Unification in the Battery Interface: the New MIPI® Alliance Specification

By:
Markus Littow, Member of Technical Staff – Analog/Mixed Signal Architecture, ST-Ericsson
Christopher Chun , Senior Staff Engineer - QCT Architecture, Qualcomm Incorporated
Stephan Schaecher, Project Leader -Battery Authentication & Management Infineon Technologies AG

Abstract

MIPI Battery Interface (BIF) is the first comprehensive battery communication interface standard for mobile devices. BIF is a robust, scalable and cost-effective single-wire communication interface between the mobile terminal and smart or low cost batteries. BIF improves mobile terminal safety and performance by providing comprehensive battery monitoring data and functions in a structured, software-friendly manner. The BIF specification is designed to replace existing proprietary battery interface solutions for mobile devices.

The Need for Unification in Mobile Battery Interfaces

For a decade, the mobile device industry has blossomed with approximately 1.5 billion mobile devices and at least 1.5 billion batteries per year using many different, non-standardized battery interfaces.

The lack of a commonly accepted battery interface has caused extra work and logistical effort throughout the industry. Mobile device manufacturers must coordinate, specify and maintain their proprietary solution for different parties in the ecosystem - themselves, mobile chipset suppliers, battery slave IC device suppliers and battery pack manufacturers.

Mobile chipset suppliers need to support several battery interfaces for different mobile device manufacturers. Each supported battery interface means more silicon area and effort in the chipset design. These interfaces may be useful to one of the customers but useless for the others. Unfortunately, each customer still pays for the useless silicon.

Battery slave IC device suppliers confront similar issues as the chipset suppliers. Additionally, the battery pack IC has more cost pressure. They may have only one customer for their chip with a customer-specific interface.

Battery pack manufacturers have to support tens of physically and electrically unique battery models for each mobile device manufacturer. The method by which the battery parameters and other required information is stored in the battery pack varies from mobile device manufacturer to manufacturer. Introducing new battery chemistries requires many interventions between battery pack manufactures and all other parties in the ecosystem on the way to the consumer’s hands. Orchestrating all of the above smoothly and efficiently requires numerous resources on many fronts. Having multiple proprietary competing battery interface solutions in the market means lower volumes for each one, and consequently higher cost especially for smart batteries. This may be the biggest single reason why the smart battery adoption rate in the mobile device industry is still relatively low, despite the benefits of smart battery technology to the mobile device manufacturers and the actual end users.

Smart Battery Benefits

Smart battery technology supports sustainable development and improves end user experience. A smart battery is a greener choice. It is designed to have a longer lifetime due to more accurate parameters, state monitoring and optimized charging events. It also enables faster adoption of new, more efficient battery chemistries available in the market. A smart battery saves overall power consumption in the mobile device by autonomously taking care of certain battery maintenance routines.

Smart battery technology improves the safety of the end users significantly by providing access to authentication functions and more accurate battery operating condition functions such as like the temperature measurement function. Battery authentication improves end user safety by eliminating the use of potentially dangerous counterfeit batteries. Battery pack temperature monitoring is vital and is required by many battery safety standards.

The new BIF interface offers a unified and unique solution for the above challenges of the mobile device industry community while keeping an eye on cost efficiency. It is designed to support both smart batteries and low cost batteries. Continuously increasing complexity in battery technology will require an expanded intelligence level in order to manage and operate the battery pack safely and in a more optimized manner. The BIF interface has been developed to meet these challenges.

The specification has been developed from the beginning to the needs of all the stakeholders in the mobile device ecosystem: consumers, carriers, mobile device suppliers, chipset platform providers, slave device providers and battery pack suppliers.

Battery Interface Specification Requirements

The MIPI BIF Working Group (WG) spent significant time to understand the fundamental technical and market requirements. The BIF standard specifies only the electrical interface requirements, leaving any physical or form factor requirements outside of the scope.

In general, a successful interface specification must offer benefits for all primary users at a minimum. For the BIF specification, the primary users are mobile device suppliers, chipset platform providers, slave device providers and battery pack suppliers.

The main required features for the BIF interface are smart and low cost battery pack support, including a fast battery pack presence detection function with a single wire interface at low cost. At the moment of writing this article, there are no other battery interface standards that can fulfill all these requirements.

The main outcomes of low cost requirement are that the pin count of the interface has to be minimized. The mechanical pins of a typical battery pack have quite high reliability requirements due to harsh operating environment, and take valuable space in the mobile device printed computer board (PCB). These pin s are relatively high cost component in a mobile device. Therefore, the BIF must work over single wire interface.

Low cost battery support is required for legacy designs, and for the simplest mobile devices. Most of the low cost batteries currently in the market have a built-in pull-down resistor in the battery pack. The measured value of the pull-down resistor usually represents certain capacity and chemistry of the battery pack. For this function, the BIF standard supports measurement of a pull-down resistor value with certain range and accuracy.

A smart battery may include multiple slave devices within the battery pack. The mobile device PCB may contain multiple slave devices, in addition to the master device. Thus, the smart battery interface needs to support a single master, multi-slave configuration. Communication speed of the interface must be easily scalable to match various available clock sources in the mobile device system under different operating conditions or data speed requirements.

Fast battery pack presence detection was required to inform system immediately if the battery pack is removed. The system software (SW) can still do the critical shutdown actions, if the physical battery interface connector is designed so that the battery communication connector pin is always the first one to disconnect. By that method, the system will still get power from the battery for a short time when the battery pack removal is noticed.

For the slave device, a cost efficient implementation was required to enable the slave device to work with an inexpensive and possibly inaccurate clock source, for example an untrimmed ring oscillator. This dictated some requirements to the protocol given that the protocol engine itself must allow a low cost implementation. The actual functions and data structures that will be built inside the slave were required to be scalable so that they don't need to waste memory bytes either in their minimum or maximum configuration.

In order to make the new interface standard even more competitive, the BIF interface was also required to support the following features:

- **Low power and low voltage signaling.** The Interface standard must enable very low power implementation for all functions and features, and enable interface implementation in the latest semiconductor processes.
- **Compliancy.** The BIF interface was also defined to be compliant with other required battery standards for mobile devices. This includes enabling certain safety related functions in the battery pack, like the authentication function, and sensor functions such as the temperature sensor.
- **Common access method.** Unified access to all functions to enable generic SW driver usage in the systems.
- **Customizable for vendor specific functions.** Allow vendor specific functions in addition to MIPI BIF-defined basic functions that enable slave device differentiation in the market and access new innovations through the same unified interface.
- **Manufacturability.** A Slave device has to support programming at different phases of the battery pack production chain and during normal use of the battery pack.

Table 1 compares the MIPI BIF v1.0 standard to the Smart Battery System (SBS) interface, which is presumably the closest relative standard to BIF. This interface was developed by the Smart Battery System Implementers Forum. It uses SMBus for data transport. Please note that SBS was not developed particularly from a mobile device viewpoint originally.

Table 1: Comparison of Battery Interfaces

Requirement	MIPI BIF v1.0	SBS Rev 1.1/SMBus 2.0
Low Cost Battery Support	Yes	No
Smart Battery Support	Yes	Yes
Fast Battery Pack Presence Detection	Yes	No
Single Wire Interface	Yes	No (two wire SMBus)
PHY Signal level VHigh_min/VHigh_max	0.9V – 3.0V	2.1V - 5.5V
Battery Authentication Function	Yes	No
Battery Temperature Monitoring	Yes	Yes
Slave Interrupt support	Yes	No (optional wire)
Manufacturer specific Function support	Yes	Yes (up to 5)
Multi-slave support	Yes	Yes
Unified SW access to all functions and data	Yes	No
Unified, scalable data structures and scalable function content in a Slave device based on need.	Yes	No

BIF Specification Delivers an Integrated, Flexible Solution

BIF architecture overview

There are two primary types of devices on a BIF bus, Master and Slave. There is only one Master on the battery communication line (BCL) of the BIF bus. There are also two types of Slaves, Primary Slave and Secondary Slave. There might be multiple Slave devices on a BCL, usually located within the Battery Pack, but they might also be located within the Host system.

The BIF master device is placed in the power management IC (PM IC) in a typical mobile device platform as illustrated in the Figure 1. It can also be placed on the digital baseband (BB) IC since the scalable electrical signaling level allows implementation on low voltage semiconductor processes. The BIF protocol allows that the master device can be implemented in hardware (HW), SW + General Purpose Input Output (GPIO) pin or a combination of HW and SW.

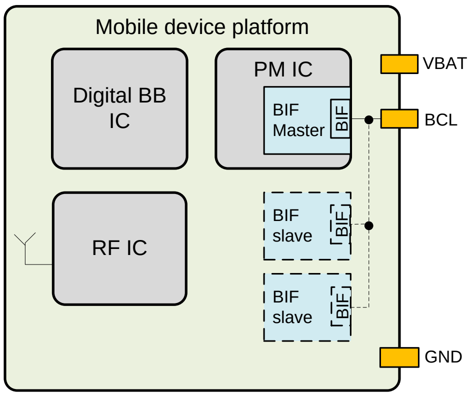


Figure 1. BIF master device placement in the mobile device platform

BIF supports low cost battery packs and smart battery packs, one connected at a time. Both of them have a pull-down resistor (R_{ID}) connected to the battery communication line (BCL) as illustrated in the block diagrams of **Figure 2**. The value of R_{ID} is used to identify whether the battery is a smart or low cost type, and to identify the battery pack electrical characteristic for low cost battery. The R_{ID} can be also used in the implementation of fast battery pack presence detection for both battery types. If a smart battery is disconnected, R_{ID} also has the important role to pull the BCL line down and consequently put the Slave device(s) in power-down mode.

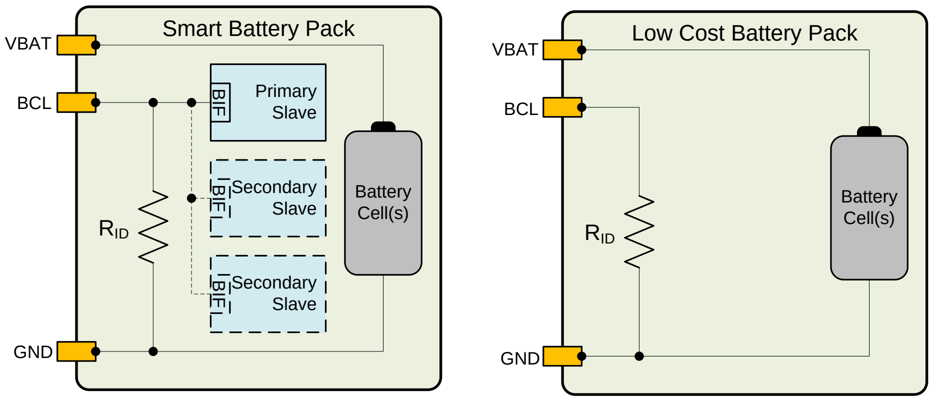


Figure 2. BIF smart battery pack and low cost battery pack

A BIF Master can address up to 256 Slave devices connected to the BCL line in total, by using so-called short 8-bit addressing. At a minimum, the Master device functions include physical layer and protocol functions, and it may implement the fast presence detection and low cost battery identification (R_{ID} value measurement) functions. Master and Slave device main functions are illustrated in Figure 3.

A Slave device is either a Primary Slave or a Secondary Slave. Both types have the protocol function and physical layer, and a small amount of standard device identification data at a minimum. Each Slave can have up to 64k bit of addressable memory space.

There can be only one Primary Slave per subsystem, i.e. battery pack or host system. The Primary Slave is able to store the Secondary Slave unique ID's of all Secondary Slaves within the same subsystem. This is the only difference between Primary and Secondary Slave. The main idea behind this arrangement was to enable the use of short (8-bit) device addresses in practical applications, and thus speed up the device discovery process in the bus and Slave device short address assignment for the Master. Otherwise, the Master has to do address search and addressing with full unique IDs (80-bits).

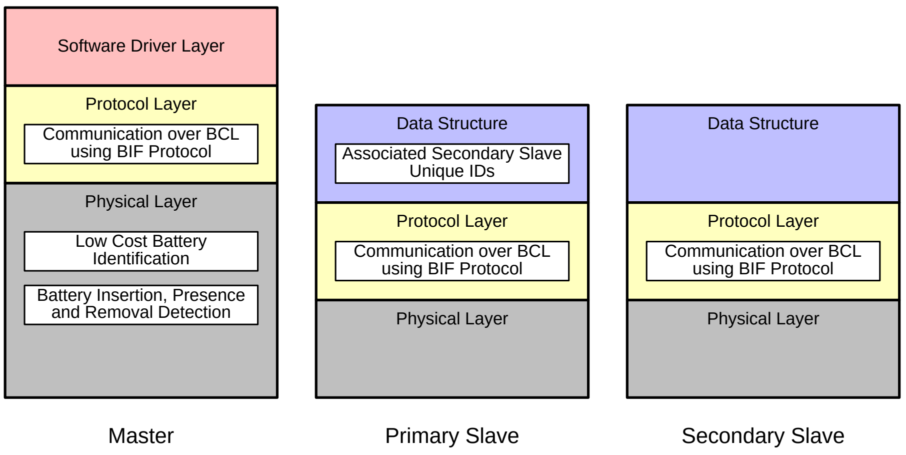

Software Driver Layer
Protocol Layer
Communication over BCL using BIF Protocol
Physical Layer
Low Cost Battery Identification
Battery Insertion, Presence and Removal Detection
Master
Data Structure
Associated Secondary Slave Unique IDs
Protocol Layer
Communication over BCL using BIF Protocol
Physical Layer
Primary Slave
Data Structure
Protocol Layer
Communication over BCL using BIF Protocol
Physical Layer
Secondary Slave

Figure 3. BIF device types

Physical Layer

BIF uses a single wire, open-drain communication interface. It supports a single low cost battery pack or single smart battery pack on the BCL at a time. The block diagram of the physical layer is illustrated in Figure 4. If the Master was designed to support low cost batteries and fast presence detection of the battery pack, then the BIF physical layer would have R_{ID} measurement and fast presence detection circuit in the Master (dashed boxes). R_{ID} value measurement requires an analog-to-digital converter (ADC) with a 10-bit resolution. The fast presence detection implementation can be done several ways. BIF specifies only the time range for the operation.

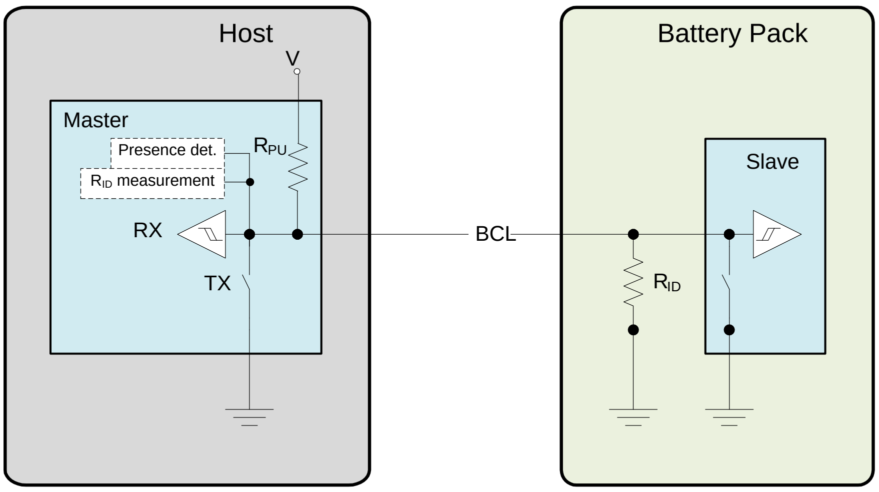


Figure 4. BIF physical layer

The communication pull-up resistor or current source is always in the host. Thus, the signal high level is set by the host and is scalable from 1.1V to 2.8V. This is important to enable the implementation of a host, even on the latest low voltage semiconductor processes. Minimum rise and fall times have been defined to limit potential electrical magnetic interference (EMI) issues.

Protocol

The BIF protocol is designed as a data transport interface. The actual battery applications, such as temperature measurement and authentication, make use of the protocol but do not interfere with it. Data transport and battery application usages are clearly separated. The challenge in BIF was to have a flexible protocol, but still have a small footprint in silicon. Gate count of a BIF implementation is expected to be approximately 1k Gates.

The main benefits of the BIF protocol are that it is either SW or HW implementable and the communication data rate is scalable between 2kbit/s – 250kbit/s (average). The minimum data rate was extended down to ~2kbit/s because in many systems there is 32.768 kHz clock available due to real time clock requirement. A 32.768 kHz clock produces about ~2kbit/s data rate in a typical BIF protocol implementation.

The maximum data rate was limited to 250kbit/s to minimize the Slave device receiver size. The maximum calculated use case suggests that even ~100kbit/s would be sufficient for some time.

BIF communication is always initiated by the Master and is based on a data word. The Master defines the communication speed in the beginning of every word and the addressed Slave must use that speed in its response. So in theory, each transaction between the Master and the Slave could happen with a different speed.

BIF Data Word

Each 17-bit data word consists of the following elements: training sequence (2 bits), payload (10), parity bits (4) and inversion bit (1). A data word can carry a command, a device or register address, read data or write data. The training sequence bits are used to tell the communication speed and also tell whether the word is a Broadcast type (intended for all Slaves) or a Unicast / Multicast type (intended for a certain Slave or Slaves only).

The payload is the actual data to be transported. The parity bits (conforming to Hamming-15 coding) are used to detect the possible communication errors. Strong detection of communication errors is important especially for a battery interface because of the physical connector on the line. Also, mobile devices are exposed to abuse and shaking by nature.

BIF protocol signaling is based on Time Distance coding, i.e. based on time between changes of the signal level. Logical "0" is an one time unit and logical "1" is a three time units, i.e. a logical "1" can be either a three time unit long high or low state signal on the bus. Between each word there must be at least 5 time units (stop). Example BIF words are illustrated in the Figure 5.

The inversion bit is included due to the nature of BIF protocol symbol signaling. If more than half of the data word bits are logical "1"'s (3 time units each), all bits in the word are inverted and then the word contains more "0"'s (1 time unit each) and overall time needed to send the whole word is shorter. This idea makes data bandwidth more efficient and less dependent on the actual data content.

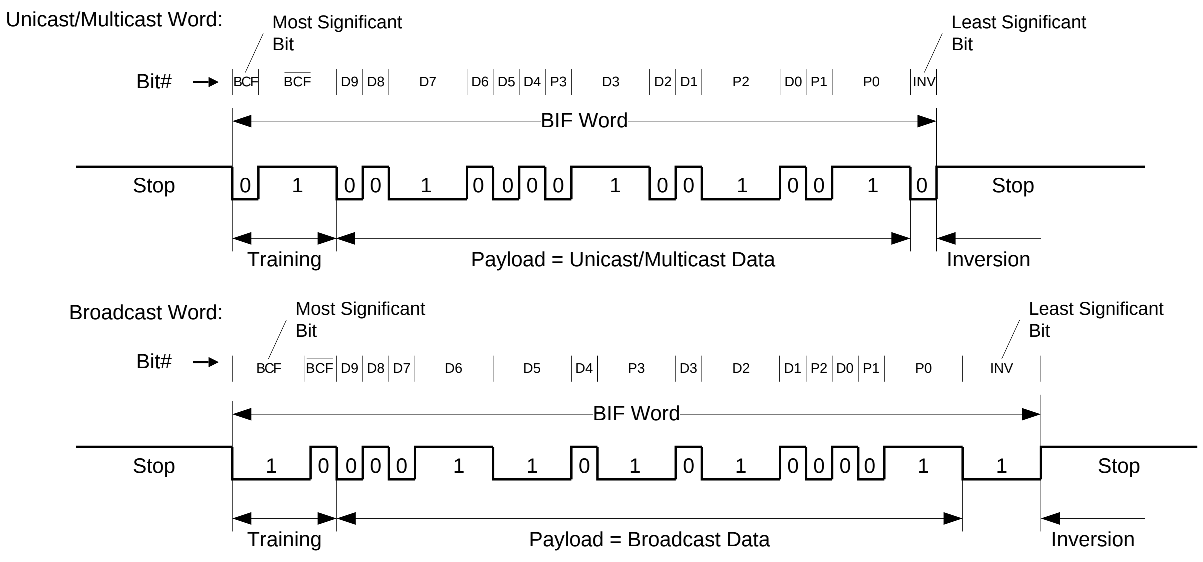


Figure 5. Example BIF Data words

Interrupts and Task Control

BIF also supports Slave interrupts and task control and has built in features to facilitate that. Task control provides information of the status (busy information etc.) of each task on the slave. This is an important feature because presumably the slave devices will be quite slow due to tight cost requirements.

At its best, interrupts can off-load the host system by, for example, taking care of battery temperature, voltage or current monitoring functions, and interrupt the host only when needed. Slaves will enter into interrupt mode when the Master sends the ""enable interrupt" command to all Slaves on the BCL. During interrupt mode there is no other communication on the BCL line.

BIF defines three power modes for Slave devices: Active, Standby and Power-down. Power-down mode has been designed so that when battery pack is removed for longer than a defined time the Slave inside it will go to power-down mode automatically.

Slave Data Structure

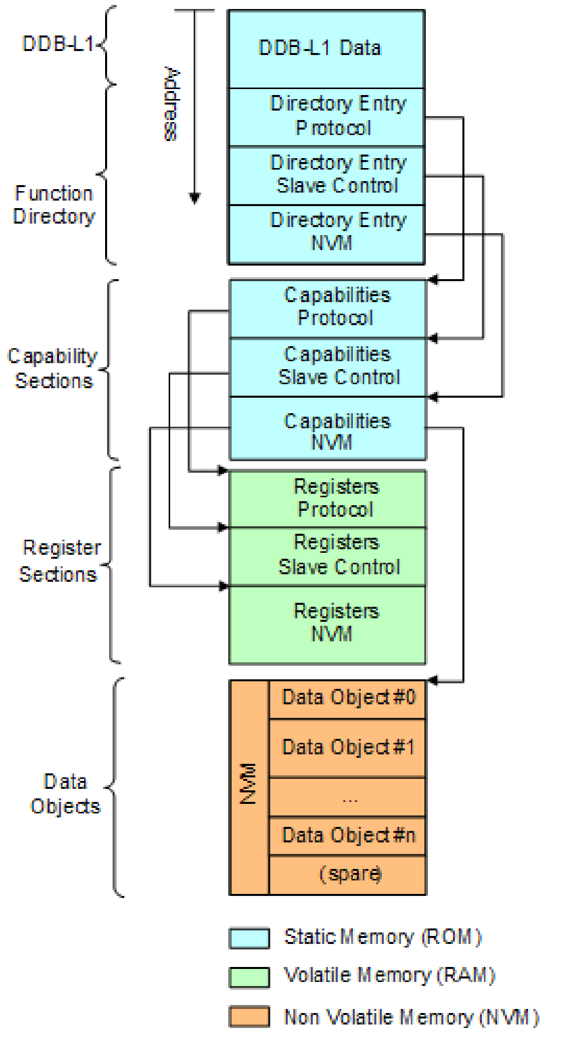
The key factors in BIF Slave device data structure definition work were cost-efficiency, unified access to all functions with a generic SW driver, manufacturability (i.e. programming support at different phases of the battery pack production chain) and support for vendor specific functions to enable Slave device differentiation in the market.

The BIF Slave data structure overview is illustrated in the **Error! Reference source not found.**. The maximum memory size addressable by BIF protocol for each BIF Slave is 64k Bytes. Memory locations can be RAM, ROM or reprogrammable NVM depending upon the need. The data map always starts with a fixed length 10-byte Layer 1 (L1) Device Descriptor Block (DDB) defined by the MIPI Alliance. DDB-L1 contains generic device identification information about the device like Manufacturer ID and Product IDs. The last two bytes of the DDB-L1 tell the total size of function directory following immediately after the DDB-L1.

Function Directory

Each function entry in the Function directory consists of four bytes. The first two bytes of a directory entry contains the Function Type and Version, which provides sufficient information for the Host to select the correct software driver for that function. The last two bytes are a pointer to the first byte of the function capability section. Each function has a capability section whose c pability information may include static information about the function or pointers to the function registers located in other memory section (volatile memory or non-volatile memory).

Fig. 6 Slave Data Structure



Functions

A function can be either a MIPI BIF pre-defined type or manufacturer defined function. The main idea is that a generic software driver can find and identify both types of functions listed in the function directory. The generic SW driver should be able to interpret and use fully any of the BIF pre-defined functions. But for the manufacturer specific functions, an additional SW driver provided by the manufacturer is needed to be able to use the function and interpret the data content.

BIF Functions

MIPI BIF has pre-defined protocol, Slave control, non-volatile memory (NVM) , temperature sensor and authentication functions. Only the protocol function is mandatory for a Slave while the others are optional.

Another main principle with all functions was to minimize the number of redundant bytes to be stored for cost efficiency. The Slave control function provides central control and status information for the tasks that are being executed on a Slave device. The total size of the Slave control function depends on how many other functions use its services in the Slave. The same principle was applied to the temperature sensor function accuracy specification with a number of different ranges. The Temperature Function also provides single and periodic measurement routines as well as interrupt triggering with thresholds.

The authentication function was defined as a BIF function type that requires, at a minimum, an equivalent of 80-bit symmetric key strength. The actual implementation of the authentication function is manufacturer specific, therefore the host software driver must have the manufacturer specific SW driver addition to control the authentication function and interpret the data.

Last but not the least is the BIF NVM function that allows reading and writing of a consecutive user NVM space of arbitrary size and independent of the physical organization of the memory. The user NVM can be used for multiple purposes such as storage of battery related information or recording usage data in the field. Parts of the user NVM can be write protected for further steps in the manufacturing process i.e. Slave IC manufacturer can write there and then the battery pack manufacturer can write and finally the mobile device can store there for example some battery pack aging information during field use.

The NVM function provides write lock control for the data written to the NVM both for the so called manufacturer NVM write and user data write. Manufacturer NVM write is used to write some reset values for registers during device manufacturing and user data write is used in later phases of battery pack life.

By being a data independent generic data transport interface, and having access to the battery pack, BIF is very suitable to control other non-battery related functions that may reside in the battery pack or even the host device.

BIF Objects

The actual data is written to and read from the user NVM in BIF-defined data objects. Reading and writing the objects is done through the NVM function. Currently BIF defines full data format for secondary Slave object and header for the battery parameter object.

Having a pre-defined battery parameter object access and battery pack manufacturer identification info and length defined in the beginning of the object enables a generic SW driver to access and identify the actual data packet. After adding the manufacturer specific SW driver, the host will be able to interpret the data packet content.

A typical battery parameter object contains, for example, information about the battery model, capacity, chemistry, charging and discharging and aging parameters. Other information may be included if needed. The idea is to provide a method for battery pack makers to pass the important parameters of the battery pack to the mobile device. A data object can contain any arbitrary data, but still it can be accessed and identified by the generic software driver.

Acknowledgements and Additional Information

For more information about MIPI Alliance and for the latest BIF specification that is available to the members of MIPI, check www.mipi.org .

The BIF specification was the joint effort of cellular handset suppliers, chipset, and slave device providers. The following companies have contributed to the BIF specification, in alphabetical order: Analog Devices, Infineon Technologies, Intel Corporation, Lattice Semiconductor Corporation, Nokia Corporation, Panasonic Corporation, Qualcomm Incorporated, Research In Motion, Sony Ericsson Mobile Communications, STMicroelectronics, ST-Ericsson, Texas Instruments Incorporated, Toshiba Corporation

The authors would like to gratefully acknowledge the contribution of all MIPI Alliance members who participated in the creation of the BIF material included in this article.