correction to the communication protocol. With a minimal increase in data payload size, the performance and robustness of the serial interfaces can be significantly improved. CPLDs and FPGAs can implement error checking and correction to the communication protocol in the device fabric with minimal latency. In comparison, microprocessor serial peripherals typically do not support error checking and correction. Adding this functionally to the protocol in software significantly increases the serial interface latency. Programmable logic devices are well suited to implement low latency, serial protocol customization for improved network performance and robustness.

Conclusion

This white paper has presented the primary functional requirements of a BMS and typical BMS architectures. BMS design challenges were discussed and opportunities for Altera programmable devices identified. Three specific opportunities were explored and performance and cost compared with existing solutions. In particular the MAX 10 FPGA family is well suited for BMS and system monitoring applications.

For further assistance selecting an Altera programmable device for BMS or other system monitoring applications refer to the “Further Information” section.

Further Information

- Altera MAX 10 FPGAs
  www.altera.com/devices/fpga/max-10/max-10-index.html
- Nios II Processor
  www.altera.com/devices/processor/nios2/ni2-index.html
- Altera Cyclone V SoC
  www.altera.com/devices/fpga/cyclone-v-fpgas/cyv-index.jsp
- Altera MAX V CPLDs
  www.altera.com/devices/cpld/max-v/mxv-index.jsp
- Altera Automotive Safety
  www.altera.com/end-markets/auto/safety/aut-safety.html
- White Paper: *Advanced System Management with Analog Non-Volatile FPGAs*
  www.altera.com/literature/wp/wp-01230-advanced-system-management-with-analog-non-volatile-fpgas.pdf
- White Paper: *Extended Kalman Filter*
  www.altera.com/literature/ds/Extended_Kalman_Filter.pdf
- White Paper: *Five Ways to Build Flexibility into Industrial Applications with FPGAs*
  www.altera.com/literature/wp/wp-01154-flexible-industrial.pdf

- White Paper: *Architecture Matters: Choosing the Right SoC FPGA for Your Application* www.altera.com/literature/wp/wp-01202-embedded-system-soc-design-considerations.pdf
- Nuvation Battery Management Systems www.nuvation.com/battery-management-system
- *Electropaedia – Battery and Energy Technologies*, Barrie Lawson www.mpoweruk.com

Acknowledgements

- Mark Bingeman, Principal Design Engineer, Nuvation (www.nuvation.com)
- Ben Jeppesen, Manager, Industrial, Automotive and Broadcast System Solution Engineering, Altera Europe

Document Revision History

Table 1 shows the revision history for this document.

Table 4. Document Revision History

Date	Version	Changes
January 2015	1.0	Initial release.