Smart Battery System Specifications

Smart Battery System Manager Specification

Revision 1.1
December 15 1998

Questions and comments regarding this specification may be forwarded to:
Email: selector@sbs-forum.org
Or: questions@sbs-forum.org

For additional information on Smart Battery System Specifications, visit the SBS Implementer's Forum (SBS-IF) at:
www.sbs-forum.org

Table of Contents

Revision History

Revision Number	Date	Author	Notes
1.0	12/15/98	S Fukatsu/R Dunstan	Version 1.0 release

1. Introduction

The Smart Battery System Manager is a specification that describes the requirements and the interface for a component or system of components that manage a number of Smart Batteries in a system. It autonomously connects one or more batteries to power the system, controls the charging of multiple batteries, reports the characteristics of the battery(s) powering the system etc. System safety largely resides in its behavior. The Smart Battery System Manager supercedes the Smart Battery Selector.

The Smart Battery Selector Specification, an adjunct to the Smart Battery Data Specification, provided a solution for many of the complexities associated with the implementation of multiple-battery systems such as notebook computers. However, it only supports batteries that charge or discharge, one-at-a-time. It has no provisions for reporting that more than one battery is powering the system at a time and its interface explicitly prohibits such an action.

Systems that operate with more than one battery pose a number of challenges for the system's designer. Since batteries and AC power can come and go, literally, at the user's whim without regard for the system's power requirements, a Smart Battery System Manager must be capable of responding to these events without compromising the integrity of the system's power supply or safe operation. Additionally, the Smart Battery System Manager (SBSM) should notify the system's power management software when ever a change takes place, such as when a battery is inserted or removed.

1.1. Scope

This specification describes the interface SBSM presents to the host system AND its minimum functional requirements. The actual SBSM implementation range from a single integrated component, emulation done using a notebook's keyboard controller, or to a micro-controller implementing SBSM functionality in addition to a Smart Battery Charger functionality. The purpose of this specification is to describe the minimum expected functionality and interface.

This document specifies the data set used by a Smart Battery System Manager and the minimal functionality that such a device must provide. The Smart Battery System Manager component's manufacturer will cover the actual electrical and mechanical specifications. This specification does not recommend or endorse any particular implementation; it just sets forth the requirements for the actual implementation.

1.2. Audience

The audience for this document includes:

- Smart Battery System component manufacturers
- Smart Battery System designers
- Designers of power management systems for Smart Battery powered portable electronic equipment

2. References

- *Smart Battery Data Specification*, Revision 1.1, SBS-Implementers Forum, December, 1998
- *Smart Battery Charger Specification*, Revision 1.1, SBS-Implementers Forum, December, 1998
- *Smart Battery Selector Specification*, Revision 1.1, SBS-Implementers Forum, December, 1998
- *System Management Bus Specification*, Revision 1.1, SBS-Implementers Forum, December, 1998
- *System Management Bus BIOS Interface Specification*, Revision 1.0, February 15, 1995
- *ACPI Specifications*, Version 1.0a, Intel Corporation, Microsoft Corporation, Toshiba Corp., July 1998 (http://www.teleport.com/~acpi)
- *The I²C-bus and how to use it*, Philips Semiconductors document #98-8080-575-01.
- *ACCESS.bus Specifications -- Version 2.2*, ACCESS.bus Industry Group, 370 Altair Way Suite 215, Sunnyvale, CA 94086 Tel (408) 991-3517

3. Definitions

- **ACPI**: Advanced Configuration and Power Interface.
- **APM:** Advanced Power Management. A BIOS interface defined to enable system-wide power management control via software.
- **Battery:** One or more cells that are designed to provide electrical power.
- **Cell:** The cell is the smallest unit in a battery. Most batteries consist of several cells connected in series.
- **EC:** Embedded Controller as defined in chapter 13 of the ACPI specification.
- **I²C-bus**: A two-wire bus developed by Phillips, used to transport data between low-speed devices.
- **Smart Battery:** A battery equipped with specialized hardware that provides present state, calculated and predicted information to its SMBus Host under software control.
- **Smart Battery Charger:** A battery charger that periodically communicates with a Smart Battery and alters its charging characteristics in response to information provided by the Smart Battery.
- **Smart Battery Selector:** A device that arbitrates between two or more Smart Batteries. It controls the power and SMBus paths between the SMBus Host, a Smart Charger and the Smart Batteries.
- **Smart Device:** An electronic device or module that communicates over the SMBus with the SMBus Host and/or other Smart Devices. For example the back light controller in a notebook computer can be implemented as a Smart Device.
- **SMBus**: The System Management Bus is a specific implementation of an I²C-bus that describes data protocols, device addresses and additional electrical requirements that is designed to physically transport commands and information between the Smart Battery, SMBus Host, Smart Battery Charger and other Smart Devices.
- **SMBus Host:** A piece of portable electronic equipment powered by a Smart Battery. It is able to communicate with the Smart Battery and use information provided by the battery.
- **Packet Error Check (PEC):** An additional byte in the SMBus protocols used to check for errors in an SMBus transmission. Refer to the System Management Bus Specification Revision 1.1. A Smart Battery System Manager indicates its ability to support PEC with the BATTERY_SYSTEM_REVISION value in BatterySystemInfo() function.

4. Smart Battery System Manager

Why is a Smart Battery System Manager (SBSM) necessary? It would seem that the obvious method for adding an additional Smart Battery to any system would be simply to allocate an additional SMBus address for the second battery. Unfortunately, this is not a practical solution; the SMBus address is merely one part of the system, defining the data path for Smart Battery data transactions between the SMBus host, the Smart Battery Charger and the Smart Battery itself. Other significant connections are required, both to report correct information and to maintain system safety.

In most multiple-battery systems today, some device or mechanism is present to arbitrate between the batteries. Additionally, this device must be able to dynamically re-configure the system's power system with sufficient speed to prevent any side effects caused by removal of the system's primary battery. A user may neither know nor care which battery is powering their system, but they do expect that their system will keep on working. For example, in a system where a slot is shared between a battery and a floppy disk drive, the user may simply remove the battery to install the floppy drive. If the battery in the shared slot was powering this example system, the device must be able to switch to the other battery quickly enough that the system's power integrity is not compromised.

The system (notebook) needs to be informed when there is a change in the battery system. For example, the presence of AC could signal the system that it is possible to start charging or a battery calibration cycle or simply change the back light settings of the LCD display. The addition of another battery should signal the system that additional operating time maybe available. The removal of a battery should signal the system to make sure that there is enough energy to maintain system and data integrity. At a higher level, the operating system (OS) needs to be informed when ever there has been a change in the battery system's state. This is required so that the OS can accurately reflect the state of the system to the user and to make appropriate power management decisions.

The SBSM described in this specification can provide this level of functionality.

4.1. Smart Battery System Manager Requirements and Considerations

There are several requirements for a system operating from battery power in a multiple-battery system. These requirements ensure system safety, data accuracy and data integrity.

In all situations:

- BatterySystemState() will always return the actual state of SBSM. Software may use this feature to verify that a previous request was valid.
- In the case of a SBSM that supports simultaneous charging/discharging, the BatterySystemState() will report each battery's presence or absence and whether the battery is powering the system or connected to the Smart Battery Charger.
- The SBSM will select the proper battery or combination of batteries to power the system immediately when external power is removed or fails. This must be done autonomously and a BatterySystemState()notification sent to the system host by whatever mechanism is used within a given implementation.
- The SBSM will have an external input that when asserted, will inhibit any and all charging activity. This input provides a hardware input that may be used prohibit charging . It has the same effect as setting the BatterySystemStateCont() CHARGING_INHIBIT bit. Note that when this input is asserted, the BatterySystemStateCont() CHARGING_INHIBIT bit will return a 1. When this input is returned to its non-asserted state, the BatterySystemStateCont() CHARGING_INHIBIT bit will be returned to its previous state.

4.1.1. Powered by the Battery Subsystem

When operating from battery power:

- The power output of unselected batteries must be isolated from the system power supply. This is necessary to prevent potentially high-current conditions that can occur if two or more batteries with different characteristics or states are simply connected together. However, legal topologies can be created where two or more batteries may be discharged simultaneously. In this case, the SBSM must have circuitry to prevent potentially dangerous current flows between the batteries in use.
- The SBSM must maintain an SMBus connection between the battery or batteries powering the system and the system host. This ensures the integrity of AlarmWarning() messages between the battery(s) in use and the system host. These warnings from battery(s) not in use may be ignored. In the case of simultaneous discharging or charging, the SBSM is responsible to create a "composite" critical alarm from the batteries in use and must send the composite critical messages to the system host. The composite critical message is the bit-wise, logical-OR of AlarmWarning() messages from the individual batteries.
- The host must be able to communicate individually with every battery in the system to determine its capacity, etc., without interfering with the normal operation of the battery(s) powering the system. This feature enables user oriented displays that show the condition of every battery in the system, not just the active one(s).
- The SBSM must provide autonomous switchover functions between batteries to prevent system power failure.
- In multiple-battery systems, when one of the batteries in use fails or is removed, the SBSM must maintain the system's power integrity. It may use one or more of the remaining batteries to power the system. These actions must take place without host intervention and without compromising safety.
- When the SBSM determines that the configuration has changed for any reason, the SMBus host must be notified that a change has occurred. There are two acceptable methods:
    1. The SBSM masters the SMBus to the host and writes its BatterySystemState() register to the SMBus host (SMBus WriteWord protocol to the SMBus Host with the command code set to the selector's address 0x14).

Note: The SBSM will issue a BatterySystemState() notification for any change in the battery/power system status, including changes in the BatterySystemStateCont(). It is the responsibility of the OS driver to query the status of both the BatterySystemState() and BatterySystemStateCont() in order to determine the exact reason for the notification (e.g. change in the AC status).

2. The SBSM may notify the system independent of the SMBus by using a signal line that would cause the host to query the SBSM's BatterySystemState() and the BatterySystemStateCont(. This signal line may be connected either to an interrupt or it may be polled by the SMBus host or other firmware on the system platform. In all cases, the host must be notified by the SBSM whenever a change occurs. The implementation is at the option of the system's designer.

4.1.2. Powered by AC

When operating from an externally supplied power source (AC), there is another set of requirements for a multiple-battery systems. As in the previous case, there are potentially harmful side-effects if any of the following are ignored while operating from external power:

- The SBSM must be able to isolate the power output of all batteries from external power. This is necessary to prevent uncontrolled currents into or out of the batteries.
- The SMBus and Safety Signal connections must be properly maintained through the SBSM so that any and all batteries in the system are charged safely.:
    1. Monitor the SMBus connection between the SBSM and each smart battery in a way that ensures proper charging voltage and charging current is applied to each battery in a safe manner. Either ChargingVoltage() and ChargingCurrent() must be received from the battery or polled by the SBSM in a regular fashion. Refer to the Smart Battery and Smart Battery Charger specification for more details.
    2. Monitor the SMBus connections in a manner that ensures all AlarmWarning() messages from the battery are received by both the system host AND the appropriate Smart Battery charger. Alternatively, the system host or EC can regularly polls all the involved Smart Battery's BatteryStatus() registers to ensure safe charging. The EC or system host must both send the AlarmWarning() messages to the host and ensure that AlarmWarning() messages are sent to the Smart Battery charger that is charging the Smart Battery sending the messages.
    3. Each battery connected to a charger must have its safety signal appropriately monitored. The safety signal is an independent mechanism used to terminate or limit charge when the SMBus fails (primary safety mechanism). Failure to connect the safety signal to the SBSM defeats this method and may result in an unsafe system. The battery safety signal(s) must be continually monitored by the charger or chargers that are sourcing power to each Smart Battery being charged. This connection ensures out-of-band charge termination information is actually coming from the Smart Battery receiving power from the charger. This does not necessarily mean that one charger may charge only one battery. In cases where one charger is used to charge more than one battery, it must respond to any charge termination signal received from the batteries being charged. The charge termination message or signal may be an AlarmWarning(), other SBS command that inhibits charging or the safety signal changing its state.

To summarize, requirements for battery system safety during charging are:

1. The Smart Battery Charger must regularly receive or poll for charging voltage and current from all batteries that are being charged.
2. The SMBus host must have the capability to receive an AlarmWarning() from a battery being charged or must poll the battery's BatteryStatus().
3. The Smart Battery's safety signal(s) must be continually monitored by the charger or chargers that are sourcing power to each battery being charged.

Any topology that is in conflict with the above, may be unsafe and is forbidden. SBSMs that allow forbidden topologies are NOT compliant with the Smart Battery System Manager specifications.

4.1.3. Implementation Guidelines

To accomplish these tasks, the SBSM can be implemented in many ways. One very simple system is a collection of five switches that control the SMBus (data), the safety signal, and power:

- Battery output power
- Charger output power
- Battery-to-charger data
- Battery-to-charger safety signal
- Host-to-battery data

A more complex SBSM could be a combination micro-controller based system that has:

- One SMBus port for each Smart Battery
- Emulates the Smart Battery Charger function
- Provides the Smart Battery System Manager functionality
- Acts as the SMBus host.

An SBSM that supports either simultaneous charging and./or discharging have additional capabilities.

SBSMs that allow simultaneous discharge must also have the capability to:

- Report data from a composite of the batteries supplying power
- Report a composite of the alarms from the batteries supplying power

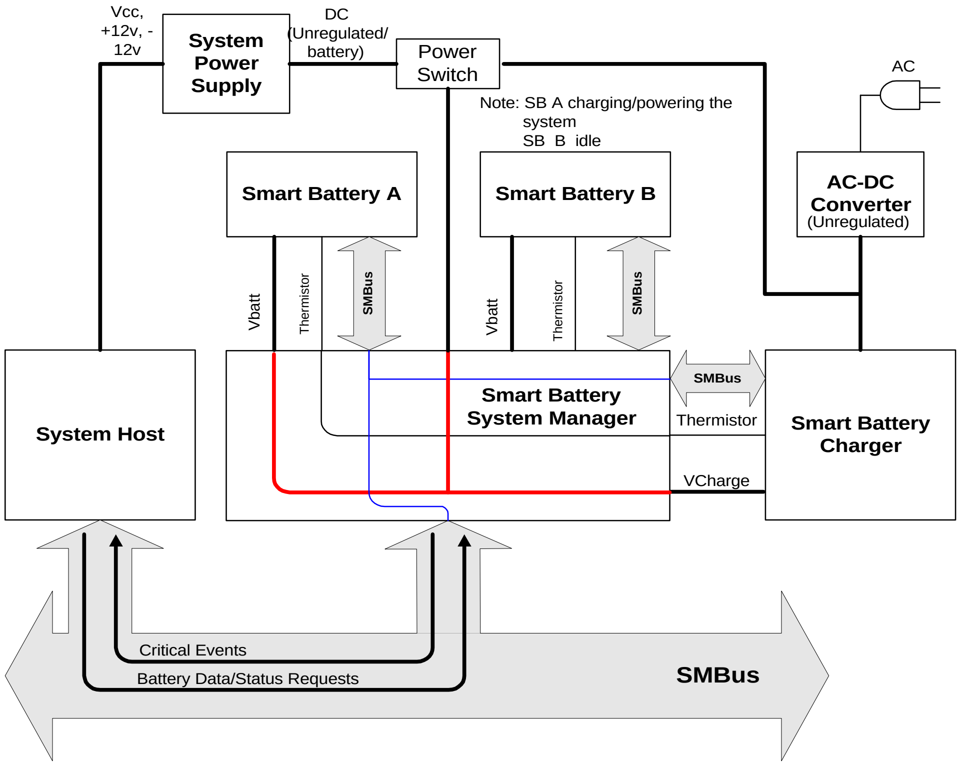
SBSMs that allow simultaneous charge must also have the capability to:

- Report a composite of the alarms from the battery(s) being charged power to both the SMBus host AND the charger(s)
- Maintain constant monitoring of the safety signal(s)
- Maintain the correct connections between the battery and charger for power, SMBus data and safety signal.

4.2. Smart Battery System Manager Model

One possible, simple Smart Battery System Manager model is a system consisting of two Smart Batteries, a Smart Battery Charger and the SBSM in a notebook computer. The diagram below illustrates the typical power and data paths when battery A is acting as the primary battery. This diagram shows a Smart Battery System Manager connecting Smart Battery A to both the system and the charger. When AC is present, the Smart Battery can opt to charge itself as required and the system will be powered by the AC source. If AC is not present, the system will be powered by Smart Battery A.

This is exactly the same model used by the Smart Battery Selector. The differences are in the reporting interface to the system.



Simple Multiple Smart Battery System

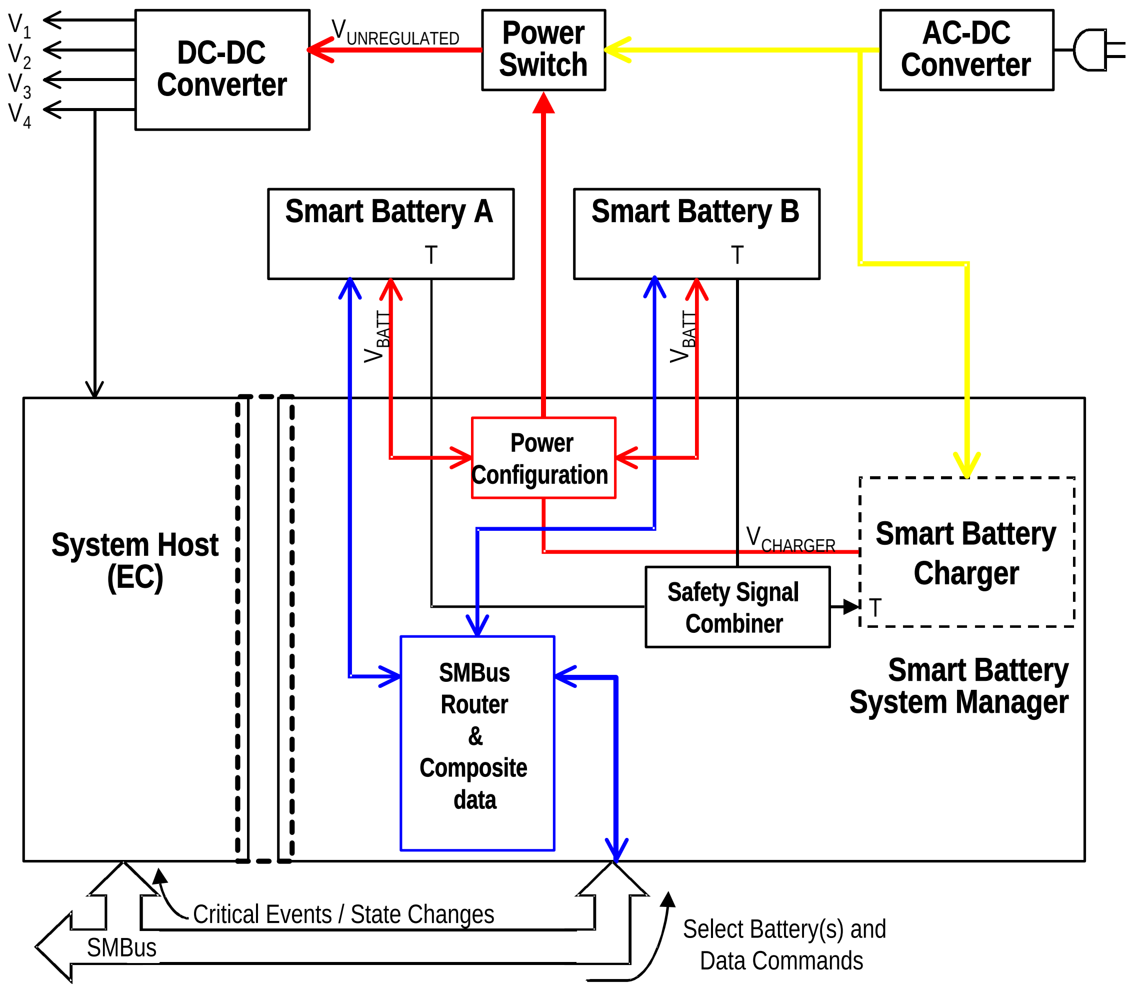
The following diagram is a block diagram of a SBSM, Smart Battery Charger and SMBus host as added functionality to the EC (embedded controller). Smart Battery A and/or B are available to power the system.

The power path configuration block is used by the SBSM to select which battery is used to power the system: Smart Battery A, Smart Battery B or a combination of both Smart Batteries A and B. The algorithm used is selected by the system's designer and is contained in the SBSM. If AC is present, the SBSM may chose to charge either Smart Battery A or B or both. Again, the algorithm used is contained entirely within the SBSM.

The safety signal combiner ensures that the Smart Battery Charger's alternative safety signaling path is always maintained.

The SMBus router ensures that the operating system can communicate with individual batteries as well as the composite of batteries being discharged simultaneously. There is no requirement for the composite battery data to be generated within the EC. Other alternatives such as a private interface between the EC and the OS that would allow a custom driver to calculate composite battery data are allowed. The SMBus router also ensures that the operating system receives all alarms from the battery or batteries being charged or discharged.

It is important to note, that while functions, such as the Smart Battery charger are shown as discrete blocks, their respective functionality may be emulated by the EC in combination with other hardware.



SBSM Combined with embedded controller

4.3. Interface Requirements

The software interface consists of the primary read/write control register, an optional read/write control register, and a read-only register. The SBSM may be implemented as an SMBus slave-only device, however it may be implemented as a master and send a BatterySystemState() notification to the SMBus host after every state change.

4.4. Smart Battery System Manager Functional Requirements

The SBSM must provide the following services:

- The SBSM will do a power-on default to connect one or more batteries to power the system if AC is not present and connect one or more batteries to their respective chargers if AC is present.
- The SBSM will monitor the discharging batteries' terminal voltages and if any falls below a preset minimum. It will autonomously switch to another battery or batteries (if any are present) or reconfigure the simultaneous discharge configuration. The SBSM will update its BatterySystemState() and the BatterySystemStateCont() to reflect any changes and notify the system the BatterySystemState() and/or the BatterySystemStateCont() has changed. In slave only implementations, this may be accomplished in several ways:
  - The SMBus host polls the SBSM's BatterySystemState() and BatterySystemStateCont() for changes

    - The SBSM uses the #SMBALERT mechanism to notify the SMBus host that a state change occurred. The SMBus host will then have to determine that the SBSM signaled the change and then read and pass on the contents of the BatterySystemState() to the system via the EC interface.
    - The SBSM uses an interrupt connected to a micro-controller such as the EC. When an interrupt is detected, it causes the BatterySystemState() to be read and passes the contents to the system via the EC interface.
- The SBSM will monitor the presence of the discharging battery(s) and, if it is removed, it will autonomously switch to another battery (or batteries) present. In systems with simultaneous discharge enabled, the SBSM will modify the configuration when any of the batteries being discharged is removed. The SBSM will update its BatterySystemState() register to reflect the change and notify the system that its BatterySystemState() has changed.
- If no batteries are powering the system as indicated by the status bits in POWERED_BY_X *and* there is an event that causes a BatterySystemState() change
    - if required the SBSM will autonomously switch to the next viable battery or simultaneous discharge configuration
    - update its BatterySystemState() register to reflect the change
    - notify the system that its BatterySystemState has changed.
- The charger connection is the sole responsibility of the Smart Battery System Manager. The SBSM must report whether AC is present or not, and if it changes, the SBSM must update its BatterySystemStateCont() register to reflect the change and notify the system that its BatterySystemState() has changed. Note: The SBSM renders the ChargerMode() and ChargerStatus() registers redundant from the Operating System's perspective.
- It is expected that the SBSM will operate in an entirely autonomous manner, independent of any high-level control such as that provided by an application or system BIOS. This autonomy allows the system to charge multiple batteries while the host intelligence is not operational (e.g., when the system is off or suspended). Since the SBSM operates autonomously, it is totally responsible for the battery system's safe operation and for maintaining the power integrity of the system.

In all cases, the SBSM's primary purpose is to ensure safe operation of the battery system and to maintain system power and minimize interruptions to that power. Its secondary purpose is to inform (or make information available to) the SMBus host about the system's BatterySystemState and changes in that BatterySystemState.

4.5. Smart Battery System Manager and Composite Battery Reporting

The SBSM is a component that owns the battery system. In some implementations, it may allow more than one battery to supply power simultaneously. This creates problems for the operating system as it does not necessarily have a means to accurately put together data from two or more batteries being discharged at the same time. The SBSM must have the ability to report composite battery data of the active batteries when more than one battery is powering the system simultaneously.

The SBSM is not required to support simultaneous discharge, but if it does, it assumes the burden to merge and report the battery system's composite data.

4.5.1. Battery Data Composition Guidelines

During simultaneous charging, the SBSM must have the ability to create a composite alarm of the individual batteries being charged. At a minimum, when any individual battery being charged issues an alarm, it must be passed through to the system host and charger. The charger(s) must subsequently stops charging if any battery's charging alarm bit defined in the Smart Battery Charger specification (including any reserved bits) is set.

During simultaneous discharging, the SBSM must logically OR the alarms of the individual batteries being discharged. At a minimum, when any individual battery being discharged issues an alarm, it must be passed through to the host including any reserved bits with the alarm values of the rest of the batteries being discharged as defined in the Smart Battery Data Specification.

During Simultaneous discharge, the SBSM should appear to the system host as either one battery or individual batteries if specific information is requested or directed to or from a particular battery. This requirement appears across the entire Smart Battery data set. Individual implementations may change the algorithms used to combine or merge data. The SBSM interface provides the means to access either the combined data or individual data.

Examples of battery data composition may be the MaxError() command. If the system host requests MaxError() and the SBSM is in a mode where battery composition is operating, the specific implementation may choose to report the RMS (root mean squared value of all MaxError() values.

4.5.2. Smart Battery System Manager Data Caching Guidelines

In order to preserve bandwidth on the system host SMBus, the SBSM may employ a cache for dynamic and static composite battery data. Dynamic data values are those that change or are expected to change during the mobile PC operation, e.g., Current(). Static values are those that remain the same as long as the battery is not removed or replaced, e.g., SerialNumber().

In order to preserve data integrity and currency, the SBSM needs to apply reasonable refresh rules to any cached values. Generally stated these rules are:

- Static values may be cached and re-read without limit as long as the SBSM has not been reset or the battery exchanged, removed or newly inserted. In any of these cases, the SBSM must refresh the static data in any cache it may maintain and notify the system host as advised in earlier in this specification.
- Dynamic data values may be cached in expectation of a system host request. Dynamic data must be refreshed after being read once. Further, the SBSM must flag an error (not respond or indicate stale data) or use any mechanism available in the physical layer to avoid or delay the response until fresh data is available and combined according to the implementation algorithm rules.
- The SBSM must refresh dynamic data at appropriate intervals suggested in the following table so that data is not stale. The timeout period for this refresh is governed by specific OEM implementations.

The following table is a guide to what battery data should be cached and when the cache should be invalidated or updated.

Function	Access	Composite	Cacheable	Cache Invalidate (in addition to battery insertion)	Cache Update (next access OR …)
ManufacturerAccess	r/w	no	No		
RemainingCapacityAlarm*	r/w	yes	Yes	On write	
RemainingTimeAlarm*	r/w	yes	Yes	On write	
BatteryMode	r/w	yes	Yes	On write	
AtRate	r/w	yes	Yes	On write	
AtRateTimeToFull	r	no	Yes	Write to AtRate or 1 minute (charge only)	
AtRateTimeToEmpty*	r	yes (discharge only)	Yes	Write to AtRate or 1 minute (discharge only)	
AtRateOK*	r	yes (discharge only)	No		

Temperature	r	Yes	Yes	1 minute (charging/discharging)) or upon a state change	
Voltage	r	yes	Yes	1 minute (charging/discharging)) or upon a state change	
Current	r	yes	No		
AverageCurrent	r	yes	No		
MaxError	r	yes	Yes	or upon a state change	
RelativeStateOfCharge	r	yes	Yes		1 minute (charging/discharging)) otherwise TBD minutes
AbsoluteStateOfCharge	r	yes	Yes		1 minute (charging/discharging)) otherwise TBD minutes
RemainingCapacity	r	yes	Yes		1 minute (charging/discharging)) otherwise TBD minutes
FullChargeCapacity	r	yes	Yes	Upon a state change	
RunTimeToEmpty*	r	yes	Yes		1 minute (charging/discharging)) otherwise TBDminutes
AverageTimeToEmpty*	r	yes	Yes		1 minute (charging/discharging)) otherwise TBD minutes
AverageTimeToFull	r	no	Yes		1 minute (charging)
ChargingCurrent	r	no	Yes		1 minute (charging)
ChargingVoltage	r	no	Yes		1 minute (charging)
BatteryStatus*	r	yes	No		
CycleCount	r	no	yes	Change from charge to discharge or discharge to charge	
DesignCapacity	r	no	yes		
DesignVoltage	r	no	yes		
SpecificationInfo	r	no	yes		
ManufactureDate	r	no	yes		
SerialNumber	r	no	yes		
reserved					
ManufacturerName	r	no	yes		
DeviceName	r	no	yes		
DeviceChemistry	r	no	yes		
ManufacturerData	r	no	no		
reserved					
OptionalMfgFunction5	r/w				

reserved					
OptionalMfgFunction4	r/w				
OptionalMfgFunction3	r/w				
OptionalMfgFunction2	r/w				
OptionalMfgFunction1	r/w				

Smart Battery Functions

4.5.3. Smart Battery System Manager Command Intercepts

In order to preserve bandwidth on the system host SMBus, the SBSM may intercept and respond directly to commands directed to either the Smart Battery or the Smart Battery Charger. When an individual battery is powering the system or is being charged or whenever individual battery data is desired, the SBSM relays the command to the battery and produces the requested data in the most efficient manner possible. The SBSM may optionally pass Smart Battery Charger commands through to the charger in the most efficient means possible.

The SBSM typically intercepts commands directed to the battery or charger whenever simultaneous charging, simultaneous discharging, or a combination of charging and discharging is taking place in the battery system. In this case, the SBSM will rely either on cached composite data values or it will have to generate composite data in real time. During these situations, the SMBus for both the battery and charger are invisible to the system host.

For instance, the SBSM may intercept the SBC ChargerMode() command when in simultaneous charging mode and make appropriate composition of the charging status. The composite data returned during simultaneous discharge should adhere to the relevant SBS specification. The method of data composition is determined by the OEM implementation.

5. Smart Battery System Manager Interface

The following functions are used by the SMBus Host to communicate with the SBSM and attached Smart Batteries.

The functions are described as follows:

FunctionName() 0xnn (command code)

Description:
A brief description of the function.
Purpose:
The purpose of the function, and an example where appropriate.
SMBus Protocol: Refer to Section 6 and to the SMBus specification for more details.
Input, Output or Input/Output: A description of the data supplied to or returned by the function.

Whenever the Smart Battery System Manager encounters a valid command with invalid data, it is expected to do nothing, just ignore the invalid data. For example, if an attempt is made to select battery A and B to simultaneously communicate with the system host, the SBSM will just ignore the request. This behavior will help to prevent errant commands from setting up conditions that might cause damage to the system, or batteries.

The following commands describe the interface the system software expects when communicating with a Smart Battery System Manager. It is acceptable for the system BIOS, or some other device in the system, to intercept and emulate some or all of the SBSM commands, provided that the SBSM interface defined below is maintained to the system software.

5.1. BatterySystemState (0x01)

Description:
This required function returns the present state of the Smart Battery System Manager and allows access to individual batteries. The information is broken into four nibbles that report:

- Which battery is communicating with the SMBus Host
- Which battery(s), if any, or AC is powering the system
- Which battery(s) is connected to the Smart Charger
- Which battery(s) are present.

The SBSM provides a mechanism to notify the system whenever there is a change in its state. Specifically, the SBSM will provides the system with a notification whenever:

- A battery is added or removed
- AC power is connected or disconnected
- The SBSM autonomously changes the configuration of the battery(s) supplying power
- The SBSM autonomously changes the configuration of the battery(s) being charged

This function provides simultaneous access to four nibble-wide registers of the following types:

- One to control the communications path between the SMBus host and a specific battery
- One to show the status of the battery(s) powering the system
- One to show the status of the battery(s) being charged
- One to show the status of the battery(s) present in the system

Purpose:
Used by the system host to determine the present state of the Smart Battery System Manager and the attached batteries. It also may be used to determine the state of the battery system after the SBSM notifies the system of a change.

SMBus Protocol: Read or Write Word
Input/Output: word -- bit flags in nibbles mapped as follows:

SMB **(r/w)** Battery(s) connected to SMBus host (comm.)				**POWER_BY** **(r)** Battery(s) Powering the System (power & comm.)				**CHARGE** **(r)** Battery(s) Charging (power & comm.)				**PRESENT** **(r)** Battery(s) Present			
D	**C**	**B**	**A**	**D**	**C**	**B**	**A**	**D**	**C**	**B**	**A**	**D**	**C**	**B**	**A**

SMB_X nibble
The read/write SMB_X nibble is used by the SMBus Host to select which individual battery to communicate with or to determine which individual battery or composite battery it is communicating with. Normally, this nibble is set to the same value as the POWER_BY_X nibble.

For example, an application that displays the remaining capacity of all batteries would write to this nibble to individually select each battery in turn and get its capacity. The bits are defined as follows:
0xf000 select all SMBus host is communicating with a composite of the battery(s) powering the system.
0x8000 SMB_D SMBus Host is communicating with Battery D
0x4000 SMB_C SMBus Host is communicating with Battery C
0x2000 SMB_B SMBus Host is communicating with Battery B
0x1000 SMB_A SMBus Host is communicating with Battery A
Note: On a write to this nibble, only one or all bits may be set. In the first case, the SMBus host communicates with the specified individual battery. In the second case, the SBSM will return a composite picture of the battery system. The battery(s) forming the composite can be determined by examining the contents of the POWER_BY_X nibble. The SBSM can not be forced to read any arbitrary set of batteries and is responsible to present composite data only for battery combinations it selects.
Note: Writes of values other than 0x1000, 0x2000, 0x4000, 0x8000, and 0xf000 are not allowed

The operating system will read this nibble to determine which battery it is actually communicating with to ensure that it is connected to the desired battery(s).
0xf000 Composite mode SMBus host is communicating with a composite of the battery(s) powering the system.
0x8000 SMB_D SMBus Host is communicating with Battery D
0x4000 SMB_C SMBus Host is communicating with Battery C
0x2000 SMB_B SMBus Host is communicating with Battery B
0x1000 SMB_A SMBus Host is communicating with Battery A
In composite mode, refer to the POWER_BY_X nibble to determine the composite batteries that make up discharging related data values. Composite data values are not returned for batteries during simultaneous charging. See the Appendix X for a classification of which data values fall into which category.

POWER_BY_X nibble
The read only POWER_BY_X nibble is used by the SMBus Host to determine which battery(s) is powering the system. All writes to this nibble will be ignored.

0x0800 POWER_BY_D System being powered by Battery D.
0x0400 POWER_BY_C System being powered by Battery C.
0x0200 POWER_BY_B System being powered by Battery B.
0x0100 POWER_BY_A System being powered by Battery A.
0x0000 POWER_BY_AC System being powered by AC.
Note: Each bit in the POWER_BY_X nibble will be set independently to indicate which batteries, if any, are powering the system. For example:
0x0300 POWER_BY_A and POWER_BY_B System being powered by battery A and B simultaneously

CHARGE_X nibble
The read only CHARGE_X nibble is used by the SMBus Host to determine which, if any, battery is being charged. All writes to this nibble will be ignored.

The bits are defined as follows:
0x0080 CHARGE_D Battery D being charged
0x0040 CHARGE_C Battery C being charged
0x0020 CHARGE_B Battery B being charged
0x0010 CHARGE_A Battery A being charged
0x0000 No Battery being charged
Note: Each bit in the CHARGE_X nibble will be set independently to indicate which batteries are actually being charged. For example:
0x0030 CHARGE_A and CHARGE_B Batteries A and B are being simultaneously charged.

An indication that multiple batteries are being charged simultaneously does not indicate that the batteries are being charged at the same rate or that they will complete their charge at the same time. To actually determine when an individual battery will be fully charged, use the SMB_X nibble to individually select the battery of interest and read the TimeToFull() value.

PRESENT_X nibble
The read only PRESENT_X nibble is used by the SMBus Host to determine how many and which batteries are present. All writes to this nibble will be ignored.

The bits are defined as follows:
0x0008 PRESENT_D Battery D is present
0x0004 PRESENT_C Battery C is present
0x0002 PRESENT_B Battery B is present
0x0001 PRESENT_A Battery A is present
Note: Each bit in the PRESENT_X nibble will be set independently to indicate the presence of a battery.

5.2. BatterySystemStateCont (0x02)

Description:
This required function returns additional state information of the Smart Battery System Manager and provides an interface to prohibit charging. This command also removes any requirement for the operating system to communicate directly with the charger to obtain AC presence information. When the SBSM is used, access to the charger address, 0x12, is blocked.

Purpose:
Used by the system host to determine the retrieve additional state information from the Smart Battery System Manager and the overall system power configuration. It may also be used by the system to prohibit any battery charging. This feature may be used in conjunction with a special power cable for example to prohibit charging in situations where charging is not allowed.

SMBus Protocol: Read or Write Word
Input/Output: word -- bit flags mapped as follows:

Field	Bits Used	Format	Allowable Values

AC_PRESENT	0	Bit flag	The AC_PRESENT bit returns a 1 when the SBSM determines that AC is present. It returns to 0 when AC is removed. Any change in this bit causes an SBSM state change. Writes to this bit have no effect
POWER_NOT_GOOD	1	Bit Flag	The POWER_NOT_GOOD bit returns 1 when the AC power is too low to supply adequate power to the system. Writes to this bit have no effect.
CALIBRATE_REQUEST_ SUPPORT	2	Bit Flag	The optional CALIBRATE_REQUEST_SUPPORT bit returns 1 when the SBSM has the ability to detect batteries needing a calibration cycle. Writes to this bit have no effect.
CALIBRATE_REQUEST	3	Bit Flag	The optional CALIBRATE_REQUEST bit returns 1 when the SBSM determines one or more batteries require a calibration cycle. Writes to this bit have no effect.
CHARGING_INHIBIT	4	Bit flag	The CHARGING_INHIBIT bit returns a 0 when charging is allowed and returns a 1 when charging is inhibited. This bit may be written to by the host to inhibit or allow charging. The default value of this bit is 0.
CHARGER_POR	5	Bit Flag	The CHARGER_POR bit always returns 0. Writing a 1 to this bit forces a power on reset of the charger.
CALIBRATE	6	Bit Flag	The optional CALIBRATE bit returns 0 when a battery calibration cycle is not in progress. This bit may be written to by the host to initiate a calibration cycle (fully discharging the battery followed by charging the battery). When the CALIBRATE_REQUEST bit is set, writing a 1 to this bit enables the SBSM to begin calibrating any battery when AC is present. The CALIBRATE bit will be reset when: • the battery begins charging • AC is removed • the battery being calibrated is removed • a 0 is written to the CALIBRATE bit The host may explicitly initiate a battery calibration cycle at any time. In this case, the host sets one of the CALIBRATE_X nibble bits and then sets the CALIBRATE bit to begin the calibration cycle.
Reserved	7	undefined	This bit is reserved and will always return zero.
CALIBRATE_A	8	Bit Flag	This optional bit may be set to select a battery for calibration. Only zero or one of the four CALIBRATE_X bits may be set at any one time. The bit is cleared when the CALIBRATE_OK bit is cleared.
CALIBRATE_B	9	Bit Flag	This optional bit may be set to select a battery for calibration. Only zero or one of the four CALIBRATE_X bits may be set at any one time. The bit is cleared when the CALIBRATE_OK bit is cleared.
CALIBRATE_C	10	Bit Flag	This optional bit may be set to select a battery for calibration. Only zero or one of the four CALIBRATE_X bits may be set at any one time. The bit is cleared when the CALIBRATE_OK bit is cleared.

CALIBRATE_D	11	Bit Flag	This optional bit may be set to select a battery for calibration. Only zero or one of the four CALIBRATE_X bits may be set at any one time. The bit is cleared when the CALIBRATE_OK bit is cleared.
Reserved	12…15	Undefined	These bits are reserved and will always return zero.

AC_PRESENT bit
The AC_PRESENT bit is a read only bit used to show the user the status AC availability to power the system. It may be used internally by the SMBus Host in conjunction with other information to determine when it is appropriate to allow a battery conditioning cycle. Whenever there is a change in the AC status, the SBSM must send a state change notification to the system. Since the AC_PRESENT bit is not part of that notification, the system has to read this register to determine the actual presence of AC.

POWER_NOT_GOOD bit
The POWER_NOT_GOOD bit is set whenever the SBSM determines that the AC (or external power source) is not supplying an adequate amount of power. The SBSM is expected to take the appropriate actions to ensure that the power integrity is maintained and notify the system of a change in the BatterySytemState. This bit will be reset when AC is removed or when the external power source is once again adequate.

CALIBRATE_REQUEST_SUPPORT bit
The CALIBRATE_REQUEST_SUPPORT bit is set to signal that the SBSM has a mechanism to determine when any of the attached batteries are in need of a calibration cycle. The bit is cleared if the SBSM does not have this capability.

CALIBRATE_REQUEST bit
The optional CALIBRATE_REQUEST bit is supported only by SBSMs that report a 1 in the CALIBRATE_REQUEST_SUPPORT bit. When this bit is set, the SBSM has determined that one or more of the connected batteries need a calibration cycle. This bit will be cleared when all batteries connected to the SBSM do not require a calibration cycle.

CHARGING_INHIBIT bit
The CHARGING_INHIBIT bit is a read/write bit that either reports the present ability of the charging system or allows the SMBus Host to inhibit the SBSM to allow batteries to be charged. When this bit is cleared, the SBSM may charge batteries as needed, when set, the SBSM MUST not allow any battery charging to occur. Changes in this bit do not cause a state change notification.

CHARGER_POR bit
The CHARGER_POR bit is used to force a charger power on reset. Writing a 1 to this bit will cause a POR, writing a 0 to this bit has no effect..

CALIBRATE bit
The optional CALIBRATE is used either to show the status of battery calibration cycles in the SBSM or to begin a calibration cycle. A calibration cycle may be done in response to a CALIBRATION_REQUEST or initiated by a user application as part a regimen to maintain battery longevity. When the CALIBRATE_REQUEST bit is set and none of the CALIBRATE_X bits are set, setting the CALIBRATE bit will start the calibration cycle on a battery selected by the SBSM. In this case, after starting the calibration, the system can read the CALIBRATE_X nibble to determine which battery is being calibrated. When the CALIBRATE_REQUEST bit is cleared, one of the CALIBRATE_X bits must be set in order for a calibration cycle to take place. Support of the battery calibration function is strongly recommended.

CALIBRATE_X bit

The CALIBRATE_X is set by the system to select a battery for a calibration cycle. After this bit is set, the system sets the CALIBRATE bit to start the calibration cycle.

5.3. BatterySystemInfo (0x04)

Description:
The SMBus system host uses this command to determine the capabilities of the Smart Battery System Manager.
Purpose:
Allows the system host to determine the number of batteries the Smart Battery System Manager supports as well as the specification revision implemented by the SBSM.
SMBus Protocol: Read Word
Input/Output: word -- bit flags in nibbles mapped as follows:

Field	Bits Used	Format	Allowable Values
BATTERIES_ SUPPORTED	0…3	bit flags	BATTERIES_SUPPORTED returns 1's in the bit positions of batteries supported by the Smart Battery System Manager. For example, a two-battery SBSM might return 0011 for this nibble, while a four-battery SBSM would return 1111 for this nibble.
BATTERY_SYSTEM_ REVISION	4…7	encoded nibble	The BATTERY_SYSTEM_REVISION reports the version of the Smart Battery System Manager specification supported: 1000 – Version 1.0. 1001 – Version 1.0 with optional PEC support All other codes reserved.
Vscale	8 .. 11	4-bit binary values	0 - 3 (multiplies voltage by 10 ^ VScale)
IPScale	12 .. 15	4-bit binary values	0 - 3 (multiplies current by 10 ^ IPScale)

Note: It is acceptable for a SBSM component to expect the system BIOS to intercept and emulate this command rather than implementing it in the actual device itself.

BATTERIES_SUPPORTED nibble
The BATTERIES_SUPPORTED nibble is used by the SMBus Host to determine how many and which batteries the SBSM can support. This specification is written to allow support for up to four batteries, but due to size or cost constraints a given SBSM may support less than this number. The bits in this nibble are individually hard-coded by the SBSM to indicate which battery positions the SBSM supports.

The bits are defined as follows:
0x0008 PRESENT_D Battery D is supported by the Smart Battery System Manager
0x0004 PRESENT_C Battery C is supported by the Smart Battery System Manager
0x0002 PRESENT_B Battery B is supported by the Smart Battery System Manager
0x0001 PRESENT_A Battery A is supported by the Smart Battery System Manager
Note: Each bit in the BATTERIES_SUPPORTED nibble will be set independently to indicate the battery positions supported by the Smart Battery System Manager.

BATTERY_SYSTEM_REVISION nibble

The BATTERY_SYSTEM_REVISION is an encoded value used to indicate which version of the Smart Battery System Manager specification the SBSM supports. For Revision 1.0 of the Smart Battery System Manager specification, the value will be 8.

6. Example Implementations

This section is intended to provide a closer look at some possible implementations of a Smart Battery System Manager.

6.1. Battery Charging Topologies

There are many possible charging topologies that may be used for simultaneous charging. In the following section three are presented. The topology used in any system is selected to meet the cost, safety and use models favored by the system designer. The SBSM specification does not have advocate a particular design, rather a consistent reporting interface and a safe implementation.

6.1.1. Single charger / multiple battery topology #1

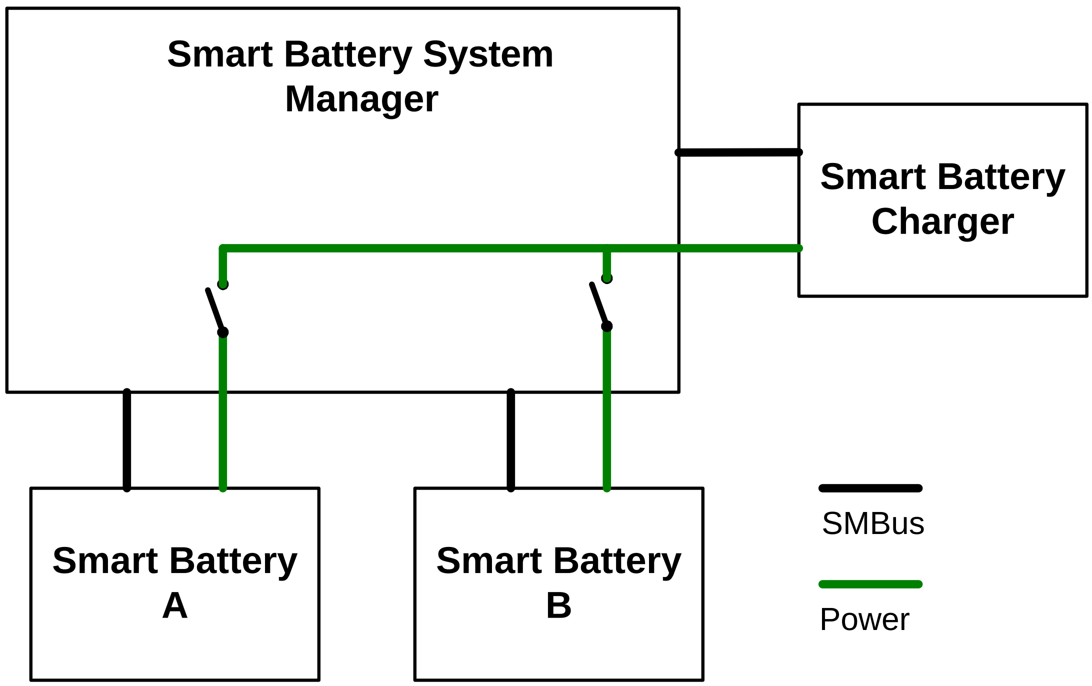


Figure 6.1 Single charger / multiple battery topology.

Figure 6.1 is a variation of the most common charger topology used by Smart Battery Selectors today. In the case of the selector version, the SMBus is switched along with the power. But to charge the batteries in parallel, the SBSM takes on additional responsibilities to ensure system safety. A few of these considerations the SBSM must take before connecting the batteries in parallel to the charger are:
The batteries are of the same chemistry. In fact, that they are from the same sub-family within a particular chemistry.
The batteries are at the same charge state.
The charging current to any individual battery does not exceed its safe limits

In the simultaneous charging topology shown in Figure 6.1, the SMBuses from each battery are logically "merged" and a composite charging request is sent to the charger. This may be done by the SBSM polling the individual battery's charging current requirements and/or their charging voltage requirements.

For example during the early stages of charging lithium batteries, the SBSM will request the charger supply a current value less than or equal to the sum of the currents requested by the individual batteries. The SBSM will then read the actual current values used by each individual batteries and ensure that none exceeds the amount that battery requested. As the batteries charge state nears full, the SBSM will transition into a constant voltage charger with the voltage set to the lower of the voltage requests from the individual batteries. The exact algorithms used will be specific to a particular platform.

6.1.2. Multiple charger / multiple battery topology

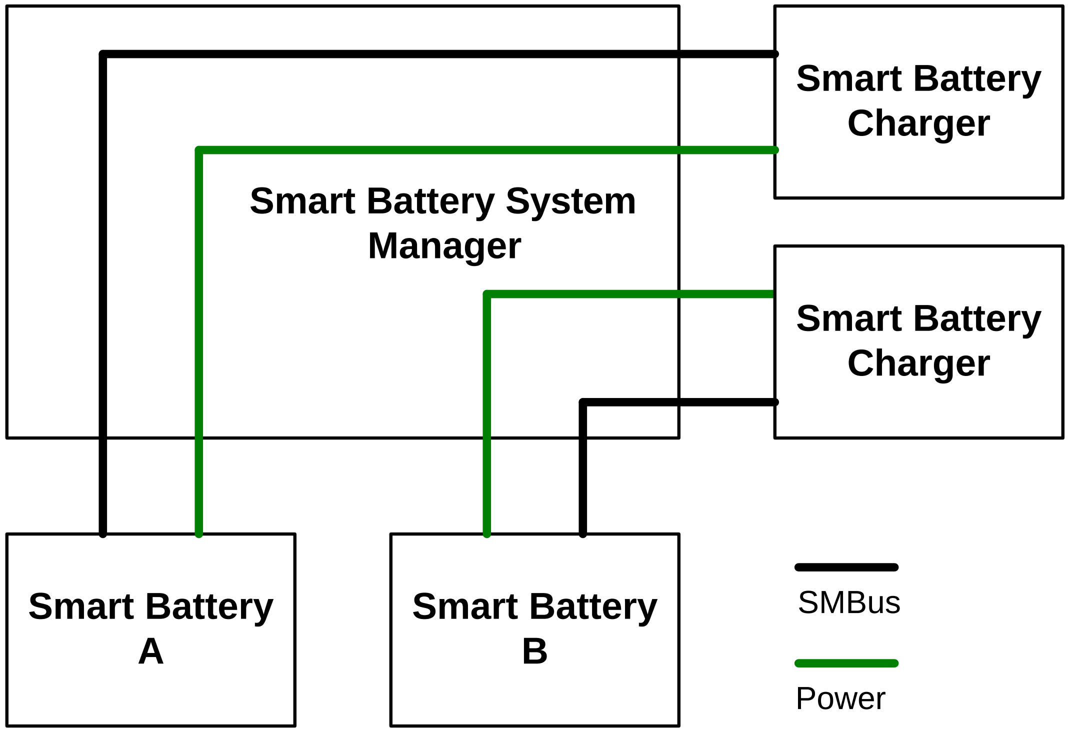


Figure 6.2 Multiple charger / multiple battery charger topology

Figure 6.2 requires a one to one correspondence between batteries being charged and chargers. This approach places little burden on the SBSM, but adds the requirement for multiple Smart Battery Chargers and may increase the BOM.

6.1.3. Single charger / multiple battery topology #2

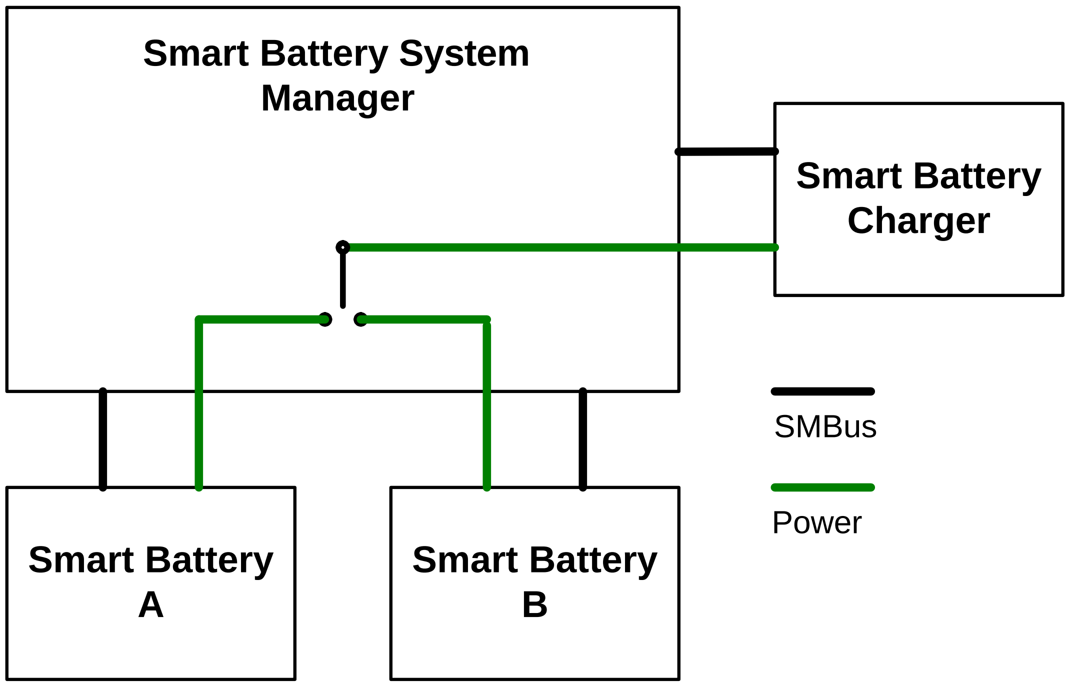


Figure 6.3 Single charger / multiple battery charger topology

Figure 6.3 shows a time –share arrangement between one charger and multiple batteries. It is similar to figure 6.1, but topologically does not allow for true simultaneous charging. This is the same approach used by a Smart Battery Selector and the SBSM must guarantee the connection of the SMBus and Safety Signal match the power connection when the switching rate between the batteries is relatively slow.

There is one major difference because the SBSM may switch rapidly between the batteries, thus appearing to charge them simultaneously. In this case, the SBSM is responsible to both program the charger and to ensure the individual battery's Safety Signals are appropriately combined to ensure safety

SMBus host is communicating with the composite of the battery(s) powering the system.

Appendix A. The command set in tabular form

In the following table, the function name, its access (r,w), data type and command are tabularized. For a Smart Battery System Manager to be recognized as compliant, it must support all the required functions described by this specification.

Smart Battery System Manager Functions

Function	Code	Access	Data
Reserved	0x00		
BatterySystemState	0x01	r/w	packed word
BatterySystemStateCont	0x02	r/w	packed word
BatterySystemInfo	0x04	R only	packed word
Reserved	0x03 - 0x2e		
OptionalMfgFunction5	0x2f	r/w	data
Reserved	0x30-0x3b		
OptionalMfgFunction4	0x3c	r/w	word
OptionalMfgFunction3	0x3d	r/w	word
OptionalMfgFunction2	0x3e	r/w	word
OptionalMfgFunction1	0x3f	r/w	word
reserved1	0x40-0xff		

Notes:
All unused function codes are reserved and may not be used
The upper two bits of all command codes are specifically reserved for future use to optionally address multiple Smart Battery System Managers

Appendix B. Intercept Command Categories

In the following table, the smart battery data set commands are categorized as dynamic or static and charger-related or discharge-related. The SBSM uses these categories to support command intercepts when combining data values and reporting the composite battery system to the host.

Table to be added.

###