Smart Batteries to the Rescue

By Ed Thompson

Blinding red lights. That's all I could see as my car slowed to a stop. Up ahead, an ambulance was pulled off to the side of the road, doors flung open, and lit up like a Christmas tree. Next to it emergency medical technicians were bending over a stretcher. One technician was busily trying to revive the accident victim, while the other monitored vital signs with portable medical equipment.

Since I had nowhere to go, I sat attentively, considering the event in front of me, and wondered. How is this life saving equipment, that's always on the go, maintained? How do they know battery power won't fail at a critical moment, like now.

SMART BATTERIES TO THE RESCUE

Maybe the answers to these questions can be found in technological advances brought about in the computer and telecommunications industries, where one area currently receiving intense activity is power management in portable systems. Our desire to edit one more document or place one last telephone call has led us to demand more staying power from our laptops and cell phones. Developments that extend performance, reliability, and safety in computers and telephones are also being adopted in a wide variety of other electronic products we have come to depend upon in our daily lives.

From semiconductor companies to software houses, component manufacturers to system integrators, a plan has been developed and actions taken place that combine information and communications within a system of portable power-related components. This system is called the Smart Battery System (SBS).

SBS is the combination of battery and electronic technologies. This article introduces the Smart Battery System Standard (version 1.0) for implementing smart battery technology, and suggests the improved performance, reliability, and safety this technology promises to bring to portable computers, medical equipment, consumer products, and more.

THE SMART BATTERY SYSTEM

Created by a group of leading companies in their effort to improve portable product performance, SBS is based on a set of standard specifications maintained by the Smart Battery Systems Implementers' Forum. As Figure 1 shows, SBS consists of several key components including:

- System Management Bus (SMBus) -- A physical medium and communication command protocols that support the transfer of information between SMBus devices.

- SMBus System Host -- The equipment (i.e., laptop PC, cellular phone, video camera, etc.) capable of communicating with SBS devices over the SMBus.

- Smart Battery -- A battery pack with added internal electronics that can measure, compute, and store battery data, and one that can communicate with other SBS devices over the SMBus.

- Smart Battery Charger -- A device that can provide a source of voltage and current for charging a Smart Battery, and that can communicate with other SBS devices over the SMBus.

- Smart Battery Selector -- A device that can provide the data and functionality used to support multiple Smart Batteries in a single system, and one that can communicate with other SBS devices over the SMBus.

SBS based products integrate these key components into a system capable of maximizing product service life, while at the same time providing accurate and timely equipment status information to the user.

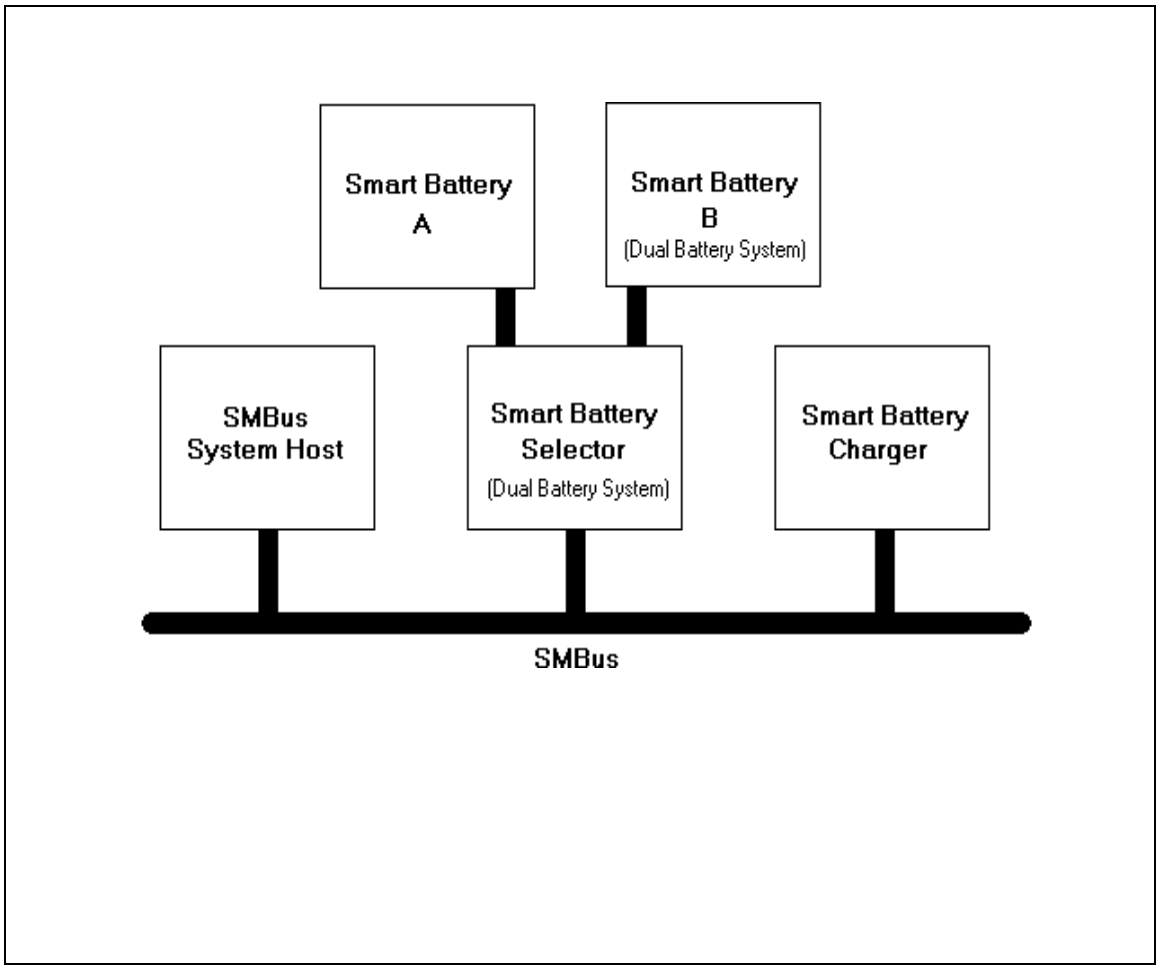

Figure 1. The minimum Smart Battery System requires a System Host, Smart Battery, and Charger. An optional Smart Battery Selector and second battery extends product run time.

SYSTEM MANAGEMENT BUS

The SMBus provides the physical medium and command protocols that support the transfer of information between SBS components. Envisioned as a low cost, low bandwidth communication link, it connects various devices within portable equipment.

SMBus includes a physical medium based on the I^2C Bus developed by Philips Semiconductors, although some of the electrical characteristics differ (refer to SMBus Specification, Section 1.5). The I^2C Bus is a two-wire open-collector multi-master/multi-drop serial bus that uses Clock and Data signals to communicate with up to 127 devices on a single bus. Using a serial bus reduces the pin count and cost of devices attached to the bus, and also reduces printed circuit board real estate requirements for pathways to connect the devices.

Devices on an SMBus may act as bus masters and/or bus slaves. A master initiates a message between itself and a slave device also attached to the bus by generating a start condition, followed by the slave address, and a read/write bit (see Figure 2). Each slave device on the bus has an assigned address. It is the responsibility of slave devices to recognize start conditions and monitor slave addresses that traverse the bus. A slave that recognizes its own address on the bus will generate an acknowledgment bit, thus signaling the master that the addressed slave device is present. Then, the master exchanges one or more data bytes and acknowledge bits with the slave, and terminates the message with a stop condition, or by initiating a new message with a repeated start.

Start	Slave Address	R/W	Ack	Data	Ack	…	Data	Ack	stoP

Start - A message initiation condition generated by a master device, where the Clock line is high while the Data line is lowered.

Slave Address - 7 bits of data on the Data line also generated by the master. Slave devices reading Data bits from the bus are synchronized with sequential pulses on the Clock line.

R/W - The Read/Write bit on the Data line is also synchronized with a pulse on the Clock line. The master generates this bit to tell the slave if the master will be reading data from, or writing data to the slave.

Ack - The Acknowledge bit that follows each byte. For the Slave Address byte, the Ack is generated by the slave device. For all other bytes, the Ack is generated by the device receiving the data.

Data - 8 bits of data on the Data line, synchronized with master generated sequential pulses on the Clock line.

StoP - The message termination condition, where the Clock line is high while the Data line is raised.

Figure 2 - SMBus relies on a basic message structure borrowed for the I^2C Bus.

Standard SMBus slave address assignments for SBS devices include:

0x10 -- SMBus System Host
0x12 -- Smart Battery Charger
0x14 -- Smart Battery Selector
0x16 -- Smart Battery

Within the structure of its messages, SMBus defines eight (8) possible command protocols (see Figure 3). These include:

- Quick Command
- Send Byte Command
- Receive Byte Command
- Write Byte/Word Command
- Read Byte/Word Command
- Process Call Command
- Block Write Command

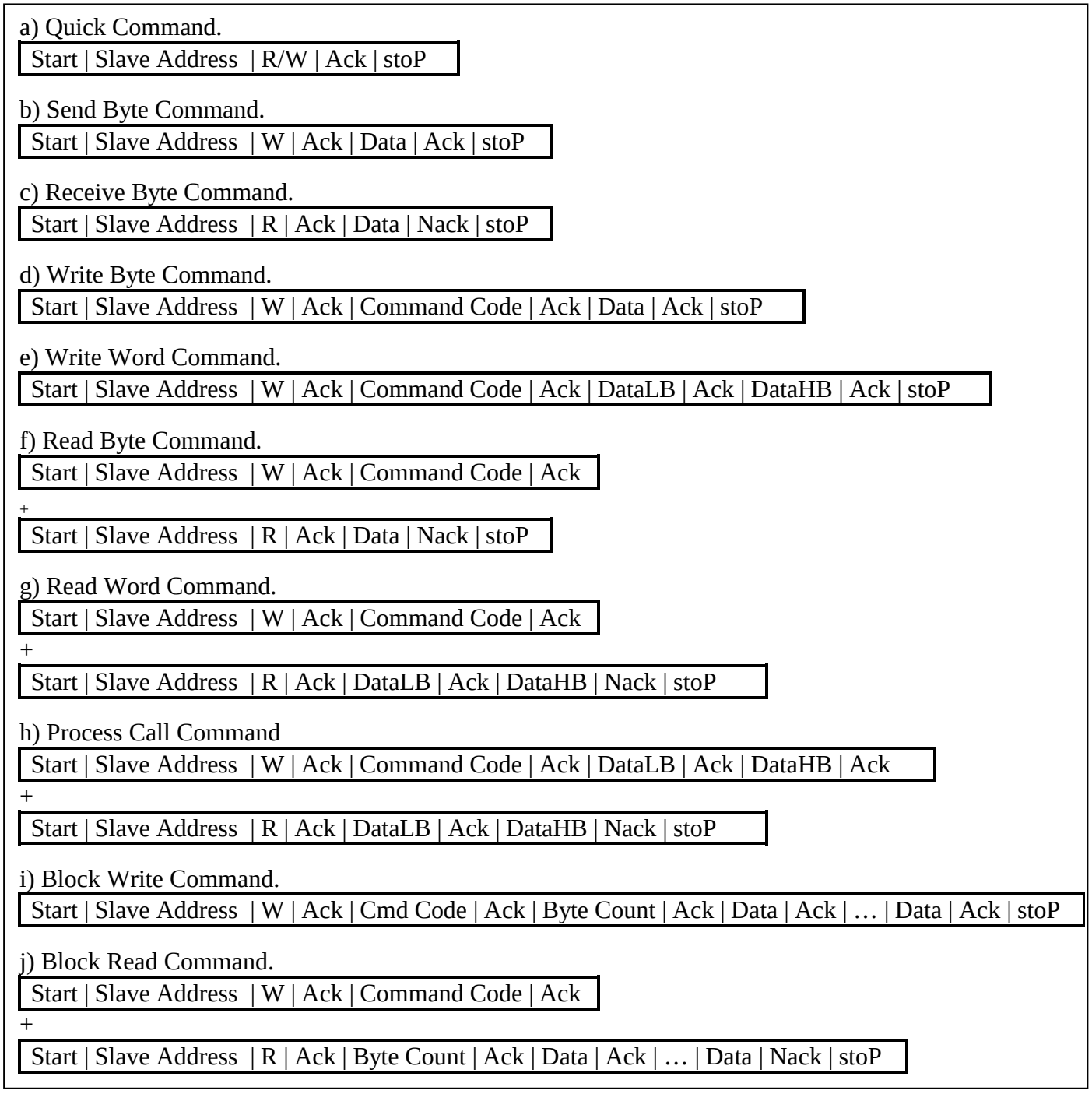

Figure 3 - Standard SMBus command protocols provide rules for communicating across the bus, and are available to all SBS devices.

These command protocols define the rules devices connected to the SMBus must follow, and provide a menu of commands to select from when implementing specific SBS device functions.

Quick commands are used to send 1 bit of data to a slave device. The value of the Read/Write bit (0 or 1) controls the state of a function in the addressed write-only slave. This command uses little bus bandwidth, and addresses the needs of simple slave devices. For example, this type of command could be used to enable or disable backlighting in an LCD controller.

Send Byte commands address slave devices that need to receive only a single byte of data. The 8-bit data byte can hold a value of 0 to 255. This data can be interpreted as the slave device sees fit. For example, a fan motor control could use this data to set motor speed. Or, an LCD backlight controller could use this data to set lamp intensity.

Receive Byte commands also involve a single data byte, but in this case the master is reading the data from the slave. The 8-bit data byte can hold a value of 0 to 255. The interpretation of this data by the master is slave device dependent. For example, the data could indicate the temperature of the Host CPU, or the status of access door interlocks.

The **Write Byte/Word** command is similar to the Send Byte command. However, this command involves a Command Code byte and 1 or 2 data bytes. The Command Code byte tells the receiving slave device how to interpret the data that follows. The 8-bit data byte can hold a value of 0 to 255. The 16-bit data word can hold a value of 0 to 65,535. For example, the message could be the Smart Battery telling the Smart Battery Charger about battery charging requirements. The Command Code byte would tell the receiving slave device, the Smart Battery charger in this case, that the data that follows is the voltage level the charger is to apply to the battery's terminals.

The **Read Byte/Word** command is similar to the Write Byte/Word command, but involves a two-step process. Here, the master must first write the Command Code to the slave device. This allows the master to tell the slave just what information the master requires. The master then, without generating a stop condition, sends a new message to the slave, reading 1 or 2 bytes of data. As with the Write Byte/Word command, the 8-bit data byte can hold a value of 0 to 255. The 16-bit data word can hold a value of 0 to 65,535. For example, this command could be used by the SMBus Host device to request battery pack temperature information from a Smart Battery. The Command Code byte would tell the slave device, the Smart Battery in this case, that the data to subsequently transmit should be the battery pack temperature.

The **Process Call** command is similar to a Write Word followed by Read Word command sequence but uses a repeated start and only a single Command Code for the entire sequence. Here, the master must first write the Command Code plus 2 data bytes (16-bits, low byte first) to the slave device. This allows the master to issue a command to the slave, and provides 2 bytes of data that can be used by the slave for internal computations. The master then, without generating a stop condition, sends a new message to the slave, reading 2 bytes of data (16-bits, low byte first). The Process Call command is typically used in more computational intensive situations.

The **Block Write** command is used to send a series of data bytes to a slave receiver device. The Command Code tells the slave receiver device how to interpret the remaining bytes in the message. The Byte Count, which has a valid range of 1 to 32, tells the slave receiver how many data bytes are to follow.

The **Block Read** command is used to read a series of data bytes from a slave transmitter. It also involves a two-step process. Here, the master must first write the Command Code to the slave device. This allows the master to tell the slave just what information the master requires. The master then, without generating a stop condition, sends a new message to the slave, reading the data. The first byte of which indicates how many data bytes are to follow.

SMBus slave devices may use any or all of the above command protocols. Supported command protocols are defined in the appropriate SBS component specification. SMBus Hosts should support all command protocols.

Although its initial use is to connect SBS devices, SMBus includes capabilities that will allow it to be used for connecting a wide variety of devices within electronic products. The protocols SMBus includes can meet most I^2C Bus communications requirements. They are worth considering even on non-SBS or SMBus related project.

SMBUS SYSTEM HOST

The SMBus System Host is the equipment powered by the Smart Battery. In a laptop PC, the System Host is the laptop's processor. In other products, the System Host might be an embedded microcontroller or microprocessor. Whatever form it takes, the System Host provides the interface between the user and the rest of the power management system. The computing power of the System Host can be used to directly or indirectly determine user power requirements. Its SMBus communication capability allows the System Host to:

- Interact with the Smart Battery to determine current and predicted power availability, and set battery mode and alarm levels.
- Set Smart Battery Charger charging current and voltage levels.
- Interact with the Smart Battery Selector to identify multiple battery availability.

The System Host can then use this information to establish a power budget for system devices that best meets user requirements.

SMART BATTERY

The Smart Battery consists of a battery pack with embedded electronics that can hold Smart Battery Data (see Table 1), measure battery operating parameters, and calculate and predict battery performance. It can also monitor alarm conditions, initiate and control battery charging algorithms, and communicate with other SMBus devices.

Placing SBS compatible electronics within the battery pack opens the door to a wide variety of battery chemistry independent power management schemes for extending product run time. Additionally, SBS-based equipment users and power management systems can access complete and accurate information even if the battery is changed.

The Smart Battery Data provides information that enables the equipment user to know how much longer a product will operate. It can also guide the power management system built into the equipment in selecting the best algorithm to extend battery life. This data includes member elements that:

- Indicate present battery operating parameters such as battery pack temperature, and terminal voltage and current.
- Predict battery operation such as remaining capacity and run time to empty.
- Control battery operation such as remaining capacity and remaining time alarm levels, and operating mode controls.
- Identify the battery, its manufacturer, and chemistry.

Although all SBS data within a Smart Battery is readable from other SBS devices, only a few alarm, mode, and rate parameters are writable from the SMBus. Most parameters, such as temperature, voltage, and current are measured or calculated within the battery pack itself. Other data elements, such as manufacturing date, serial number, and device chemistry are programmed into the battery pack electronics during manufacturing.

SMART BATTERY CHARGER

The Smart Battery Charger provides a source of voltage and current for charging a Smart Battery. The charger circuit communicates with the Smart Battery over the SMBus. The charger can become a bus master and actively poll the battery for current and voltage requirements. A passive charger could act as a slave only device, and receive charging information from the battery, or System Host, if the battery is unable to provide this information directly. The charger may also receive notification of critical battery events such as over-charging, over-voltage, and over-temperature conditions.

With SBS compatible electronics now within the battery pack, the Smart Battery Charger can be reduced to a slave device that supplies charging voltage and current to the battery independent of battery chemistry.

SMART BATTERY SELECTOR

The Smart Battery Selector meets the need for multiple battery packs within a system. To support this function, the Selector maintains configuration data, provides system power switching, battery charge switching, and has SMBus communication capabilities.

Selector configuration data identifies:

- Which battery is connected to the SMBus System Host.
- What is the current system power source.
- Which battery is connected to the charger.
- What batteries are present.

Smart Battery Data Smart Battery Data Specification, Revision 1.0	
Name	Description
ManufacturerAccess	Content determined by Battery Manufacturer.
RemainCapacityAlarm	When the RemainingCapacity falls below this value, battery sends AlarmWarning to SMBus Host with the REMAINING_CAPACITY_ALARM bit set.
RemainTimeAlarm	When the AverageTimeToEmpty falls below this value, battery sends AlarmWarning to SMBus Host with the REMAINING_TIME_ALARM bit set.
BatteryMode	Controls battery operating modes and reporting capabilities.
AtRate	Sets charge or discharge rate for AtRateTimeToFull, AtRateTimeToEmpty, and AtRateOK functions. Specified in milliamp if BatteryMode.CAPACITY_MODE bit = 0, else 10milliwatts.
AtRateTimeToFull	Predicted remaining time to full charge at AtRate value.
AtRateTimeToEmpty	Predicted remaining operating time at AtRate value.
AtRateOK	Indicates if battery can deliver the current AtRate value for 10 seconds.
Temperature	Cell-pack's internal temperature in degrees Kelvin.
Voltage	Cell-pack voltage in millivolts.
Current	Current being supplied (or accepted) through battery's terminals in milliamps.
AverageCurrent	One-minute rolling average current being supplied (or accepted) through battery's terminals in milliamps.
MaxError	Expected margin of error (%) in the state of charge calculations.
RelStateOfCharge	Predicted remaining battery capacity as a percentage of FullChargeCapacity.
AbsStateOfCharge	Predicted remaining battery capacity as a percentage of DesignCapacity.
RemainingCapacity	Predicted remaining battery capacity in milliamp Hours if BatteryMode.CAPACITY_MODE bit = 0, else in 10milliwatt Hours.
FullChargeCapacity	Predicted pack capacity when fully charged in milliamp Hours if BatteryMode.CAPACITY_MODE bit = 0, else in 10milliwatt Hours.
RunTimeToEmpty	Predicted remaining battery life at the present rate of discharge in minutes.
AveTimeToEmpty	One-minute rolling average of the predicted remaining battery life in minutes.
AveTimeToFull	One-minute rolling average of the predicted remaining time to full charge in minutes.
ChargingCurrent	Desired charging rate in milliamps.
ChargingVoltage	Desired charging voltage in millivolts.
BatteryStatus	Battery status word (flags).
CycleCount	Number of charge/discharge cycles the battery has experienced.
DesignCapacity	Theoretical capacity of a new pack in milliamp Hours if BatteryMode.CAPACITY_MODE bit = 0, else in 10milliwatt Hours.
DesignVoltage	Theoretical voltage of a new pack in millivolts.
SpecificationInfo	Smart Battery specification version supported, voltage and current scaling information.
ManufacturerDate	Date the cell pack was manufactured.
SerialNumber	Battery serial number.
ManufacturerName	Battery's manufacturer's name.
DeviceName	Battery's name.
DeviceChemistry	Battery's chemistry.
ManufacturerData	Content determined by Battery Manufacturer.

Table 1 - Smart Battery Data describes the actual and predicted operation of an SBS Smart Battery. By being located within the battery itself, this information is accurate even if the battery is changed.

This data allows the System Host to determine when:

- a Smart Battery has been added or removed.
- AC power is connected or disconnected.
- the Selector switches which battery is powering the system.

A Selector may act as a slave-only device, responding to polls from the System Host, as a master device, initiating commands to the System Host, or a combination of both.

POWERFUL DEMANDS

The Smart Battery System is destined to move beyond portable computers and cell phones, and into a widening range of portable electronic products. It offers extended product run-times, the ability to accurately predict product performance, and improved charging safety for a wide range of battery-powered equipment. These are product features that are highly visible to users, and ones they will surely demand.

Portable medical equipment is one of the product categories that will soon benefit from SBS technology. Others will certainly follow. If you are responsible for portable product design and development for your company, maybe you should consider this technology, too.

Hopefully, the next time you see an emergency team at work, they will know that their lifesaving equipment has the power to handle the job.

Biography

Ed Thompson is the president of Micro Computer Control Corporation. For the last six years he has concentrated on I^2C/SMBus applications and development tools. You can reach him at Ed.Thompson@mcc-us.com.