

## Table of Contents

Section 1 : HESC Overview.....	3
Section 2 : HESC-SER SerBus Communications .....	4
A. Description: .....	4
B. Master Mode: .....	5
C. Slave Mode: .....	6
Section 3 : HESC104 PC/104 Bus Communications .....	7
A. Description: .....	7
B. Master Mode: .....	8
C. Slave Mode: .....	9
Section 4 : HESC I <sup>2</sup> C/SMBus Bus Communications.....	10
A. Description: .....	10
B. Master Mode: .....	11
C. Slave Mode: .....	12
Section 5 : SerBus, SMBus, and PC/104 Command Functions.....	13
A. List of Command Functions .....	13
B. HESC Function Command Details .....	14
1. BattTempCmd() 0x08.....	14
2. BattVCmd() 0x09 .....	14
3. BattICmd() 0x0A .....	14
4. BattIavgCmd() 0x0B.....	15
5. BattRemCapCmd() 0x0F .....	15
6. ChargerSpecInfoCmd() 0x11 .....	15
7. ChargerModeCmd() 0x12 .....	16
8. ChargerStatusCmd() 0x13 .....	16
9. ChargingCurrentCmd() 0x14 .....	18
10. ChargingVoltageCmd() 0x15.....	19
11. BatteryStatusCmd() 0x16 .....	19
12. GetVersionCmd() (0x3E) .....	20
13. SerialNumber() (0x3F).....	20
14. GetI2CTempCmd0() (0x80) to GetI2CTempCmd15() (0x8F).....	20
15. TempThCmd() (0x90) .....	20
16. MainICmd() (0x92) .....	21
17. InputPwrCmd() (0x93).....	21
18. BattPwrCmd() (0x94) .....	21
19. ChCycleCmd() (0x95) .....	22
20. ChTermLastCmd() (0x96).....	22
21. ShutDownCmd() (0x97) .....	22
22. PowerSupplyStatusCmd() (0x98).....	23
23. SDSUCauseCmd() (0x99) .....	24
24. ActiveEEcmd() (0xA0).....	25
25. EEPROMCmd () (0xA1) .....	25
Section 6 : EEPROM Setpoints and Configuration Variables .....	26
A. List of EEPROM Setpoints and Configuration Variable .....	26
B. SerBus EEPROM Details .....	28
1. ChTerm - EEPROM Locations 0x00, 0x20, 0x40, 0x60: .....	28
2. Spare - EEPROM Locations 0x02 0x22 0x42 0x62.....	28
3. Spare - EEPROM Locations 0x04 0x24 0x44 0x64.....	28
4. BattVmaxDef - EEPROM Locations 0x06, 0x26, 0x46, 0x66: .....	28

## HESC - High Efficiency & Smart Charging Firmware Manual

5.	BattVmaxTimeDef - EEprom Locations 0x08, 0x28, 0x48, 0x68: .....	29
6.	BattVdeltaDef - EEprom Locations 0x0A, 0x2A, 0x4A, 0x6A:.....	29
7.	TimeMaxDef - EEprom Locations 0x0C, 0x2C, 0x4C, 0x6C: .....	29
8.	BattIminDef - EEprom Locations 0x0E, 0x2E, 0x4E, 0x6E:.....	29
9.	BattImaxDef - EEprom Locations 0x10, 0x30, 0x50, 0x70: .....	29
17.	ChFlags - EEprom Locations 0x80;.....	30
18.	BattTempMinDef - EEprom Locations 0x8A: .....	30
19.	BattTempMaxDef - EEprom Locations 8C: .....	30
20.	BattVminDef - EEprom Locations 0x8E: .....	30
21.	BattSelDef - EEprom Locations 0x9E: .....	30
Section 7 : Temperature Sensor, TS-IC and Controller, TC-IC .....		31
Section 8 : Start-up and Shut-down settings, delays, and functions.....		32
Section 9 : Battery Charging Algorithms and Strategies.....		33
A. Overview.....		33
B. Sealed Lead Acid, SLA .....		33
1.	Single Stage Charging .....	33
2.	Dual Stage Charging .....	33
Table 1, List of HESC Command Functions.....		13
Table 2, ChargerSpecInfoCmd bit map .....		15
Table 3, ChargerModeCmd bit map .....		16
Table 4, ChargerStatusCmd bit map .....		17
Table 5, ChTermLastCmd bit map.....		22
Table 6, PowerSupplyStatusCmd bit map .....		23
Table 7, SDSUCauseCmd bit map .....		24
Table 8, HESC Setpoints and Configuration Variables .....		26
Table 9, ChTerm bit map .....		28
Table 10, ChFlags bit map.....		30
Table 11, Temperature Sensor and Temperature Controller bit map.....		31
Table 12, SLA Single Stage Charging Set-up .....		33
Table 13, SLA Two Stage Charging Set-up .....		34

## Section 1 : HESC Overview

The High Efficiency & Smart Charging (HESC) product line includes the HESC-SER and HESC104. Command, control, monitoring and datalogging via the SMBus are the same for both the HESC-SER and HESC104. However, the HESC-SER also has a RS232 serial port and the HESC104 has a PC/104 bus interface. The serial port and PC/104 bus interface are in most instances connected only to the Host CPU, however other devices may also make use of these ports. For example, it is possible that a battery pack could be built with an RS232 serial port and it could be connected to the HESC-SER serial port.

Throughout this manual, use of the term HESC refers to any of the HESC products, while use of the name HESC-SER or HESC104 refers strictly to a particular product. From the command function point of view, there are few differences between the HESC-SER and the HESC104 even they have different Host interfaces (the SMBus is usually thought of as a battery interface, but it may also interface with a Host CPU or a microcontroller). Whenever there are any differences in the commands functions for the different HESC models, they will be noted in the command function description.

The HESC can be set-up to provide up to four stages of charging for standard battery packs. Changing from one cycle to the next is accomplished by setting the charge termination flags and values. When one of the charge termination methods is satisfied, the next charging stage values are retrieved from the EEPROM. If the current cycle is the last cycle enabled, then charging will be terminated. Additionally, the Host can directly command the HESC to change charging cycles.

## Section 2 : HESC-SER SerBus Communications

### **A. Description:**

The HESC-SER communicates with the Host through the asynchronous serial bus. Commands and data are sent and received using a full duplex 8 bit, no parity, 9600 baud, and one stop bit format. The HESC-SER uses two modes, "master", and "slave" mode. The "master" mode for the HESC-SER is defined as communication initiated by the HESC-SER. When the Host initiates the communications, the HESC-SER operates in "slave" mode.

The master mode for the HESC-SER is used to alert the Host of a change in status or an alarm in the HESC. The HESC-SER alerts the Host by sending the ChargerStatus() value. This includes when main power is applied or removed, battery pack inserted/removed, battery fully charged or fully discharge, shutdown activated or de-activated, or temperature alarm. When the HESC-SER alerts the Host (or other device), the HESC-SER places its address 0x12 in the command byte. The HESC-SER alert communications format would then appear as 0x10, 0x12, datalow, datahigh. This allows the Host to identify the source of the data, and to interpret it correctly. For instance, if a battery pack was also on the SerBus and using it's Master mode to talk to the Host, it would place the value 0x16 (battery address value) into the command byte so the Host would know the battery sent the data.

- Table 1 lists the commands the HESC supports.

To ensure reliable communication, an acknowledge byte is returned after each address, command or data byte transmitted. The receiving device (HESC-SER or Host) must acknowledge receipt of each byte, unless the bus timer is turned off. If an acknowledge byte doesn't match the acknowledge number expected then a "collision" is deemed to have occurred. The transaction is aborted immediately and the result byte set accordingly.

An "enhanced" communication mode is available by adding an optional checksum value. If the HESC-SER receives a Read command from the Host with the acknowledge set to 0x03 after sending [databyteR high], it will switch to enhanced mode for all future communications. Communication will return to non-checksum mode when the Host CPU sends an acknowledge 0xFF after [databyteR high].

## B. Master Mode:

1. Commands send from the HESC-SER and data sent to or received from the to Host CPU over the SerBus. The transaction is invalid and commands/data are not to be used until:

- the final acknowledge 0xFF is received
- the checksum matches the transmitted data (checksum is optional, but if sent must be used)

2. Note: Read and Write is defined as the action the command places on the HESC-SER RAM and EEPROM. Therefore, Read and Write have the same meaning for Master and Slave modes:

3a. Read command without checksum acknowledge and with bus timer enabled.

HESC-SER: <addressR>      <command>      <databyteR low>      <databyteR high>  
 HOST:                      [0x00]                      [0x01]                      [0x02]                      [0xFF]

3b. Read command with checksum acknowledge and with bus timer enabled.

HESC-SER: <addressR>      <command>      <databyteR low>      <databyteR high>      <checksum>  
 HOST:                      [0x00]                      [0x01]                      [0x02]                      [0x03]                      [0xFF]

3c. Read command without checksum acknowledge and without bus timer enabled.

HESC-SER: <addressR> <command> <databyteR low > <databyteR high>  
 HOST: {no response from host}

4a. Write command without checksum acknowledge and with bus timer enabled.

HESC-SER: <addressW>      <command>                      <0x02>                      <0xFF>  
 HOST:                      [0x00]                      [databyteW low]                      [databyteW high]

4b. Write command with checksum acknowledge and with bus timer enabled.

HESC-SER: <addressW>      <command>                      <0x02>                      <0x03>                      <0xFF>  
 HOST:                      [0x00]                      [databyteW low]                      [databyteW high]                      [checksum]

4c. Write command without checksum acknowledge and without bus timer enabled.

HESC-SER: <addressW> <command>                      <0x02>\*\*                      <0xFF>\*\*  
 HOST:    [databyteW low] [databyteW high]  
 \*\*{HOST does not need to wait for Ack from HESC-SER}

where "host" addressR = 0001 000 + 0 (R/W bit) = 0x10

"host" addressW = 0001 000 + 1 (R/W bit) = 0x11

The checksum is a two digit hexadecimal checksum that is the two's complement of the sum of all preceding bytes. For example the data <0x10> <0x12> <0xC0> <0x03> has the checksum 0x1B.

### C. Slave Mode:

1. Commands received from the Host and data sent to or received from the to Host CPU over the SerBus. The transaction is invalid and commands/data are not to be used until:

- the final acknowledge 0xFF is received
- the checksum matches the transmitted data (checksum is optional, but if sent must be used)

2. Note: Read and Write is defined as the action the command places on the HESC-SER RAM and EEPROM. Therefore, Read and Write have the same meaning for Master and Slave modes:

3a. Write command without checksum acknowledge.

```
HOST: <addressW>    <command>    <databyteW low >    <databyteW high >
HESC-SER:    [0x00]          [0x01]          [0x02]          [0xFF]
```

3b. Write command with checksum acknowledge.

```
HOST: <addressW>    <command>    <databyteW low >    <databyteW high >    <checksum>
HESC-SER:    [0x00]          [0x01]          [0x02]          [0x03]          [0xFF]
```

3c. Write command without checksum acknowledge and without bus timer enabled.

```
HOST: <addressW> <command> <databyteW low > <databyteW high >
HESC-SER:    [0x00]**    [0x01]**    [0x02]**    [0xFF]**
```

4a. Read command without checksum acknowledge.

```
HOST: <addressR>    <command>          <0x02>          <0xFF>
HESC-SER:    [0x00]          [databyteR low]    [databyteR high]
```

4b. Read command with checksum acknowledge.

```
HOST: <addressR>    <command>          <0x02>          <0x03>          <0xFF>
HESC-SER:    [0x00]          [databyteR low]    [databyteR high]    [checksum]
```

4c. Read command without checksum acknowledge and without bus timer enabled.

```
HOST: <addressR> <command>
HESC-SER:    [0x00]**    [databyteR low] [databyteR high]
    **{HOST does not need to wait for Ack from HESC-SER}
```

where "host" addressW = 0001 001 + 0 (R/W bit) = 0x12  
 "host" addressR = 0001 001 + 1 (R/W bit) = 0x13

The checksum is a two digit hexadecimal checksum that is the two's complement of the sum of all preceding bytes. For example the data <0x10> <0x12> <0xC0> <0x03> has the checksum 0x1B.

## Section 3 : HESC104 PC/104 Bus Communications

### **A. Description:**

The HESC104 communicates with the Host through the PC/104 bus. Commands and data are sent and received using a 8 bit, I/O memory mapped I/O address. The address lines A0 to A9, & AEN are decoded to provide four addresses that are jumper selectable. An I/O write to the decoded address will "strobe" the data into the HESC, and an I/O read will read the data from the HESC104. Whenever the HESC104 has data in it's output port that requires the Host CPU to perform an I/O read, it generates a PC/104 bus interrupt (IRQ5 or IRQ7, see section on setting jumpers). The PC/104 bus interrupt is removed after an I/O read from the HESC104.

The HESC104 uses two modes, "master", and "slave" mode. The "master" mode for the HESC104 is defined as communication initiated by the HESC104. When the Host initiates the communications, the HESC104 operates in "slave" mode.

The master mode for the HESC104 is used to alert the Host of a change in status or an alarm in the HESC. The HESC104 alerts the Host by sending the ChargerStatus() value. This includes when main power is applied or removed, battery pack inserted/removed, battery fully charged or fully discharge, shutdown activated or de-activated, or temperature alarm. When the HESC104 alerts the Host, the HESC104 places its address 0x12 in the command byte. The HESC104 alert communications format would then appear as 0x10, 0x12, datalow, datahigh. This is to maintain compatible with the SMBus and SerBus formats.

- Table 1 lists the commands the HESC supports.

To ensure reliable communication, an acknowledge byte is returned after each address, command or data byte transmitted. The receiving device (HESC104 or Host) must acknowledge receipt of each byte. This is true even if the bus timer is turned off. (*The HESC104 and HESC-SER differ in that if the HESC-SER bus timer is off an acknowledge byte is not issued.*) If an acknowledge byte doesn't match the acknowledge number expected then a "collision" is deemed to have occurred. The transaction is aborted immediately and the result byte set accordingly.

An "enhanced" communication mode is available by adding an optional checksum value. If the HESC104 receives a Read command from the Host with the acknowledge set to 0x03 after sending [databyteR high], it will switch to enhanced mode for all future communications. Communication will return to non-checksum mode when the Host CPU sends an acknowledge 0xFF after [databyteR high].

## **B. Master Mode:**

1. Commands send from the HESC104 and data sent to or received from the to Host CPU over the PC/104 Bus. The transaction is invalid and commands/data are not to be used until:

- the final acknowledge 0xFF is received
- the checksum matches the transmitted data (checksum is optional, but if sent must be used)

2. Note: Read and Write is defined as the action the command places on the HESC104's RAM and EEprom. Therefore, Read and Write have the same meaning for Master and Slave modes:

3a. Read command without checksum acknowledge and with bus timer enabled.

HESC104: <addressR>      <command>      <databyteR low>      <databyteR high>  
 HOST:                      [0x00]                      [0x01]                      [0x02]                      [0xFF]

3b. Read command with checksum acknowledge and with bus timer enabled.

HESC104: <addressR>      <command>      <databyteR low>      <databyteR high>      <checksum>  
 HOST:                      [0x00]                      [0x01]                      [0x02]                      [0x03]                      [0xFF]

4a. Write command without checksum acknowledge and with bus timer enabled.

HESC104: <addressW>      <command>                      <0x02>                      <0xFF>  
 HOST:                      [0x00]                      [databyteW low]                      [databyteW high]

4b. Write command with checksum acknowledge and with bus timer enabled.

HESC104: <addressW>      <command>                      <0x02>                      <0x03>                      <0xFF>  
 HOST:                      [0x00]                      [databyteW low]                      [databyteW high]                      [checksum]

where "host" addressR = 0001 000 + 0 (R/W bit) = 0x10

"host" addressW = 0001 000 + 1 (R/W bit) = 0x11

The checksum is a two digit hexadecimal checksum that is the two's complement of the sum of all preceding bytes. For example the data <0x10> <0x12> <0xC0> <0x03> has the checksum 0x1B.

### C. Slave Mode:

1. Commands received from the Host and data sent to or received from the to Host CPU over the PC/104 bus. The transaction is invalid and commands/data are not to be used until:

- the final acknowledge 0xFF is received
- the checksum matches the transmitted data (checksum is optional, but if sent must be used)

2. Note: Read and Write is defined as the action the command places on the HESC104's RAM and EEPROM. Therefore, Read and Write have the same meaning for Master and Slave modes:

3a. Write command without checksum acknowledge.

```
HOST: <addressW>    <command>    <databyteW low >    <databyteW high >
HESC104:          [0x00]          [0x01]          [0x02]          [0xFF]
```

3b. Write command with checksum acknowledge.

```
HOST: <addressW>    <command>    <databyteW low >    <databyteW high >    <checksum>
HESC104:          [0x00]          [0x01]          [0x02]          [0x03]          [0xFF]
```

4a. Read command without checksum acknowledge.

```
HOST: <addressR>    <command>          <0x02>          <0xFF>
HESC104:          [0x00]          [databyteR low]    [databyteR high]
```

4b. Read command with checksum acknowledge.

```
HOST: <addressR>    <command>          <0x02>          <0x03>          <0xFF>
HESC104:          [0x00]          [databyteR low]    [databyteR high]    [checksum]
```

where "host" addressW = 0001 001 + 0 (R/W bit) = 0x12

"host" addressR = 0001 001 + 1 (R/W bit) = 0x13

The checksum is a two digit hexadecimal checksum that is the two's complement of the sum of all preceding bytes. For example the data <0x10> <0x12> <0xC0> <0x03> has the checksum 0x1B.

## Section 4 : HESC I<sup>2</sup>C/SMBus Bus Communications

### A. Description:

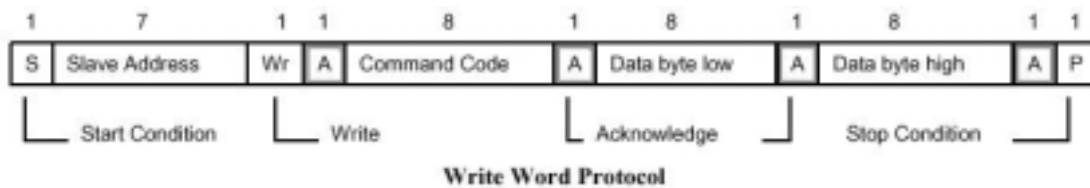
The HESC can communicate with System Management Bus (SMBus) batteries, I<sup>2</sup>C digital temperature sensors, and Hosts and microprocessors through the I<sup>2</sup>C/SMBus. The SMBus is a two-wire interface through which the HESC can communicate to I<sup>2</sup>C/SMBus devices. The HESC supports I<sup>2</sup>C/SMBus multi-master bus capability, meaning that other devices capable of controlling the bus can be connected to it. The HESC transfers data by one I<sup>2</sup>C/SMBus device acting as a master, and another I<sup>2</sup>C/SMBus device acting as a slave (with one of the devices being the HESC). A master device initiates a bus transfer and provides the clock signals (SCL). A slave device can receive data (SDA) provided by the master or it can provide data to the master. Since more than one device may attempt to take control of the bus as a master, I<sup>2</sup>C/SMBus provides an arbitration mechanism, relying on the wired-AND connection of all I<sup>2</sup>C/SMBus interfaces to the I<sup>2</sup>C/SMBus.

**NOTE:** Care should be taken in the design of both the input and output stages of SMBus devices, in order not to load the bus when their power plane is turned off.

The HESC uses the master to alert the Host of a change in status or of alarm in the HESC. The HESC alerts the Host by sending the ChargerStatus() value. This occurs when main power is applied or removed, battery pack inserted/removed, battery fully charged or fully discharge, shutdown activated or de-activated, or temperature alarm. When the HESC alerts the Host, the HESC places its address 0x12 in the command byte. The HESC alert communications format would then appear as 0x10, 0x12, datalow, datahigh.

- Table 1 lists the commands the HESC I<sup>2</sup>C/SMBus supports.

To ensure reliable communication, an acknowledge bit is returned after each address, command or data byte transmitted. The receiving device must acknowledge receipt of each byte. If an acknowledge bit is not received then the transfer is aborted immediately and the result byte set accordingly.



***B. Master Mode:***

***C. Slave Mode:***

## Section 5 : SerBus, SMBus, and PC/104 Command Functions

### A. List of Command Functions

1. The SerBus and PC/104 Bus command functions are similar to many of the SMBus command functions. Where possible, the functions are the same as the SMBus functions. The following table list the HESC command functions, access, units and range of the data.

2. There are two types of command functions, RAM and EEPROM. RAM variables are used to actively monitor and control the HESC. However, RAM is volatile memory and loses its contents on power loss. The EEPROM is used to store setpoints and configuration for the HESC. Separate functions allow easy access to the RAM variables.

3. Since setting up the HESC is only done infrequently, only two commands are provided. Function ActiveEEcmd() sets the location for the next EEPROM read or write, and if the location should auto increment. Function EEPROMCmd() reads or writes the EEPROM location as set by ActiveEEcmd().

Table 1, List of HESC Command Functions

Code	Command Function	SerBus	SMBus	PC/104	Access	Units	Range	Description
0x08	BattTempCmd()	✓	✓	✓	R	0.1K *	0 to 6553.5	Read battery temperature
0x09	BattVCmd()	✓	✓	✓	R	mV	0 to 65535	Read battery voltage
0x0A	BattICmd()	✓	✓	✓	R	mA	0 to 32767 0 to -32768	Read battery current
0x0B	BattIavgCmd()	✓	✓	✓	R	mA	0 to 32767 0 to -32768	Read 1 minute rolling average battery current
0x0F	BattRemCapCmd()	✓	✓	✓	R	mAh or 10mWh **	0 to 65535	Read remaining battery capacity
0x11	ChargerSpecInfoCmd()	✓	✓	✓	R	Bit flags	N/A	Read SMBus specification compatibility
0x12	ChargerModeCmd()	✓	✓	✓	W	Bit flags	N/A	Set Charger Mode
0x13	ChargerStatusCmd()	✓	✓	✓	R	Bit flags	N/A	Read charger status
0x14	ChargingCurrentCmd()	✓	✓	✓	R/W	mV	0 to 65535	Read battery charging voltage setpoint
0x15	ChargingVoltageCmd()	✓	✓	✓	R/W	mA	0 to 65535	Read battery charging current setpoint
0x16	AlarmWarningCmd()	✓	✓	✓	R/W	Bit flags	N/A	Alarm Notification
0x3E	GetVersionCmd()	✓	✓1	✓	R	High/Low byte***	N/A	Read charger firmware revision
0x3F	GetChSerNumCmd()	✓	✓1	✓	R	8 byte	N/A	Read charger serial number
0x80	GetI2Ctemp0()	✓	✓2	✓	R	0.1K *	0 to 6553.5	Read I2C temperature
	to							
0x8F	GetI2Ctemp15()	✓	✓2	✓	R	0.1K *	0 to 6553.5	Read I2C temperature
0x90	TempThCmd()	✓	✓2	✓	R	Word	0 to 65535	Read TH (thermistor) value
0x91	MainVCmd()	✓	✓2	✓	R	mV	0 to 65535	Read main input voltage
0x92	MainICmd()	✓	✓2	✓	R	mA	0 to 65535	Read input current
0x93	InputPwrCmd()	✓	✓2	✓	R	10mW	0 to 65535	Read input power
0x94	BattPwrCmd()	✓	✓2	✓	R	10mW	0 to 65535	Read charging power
0x95	ChCycleCmd()	✓	✓2	✓	R/W	integer	0 to 3	Read/Set charging cycle
0x96	ChTermLast()	✓	✓2	✓	R	Bitflags	N/A	Read last termination method
0x97	ShutDownCmd()	✓	✓2	✓	R/W	sec	0 to 65535	A write will shut down the power supply in X seconds. A read will return time left until shutdown. Use command PowerSupplyStatusCmd() to determine if a shutdown is in progress.
0x98	PowerSupplyStatusCmd()	✓	✓2	✓	R/W	Bitflags	N/A	Read/Write power supply flags
0x99	SDSUCauseCmd()	✓	✓2	✓	R	Bitflags	N/A	Read cause of SD and SU requests
0x9A	I2CLowAlarmsCmd	✓	✓2	✓	R	Bitflags	N/A	
0x9B	I2CHighAlarmsCmd	✓	✓2	✓	R	Bitflags	N/A	
0xA0	ActiveEEcmd()	✓	✓2	✓	R/W	byte.byte	****	Read/Set active EEPROM location and auto increment
0xA1	EEPROMCmd()	✓	✓2	✓	R/W	Word	0 to 65535	Read/Write value to active EEPROM location

✓1 SMBus Optional Manufacturer Functions

✓2 Function not defined by SMBus standards

\* 0 degrees Kelvin = -273.2C

\*\* The BattRemCapCmd() capacity value is expressed in either current (mAh at a C/5 discharge rate) or power (10mWh at a P/5 discharge rate) depending on the setting of the capacity mode bit.

\*\*\* High byte = major version, Low byte = minor version

\*\*\*\* Low byte = location 0 to 127, High byte: no auto increment = 0; auto increment = 1

## **B. HESC Function Command Details**

### **1. BattTempCmd() 0x08**

**Description:**

Returns the batteries internal temperature (°K). The actual operational temperature range will typically be in the range of -20°C to +75°C.

**Purpose:**

The BattTempCmd() function provides accurate cell temperatures for use by HESC-SER and Host management system. The HESC is able to use the temperature as a safety check and the Host may use the temperature in it's thermal management. (Kelvin units are used to facilitate simple unsigned handling of temperature information and to permit easy conversion to other units.)

**Protocol:** Read Word

**Output:** unsigned int -- cell temperature in tenth degree Kelvin increments

Units: 0.1°K

Range: 0 to +6553.5°K

### **2. BattVCmd() 0x09**

**Description:**

Returns the battery voltage (mV).

**Purpose:**

The BattVCmd() function provides the Host power management system with an accurate battery voltage. The Host management system can use this voltage, along with battery current information to help enable intelligent, adaptive power management systems.

**Protocol:** Read Word

**Output:** unsigned int -- battery terminal voltage in milli-volts

Units: mV

Range: 0 to 65,535 mV

### **3. BattICmd() 0x0A**

**Description:**

Returns the current being supplied (or accepted) through the battery (mA).

**Purpose:**

The BattICmd() function provides a snapshot for the Host power management system of the current flowing into or out of the battery. This information will be of particular use in the Host power management system because it can characterize individual devices and "tune" their operation to actual system power behavior.

**Protocol:** Read Word

**Output:** signed int -- charge/discharge rate in mA increments - positive for charge, negative for discharge

Units: mA

Range: 0 to 32,767 mA for charge or

0 to -32,768 mA for discharge

#### 4. BattlavgCmd() 0x0B

**Description:**

Returns a one-minute rolling average based on the current being supplied (or accepted) through the battery(mA).

**Purpose:**

The BattlavgCmd() function provides the average current flowing into or out of the battery for the Host power management system.

**Protocol:** Read Word

**Output:** signed int -- charge/discharge rate in mA increments - positive for charge, negative for discharge.

Units: mA

Range: 0 to 32,767 mA for charge or  
0 to -32,768 mA for discharge

#### 5. BattRemCapCmd() 0x0F

**Description:**

Returns the predicted remaining battery capacity. The BattRemCapCmd() capacity value is expressed in either current (mAh at a C/5 discharge rate) or power (10mWh at a P/5 discharge rate) depending on the setting of the capacity\_mode bit.

**Purpose:**

The BattRemCapCmd() function returns the battery's remaining capacity in absolute terms but relative to a specific discharge rate.

**Protocol:** Read Word

**Output:** unsigned int -- remaining charge in mAh or 10mWh

#### 6. ChargerSpecInfoCmd() 0x11

**Description:**

The Host uses this command to read the charger's extended status bits.

**Purpose:**

Allows the Host to determine the specification revision the charger supports as well as other extended status information.

**Protocol:** Read Word

**Output:** unsigned int - bit mapped - see below

Units: not applicable

Range: not applicable

Table 2, ChargerSpecInfoCmd bit map

Field	Bits Used	Format	Allowable Values
CHARGER_SPEC	0...3	Bit flag	The ChargerSpecInfoCmd() reports the version of the Smart Battery Charger specification the HESC-SER charger supports. All other values reserved. 0001 – Version 1.0 0010 – Version 1.1 0011 – Version 1.1
SELECTOR_SUPPORT	4	Bit flag	0 - HESC-SER does not support the optional Smart Battery Selector commands
Reserved	5...15	Bit flag	These bits are reserved and will return zero.

## 7. ChargerModeCmd() 0x12

**Description:**

The Host uses this command to set the various charger modes. The default values are set to allow a Smart Battery and the Smart Charger to work in concert without requiring a host.

**Purpose:**

Allows the System Host to configure the charger and change the default modes. This is a write only function, but the value of the "mode" bits: INHIBIT\_CHARGE and ENABLE\_POLLING may be determined using the ChargerStatusCmd() function.

**Protocol:** Write Word

**Input:** unsigned int - bit mapped - see below

Units: not applicable

Range: not applicable

Table 3, ChargerModeCmd bit map

Field	Bits Used	Format	Allowable Values
INHIBIT_CHARGE	0	bit flag	0 - enable charging (power-on default) 1 - inhibit charging
ENABLE_POLLING	1	bit flag	0 - disable polling (power-on default for Level 2 chargers) 1 - enable polling (power-on default for Level 3 Smart Battery Chargers)
POR_RESET	2	bit flag	0 - mode unchanged (default) 1 - set charger to power-on defaults
RESET_TO_ZERO	3	bit flag	0 - charging values unchanged (default) 1 - set charging values to zero
Reserved	4...15		These bits are reserved and will return zero.

The INHIBIT\_CHARGE bit allows charging to be inhibited without changing the ChargingCurrentCmd() and ChargingVoltageCmd() values. Only the Host may set this bit while a battery is charging to inhibit charge. The Smart Battery is not allowed to write to this bit. The charging may be resumed by clearing the bit. This bit is automatically cleared when power is re-applied or when a battery is re-inserted.

The ENABLE\_POLLING bit enables the polling feature of the HESC in Level 3 Smart SMBus mode. This bit is set at power on for the HESC.

The POR\_RESET bit sets the Smart Battery Charger to its power-on default conditions.

The RESET\_TO\_ZERO bit sets the ChargingCurrentCmd() and ChargingVoltageCmd() values to zero. This function ALWAYS clears the ChargingVoltageCmd() and ChargingCurrentCmd() values to zero even if the INHIBIT\_CHARGE bit is set.

The ChargerModeCmd() bits are defined as write only. Appropriate actions will take place when writing these command bits, however reading will return undefined values.

## 8. ChargerStatusCmd() 0x13

**Description:**

The Host uses this command to read the charger's status bits.

**Purpose:**

Allows the Host to determine the status and level of the charger.

**Protocol:** Read Word

**Output:** unsigned int - bit mapped - see below

Units: not applicable

Range: not applicable

Table 4, ChargerStatusCmd bit map

Field	Bits Used	Format	Allowable Values
CHARGE_INHIBITED	0	bit flag	0 - charger is enabled 1 - charger is inhibited
POLLING_ENABLED	1	bit flag	0 - charger is in slave-mode ( polling disabled) 1 - charger is in master-mode (polling enabled)
VOLTAGE_NOTREG	2	bit flag	0 - charger's output voltage is in regulation 1 - requested ChargingVoltage() is not being met
CURRENT_NOTREG	3	bit flag	0 - charger's output current is in regulation 1 - requested ChargingCurrent() is not being met
LEVEL_2	4	bit flag	00 is reserved
LEVEL_3	5	bit flag	01 - charger is a Level 2 10 - reserved 11 - charger is a Level 3
CURRENT_OR	6	bit flag	0 - ChargingCurrent() value is valid 1 - ChargingCurrent() value is invalid
VOLTAGE_OR	7	bit flag	0 - ChargingVoltage() value is valid 1 - ChargingVoltage() value is invalid
RES_OR	8	bit flag	0 - Safety Signal not over-range 1 - Safety Signal over-range
RES_COLD	9	bit flag	0 - Safety Signal not cold 1 - Safety Signal cold
RES_HOT	10	bit flag	0 - Safety Signal not hot 1 - Safety Signal hot
RES_UR	11	bit flag	0 - Safety Signal not under-range 1 - Safety Signal under-range
ALARM_INHIBITED	12	bit flag	0 - charger not alarm inhibited 1 - charger alarm inhibited
POWER_FAIL	13	bit flag	0 - input voltage is not low 1 - input voltage is low
BATTERY_PRESENT	14	bit flag	0 - battery is not present 1 - battery is present
AC_PRESENT	15	bit flag	0 - charge power is not available 1 - charge power is available

CHARGE\_INHIBITED bit reflects the status of the charger set by the INHIBIT\_CHARGE bit.

POLLING\_ENABLED bit is set/reset in the HESC-SER charger with the ENABLE\_POLLING bit of ChargerModeCmd() set.

VOLTAGE\_NOTREG bit is set when the HESC detects that the requested voltage in the ChargingVoltageCmd() register is not in regulation. The VOLTAGE\_NOTREG bit typically is set during constant current charging unless the battery voltage reaches the value set in ChargingVoltageCmd() and the HESC begins to voltage regulate to ChargingVoltageCmd() value. VOLTAGE\_NOTREG is cleared when the HESC is regulating to the ChargingVoltageCmd() value. VOLTAGE\_NOTREG is not defined when the charger is disabled.

CURRENT\_NOTREG bit is set when the HESC detects that the requested current in the ChargingCurrentCmd() register is not in regulation. The CURRENT\_NOTREG bit will typically be set during constant voltage charging unless the battery current is near the value set in ChargingCurrentCmd() and the HESC begins to current regulate to ChargingCurrentCmd() value. CURRENT\_NOTREG is cleared when the HESC is regulating to the ChargingCurrentCmd() value. CURRENT\_NOTREG is not defined when the HESC is disabled.

LEVEL\_2 bit is defined to always be set.

LEVEL\_3 bit is set. The HESC is a Level 3 Smart Battery Charger. Note: The HESC operates as a LEVEL\_2 charger when the ENABLE\_POLLING bit is cleared.

CURRENT\_OR bit is set only when ChargingCurrentCmd() is set to a value outside the current regulation range of the HESC. When ChargingCurrentCmd() is set to the programmatic maximum current + 1mA or more, the CURRENT\_OR bit will be set.

VOLTAGE\_OR bit is set only when ChargingVoltageCmd() is set to a value outside the voltage regulation range of the HESC. When ChargingVoltageCmd() is set to the programmatic maximum voltage + 1mV or more, the VOLTAGE\_OR bit will be set.

RES\_OR bit is set when the Th (Safety Signal) resistance value is > 95k ohms. The HESC considers the Th signal as an open circuit.

RES\_COLD bit is set when the Th (Safety Signal) resistance value is > 28,500 ohms. The Th signal indicates a cold battery.

RES\_HOT bit is set when the Th (Safety Signal) resistance value is < 3150 ohms, which indicates a hot battery.

RES\_UR bit is set when the Th (Safety Signal) resistance value is < 575 ohms.

**Notes:**

- Multiple bits may be set depending on the value of the Th (Safety Signal) (e.g., a Th signal resistance that is 400 ohms will cause both the RES\_HOT *and* the RES\_UR bits to be set).
- A Smart Battery can signal some or all of the Safety Signal ranges using fixed value resistors. In battery packs that do not require the Safety Signal as a secondary fail-safe indicator, a single, fixed resistor, may be used to select the Safety Signal range which allows indefinite “wake-up” charging or “wake-up” charging only for the time-out period.
- In all cases, it is the responsibility of the battery pack to manipulate the Safety Signal to obtain correct charger behavior.

ALARM\_INHIBITED bit is set if a valid AlarmWarningCmd() message has been received and charging is inhibited as a result. This bit is cleared if both ChargingVoltageCmd() and ChargingCurrentCmd() are re-written to the charger, power is removed, or if a battery is removed.

POWER\_FAIL bit is set if the input is below the set threshold.

BATTERY\_PRESENT is set if a battery is present otherwise it is cleared.

AC\_PRESENT is set if a source of power for charging is available otherwise it is cleared.

## 9. ChargingCurrentCmd() 0x14

**Description:**

The Host sends the desired charging rate (mA).

**Purpose:**

The HESC uses the ChargingCurrentCmd() function to establish the charging current. In combination with the ChargingVoltageCmd() function and the battery's internal impedance, this function determines the HESC's operating point. Together, these functions permit the HESC to dynamically adjust its charging profile (current/voltage) for optimal charge. The Host can effectively turn off the HESC by returning 0 for this function. The HESC can be operated as a constant voltage source by returning a ChargingCurrentCmd() value of 65535.

**Protocol:** Write Word

**Output:** unsigned int -- maximum charger output current in mA

Units: mA

Range: data range is 0 to 65,535 mA.

## 10. ChargingVoltageCmd() 0x15

**Description:**

The Host sends the desired charging voltage (mV).

**Purpose:**

The HESC uses the ChargingVoltageCmd() function to establish the charging voltage. In combination with the ChargingCurrentCmd() function and the battery's internal impedance, this function determines the HESC's operating point. Together, these functions permit the HESC to dynamically adjust its charging profile (current/voltage) for optimal charge. The Host can effectively turn off the HESC by returning 0 for this function. The HESC can be operated as a constant current source by returning a ChargingVoltageCmd() value of 65535.

**Protocol:** Write Word

**Output:** unsigned int -- maximum charger output voltage in mV

Units: mV

Range: data range is 0 to 65,535 mV

## 11. BatteryStatusCmd() 0x16

**Description:**

If an SMBus Battery is connected to the HESC, this command notifies the HESC that the one or more alarm conditions exist. Alarm and status indications are encoded as bit fields. The HESC will, acting in Master Mode, send the AlarmWarningCmd() to the Host over the SerBus or the PC/104 bus. If the HESC is charging a Standard Battery pack, it will set the appropriate fields, and broadcast it to the Host over the SerBus or the PC/104 bus.

**Purpose:**

The AlarmWarningCmd() function is used by the power management system to get alarm and status bits, as well as error codes from the HESC.

**Protocol:** Read Word

**Output:** unsigned int - Status Register with alarm conditions bit mapped as follows:

\*\*\*\*\* Alarm Bits \*\*\*\*\*

0x8000 OVER\_CHARGED\_ALARM

0x4000 TERMINATE\_CHARGE\_ALARM

0x2000 Reserved

0x1000 OVER\_TEMP\_ALARM

0x0800 TERMINATE\_DISCHARGE\_ALARM

0x0400 Reserved

0x0200 REMAINING\_CAPACITY\_ALARM

0x0100 REMAINING\_TIME\_ALARM

\*\*\*\*\* Status Bits \*\*\*\*\*

0x0080 INITIALIZED

0x0040 DISCHARGING

0x0020 FULLY\_CHARGED

0x0010 FULLY\_DISCHARGED

\*\*\*\*\* Error Code \*\*\*\*\*

0x0000-0x000f Reserved

## 12. GetVersionCmd() (0x3E)

**Description:**

This function returns the firmware version number of the HESC.

**Purpose:**

The GetVersionCmd() function is used to identify a particular battery. This may be important in systems that take advantage of enhanced or custom features of a particular HESC revision.

**Protocol:** Write Word

**Output:** High byte - major revision  
Low byte - minor revision

## 13. SerialNumber() (0x3F)

**Description:**

This function returns the serial number of the HESC and the model name. For the HESC-SER models, "HESC-SER" is returned, and for the HESC104 models, "HESC104 " is returned. The eight digit serial number is appended to the end of the model name.

**Purpose:**

The SerialNumber() function identifies a particular HESC.

**Protocol:** Read Block

**Output:** string – Eight character string of the model name followed by an eight character string of the serial number.

## 14. GetI2CTempCmd0() (0x80) to GetI2CTempCmd15() (0x8F)

**Description:**

The HESC returns the requested I2C temperature to the Host CPU.

**Purpose:**

The Host can perform power management and control functions with this information. Remote cooling fans or heater can be started if temperatures exceed safe limits.

**Protocol:** Read Word

**Output:** unsigned int – temperature in 0.1K  
Units: 0.1K  
Range: data range is 0 to 65,535

## 15. TempThCmd() (0x90)

**Description:**

The HESC returns the thermistor value to the Host CPU.

**Purpose:**

The thermistor monitors the battery temperature, and for the SMBus is a secondary safety device. The Host CPU can use this information to determine the battery performance. The high byte is used by the HESC to determine the battery state, such as BattNew, BattGone, etc.

**Protocol:** Read Word

**Output:** unsigned int – value from ADC converter  
Units: ADC converter value  
Range: data range is 0 to 65,535  
**Description:**  
Returns the main input voltage (mV).

**Purpose:**

The MainVCmd() function provides the Host power management system with an accurate main input voltage. The Host management system can use this voltage, along with main input current information to help enable intelligent, adaptive power management systems.

**Protocol:** Read Word

**Output:** unsigned int – main input voltage in mV

Units: mV

Range: 0 to 65,535

## 16. MainICmd() (0x92)

**Description:**

Returns the current being supplied to the HESC (mA).

**Purpose:**

The MainICmd() function provides the Host power management system an accurate measure of the current flowing into the HESC. The current being report by the MainICmd() will be either from the main input or battery or some from both inputs if both the main input and battery are the same potential. Compare the voltages returned by BattIVCmd() and the MainVCmd() to determine the source of the power.

**Protocol:** Read Word

**Output:** unsigned int – input current rate in mA increments

Units: mA

Range: 0 to 65,535

## 17. InputPwrCmd() (0x93)

**Description:**

Returns the power being supplied to the HESC.

**Purpose:**

The InputPwrCmd() function provides the Host power management system a measure of the power flowing into the HESC-SER. The power being report by the InputPwrCmd() will be either from the main input or battery or some from both inputs if both the main input and battery are the same potential. Compare the voltages returned by BattIVCmd() and the MainVCmd() to determine the source of the power.

**Protocol:** Read Word

**Output:** unsigned int – input power in 10mW increments

Units: 10mW

Range: 0 to 65,535

## 18. BattPwrCmd() (0x94)

**Description:**

Returns the power being charged/dischARGE to the battery through the HESC.

**Purpose:**

The BattPwrCmd() function provides the Host power management system a measure of the power charging/discharging into the battery through the HESC. The power being reported by the BattPwrCmd() will be positive for charging, and negative for discharging.

**Protocol:** Read Word

**Output:** signed int – power in 10mW increments

Units: 10mW

Range: 0 to 32,767 charging

0 to -32768 discharging

## 19. ChCycleCmd() (0x95)

**Description:**

Sets or returns the current charging cycle of the HESC.

**Purpose:**

The ChCycleCmd() function allows the Host power management system to detect or change the current charge cycle. The Host CPU chooses to do this to minimize input power, or place the battery pack into a different charge cycle.

**Protocol:** Read/Write Word

**Output:** unsigned int – charge cycle

Units: integer

Range: 0 to 3

## 20. ChTermLastCmd() (0x96)

**Description:**

This function returns the cause of the last charge termination method for Standard Battery packs.

**Purpose:**

The Host can determine how effective the charging parameters are by the ChTermLastCmd() for an individual battery. The ChTermLastCmd() will also allow the Host to determine the condition of the battery.

**Protocol:** Read Word

Table 5, ChTermLastCmd bit map

Bit#	Bit Name	Description
0	TimeMaxEn	Charging time exceeded TimeMaxDef minutes.
1	BattTempMaxEn	Battery temperature was above the BattTempMaxDef 0.1K.
2	BattIminEn	Charging current was below BattIminDef mA.
3	BattVmaxEn	Charging voltage was above the BattVmaxDef mV.
4	BattVmaxTimeEn	The battery voltage did not increase for BattVmaxTimeDef minutes.
5	BattVdeltaEn	The battery voltage decreased by BattVdeltaDef mV.
6	BattTempRateEn	The temperature increased at a rate of BattTempRateDef 0.1K/minute or greater.
7 to 15	future	

## 21. ShutDownCmd() (0x97)

**Description:**

Shut down the HESC outputs in “X” seconds.

**Purpose:**

Allows the Host CPU to do an orderly shut down of files and it's operating system before power the HESC outputs are turned off. A read using ShutDownCmd() will return the number of seconds until the HESC outputs are turned off. If a shutdown is not in-progress the ShutDownCmd() will return the predefined number of seconds for shut down. ShutDownCmd() does not over-right the EEPROM default values.

**Protocol:** Write/Read Word

**Output:** unsigned int – shut down time in seconds

Units: seconds

Range: data range is 0 to 65,535 mV

## 22. PowerSupplyStatusCmd() (0x98)

**Description:**

The Host uses this command to read or set the various power supply modes.

**Purpose:**

Allows the System Host to configure the power supply and change the default modes.

**Protocol:** Write Word

**Input:** unsigned int - bit mapped - see below

Units: not applicable

Range: not applicable

Table 6, PowerSupplyStatusCmd bit map

Bit#	Bit Name	Description
0	BattAutoStartEn*	Charging is to autostart when the HESC is reset, main power is removed then re-applied, or when a new battery is inserted. 1 = enable, 0 = disable
1	TermEn*	Charge termination is enabled when TermEn is set. 1 = enable, 0 = disable
2	SMBactiveEn	The HESC to function as a level 3 SMBus charger. 1 = enable, 0 = disable
3	IgnHiOffEn	When IgnHiOff is set, the HESC-SER will begin shutdown procedures when the Ign pin is high. If IgnHiOff is low, the HESC-SER will begin shutdown procedures when the Ign pin is low. The shutdown procedure will finish once shutdown is started.
4	BattIsolateEn*	The HESC de-activates the battery enable line (BE) after the power supply enters the shutdown mode.
5	SMBtimingEn	Observes the minimum timing requirements for SMBus compatibility. 1 = enable, 0 = disable
6	Then*	Thermistor monitoring select, 1 = enable, 0 = disable
7	SU_Req	Start up request
8	SD_Req	Shut down request
9	I2CLowAlarm	A I2C sensor/controller is in low alarm = 1, normal = 0
10	I2CHighAlarm	A I2C sensor/controller is in high alarm = 1, normal = 0
11	IgnInput	Status of ignition input
12	SDinput	Status of Shut Down input
13	CHen	Charger is enabled and ready to charge (read only)
14	LowPwrMode	HESC is in low or limited power mode
15	ChecksumEn	Checksum protocol in effect on SerBus or PC/104 bus

### 23. SDSUCauseCmd() (0x99)

**Description:**

The Host uses this command to read the cause(s) of the current SDreq and SUreq.

**Purpose:**

Allows the System Host to determine how to adjust shut-down or start-up procedures.

**Protocol:** Write Word

**Input:** unsigned int - bit mapped - see below

Units: not applicable

Range: not applicable

Table 7, SDSUCauseCmd bit map

Bit#	Action	Bit Name	Cause of Start-Up request
0	SUreq = 1	ChFlags.SUreq	Power applied to a de-energized HESC with ChFlags.SUreq set
1	SUreq = 1	PowerSupplyStatusCmd (0x98)	Bit 7 set during PowerSupplyStatusCmd write
2	SUreq = 1	IGN (SD)	Received a start-up request from the IGN input
3	SUreq = 1	SD-PB	Received a start-up request from the pushbutton input
4			
5			
6			
7			

Bit#	Action	Bit Name	Cause of Shut-Down request
8	SDreq = 1	Main Power	Loss of main power
9	SDreq = 1	PowerSupplyStatusCmd (0x98)	Bit 8 set during PowerSupplyStatusCmd write
10	SDreq = 1	IGN (SD)	Received a shut-down request from the IGN input
11	SDreq = 1	SD-PB	Received a shut-down request from the pushbutton input
12	SDreq = 1	ShutDownCmd (0x97)	Command received from Host
13			
14			
15			

## 24. ActiveEEcmd() (0xA0)

**Description:**

The Host uses this command to set the address for reading/writing to EEPROM.

**Purpose:**

Allows the System Host to change charging profiles and HESC setup.

**Protocol:** Write Word

**Input:** unsigned int, Low byte = location 0 to 127, High byte: no auto increment = 0; auto increment = 1  
Units: not applicable

## 25. EEPROMCmd () (0xA1)

**Description:**

The Host uses this command to read/write the data in the EEPROM. The read/write address in the EEPROM will increment after each read/write if the auto increment byte is set = 1 through command function ActiveEEcmd() (0xA0).

**Purpose:**

Allows the System Host to change charging profiles and HESC setup.

**Protocol:** Write Word

**Input:** unsigned int, 16 bit value. See [List of EEPROM Setpoints and Configuration Variable](#)  
Units: not applicable

## Section 6 : EEprom Setpoints and Configuration Variables

### A. List of EEprom Setpoints and Configuration Variable

1. The EEprom is used to store setpoints and configuration for the HESC. The EEprom is non-volatile and will not lose its contents on power loss and has an endurance for a minimum of 100,000 cycles of writes or erases
2. Variables or setpoints are stored with their least significant byte stored in the lower memory location, and the most significant byte in the high location.

Table 8, HESC Setpoints and Configuration Variables

EEprom start address				Variable Size	Variable/Setpoint Name	Units	Range	Description
1	2	3	4					
Charge Cycle								
0x00	0x20	0x40	0x60	Word	ChTerm	Bit flags	N/A	Charge termination enable flags
0x02	0x22	0x42	0x62	Word	Future1			
0x04	0x24	0x44	0x64	Word	Future2			
0x06	0x26	0x46	0x66	Word	BattVmaxDef	mV	0 to 65535	Maximum battery charging voltage
0x08	0x28	0x48	0x68	Word	BattVmaxTimeDef	Min	0 to 65535	Maximum time since peak battery voltage detected
0x0A	0x2A	0x4A	0x6A	Word	BattVdeltaDef	mV	0 to 65535	Charge termination negative delta V
0x0C	0x2C	0x4C	0x6C	Word	timeMaxDef	Min	0 to 65535	Maximum time for charge cycle
0x0E	0x2E	0x4E	0x6E	Word	BattIminDef	mA	0 to 65535	Minimum charge current allowed
0x10	0x30	0x50	0x70	Word	BattImaxDef	mA	0 to 65535	Maximum charge current allowed
0x12	0x32	0x52	0x72	Word	timeTermEnDef	Min	0 to 65535	Minimum time before charge termination allowed
0x14	0x34	0x54	0x74	Word	BattTempCompDef	mV/degK	0 to 65535	Temperature compensation applied to BattVDef
0x16	0x36	0x56	0x76	Word	BattVDef	mV	0 to 65535	Charging voltage set point
0x18	0x38	0x58	0x78	Word	BattIDef	mA	0 to 65535	Charging current set point
0x1A	0x3A	0x5A	0x7A	Word	BattTempRateDef	0.1K/Min	0 to 6553.5	Maximum rate of battery temperature increase allowed
0x1C	0x3C	0x5C	0x7C	Word	BattTrickleDef	mA	0 to 65535	Trickle charge current if below min temp or voltage
0x1E	0x3E	0x5E	0x7E	Word	BattTrickleTimeDef	Min	0 to 65535	Maximum time allowed in Trickle charge mode
0x80				Word	ChFlags	Bit flags	N/A	Charger/power supply enable flags
0x82				Word	SDdef	Sec	0 to 65535	Delay in turning off power supply outputs
0x84				Word	SUdef	Sec	0 to 65535	Delay in turning on power supply outputs
0x86				Word	MainPwrMaxDef	10mW	0 to 65535	Maximum input power allowed
0x88				Byte	MaxBusTime	Timer Ticks	0 to 255	Maximum time before communications timeout
0x89				Byte	CHCycleMax	Cycle	1 to 4	Defines how many charge cycles to use
0x8A				Word	BattTempMinDef	0.1K *	0 to 6553.5	Minimum battery charging temperature
0x8C				Word	BattTempMaxDef	0.1K *	0 to 6553.5	Maximum battery charging temperature
0x8E				Word	BattVminDef	mV	0 to 65535	Minimum battery charging voltage
0x90				Byte	ChTempSelect	Bit flags	0 to 16	I2C device to use for battery temp, if zero use Th
0x91				Byte	ChAmbientSelDef	Bit flags	0 to 16	I2C device to use for ambient temp, if zero use Th
0x92				Word	I2CpollTimeDef	Sec	0 to 65535	Rate at which I2C devices are polled
0x94				Word	I2CtsICenDef	Bit flags	N/A	Enables polling for selected I2C device
0x96				Word	BattSelDef	Bit flags	N/A	Selects multi-battery packs, if zero BE selects battery pack
0x9A								
0x9C								
0x9E								

## HESC - High Efficiency & Smart Charging Firmware Manual

0xA0, 0xA1	I2Cconfig0	Bit flags	N/A	Defines I2C sensor 0 address, and operating modes
0xA2, 0xA3	I2CSetPoint0	0.1K *	0 to 6553.5	I2C Sensor 0 low temp alarm setting
0xA4, 0xA5	I2CHiLoAlarm0	0.1K *	0 to 6553.5	I2C Sensor 0 high temp alarm setting
0xA6, 0xA7	I2Cconfig1	Bit flags	N/A	Defines I2C sensor 1 address, and operating modes
0xA8, 0xA9	I2CSetPoint1	0.1K *	0 to 6553.5	I2C Sensor 1 low temp alarm setting
0xAA, 0xAB	I2CHiLoAlarm1	0.1K *	0 to 6553.5	I2C Sensor 1 high temp alarm setting
0xAC, 0xAD	I2Cconfig2	Bit flags	N/A	Defines I2C sensor 2 address, and operating modes
0xAE, 0xAF	I2CSetPoint2	0.1K *	0 to 6553.5	I2C Sensor 2 low temp alarm setting
0xB0, 0xB1	I2CHiLoAlarm2	0.1K *	0 to 6553.5	I2C Sensor 2 high temp alarm setting
0xB2, 0xB3	I2Cconfig3	Bit flags	N/A	Defines I2C sensor 3 address, and operating modes
0xB4, 0xB5	I2CSetPoint3	0.1K *	0 to 6553.5	I2C Sensor 3 low temp alarm setting
0xB6, 0xB7	I2CHiLoAlarm3	0.1K *	0 to 6553.5	I2C Sensor 3 high temp alarm setting
0xB8, 0xB9	I2Cconfig4	Bit flags	N/A	Defines I2C sensor 4 address, and operating modes
0xBA, 0xBB	I2CSetPoint4	0.1K *	0 to 6553.5	I2C Sensor 4 low temp alarm setting
0xBC, 0xBD	I2CHiLoAlarm4	0.1K *	0 to 6553.5	I2C Sensor 4 high temp alarm setting
0xBE, 0xBF	I2Cconfig5	Bit flags	N/A	Defines I2C sensor 5 address, and operating modes
0xC0, 0xC1	I2CSetPoint5	0.1K *	0 to 6553.5	I2C Sensor 5 low temp alarm setting
0xC2, 0xC3	I2CHiLoAlarm5	0.1K *	0 to 6553.5	I2C Sensor 5 high temp alarm setting
0xC4, 0xC5	I2Cconfig6	Bit flags	N/A	Defines I2C sensor 6 address, and operating modes
0xC6, 0xC7	I2CSetPoint6	0.1K *	0 to 6553.5	I2C Sensor 6 low temp alarm setting
0xC8, 0xC9	I2CHiLoAlarm6	0.1K *	0 to 6553.5	I2C Sensor 6 high temp alarm setting
0xCA, 0xDB	I2Cconfig7	Bit flags	N/A	Defines I2C sensor 7 address, and operating modes
0xCC, 0xCD	I2CSetPoint7	0.1K *	0 to 6553.5	I2C Sensor 7 low temp alarm setting
0xCE, 0xCF	I2CHiLoAlarm7	0.1K *	0 to 6553.5	I2C Sensor 7 high temp alarm setting
0xD0, 0xD1	I2Cconfig8	Bit flags	N/A	Defines I2C sensor 8 address, and operating modes
0xD2, 0xD3	I2CSetPoint8	0.1K *	0 to 6553.5	I2C Sensor 8 low temp alarm setting
0xD4, 0xD5	I2CHiLoAlarm8	0.1K *	0 to 6553.5	I2C Sensor 8 high temp alarm setting
0xD6, 0xD7	I2Cconfig9	Bit flags	N/A	Defines I2C sensor 9 address, and operating modes
0xD8, 0xD9	I2CSetPoint9	0.1K *	0 to 6553.5	I2C Sensor 9 low temp alarm setting
0xDA, 0xDB	I2CHiLoAlarm9	0.1K *	0 to 6553.5	I2C Sensor 9 high temp alarm setting
0xDC, 0xDD	I2Cconfig10	Bit flags	N/A	Defines I2C sensor 10 address, and operating modes
0xDE, 0xDF	I2CSetPoint10	0.1K *	0 to 6553.5	I2C Sensor 10 low temp alarm setting
0xE0, 0xE1	I2CHiLoAlarm10	0.1K *	0 to 6553.5	I2C Sensor 10 high temp alarm setting
0xE2, 0xE3	I2Cconfig11	Bit flags	N/A	Defines I2C sensor 11 address, and operating modes
0xE4, 0xE5	I2CSetPoint11	0.1K *	0 to 6553.5	I2C Sensor 11 low temp alarm setting
0xE6, 0xE7	I2CHiLoAlarm11	0.1K *	0 to 6553.5	I2C Sensor 11 high temp alarm setting
0xE8, 0xE9	I2Cconfig12	Bit flags	N/A	Defines I2C sensor 12 address, and operating modes
0xEA, 0xEB	I2CSetPoint12	0.1K *	0 to 6553.5	I2C Sensor 12 low temp alarm setting
0xEC, 0xED	I2CHiLoAlarm12	0.1K *	0 to 6553.5	I2C Sensor 12 high temp alarm setting
0xEE, 0xEF	I2Cconfig13	Bit flags	N/A	Defines I2C sensor 13 address, and operating modes
0xF0, 0xF1	I2CSetPoint13	0.1K *	0 to 6553.5	I2C Sensor 13 low temp alarm setting
0xF2, 0xF3	I2CHiLoAlarm13	0.1K *	0 to 6553.5	I2C Sensor 13 high temp alarm setting
0xF4, 0xF5	I2Cconfig14	Bit flags	N/A	Defines I2C sensor 14 address, and operating modes
0xF6, 0xF7	I2CSetPoint14	0.1K *	0 to 6553.5	I2C Sensor 14 low temp alarm setting
0xF8, 0xF9	I2CHiLoAlarm14	0.1K *	0 to 6553.5	I2C Sensor 14 high temp alarm setting
0xFA, 0xFB	I2Cconfig15	Bit flags	N/A	Defines I2C sensor 15 address, and operating modes
0xFC, 0xFD	I2CSetPoint15	0.1K *	0 to 6553.5	I2C Sensor 15 low temp alarm setting
0xFE, 0xFF	I2CHiLoAlarm15	0.1K *	0 to 6553.5	I2C Sensor 15 high temp alarm setting

## B. SerBus EEprom Details

### 1. ChTerm - EEprom Locations 0x00, 0x20, 0x40, 0x60:

**Description:**

ChTerm is a set of bit flags that enable/disable charging termination and charging functions. Each ChTerm location is active only when its charging cycle is active.

**Purpose:**

Allow the Host to configure the HESC-SER to charge a particular battery type.

**Structure:**

Table 9, ChTerm bit map

Bit#	Bit Name	Description
0	BattTempMinEn	Charge termination method that terminates charging if the battery temperature is below the BattTempMinDef. 1 = enable, 0 = disable
1	BattTempMaxEn	Charge termination method that terminates charging if the battery temperature is above the BattTempMaxDef. 1 = enable, 0 = disable
2	BattVminEn	Charge termination method that terminates main charging if the battery voltage is below the BattVminDef. Trickle charging may continue until the Battery voltage reaches BattVminDef. 1 = enable, 0 = disable
3	BattVmaxEn	Charge termination method that terminates charging if the battery voltage is above the BattVmaxDef 1 = enable, 0 = disable
4	BattVmaxTimeEn	Charge termination method that terminates charging if the battery voltage has not increased for BattVmaxTimeDef minutes. 1 = enable, 0 = disable
5	BattVdeltaEn	Charge termination method that terminates charging if the battery voltage has decreased by BattVdeltaDef. 1 = enable, 0 = disable
6	TimeMaxEn	Charge termination method that terminates charging if charging has taken place for TimeMaxDefEn minutes. 1 = enable, 0 = disable
7	BattIminEn	Charge termination method that terminates charging if charging current is below BattIminDef. 1 = enable, 0 = disable
8	TimeTermEn	Prevents charge termination for TimeTermEnDef minutes. 1 = enable, 0 = disable
9	BattTempCompEn	Compensates the BattVDef voltage for the temperature. Compensation is in mV starting at 298.2K (25C). BattVDef decreases when temperature increases above the 298.2K, and increases when the temperature decreases below 298.2K. 1 = enable, 0 = disable
10	BattTempRateEn	Charge termination method that terminates charging if the temperature increases at a rate of BattTempRateDef 0.1K/minute
11	BattTrickleTimeEn	Charge termination method that terminates trickle charging is trickle charging has continued for BattTrickleTimeDef minutes. 1 = enable, 0 = disable
12	Future	
13	Future	
14	Future	
15	Future	

### 2. Spare - EEprom Locations 0x02 0x22 0x42 0x62

### 3. Spare - EEprom Locations 0x04 0x24 0x44 0x64

### 4. BattVmaxDef - EEprom Locations 0x06, 0x26, 0x46, 0x66:

**Description:**

When BattVmaxEn is enabled, the HESC-SER will terminate charging a standard battery pack when the battery voltage rises above BattVminDef mV.

**Purpose:**

Prevent damage to batteries that can occur from being overcharged.

## 5. BattVmaxTimeDef - EEprom Locations 0x08, 0x28, 0x48, 0x68:

### Description:

When BattVmaxTimeEn is enabled, the HESC-SER will terminate charging a standard battery pack when the length of time the HESC-SER has been charging since the last increase in battery voltage exceeds BattVmaxTimeDef minutes.

### Purpose:

Prevent damage to batteries that can occur from being overcharged. Some batteries, like NiMH, have a very small negative delta V that may not be detected.

## 6. BattVdeltaDef - EEprom Locations 0x0A, 0x2A, 0x4A, 0x6A:

### Description:

When BattVdeltaEn is enabled, the HESC-SER will terminate charging a standard battery pack when the battery voltage has reduced BattVdeltaDef mV from the maximum peak voltage.

### Purpose:

Prevent damage to batteries that can occur from being overcharged. Some batteries, like NiMH, have a very small negative delta V that may not be detected.

## 7. TimeMaxDef - EEprom Locations 0x0C, 0x2C, 0x4C, 0x6C:

### Description:

When TimeMaxEn is enabled, the HESC-SER will terminate charging a standard battery pack when the length of time the HESC-SER has been charging (this charge cycle) exceeds TimeMaxDef minutes.

### Purpose:

Prevent damage to batteries that can occur from being overcharged. This puts a limit on how long the charger will operate.

## 8. BattlminDef - EEprom Locations 0x0E, 0x2E, 0x4E, 0x6E:

### Description:

When BattlminEn is enabled, the HESC-SER will terminate charging a standard battery pack when the charge current is below BattlminDef mA.

### Purpose:

Prevent damage to batteries that can occur from being low charge currents. Some batteries, like NiMH, should not be charged below a minimum rate or the batteries may overheat.

## 9. BattlmaxDef - EEprom Locations 0x10, 0x30, 0x50, 0x70:

### Description:

When BattlmaxEn is enabled, the HESC-SER will terminate charging a standard battery pack when the charge current exceeds BattlminDef mA.

### Purpose:

Prevent damage to batteries that can occur from excessive charge currents. Excessive current can cause serious overheating, and possible venting.

## 17. ChFlags - EEprom Locations 0x80;

### Description:

ChFlags is a set of bit flags that enable/disable charging and power supply functions.

### Purpose:

ChFlags allows the Host to configure the HESC-SER to the type of battery pack, and system it is installed into.

Table 10, ChFlags bit map

Bit#	Bit Name	Description
0	BattAutoStartEn*	Charging is to autostart when the HESC-SER is reset, main power is removed then re-applied, or when a new battery is inserted. 1 = enable, 0 = disable
1	TermEn*	Charge termination is enabled when TermEn is set. 1 = enable, 0 = disable
2	SMBActiveEn	The HESC-SER to function as a level 3 SMBus charger. 1 = enable, 0 = disable SMBus timing in effect when SMBActiveEn = 1
3	IgnHiOffEn	When IgnHiOff is set, the HESC-SER will begin shutdown procedures when the Ign pin is high. If IgnHiOff is low, the HESC-SER will begin shutdown procedures when the Ign pin is low. The shutdown procedure will finish once shutdown is started.
4	BattIsolateEn*	The HESC-SER de-activates the battery enable line (BE) after the power supply enters the shutdown mode.
5	MultiBattEn*	Enables Multi battery pack charging, 1 = enable, 0 = disable
6	Then*	Thermistor monitoring select, 1 = enable, 0 = disable
7	SUreq	Startup request when power applied when HESC-SER is hard-off**, 1 = off, 0 = on

\* For standard battery packs only (ie. Not for SMBus battery packs)

## 18. BattTempMinDef - EEprom Locations 0x8A:

### Description:

When BattTempMinEn is enabled, the HESC-SER will not charge a standard battery pack when the temperature is below BattTempMinDef degrees.

### Purpose:

Prevent damage to batteries that can occur when being charged at low temperatures.

## 19. BattTempMaxDef - EEprom Locations 8C:

### Description:

When BattTempMaxEn is enabled, the HESC-SER will terminate charging a standard battery pack when the battery temperature rises above BattTempMaxDef degrees.

### Purpose:

Prevent damage to batteries that can occur from being over charged or charged at high ambient temperatures.

## 20. BattVminDef - EEprom Locations 0x8E:

### Description:

When BattVminEn is enabled, the HESC-SER will not charge or terminate charging a standard battery pack when the battery voltage is below BattVminDef mV.

### Purpose:

To prevent damage to batteries that can occur from being charged when in an excessively low charge condition, or to prevent normal charging if one or more cells are faulty.

## 21. BattSelDef - EEprom Locations 0x9E:

### Description:

BattSelDef is a set of bitflags that define the battery packs in the system. Up to 16 battery packs can be used with the remote battery multiplexer. If BattSelDef = 0, then two battery packs are charged using BE as the battery selector. The MultiBattEn bit in the ChFlags EEprom (location 0x80) must be set before the BattSelDef will be used.

### Purpose:

To allow charging multiple battery packs.

## Section 7 : Temperature Sensor, TS-IC and Controller, TC-IC

The HESC can support up to a total of 16 TS-IC sensors and TC-IC controllers on the I2C bus. A block of six EEPROM bytes are reserved for each device. The six EEPROM bytes have the following names and functions:

1. Byte 0        I2Cconfig#  
- See table below
2. Byte 1        Address byte#  
- Bits 1 to 7 forms the I2C address, bit 0 forms R/W bit. For a read set bit 0 = 1 and for a write set bit 0 = 0.
3. Byte 2 & 3    I2CSetpoint#  
For TC-IC: Setpoint value that will be sent. Value = 0 to 65535.  
For TS-IC: Low alarm value. Alarm range = 0 to 65535.
4. Byte 4 & 5    I2ChiLoAlarm#.  
For TC-IC: Low byte = low alarm value, High byte = high alarm value. Value = 0 to 255.  
Low alarm setting = I2CSetpoint# - I2ChiLoAlarm# (low byte).  
High alarm setting = I2CSetpoint# + I2ChiLoAlarm#(high byte).  
Note: if I2Csetpoint < I2ChiLoAlarm#(low byte), then Low alarm setting = 0;  
if I2Csetpoint > (65505 - I2ChiLoAlarm#(high byte)), then High alarm setting = 65535;  
  
For TS-IC: High alarm value. Alarm range = 0 to 65535.

Table 11, Temperature Sensor and Temperature Controller bit map

Bit#	Bit Name	Description
0	Cmd bit 0	Bit 0 and 1 selects data and/or commands within the TC-IC address. Refer to TC-IC for more details.
1	Cmd bit 1	For TS-IC, Cmd bit 0 & 1 should be zero. For a write operation the contents of I2CSetpoint# will be sent as the data. The data returned from a read operation is compared to the alarm settings.
2	MainPwr	Indicates to TC-IC devices that main power is available when = 1. When operating on battery power = 0. Works for read or write to TC-IC devices. This bit is a don't care for TS-ICs.
3	LimitedPwr	Indicates to TC-IC devices that limited power (main or battery) is available. Works for read or write to TC-IC devices. This bit is a don't care for TS-ICs.
4	ShutDown	Indicates to TC-IC devices that the HESC is in timed shutdown mode. HESC outputs will turn off after the timed shutdown finishes. Works for read or write to TC-IC devices. This bit is a don't care for TS-ICs.
5	AlarmLoEn	Enables low alarm reporting = 1, Disables low alarm reporting = 0.
6	AlarmHiEn	Enables high alarm reporting = 1, Disables high alarm reporting = 0;
7	TCICSel	Selects temperature controller (TC-IC) or temperature sensor (TS-IC): For TC-IC set bit to 1, for TS-IC set bit to 0

Note: The "#" represent the number (0 to 15) of the TC-IC or TS-IC device.

## Section 8 : Start-up and Shut-down settings, delays, and functions

The HESC has a power management system that assists in interfacing with advanced operating systems such as Windows, and Linux. It is imperative with the advanced operating systems that an orderly shutdown of the files, application program and the operating system itself occur. Failure to allow for orderly shutdown can cause data loss, intermittent crashes, or a complete unrecoverable system failure.

The HESC has two internal modes to control startup and shutdown. The startup mode is entered when the SUreq is set. The shutdown mode is entered when the SDreq is set. It is possible that both the SUreq and the SDreq will be set at the same time. The methods for entering each mode is listed below:

The Startup mode is set:

- a) When the HESC has power applied after being completely de-energized, either through the main input, or through the battery input.  
- Set bit 7 of EEPROM location 0x80.
- b) When the Host CPU sends command PowerSupplyStatusCmd (0x98) with bit 7 set, either through the SerBus, PC/104 bus or the I2C/SMBus.
- c) When the SD input of the HESC-SER is taken to the "On" state. The polarity of the SD is programmable, so the physical state of the input (high or low) will depend on the polarity setting.
- d) When the SD-PB input (separate push-button input for the non-PC/104 version of the HESC) is energized, and the HESC-SER is off, or is in the shutdown mode.

The Shutdown mode is set:

- a) When main input power fails or is removed.
- b) When the Host CPU sends command PowerSupplyStatusCmd (0x98) with bit 8 set, either through the SerBus or the I2C/SMBus.
- c) When the SD input of the HESC-SER is taken to the "Off" state. The polarity of the SD is programmable, so the physical state of the input (high or low) will depend on the polarity setting.
- d) When the SD-PB input (separate push-button input for the non-PC/104 version of the HESC) is energized, and the HESC-SER is on, or is in the startup mode.
- e) When the Host CPU send command ShutDownCmd (0x97).

The Shutdown mode, once started, can only be cancelled through the Host CPU. The Host CPU can cancel the Shutdown mode with command PowerSupplyStatusCmd (0x98) with bit 8 cleared.

A typical situation where the Host CPU might cancel the Shutdown mode is if the HESC-SER SD input was turned off briefly, then turned back on. The Host CPU recognizing that the SD input had returned to the On state could halt it's orderly shutdown and continue normal operations. However, in the same situation, if the Host CPU was too far advanced into its orderly shutdown, it would be unable to cancel the shutdown cycle and the HESC-SER would complete the Shutdown. Following the Shutdown, the Startup mode would be initiated as a result of the SD input being returned to the ON state.

The Startup request can be cancelled if the HESC-SER outputs are turned off, or the HESC-SER is currently in the Shutdown mode, and another Shutdown request is received. This prevents an unnecessary cycle where the HESC-SER outputs are turned off, then briefly on, then off again.

EEProm locations are used to set the default Shutdown delays (0x82 & 0x83), and Startup delays (0x84 & 0x85). The ShutDownCmd (0x97) can also be used to set the Shutdown delay while the HESC-SER is operating.

## Section 9 : Battery Charging Algorithms and Strategies

### A. Overview

### B. Sealed Lead Acid, SLA

#### 1. Single Stage Charging

Many SLA batteries require only a single stage of charging. In single stage charging, the battery is charged with constant current until the battery voltage approaches the "float" voltage. The float voltage is the voltage across the battery when a small trickle current is used to maintain the full charge state of the SLA battery. The float voltage is very close to the "open circuit" voltage of the HESC charger output, and in most instances, is an acceptable method of checking the HESC float voltage setting. When the battery voltage rises close to the float voltage the charger transitions from constant current to constant voltage charging. The transition from constant current to constant voltage is not an instantaneous change, but gradual as the net potential difference between the HESC charger output and the battery voltage is not sufficient to maintain the current in constant current mode.

The following is an example charging parameters, and charge termination methods that are suitable for SLA batteries requiring single stage charging.

Ex. 12V SLA (6 cell), 4.5amp-hour, ambient temperature conditions -20C to 35C

Table 12, SLA Single Stage Charging Set-up

EEprom Enable Values	EEprom Variable Values	Description
BattTempMaxEn = 1	BattTempMax = 3182*	Terminates charging if battery temperature above BattTempMax. Set for 45C
BattVmaxEn = 1	BattVmax = 13900	Terminates charging if battery voltage above BattVmaxDef. Set for 13.9V
BattTempCompEn = 1	BattTempComp = 18	Compensates the BattVDef voltage for the ambient temperature. Compensation is in mV starting at 298.2K (25C).
N/A	BattVDef = 13700	"Float" voltage set-up. Set for 13.7 volts
N/A	BattIDef = 2500	Maximum charge current. Set for 2500mA
CHCycleMax = 1		Set for single stage charging
BattAutoStartEn = 1		Charging auto starts when the HESC is reset (main power is removed then re-applied), or when a new battery is inserted
TermEn = 1		Charge termination is enabled
BattIsolateEn = 1		HESC de-activates the battery enable line (BE) after the power supply enters the shutdown mode.
Then = 1		Thermistor monitoring enabled

\*Temperature in 0.1degK, absolute zero (0K) = -273.2C

#### 2. Dual Stage Charging

A disadvantage of single stage charging is the charging current drops as the battery voltage approaches the float voltage. A two stage charging algorithm maintains full charging current throughout the first stage, and switches to constant float voltage charging in the second stage. The termination of stage 1 charging is when the battery voltage reaches the stage one BattVmax voltage. BattVmax is usually higher than the float voltage (2.45V/cell vs 2.3V/cell). Since the HESC is required to operate in constant current mode throughout stage one, the BattVDef has to be set higher than BattVMax. In theory, BattVDef could be set to its maximum, but this is not a good practice. A better practice is to set BattVDef just high enough to allow the HESC to remain in constant current mode throughout stage one charging.

Table 13, SLA Two Stage Charging Set-up

EEprom Enable Values	EEprom Variable Values	Stage	Description
BattTempMaxEn = 1	BattTempMax = 3182*	1	Terminates charging if battery temperature above BattTempMax. Set for 45C
BattVmaxEn = 1	BattVmax = 14700	1	Terminates charging if battery voltage above BattVmaxDef. Set for 13.9V
BattTempCompEn = 1	BattTempComp = 18	1	Compensates the BattVDef voltage for the ambient temperature. Compensation is in mV starting at 298.2K (25C).
N/A	BattVDef = 15700	1	"Float" voltage set-up. Set for 13.7 volts
N/A	BattIDef = 2500	1	Maximum charge current. Set for 2500mA
BattTempMaxEn = 0	N/A	2	Terminates charging if battery temperature above BattTempMax. Set for 45C
BattVmaxEn = 0	N/A	2	Terminates charging if battery voltage above BattVmaxDef. Set for 13.9V
BattTempCompEn = 1	BattTempComp = 18	2	Compensates the BattVDef voltage for the ambient temperature. Compensation is in mV starting at 298.2K (25C).
N/A	BattVDef = 13700	2	"Float" voltage set-up. Set for 13.7 volts
N/A	BattIDef = 2500	2	Maximum charge current. Set for 2500mA
CHCycleMax = 2			Set for two stage charging
BattAutoStartEn = 1			Charging auto starts when the HESC is reset (main power is removed then re-applied), or when a new battery is inserted
TermEn = 1			Charge termination is enabled
BattIsolateEn = 1			HESC de-activates the battery enable line (BE) after the power supply enters the shutdown mode.
Then = 1			Thermistor monitoring enabled