HESC-UPS18 Manual

High Efficiency & Smart Charging Uninterruptible Power Supply Firmware Manual

For HESC products featuring
Microchip PIC18F microprocessors

Designed by
TRI-M ENGINEERING
Engineered Solutions for Embedded Applications

Technical Manual

P/N: HESC-UPS18
Revision: 17-September-09

Unit 100, 1407 Kebet Way
Port Coquitlam, British Columbia, V3C 6L3
Canada
http://www.Tri-M.com
Tel 604.945.9565 - North America 800.665.5600
Fax 604.945.956

PREFACE

This manual is for integrators of applications of embedded systems. It contains information on hardware and software requirements and interconnection to other embedded electronics.

DISCLAIMER

Tri-M Engineering makes no representations or warranties with respect to the contents of this manual, and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. Tri-M Engineering shall under no circumstances be liable for incidental or consequential damages or related expenses resulting from the use of this product, even if it has been notified of the possibility of such damages. Tri-M Engineering reserves the right to revise this publication from time to time without obligation to notify any person of such revisions. If errors are found, please contact Tri-M Engineering at the address listed on the title page of this document.

Table of Contents

List of Tables 5
Section 1 : HESC-UPS18 Introduction 6
Section 2 : Configuring the HESC. 7
Section 3 : Start-up and Shutdown settings, delays, and functions 7
Section 4 : HESC-SER(D) SerBus Communications 9
A. Description: 9
B. Master Mode: 9
C. Slave Mode: 10
Section 5 : HESC104(D) PC/104 Bus Communications 12
A. Description: 12
B. Master Mode: 12
C. Slave Mode: 13
Section 6 : HESC I²C/SMBus Bus Communications 14
A. Description: 14
B. Master Mode: 15
Section 7 : SerBus, SMBus, and PC/104 Command Functions 15
A. List of Command Functions 15
B. HESC Function Command Details 17
1. Function0Cmd() 0x00 17
2. BattTempCmd() 0x08 22
3. BattVCmd () 0x09 22
4. BattICmd () 0x0A 22
5. BattIavgCmd () 0x0B 22
6. BattRemCapCmd () 0x0F 23
7. ChargerSpecInfoCmd () 0x11 23
8. ChargerModeCmd () 0x12 23
9. ChargerStatusCmd () 0x13 24
10. ChargingCurrentCmd () 0x14 26
11. ChargingVoltageCmd () 0x15 27
12. AlarmWarningCmd () 0x16 27
13. GetVersionCmd () (0x3E) 28
14. SerialNumber () (0x3F) 28
15. GetI2CTempCmd0 () (0x80) to GetI2CtempCmd7() (0x87) 29
16. TempThCmd () (0x90) 29
17. MainVCmd () (0x91) 29
18. MainICmd () (0x92) 29
19. InputPwrCmd () (0x93) 30
20. BattPwrCmd () (0x94) 30
21. ChCycleCmd () (0x95) 30
22. ChTermLastCmd () (0x96) 30
23. ShutDownCmd () (0x97) 31
24. PowerSupplyStatusCmd () (0x98) 31
25. SDSUCauseCmd () (0x99) 32
26. I2ClowAlarmsCmd () (0x9A) 33
27. I2ChighAlarmsCmd () (0x9B) 33
28. V5Cmd () 0x9C, V12Cmd() 0x9D, V3Cmd() 0x9E 33
29. ActiveEECmd() (0xA0) 34
30. EEpromCmd () (0xA1) 34
31. ActiveEEI2CCmd () (0xA2) 34
32. EEPromI2CCmd () (0xA3) 35
33. I2CLogPointerCmd () (0xA4) 36
34. I2CFcnCmd () (0xA5) 36
35. I2CFcnDataCmd () (0xA6) 37
36. ProfileCmd () (0xA7) 37
37. ProfileBankCmd() (0xA8) 38
38. RTCloCmd () (0xB0), RTChiCmd() (0xB1), [32-bit name is RTC32] 38
39. RTCSUloCmd () (0xB2), RTCSUhiCmd() (0xB3), RTCSDloCmd () (0xB4), RTCSDhiCmd() (0xB5), [32-bit names are RTCSU32, RTCSD32] 39
Section 8 : EEprom Setpoints and Configuration Variables 40

A. List of EEprom Setpoints and Configuration Variable 40
B. SerBus EEprom Details 42
1. ChTermDef - EEprom Locations 0x00, 0x20, 0x40, 0x60: 42
2. BattMaxCapDef - EEprom Locations 0x02 0x22 0x42 0x62 43
3. Future - EEprom Locations 0x04 0x24 0x44 0x64 43
4. BattVmaxDef - EEprom Locations 0x06, 0x26, 0x46, 0x66: 43
5. BattVmaxTimeDef - EEprom Locations 0x08, 0x28, 0x48, 0x68: 43
6. BattVdeltaDef - EEprom Locations 0x0A, 0x2A, 0x4A, 0x6A: 43
7. TimeMaxDef - EEprom Locations 0x0C, 0x2C, 0x4C, 0x6C: 44
8. BattIminDef - EEprom Locations 0x0E, 0x2E, 0x4E, 0x6E: 44
9. BattImaxDef - EEprom Locations 0x10, 0x30, 0x50, 0x70: 44
10. TimeTermEnDef - EEprom Locations 0x12, 0x32, 0x52, 0x72: 44
11. BattTempCompDef - EEprom Locations 0x14, 0x34, 0x54, 0x74: 45
12. BattVDef - EEprom Locations 0x16, 0x36, 0x56, 0x76: 45
13. BattIDef - EEprom Locations 0x18, 0x38, 0x58, 0x78: 45
14. BattTempRateDef - EEprom Locations 0x1A, 0x3A, 0x5A, 0x7A: 46
15. BattTrickleDef - EEprom Locations 0x1C, 0x3C, 0x5C, 0x7C: 46
16. BattTrickleTimeDef - EEprom Locations 0x1E, 0x3E, 0x5E, 0x7E: 46
17. ChFlagsDef - EEprom Locations 0x80, 0x81; 46
18. BattLowVoltageDef – EEprom Locations 0x82, 0x83 47
19. BattLowCapacityDef – EEprom Locations 0x84, 0x85 47
20. MainPwrMaxDef – EEprom Locations 0x86, 0x87 48
21. MaxBusTimeDef – EEprom Locations 0x88 48
22. CHCycleMaxDef – EEprom Locations 0x89 48
23. BattTempMinDef - EEprom Locations 0x8A: 48
24. BattTempMaxDef - EEprom Locations 0x8C: 49
25. BattVminDef - EEprom Locations 0x8E: 49
26. ChTempSelectDef (0x90), Ch2TempSelectDef (0x92): 49
27. I2CpollTimeDef - EEprom Locations 0x92, 0x93: 50
28. I2CTsICenDef - EEprom Locations 0x94: 50
29. I2CDevEnDef - EEprom Locations 0x95: 50
30. I2CTsAlmDef - EEprom Locations 0x9A, 0x9B: 51
31. I2CTsNormDef - EEprom Locations 0x9C, 0x9D: 51
32. ChFlags_ExtDef – EEprom Locations 0x9E, 0x9F: 52
33. PWRSDdebDef (0xD0), PWRSUdebDef (0xD2), IGNSDdebDef (0xD8), IGNSUdebDef (0xDA) 52
34. PWRSDDef (0xD4), IGNSDDef (0xDC), PBSDDef (0xE0), BATTSDDef (0xE4), Cmd98SDDef (0xE6), TempSDDef (0xEA) 53
35. PWRSUDef (0xD6), IGNSUDef (0xDE), PBSUDef (0xE2), Cmd98SUDef (0xE8), TempSUDef (0xEC) 53
36. RTCSDDef(0xEE) 53
37. RTCSDEloDef(0xF0), RTCSDEhiDef(0xF2), [32-bit name is RTCSDE32Def] 53
38. RTCSUloDef(0xF4), RTCSUhiDef(0xF6) , [32-bit name is RTCSUE32Def] 54
39. BattMaxCapDef – EEprom Locations 0xFA, 0xFB 54
40. BattRemCapDef – EEprom Locations 0xFC, 0xFD 54
41. BattTime2RechargeDef - EEprom Locations 0xFE, 0xFF: 55
Section 9 : Temperature Sensor, TS-IC and Controller, TC-IC 56
Section 10 : Battery Charging Algorithms and Strategies 57
A. Sealed Lead Acid, SLA 57
1. Single Stage Charging 57
2. Dual Stage Charging 57
B. Dual Battery Support – HESC-SERD, HESC104+, and V5SC 58
Section 11 : HESC-UPS18 FIRMWARE UPDATE PROCEDURE 59
Section 12 : Implementing a smart “Watchdog Timer” on the Host CPU 60
Section 13 : HESC-UPS LED Operation 61
Section 14 : Adding an External Real Time Clock to a HESC 62
Section 15 CRC-8 Firmware Implementation 63

List of Tables

Table 1 : List of HESC Command Functions 16
Table 2 : Power Supply Supported Calibration 17
Table 3 : Function 0 Bit map values 18
Table 4 : Index for calibration values 19
Table 5 : Calibration Range Values 20
Table 6 ChargerSpecInfoCmd bit map 23
Table 7 ChargerModeCmd bit map 24
Table 8 ChargerStatusCmd bit map 25
Table 9 AlarmWarningCmd bit map 27
Table 10 HESC Model String Data 28
Table 11 ChTermLastCmd bit map 31
Table 12 PowerSupplyStatusCmd bit map 31
Table 13 SDSUCauseCmd bit map 32
Table 14 I2ClowAlarmsCmd() (0x9A) 33
Table 15 I2ChighAlarmsCmd() (0x9B) 33
Table 16 ActiveEEI2CCmd() (0xA2) 35
Table 17 : Write Function 36
Table 18 Read Function: 36
Table 19 I2CFcnCmd() High byte: 36
Table 20 ProfileBankCmd() (0xA8) 38
Table 21 HESC Setpoints and Configuration Variables 40
Table 22 ChTerm bit map 42
Table 23 ChFlags bit map 47
Table 24 Temperature device selection values: 49
Table 25 TS-IC and TC-IC device enable settings: 50
Table 26 I2CdevEnDef bit flags: 50
Table 27 I2CTsAlmDef SD & SU action definitions. 51
Table 28 I2CTsAlmDef SD & SU action definitions. 51
Table 29 ChFlags_Ext bit map 52
Table 30 Temperature Sensor and Temperature Controller bit map 56
Table 31 SLA Single Stage Charging Set-up 57
Table 32 SLA Two Stage Charging Set-up 58
Table 33 Heartbeat LED toggle rates 61
Table 34 Charger LED Operation 61
Table 35 Location of Interface Signals on HESC 62

Section 1 : HESC-UPS18 Introduction

The HESC-UPS18 is the third generation of the firmware for the HESC products. The HESC-UPS18 operates on HESC products with built-in Microchip PIC18 microprocessors. The HESC-UPS18 firmware supports all the features of the HESC-UPS firmware and adds many new functions. The HESC-UPS18 firmware cannot be loaded into a PIC16 processor, however the PIC16 on the HESC104 can be replaced with a PIC18 that does support the HESC-UPS18 firmware. New smart power supply products such as the V5SC-SER, V5SC104, HESC104+, HESC-SERD, HPSC104-SER are supplied standard with the PIC18 processor.

Many new advanced features, and improved performance levels are provided with the HESC-UPS18 firmware. Features such as Real-time-clock (RTC) support, dual-battery support, Watchdog mode, improved measurements on voltage and current readings, data storage in external EEproms and start-up and shutdown on temperatures measured with digital temperature sensors. The Master Mode alert, which was first available in the HESC-FW and removed in the HESC-UPS, is returned with the HESC-UPS18 firmware.

The High Efficiency and Smart Charging (HESC) product line that the HESC-UPS18 firmware supports includes the HESC-SER, HESC-SERD, HESC104+, V5SC-SER, V5SC104, HESC104 and HPSC104-SER. Command, control, monitoring and datalogging via the SMBus are the same for all the models. In addition, the HESC-SER, HESC-SERD, V5SC-SER, HPSC104-SER and the HESC104+ (through an on-board Uart) have an RS232 serial port. The HESC104, V5SC104 and the HPSC104-SER have a PC/104 bus interface (Note: The HPSC104-SER has both an RS232 and a PC/104 serial port). The serial port and PC/104 bus interfaces are in most instances connected only to the Host CPU, however other devices may also make use of these ports. For example, the HPSC104-SER has both an RS232 and a PC/104 bus. The Host PC/104 CPU can communicate with the HPSC104-SER via the PC/104 bus, and a remote monitoring system (laptop for example), could connect to the HPSC104-SER via the RS232 serial port. Both the Host PC/104 CPU and the remote Laptop would have full access and control of the HPSC104-SER.

Throughout this manual, use of the term HESC refers to any of the HESC products, HESC-SER(D) refers to any of the serial port models, and HESC104(D) refers to any of the PC/104 connected models. When reference to a specific model is made, the specific model name will be used. From the command function point of view; there are few differences between the different models even though they have different Host interfaces. 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 programmed cycle, then charging will be terminated. Additionally, the Host can directly command the HESC to change charging cycles.

A powerful feature of the HESC is the separate Shutdown (SD) and Start-Up (SU) interval timers. The SD and SU interval timers can be used separately, but when used together create a special situation where the HESC is commanded to shutdown after the SD interval, stay off for the SU interval and then restart the outputs. The HESC-UPS18 uses this feature to create a "Watchdog" mode by setting both SD and SU active. If the Host CPU doesn't "reset" the SD interval regularly the HESC will do a complete shutdown and restart. A remote data logging system could take advantage of the Watchdog mode by setting the SD interval short, and the SU interval long. The remote data logging system would start-up, complete it's measurement and control actions, and then the system would shut down and the process would be repeated endlessly.

New in the HESC-UPS18 firmware is support for an external Real Time Clock (RTC). The HESC can be awakened or put to sleep precisely at any second up to the year 2136AD. In addition, the Host CPU can read or set the RTC, RTC start-up time or RTC shutdown time through the SerBus or through the PC/104 bus. The ability to wake up at exactly the right time will decrease energy requirements. And while the HESC is "hibernating", the total system energy will be only that of the RTC, and that typically is less than 2microamperes.

Also new in the HESC-UPS18 firmware is support for external I2C EEproms. Up to seven 64Kbytes or three 128Kbytes devices are supported and with the 512bytes in the RTC a total of 459,256bytes non-volatile bytes of storage. The Host CPU can read or write any of to any of the EEprom storage. The HESC-UPS18 firmware can use this EEprom storage for datalogging HESC operational data and events for later analysis.

The HESC-UPS18 temperature monitoring has alarm setpoints for high and low temperatures. When the Temperature Shutdown or Start-Up interval timers are set, the HESC can turn-off and turn-on its outputs in response to high and low temperatures.

Section 2 : Configuring the HESC.

The Profile Setup Wizard (PSW.exe) provides an easy method to create a profile to configure the smart power management features and the battery charging functions of the HESC power supplies. The profile is created step by step and a full explanation is provided for each step. Existing profiles can be reviewed and also edited. The profiles are saved in the PSD hex file format, which was introduced with the SCU.exe version 3.2. The PSW.exe is closely linked with the SCU.exe (V3.2). The SCU.exe will open the PSW.exe when a request to create a new profile (FILE, NEW). The PSW.exe can also be called by the SCU.exe through TOOLS, PSW. When the PSW starts up, the existing profile is the default item in the load menu.

Section 3 : Start-up and Shutdown settings, delays, and functions

The HESC-UPS18 firmware is a sophisticated power management system that is loaded into the HESC products utilizing the Microchip PIC18F processors. The HESC-UPS18 works well with advanced operating systems such as Windows, and Linux. It is imperative with advanced operating systems that orderly shutdown of the files is done by the application program and operating system. Failure to allow for orderly shutdown can cause data loss, intermittent crashes, or a complete unrecoverable system failure.
The HESC-UPS18 firmware has three internal modes that control the start up and shut down of the HESC outputs.

1. Debounce
   HESC-UPS18 firmware "debounces" the main power and the IGN input start up and shut down requests by requiring the cause of the request to remain valid for the debounce interval. There is separate shut down and start up debounce definitions for both main power and the IGN input. The debounce time can be set from zero to 65535 seconds. The HESC-UPS18 provide this debouncing to prevent nuisance shut downs or start-ups.

2. Start-Up
   The HESC-UPS18 firmware sets the Start-Up request flag, and begins the Start-Up mode after receiving a start up request. (The Start-Up request flag can be read with PowerSupplyStatusCmd.) The Start-Up mode is initiated by loading the Start-Up timer with the Start-Up interval assigned to the Start-Up request cause. If the HESC is already in Start-Up mode when a new Start-Up request is received, the Start-Up timer will be loaded with the new Start-Up interval if it is less than the existing Start-Up timer count. When the Start-Up timer expires, the HESC outputs are turned on. During Start-Up mode, the CPU LED will blink rapidly (2 Hz). If a Shutdown request is received while a Start-Up request is active, the current active Start-Up request will be cancelled.

   Start-Up requests are generated by:
   - When the HESC has main power applied.
   - When the IGN input of the HESC-SER(D) is taken to the "On" state. The polarity of the IGN is programmable, so the state of the input (high or low) will depend on the polarity setting.
   - When the SD input of the HESC104 is taken to the "On" state. The polarity of the SD is programmable, so the state of the input (high or low) will depend on the polarity setting.
   - When the Pushbutton Shutdown (SD) input is energized, and the HESC-SER(D) is off or is in the shutdown mode.
   - When the Pushbutton Shutdown (PS) input is pulled to Common, and the HESC104 is off or is in the shutdown mode.
   - When the Host CPU writes command PowerSupplyStatusCmd with bit 7 set, either through the SerBus, PC/104 bus or the I2C/SMBus.

- o A temperature measured with an I2C digital sensor going into an alarm condition, or returning to normal from an alarm condition.
- o When the “real-time”, as kept by an RTC attached to the HESC, reaches its Start-up time.

Note: Setting a Start-Up interval to zero prevents a Start-Up request being issued.

3. Shutdown

The HESC-UPS18 firmware sets the Shutdown request flag, and begins the Shutdown mode after receiving a shut down request. (The Shutdown request flag can be read with PowerSupplyStatusCmd.) The Shutdown mode begins by loading the Shutdown timer with the Shutdown interval assigned to the Shutdown request cause. If the HESC is already in Shutdown mode when a new Shutdown request is received, the Shutdown timer will be loaded with the new Shutdown interval if it is less than the existing Shutdown timer count. When the Shutdown timer expires, the HESC outputs are turned off. During Shutdown mode, the CPU LED will blink slowly (0.5 Hz).). If a Start-Up request is received while a Shutdown request is active, the Start-Up request will be "registered" and take effect after the Shutdown mode is complete. The only method to cancel a Shutdown mode in progress is by a Host CPU using command PowerSupplyStatusCmd with bit 8 cleared.

Shutdown requests are generated by:

- o When the HESC main power is removed.
- o When the IGN input of the HESC-SER(D) is taken to the "Off" state. The polarity of the IGN is programmable, so the state of the input (high or low) will depend on the polarity setting.
- o When the SD input of the HESC104 is taken to the "Off" state. The polarity of the IGN is programmable, so the state of the input (high or low) will depend on the polarity setting.
- o When the Pushbutton Shutdown (SD) input is energized, and the HESC-SER(D) is on.
- o When the Pushbutton Shutdown (PS) input is pulled to Common, and the HESC104 is on.
- o When the Host CPU writes command PowerSupplyStatusCmd with bit 8 set, either through the SerBus, PC/104 bus or the I2C/SMBus.
- o When the Host CPU writes command ShutDownCmd to the HESC through the SerBus, PC/104 bus or the I2C/SMBus.
- o A temperature measured with an I2C digital sensor going into an alarm condition, or returning to normal from an alarm condition.
- o When the “real-time”, as kept by an RTC attached to the HESC, reaches its Shutdown time.

When the battery voltage or battery capacity falls below the BattLowVoltageDef or BattLowCapacityDef levels.
Note: Setting a shut down interval to zero prevents a Shutdown request being issued.

Section 4 : HESC-SER(D) SerBus Communications

A. Description:

The HESC-SER(D) 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. Normally, the HESC-SER(D) operates in a "slave" mode where the Host initiates all the communications between the HESC-SER(D) and the Host. By setting the MasterModeEn bit in the ChFlags default EEprom register, the HESC-SER(D) can use the Master Mode to alert the Host CPU of a status change. Please refer to “Section 7B, Paragraph 9” for more details on Master Mode alerts to the Host CPU.

- 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 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(D) 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 are sent from the HESC-SER(D) and data sent to the 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: A Write is defined as the action the command places on the HESC-SER(D) RAM and EEprom. Therefore, Read and Write have the same meaning for Master and Slave modes:

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

HESC-SER(D): <addressW>	<command>	< databyteW low >	< databyteW high>
HOST: [0x00]	[0x01]	[0x2]	[0xFF]

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

HESC-SER(D): <addressW>	<command>	< databyteW low >	< databyteW high>	< checksum>
HOST: [0x00]	[0x01]	[0x2]	[0x03]	[0xFF]

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

HESC-SER(D): <addressW> <command> <databyteW low> <databyteW high>
HOST: ***HOST does not Ack*

where "host" addressR = 0001 000 + 0 (R/W bit) = 0x10
"host" addressW = 0001 000 + 1 (R/W bit) = 0x11

Refer to Section 15 for details on the 8-bit CRC checksum.

C. Slave Mode:

1. Commands are sent from the Host to the HESC-SER(D) and data is sent to or received from the Host CPU to the HESC-SER(D) over the SerBus. The transaction is invalid and commands/data are not to be used until:
   a. the final acknowledge 0xFF is received
   b. the checksum matches the transmitted data (checksum is optional, but if sent must be used)

Communication format for a write communication transaction (format shown is for bus timer enabled and with checksum).

HOST:	<addressW>	<command>	<databyteW low >	<databyteW high >	<checksum>
HESC-SER(D):	[ACK0]	[ACK1]	[ACK2]	[ACK3]	[TACK]

Where:
AddressW = 0001 001 + 0 (R/W bit) = 0x12.

Acknowledge handshaking (ACK0, ACK1, ACK2 & ACK3).
ACK0 = 0x00 if valid addressW (or addressR, see Read format below) detected, otherwise ACK0 = 0xF0.
ACK1 = 0x01, or 0xF0 if bus timer expired.
ACK2 = 0x02, or 0xF0 if bus timer expired.
ACK3 = 0x03, or 0xF0 if bus timer expired.
Checksum = 8-bit CRC checksum. Refer to Section 15 for details.

And:
Termination acknowledge values (TACK).
TACK = 0xFF if transaction completes successfully.
TACK = 0xF0 if bus timer expired or incorrect checksum detected.
TACK = 0xFE if an invalid command was attempted, or any invalid range specified.

Communication format for a read communication transaction (format shown is for bus timer enabled and with checksum).

HOST:	<addressR>	<command>	<ACK2>	<ACK3>	<TACK>
HESC-SER(D):	[ACK0]	[databyteR low]	[databyteR high]	[checksum]	

Where:
AddressR = 0001 001 + 1 (R/W bit) = 0x13.

Acknowledge handshaking (ACK0, ACK1, ACK2 & ACK3)
ACK0 = 0x00 if addressR (or addressW, see Write format above) detected, otherwise ACK0 = 0xF0.
ACK2 = 0x02, host may cancel a read communication transaction by setting ACK2 != 0x02
ACK3 = 0x03, host may cancel a read communication transaction by setting ACK3 != 0x03
Checksum = 8-bit CRC checksum of address, command & data. If requested command is not supported by HESC-SER(D), then checksum will be returned with an “inverted” checksum, and the data returned will be zero. (each bit inverted, ex: 0xBF = ~0x40) Refer to Section 15 for details of the 8-bit CRC checksum.

And:
Termination acknowledge values (TACK)
TACK = 0xFF if transaction completes.

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

3a. Write command without checksum acknowledge.

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

3b. Write command with checksum acknowledge.

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

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

HOST:	<addressW> <command> <databyteW low > <databyteW high >				**{HOST does not need to wait for Acks}*
HESC-SER(D):	[0x00]**	[0x01]**	[0x02]**	{0xFF]	

4a. Read command without checksum acknowledge.

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

4b. Read command with checksum acknowledge.

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

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

HOST:	<addressR> <command>	**{HOST does not need to wait for ACK0}*	
HESC-SER(D):	[0x00]**	[databyteR low] [databyteR high]	

5a. Block Read command (SerialNumber(0x3F)) without checksum acknowledge.

HOST:	<addressR>	<0x3F>	<0x02>	<0x03>…	…<0x10>	<0xFF>
HESC-SER(D):	[0x00]	[SerByte1]	[SerByte2]...	…[SerByte15]	[SerByte16]	

5b. Block Read command (SerialNumber(0x3F)) with checksum acknowledge.

HOST:	<addressR>	<0x3F>	<0x02>	<0x03>…	…<0x10>	<0x11>	<0xFF>
HESC-SER(D):	[0x00]	[SerByte1]	[SerByte2]...	…[SerByte15]	[SerByte16]	[checksum]	

5a. Block Read command (SerialNumber(0x3F)) without checksum acknowledge and without bus time enabled.

HOST:	<addressR><0x3F>	**{HOST does not need to wait for ACK0}*	
HESC-SER(D):	[0x00]**	[SerByte1] [SerByte2]... …[SerByte15] [SerByte16]	

Section 5 : HESC104(D) PC/104 Bus Communications

A. Description:

The HESC104(D) 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 PC/104 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 HESC104(D), and an I/O read will read the data from the HESC104(D). Whenever the HESC104(D) puts data in it's output port for the Host PC/104 computer, it generates a PC/104 bus interrupt (IRQ5 or IRQ7, see section on setting jumpers). The HESC104(D) will remove the interrupt signal after the Host performs I/O read on the HESC104(D) I/O port. Normally, the HESC104(D) operates in a "slave" mode where the Host initiates all the communications between the HESC104(D) and the Host. By setting the MasterModeEn bit in the ChFlags default EEprom register, the HESC104(D) can use the Master Mode to alert the Host CPU of a status change. Please refer to “Section 7B, Paragraph 12” for more details on Master Mode alerts to the Host CPU.

- 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(D) or Host) must acknowledge receipt of each byte. *(The HESC104(D) and HESC-SER(D) differ in that the HESC104(D) PC/104 bus communications does not have a non-bus timer mode.)* If an acknowledge byte doesn't match the acknowledge number expected then the communication transaction is aborted immediately.

An "enhanced" communication mode is available by adding an optional checksum value. If the HESC104(D) 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. The communication will return to non-checksum mode when the Host CPU sends an acknowledge 0xFF after [databyteR high] instead of the 0x03.

B. Master Mode:

1. Commands are sent from the HESC104(D) and data sent to the 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: A Write is defined as the action the command places on the HESC1-4(D) 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(D):	<addressR>	<command>	<databyteR low>	<databyteR high>
HOST:	[0x00]	[0x01]	[0x02]	[0xFF]

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

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

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

HESC104(D):	<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

Refer to Section 15 for details on the 8-bit CRC checksum.

C. Slave Mode:

1. Commands are sent from the Host to the HESC-SER(D) and data is sent to or received from the Host CPU to the HESC-SER(D) over the PC/104 Bus. The transaction is invalid and commands/data are not to be used until:
   a. the final acknowledge 0xFF is received
   b. the checksum matches the transmitted data (checksum is optional, but if sent must be used)

Communication format for a write communication transaction (format shown is for bus timer enabled and with checksum).

HOST:	<addressW>	<command>	<databyteW low >	<databyteW high >	<checksum>
HESC-SER(D):	[ACK0]	[ACK1]	[ACK2]	[ACK3]	[TACK]

Where:
AddressW = 0001 001 + 0 (R/W bit) = 0x12.

Acknowledge handshaking (ACK0, ACK1, ACK2 & ACK3).
ACK0 = 0x00 if valid addressW (or addressR, see Read format below) detected, otherwise ACK0 = 0xF0.
ACK1 = 0x01, or 0xF0 if bus timer expired.
ACK2 = 0x02, or 0xF0 if bus timer expired.
ACK3 = 0x03, or 0xF0 if bus timer expired.
Checksum = 8-bit CRC checksum. Refer to Section 15 for details.

And:
Termination acknowledge values (TACK).
TACK = 0xFF if transaction completes successfully.
TACK = 0xF0 if bus timer expired or incorrect checksum detected.
TACK = 0xFE if an invalid command was attempted, or any invalid range specified.

Communication format for a read communication transaction (format shown is for bus timer enabled and with checksum).

HOST:	<addressR>	<command>	<ACK2>	<ACK3>	<TACK>
HESC-SER(D):	[ACK0]	[databyteR low]	[databyteR high]	[checksum]	

Where:
AddressR = 0001 001 + 1 (R/W bit) = 0x13.

Acknowledge handshaking (ACK0, ACK1, ACK2 & ACK3)
ACK0 = 0x00 if addressR (or addressW, see Write format above) detected, otherwise ACK0 = 0xF0.
ACK2 = 0x02, host may cancel a read communication transaction by setting ACK2 != 0x02
ACK3 = 0x03, host may cancel a read communication transaction by setting ACK3 != 0x03
Checksum = 8-bit CRC checksum of address, command & data. If requested command is not supported by HESC-SER(D), then checksum will be returned with an “inverted” checksum, and the data returned will be zero. (each bit inverted, ex: 0xBF = ~0x40) Refer to Section 15 for details of the 8-bit CRC checksum.

And:
Termination acknowledge values (TACK)
TACK = 0xFF if transaction completes.

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 >	
HESC-SER(D):	[0x00]	[0x01]	[0x02]	[0xFF]	

3b. Write command with checksum acknowledge.

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

4a. Read command without checksum acknowledge.

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

4b. Read command with checksum acknowledge.

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

5a. Block Read command (SerialNumber(0x3F)) without checksum acknowledge.

HOST:	<addressR>	<0x3F>	<0x02>	<0x03>…	…<0x10>	<0xFF>
HESC-SER(D):	[0x00]	[SerByte1]	[SerByte2]...	…[SerByte15]	[SerByte16]	

5b. Block Read command (SerialNumber(0x3F)) with checksum acknowledge.

HOST:	<addressR>	<0x3F>	<0x02>	<0x03>…	…<0x10>	<0x11>	<0xFF>
HESC-SER(D):	[0x00]	[SerByte1]	[SerByte2]...	…[SerByte15]	[SerByte16]	[checksum]	

Section 6 : HESC I²C/SMBus Bus Communications

A. Description:

The HESC-SER(D) can communicate with System Management Bus (SMBus) batteries, I²C digital temperature sensors, and Hosts and microprocessors through the I2C/SMBus. The SMBus is a two-wire interface through which the HESC-SER(D) can communicate to I²C/SMBus devices. The HESC supports I²C/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 I²C/SMBus device acting as a master, and another I²C/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, I²C/SMBus provides an arbitration mechanism, relying on the wired-AND connection of all I²C/SMBus interfaces to the I²C/SMBus.

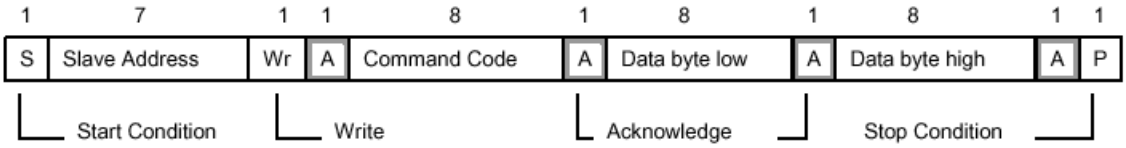
NOTE: *Care should be taken in the design of both the input and output stages of SMBus devices, in order not to load the bus when their power plane is turned off.*

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 I²C/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.

Figure 1 : **SMBus Write Word Protocol**



B. Master Mode:

The HESC charger acts in master mode when an SMBus battery is attached and the polling variable (I2CpollTimeDef) is set to non-zero. The HESC then has the ability to query the Smart Battery to determine the Smart Battery's charging requirements. The HESC charger polls the battery using the ChargingVoltageCmd(), ChargingCurrentCmd() and the ChargerStatusCmd() functions to determine the charging requirements. Polling must be set for at least once per minute in order to maintain safe charging. Polling more frequently is recommended, however performance may not improve when polling more frequently than every five seconds.

Note: In order to ensure proper charging of ALL battery chemistries, it is required that a minimum polling frequency of once per minute be used.

Section 7 : 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 it 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
0x00	Function0Cmd()	✓	✓	✓	R/W	Bit Flags	N/A	Enable Calibration Mode, override Bus Timer mode
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	-32768 to 32767	Read battery current
0x0B	BattIavgCmd()	✓	✓	✓	R	mA	-32768 to 32767	Read 1 minute rolling average battery current
0x0F	BattRemCapCmd()	✓	✓	✓	R	mAh	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	mA	0 to 65535	Read battery charging current setpoint
0x15	ChargingVoltageCmd()	✓	✓	✓	R/W	mV	0 to 65535	Read battery charging voltage 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	GetI2CtempCmd0()	✓	✓2	✓	R	0.1K *	0 to 6553.5	Read I2C temperature
to								
0x87	GetI2CtempCmd7()	✓	✓2	✓	R	0.1K *	0 to 6553.5	Read I2C temperature
0x8F	FanSpeedPercent()	✓	✓	✓	R	percent	0 to 100	Read fan speed in percent
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	ChTermLastCmd()	✓	✓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.
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	Read Low alarm status of I2C temperature sensors
0x9B	I2ChighAlarmsCmd()	✓	✓2	✓	R	Bitflags	N/A	Read High alarm status of I2C temperature sensors
0x9C	V5Cmd()	✓	✓2	✓	R	mV	0 to 65535	Read +5 volt output; HESC-SERD, HESC104+, V5SC
0x9D	V12Cmd()	✓	✓2	✓	R	mV	0 to 65535	Read +12 volt output; HESC-SERD, HESC104+
0x9E	V3Cmd()	✓	✓2	✓	R	mV	0 to 65535	Read +3.3 volt output; HESC104+
0xA0	ActiveEEcmd()	✓	✓2	✓	R/W	bitflag.byte	0 to 256	Read/Set active EEprom location and auto increment
0xA1	EEPromCmd()	✓	✓2	✓	R/W	Word	0 to 65535	Read/Write value to active EEprom location
0xA2	ActiveEEI2CCmd ()	✓	✓2	✓	R/W	Bitflags	N/A	Enables I2C EEprom and auto increment
0xA3	EEPromI2CCmd ()	✓	✓2	✓	R/W	Word	0 to 65535	Read/Write I2C EEprom address and data
0xA4	I2ClogPonterDef()	✓	✓2	✓	R/W	Word	0 to 65535	Read/Write I2C EEprom Log pointer
0xA5	I2CFcnCmd()	✓	✓2	✓	R/W	byte.byte	N/A	I2C Function, I2C Address, R/W flag
0xA6	I2CFcnDataCmd()	✓	✓2	✓	R/W	Word	0 to 65535	I2C Data
0xA7	ProfileCmd()	✓	✓2	✓	R/W	Bitflags	N/A	Save/restore customized profile versions.
0xA8	ProfileBankCmd()	✓	✓2	✓	R/W	integer	0 to 3	Select EEprom profile bank
0xB0	RTClo()	✓	✓2	✓	R/W	Word	0 to 65535	Read/Set the RTC time (low 16-bits)
0xB1	RTChi()	✓	✓2	✓	R/W	Word	0 to 65535	Read/Set the RTC time (high 16-bits)
0xB2	RTCSUlo()	✓	✓2	✓	R/W	Word	0 to 65535	Read/Set the RTC SU time (low 16-bits)
0xB3	RTCSUhi()	✓	✓2	✓	R/W	Word	0 to 65535	Read/Set the RTC SU time (high 16-bits)
0xB4	RTCSDlo()	✓	✓2	✓	R/W	Word	0 to 65535	Read/Set the RTC SD time (low 16-bits)
0xB5	RTCSDhi()	✓	✓2	✓	R/W	Word	0 to 65535	Read/Set the RTC SD time (high 16-bits)

✓1 SMBus Optional Manufacturer Functions
✓2 Function not defined by SMBus standards
*Kelvin = -273.2°Celsius

B. HESC Function Command Details

1. Function0Cmd() 0x00

Description:
Function0Cmd() is used to override the bus timer mode. Function0Cmd() can also change the mode of operation of ActiveEEcmd() and EEPromCmd(). The "Normal Mode" of operation for ActiveEEcmd() and EEPromCmd() is for accessing the power supply profile defaults stored in the EEprom. The Function0Cmd() can change the mode of operation for ActiveEEcmd() and EEPromCmd() to the "Calibration Mode" (see Table 2 for supported Tri-M Engineering Smart power supplies) or to the "RAM Access Mode" to read/write select profile defaults that have been loaded from the EEprom storage into the microprocessor RAM registers. The HESC-UPS18 firmware always boots to the Normal Mode of operation.

Purpose:
Bus Timer Override: Disabling the Bus Timer allows the User to manually enter and read communication values without the Bus Timer timing out and resetting the communication in progress to "idle" mode. When the Bus Timer mode is disabled, a serial terminal program (one that supports hex values) can be used for a serial based power supply, or a debug program (such as comes with DOS) for a PC/104 based power supply.

Calibration Mode: The Tri-M Engineering power supplies that support the HESC-UPS18 firmware have output voltages that in some models can be "calibrated" by a PWM (pulse width modulated) signal from the on-board microprocessor (refer to Table-2 that lists each model and the outputs available for calibrating). The calibration values are stored in the top 64 bytes of flash memory in the on-board microprocessor.

Table 2 ***: Power Supply Supported Calibration***

Power Supply Model	Outputs that support Calibration Mode			
	Battery Charger	3.3 volt	5 volt	12 volt
V5SC-SER V4	Yes	N/A	Yes	N/A
V5SC104	Yes	N/A	Yes	N/A
HESC104 V6	Yes	N/A	No	Yes*
HESC104+	Yes	No	No	No
HESC-SER V3.2	Yes	N/A	Yes	Yes
HESC-SERD V3.2	Yes	N/A	Yes	Yes
HPSC104-SER	Yes	Yes	Yes	Yes
HPS3512 V3	N/A	Yes	Yes	Yes

* Note: HESC104 V6 12V output has a one stage ~100mV calibration boost adjustment.

RAM Access Mode: (Added in HESC-UPS18 version 4.10) In addition, the power supply "profile" that is stored in EEprom is loaded into RAM (random access memory) registers when the power supply "boots". These registers configure how the power supply will operate and are not accessible during Normal Mode operation. Some of these registers can be accessed using the RAM Access Mode.

Protocol: Read/Write Word

Range: 16-bit word - bit mapped

Table 3 **: Function 0 Bit map values**

Bit#	Field	Description
0	Bus Time Mode	Overrides the EEprom Bus Timer mode. When the power supply is "rebooted" the Bus Timer mode defined by the EEprom value will be restored. Note: The Bus Timer bit-0 must be set to the desired value whenever the Function0Cmd(0x00) is written. 0 = no Bus Timer disabled 1= Bus Timer enabled
1 2	Calibration and RAM Access Modes	Enables/Disables Calibration and RAM access mode. - Return to Normal Mode to save calibration values to Flash memory. 00 = Normal Mode (Calibration Mode disabled, RAM Access Mode disable) 01 = Calibration Mode enabled (RAM Access Mode disabled and Normal Mode disabled) 10 = RAM Access Mode enabled (Calibration Mode disabled and Normal Mode disabled) 11 = Future

a. Calibration mode.

The Calibration Mode is used to read/modify the calibration values stored in the Flash memory in the on-board microprocessors. The steps required to calibrate an output are:

i. Enable Calibration Mode through Function0Cmd(0x00).
ii. Set the Calibration Index Pointer to the desired calibration value through ActiveEECmd(0xA0).
iii. Write the calibration value through EEpromCmd(0xA1) to a RAM register.
iv. Repeat step iii) until output value is meets calibration requirement.
v. Repeat step ii) and iii) until all outputs calibrated.
vi. Save the calibration values by disabling the Calibration Mode through Function0Cmd(0x00). If power is recycled, or a power supply reboot is requested before saving the calibration values, they will not be saved to Flash and the calibrations will be lost!

Command to enter into Calibration Mode (non-Bus Timer mode set):

HOST:	<0x12>	<0x00>	<0x02 >	<0x00>
HESC:	[0x00]	[0x01]	[0x02]	[0xFF]

Command to enter into Calibration Mode (Bus Timer mode set):

HOST:	<0x12>	<0x00>	<0x03 >	<0x00>
HESC:	[0x00]	[0x01]	[0x02]	[0xFF]

Command to return to Normal Mode (non-Bus Timer mode set):

HOST:	<0x12>	<0x00>	<0x00 >	<0x00>
HESC:	[0x00]	[0x01]	[0x02]	[0xFF]

Command to return to Normal Mode (Bus Timer mode set):

HOST:	<0x12>	<0x00>	<0x01 >	<0x00>
HESC:	[0x00]	[0x01]	[0x02]	[0xFF]

Command to read the Function 0 mode:

HOST:	<0x13>	<0x00>	<0x02 >	<0xFF>
HESC:	[0x00]	[DataLo]	[DataHi]	

b. Calibration Index Pointer Register.

The Calibration Mode uses ActiveEECmd(0xA0) as an index pointer to calibration values.
Note: ActiveEECmd(0xA0) is only an index pointer to calibration values when in the Calibration Mode.

Command to set the index pointer to a calibration value:

HOST:	<0x12>	<0xA0>	<Index >	<0x00>	
HESC:	[0x00]	[0x01]	[0x02]	[0xFF]	

Command to read the Calibration Index Pointer:

HOST:	<0x13>	<0xA0>	<0x02 >	<0xFF>
HESC:	[0x00]	[Index]	[0x00]	

Table 4 **: Index for Calibration Values**

Index	*Calibration Values*
0x00	5V Output
0x01	12V Output
0x02	Battery Charger Output Voltage
0x03	3.3V Output

c. Read/Write calibration values.

The Calibration Mode uses EEpromCmd(0xA1) to read/modify the selected calibration value. Each calibration value is a 16-bit register. When a calibration value is written, the HESC-UPS18 firmware will test the range of the calibration value, and if in range will save the calibration value to a RAM register. If the calibration value is not in range the HESC-UPS18 firmware will return an ACK of 0xFE. The calibration values will only be saved to Flash when the Calibration Mode is disabled.

Command to write a calibration value:

HOST:	<0x12>	<0xA1>	<DataLo >	<DataHi>	
HESC:	[0x00]	[0x01]	[0x02]	[0xFF]	

Command to read a calibration value:

HOST:	<0x13>	<0xA1>	<0x02 >	<0xFF>
HESC:	[0x00]	[DataLo]	[DataHi]	

Example of writing an illegal value where the function error acknowledge is returned:

HOST:	<0x12>	<0xA1>	<DataLo >	<DataHi>
HESC:	[0x00]	[0x01]	[0x02]	[0xFE] ← Function error value returned to host

Table 5 **: Calibration Range Values**

Power Supply Model	Valid outputs ranges			
	Battery Charger	3.3 volt	5 volt	12 volt
V5SC-SER V4	0xFE70 -0x01F4 (-400 to 500)	N/A	0 – 0x03FF	N/A
V5SC104	0xFE70 -0x01F4 (-400 to 500)	N/A	0 – 0x03FF	N/A
HESC104 V6	0xFE70 -0x01F4 (-400 to 500)	N/A	No	0, 1*
HESC104+	0xFE70 -0x01F4 (-400 to 500)	No	No	No
HESC-SER V3.2	0xFE70 -0x01F4 (-400 to 500)	N/A	0 – 0x03FF	0 – 0x03FF
HESC-SERD V3.2	0xFE70 -0x01F4 (-400 to 500)	N/A	0 – 0x03FF	0 – 0x03FF
HPSC104	0xFE70 -0x01F4 (-400 to 500)	0 – 0x03FF	0 – 0x03FF	0 – 0x03FF
HPS3512 V3	N/A	0 – 0x03FF	0 – 0x03FF	0 – 0x03FF

* Note: HESC104 V6 12V output has a one stage ~100mV calibration boost adjustment.

d. RAM Access Mode.

The RAM Mode is used to read/modify the select power supply profile values stored in the RAM registers of the on-board microprocessor. The steps required to access one of these registers:

i. Enable RAM Access Mode through Function0Cmd(0x00).
ii. Set the RAM Index Pointer to the desired RAM register index value through ActiveEECmd(0xA0).
iii. Write the RAM register value through EEpromCmd(0xA1) to a RAM register.
iv. Exit RAM Access Mode and return to Normal Mode operation.

Command to enter into RAM Access Mode (non-Bus Timer mode set):

HOST: <0x12> <0x00> <0x04 > <0x00>
HESC: [0x00] [0x01] [0x02] [0xFF]

Command to enter into RAM Access Mode (Bus Timer mode set):

HOST: <0x12> <0x00> <0x05 > <0x00>
HESC: [0x00] [0x01] [0x02] [0xFF]

Command to return to Normal Mode (non-Bus Timer mode set):

HOST: <0x12> <0x00> <0x00 > <0x00>
HESC: [0x00] [0x01] [0x02] [0xFF]

Command to return to Normal Mode (Bus Timer mode set):

HOST: <0x12> <0x00> <0x01 > <0x00>
HESC: [0x00] [0x01] [0x02] [0xFF]

Command to read the Function 0 mode:

HOST: <0x13> <0x00> <0x02 > <0xFF>
HESC: [0x00] [DataLo] [DataHi]

e. RAM Access Index Pointer Register.

The RAM Access Mode uses ActiveEECmd(0xA0) as an index pointer to RAM registers.
Note: ActiveEECmd(0xA0) is only an index pointer to RAM registers when in the RAM Access Mode.

Command to set the RAM Access Index Pointer to a RAM register:

HOST:	<0x12>	<0xA0>	<Index >	<0x00>
HESC:	[0x00]	[0x01]	[0x02]	[0xFF]

Command to read the RAM Access Index Pointer:

HOST:	<0x13>	<0xA0>	<0x02 >	<0xFF>
HESC:	[0x00]	[Index]	[0x00]	

Table 6 ***: Index for RAM Access Registers***

Index	Register Name	Units	Range	Description
0 (0x00)	PWRSDdebDef	Sec	0 to 65535	Debounce interval for main power loss
1 (0x01)	PWRSUdebDef	Sec	0 to 65535	Debounce interval for main power applied
2 (0x02)	PWRSDDef	Sec	0 to 65535	Shutdown delay for main power loss
3 (0x03)	PWRSUDef	Sec	0 to 65535	Start-Up delay for main power applied
4 (0x04)	IGNSDdebDef	Sec	0 to 65535	Debounce interval for IGN input turned off
5 (0x05)	IGNSUdebDef	Sec	0 to 65535	Debounce interval for IGN input turned on
6 (0x06)	IGNSDDef	Sec	0 to 65535	Shutdown delay for IGN input turned off
7 (0x07)	IGNSUDef	Sec	0 to 65535	Start-Up delay for IGN input turned on
8 (0x08)	PBSDDef	Sec	0 to 65535	Shutdown delay for Pushbutton off
9 (0x09)	PBSUDef	Sec	0 to 65535	Start-Up delay for Pushbutton on
10 (0x0A)	BATTSDDef	Sec	0 to 65535	Shutdown delay for low battery voltage or capacity
11 (0x0B)	Cmd98SDDef	Sec	0 to 65535	Shutdown delay for command PowerSupplyStatusCmd
12 (0x0C)	Cmd98SUDef	Sec	0 to 65535	Start-Up delay for command PowerSupplyStatusCmd
13 (0x0D)	TempSDDef	Sec	0 to 65535	Shutdown delay for alarm temperature condition.
14 (0x0E)	TempSUDef	Sec	0 to 65535	Start-Up delay for return to normal temperature condition.
15 (0x0F)	RTCSDDef	Sec	0 to 65535	Shutdown delay after RTC time reached.
16 (0x10)	RTCSDloDef	Sec	0 to 65535	Shutdown time in seconds from January 1, 2001, 00:00, lo word.
17 (0x11)	RTCSDhiDef	Sec	0 to 65535	Shutdown time in seconds from January 1, 2001, 00:00, hi word.
18 (0x12)	RTCSUloDef	Sec	0 to 65535	Start-Up time in seconds from January 1, 2001, 00:00, lo word.
19 (0x13)	RTCSUhiDef	Sec	0 to 65535	Start-Up time in seconds from January 1, 2001, 00:00, hi word.

f. Read/Write RAM Access Register Values.

The RAM Access Mode uses EEpromCmd(0xA1) to read/modify the selected RAM register. Each RAM Access value is a 16-bit register.

Command to write to a RAM Access Register:

HOST:	<0x12>	<0xA1>	<DataLo >	<DataHi>
HESC:	[0x00]	[0x01]	[0x02]	[0xFF]

Command to read a RAM Access Register:

HOST:	<0x13>	<0xA1>	<0x02 >	<0xFF>
HESC:	[0x00]	[DataLo]	[DataHi]	

2. BattTempCmd() 0x08

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

Purpose:
The BattTempCmd() function provides accurate cell temperatures for use by HESC and Host management system. The HESC is able to use the temperature as a safety check and the Host may use the temperature for 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

Range: 0 to +6553.5°K -- cell temperature in tenth degree Kelvin (0.1°K) increments

3. BattVCmd () 0x09

Description:
Returns the battery voltage.

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

Range: 0 to 65,535 mV -- battery voltage measured at HESC terminals in mV increments.

4. BattICmd () 0x0A

Description:
Returns the current being supplied (or accepted) through the HESC.

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

Range: -32,768 to 32,767mA – in mA increments (0 to -32,768 mA for discharge or 1 to 32,767 mA for charge).

5. BattIavgCmd () 0x0B

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

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

Protocol: Read Word

Range: -32,768 to 32,767mA – in mA increments (0 to -32,768 mA for discharge or 1 to 32,767 mA for charge).

6. BattRemCapCmd () 0x0F

Description:
Returns the predicted remaining battery capacity. The BattRemCapCmd() capacity value is expressed in current.

Purpose:
The BattRemCapCmd() function returns the battery's remaining capacity.

Protocol: Read/Write Word

Range: 0 to 65,5535 x 10mWh -- remaining charge in 10mWh increments

7. 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

Range: 16-bit word - bit mapped – see Table 7.

Table 7 ChargerSpecInfoCmd bit map

Bit#	Field	Description
0…3	CHARGER_SPEC	The ChargerSpecInfoCmd() reports the version of the Smart Battery Charger specification the HESC charger supports. All other values reserved. 0001 – Version 1.0 0010 – Version 1.1 0011 – Version 1.1
4…15	Reserved	These bits are reserved and will return zero.

8. ChargerModeCmd () 0x12

Description:
The Host uses this command to set the various charger modes.

Purpose:
Allows the System Host to configure the charger and change the default modes. The default values are set to allow an SMBus Battery and the HESC to work in concert without requiring a host. 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. Reading will return undefined values.

Protocol: Write Word

Range: 16-bit word - bit mapped - see Table 8.

Table 8 ChargerModeCmd bit map

Bit#	Field	Description
0	INHIBIT_CHARGE	0 - enable charging (power-on default) 1 - inhibit charging
1	ENABLE_POLLING	0 - disable polling (power-on default for Level 2 chargers) 1 - enable polling (power-on default for Level 3 Smart Battery Chargers
2	POR_RESET	0 - mode unchanged (default) 1 - set charger to power-on defaults
3	RESET_TO_ZERO	0 - charging values unchanged (default) 1 - set charging values to zero
4...15	Reserved	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 HESC 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.

9. ChargerStatusCmd () 0x13

Description:
The Host uses this command to read the HESC charger status bits.

Purpose:
Allows the Host to determine the status of the SMBus safety signal, thermistor or temperature sensor, HESC charger status and SMBus level compatibility of the HESC charger. ChargerStatusCmd() function is used by the power management system to get alarm and status bits, as well as error codes from the HESC. It is also used the by the HESC in Master Mode to alert the Host of a change in status or an alarm in the HESC. When the HESC sends the ChargerStatusCmd() to the Host CPU under Master Mode, the communications format is: <0x10>, <0x12>, <datalow>, <datahigh>. This allows the Host to identify the source of the data, and to interpret it correctly.

Protocol: Read Word

Range: 16-bit word - bit mapped - see Table 9.

Table 9 ChargerStatusCmd bit map

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

CHARGE_INHIBITED bit reflects the status of the HESC charger set by the INHIBIT_CHARGE bit.

POLLING ENABLED bit is set/reset in the HESC 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.

10. ChargingCurrentCmd () 0x14

Description:
The Host sends the desired charging rate or reads the charging current setpoint.

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: Read/Write Word

Range: 0 to 65,535 mA -- charger output current set point in mA increments.

11. ChargingVoltageCmd () 0x15

Description:
The Host sends the desired charging voltage or reads the charging voltage setpoint.

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: Read/Write Word

Range: 0 to 65,535 mV – charger output voltage set point in mV increments.

12. AlarmWarningCmd () 0x16

Description:
If an SMBus Battery is connected to the HESC, AlarmWarningCmd() is used by the SMBus battery to notify the HESC that one or more alarm conditions exist. Alarm and status indications are encoded as bit fields. If the HESC is charging a Standard Battery pack, it will set the appropriate fields based on the thermistor or digital temperature sensor.

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: 16-bit word - bit mapped - Status Register with alarm conditions bit mapped as in Table 10:

Table 10 AlarmWarningCmd bit map

Bit#	Field
0	Reserved
1	Reserved
2	Reserved
3	Reserved
4	FULLY_DISCHARGED
5	FULLY_CHARGED
6	DISCHARGING
7	INITIALIZED
8	REMAINING_TIME_ALARM
9	REMAINING_CAPACITY_ALARM
10	Reserved
11	TERMINATE_DISCHARGE_ALARM
12	OVER_TEMP_ALARM
13	Reserved
14	TERMINATE_CHARGE_ALARM
15	OVER_CHARGED_ALARM

13. GetVersionCmd () (0x3E)

Description:
This function returns the firmware version number of the HESC.

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

Protocol: Read Word

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

14. SerialNumber () (0x3F)

Description:
This function returns the serial number of the HESC and the model name. 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-byte serial number**.** Typically the serial number will be made of bytes of value 0x00 to 0x09, but any value from 0x00 to 0xFF is possible.

Table 11 HESC Model String Data

Power Supply Model	Model String Data
V5SC-SER-UPS V4	**"VSC18SU#"**
V5SC-SER V4 (without built-in battery pack)	**"VSC18S^#"**
V5SC104-UPS	**"VSC18PU#"**
V5SC104 (without built-in battery pack)	**"VSC18P^#"**
HESC104 V6	**"HESC18^#"**
HESC104+	**"PCI18^^#"**
HESC-SER V3.2	**"SER18^^#"**
HESC-SERD V3.2	**"SERD18^#"**
HPSC104	**"HPSC18^#"**
HPS3512 V3	**"HPS3512#"**

Note: The "^" represents a space character.

15. GetI2CTempCmd0 () (0x80) to GetI2CtempCmd7() (0x87)

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 heaters can be started if temperatures exceed safe limits. Temperature is a primary charging termination for NiCd and NiMh batteries. The HESC supports both digital I2C sensors and thermistors, and either may be used for battery temperature monitoring.

Protocol: Read Word

Range: 0 to 65,535 x 0.1°K – temperature in 0.1K increments (Kelvin: 0 degC = 273.2°K).

16. 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. Temperature is a primary charging termination for NiCd and NiMh batteries. The HESC supports both digital I2C sensors and thermistors, and either may be used for battery temperature monitoring. The Host CPU may use this information to determine the HESC and battery performance.

Protocol: Read Word

Range: 0 to 65,535 – value from ADC converter.

17. MainVCmd () (0x91)

Description:
Returns the Main Input voltage.

Purpose:
The MainVCmd() function provides the Host power management system with an accurate measure of main input voltage. The HESC compares this value with the value of BattVCmd() to determine if power is supplied from the Main Input or the Battery Input supplies.

Protocol: Read Word

Range: 0 to 65,535 mV – main input voltage in mV increments.

18. MainICmd () (0x92)

Description:
Returns the current being supplied to the HESC.

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 zero if BattIVCmd() is greater than MainVCmd().

Protocol: Read Word

Range: 0 to 65,535 mA – input current rate in mA increments

19. InputPwrCmd () (0x93)

Description:
Returns the power being supplied to the HESC.

Purpose:
The InputPwrCmd() function provides a measure of the power flowing into the HESC. The power being reported by the InputPwrCmd() will be from the values of MainVCmd() and MainICmd(), or zero if BattVCmd() > MainVCmd().

Protocol: Read Word

Range: 0 to 65,535 x 10mW – input power in 10mW increments.

20. 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

Range: -32,768 to 32,767 x 10mW – in 10mW increments (0 to -32,768 x 10mW for discharge or 1 to 32,767 x 10mW for charge).

21. 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

Range: 0 to 3 – charge cycle

22. 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: 16-bit word - bit mapped – Charge termination bits flags per Table 12.

Table 12 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	RemBattCapEn	The capacity of the battery exceeded battery capacity limit. 10mWh
8 to 15	future	

23. ShutDownCmd () (0x97)

Description:
Shut down the HESC outputs in "X" seconds.

Purpose:
Allows the Host CPU to command the HESC to turn off its outputs after a delay. Typically, the Host CPU would use this after executing an ordering shutdown of its files and operating system.

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 0xFFFF.

The default EEprom value Cmd98SDDef acts as a maximum limit ShutDownCmd() can request for a shutdown time. If ShutDownCmd() requests a shutdown greater than Cmd98SDDef, the value in Cmd98SDDef will be used.

A write with value equal to zero results in the shortest shut down. ShutDownCmd() does not over-right the EEprom default values

Protocol: Write/Read Word

Range: 0 to 65,535 sec – shut down time in seconds

24. 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

Range: 16-bit word - bit mapped - see Table 13.

Table 13 PowerSupplyStatusCmd bit map

Bit#	Bit Name	Description
0	BattAutoStartEn*	Charging is to auto start 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	SDserHiLo	Define polarity of SD (shut down) line on RS232 connector SDser = 0; SD RS232 input transitioning from +5V to -5V generates shut down request. SDser = 1; SD RS232 input transitioning from -5V to +5V generates shut down request.
6	Then*	Thermistor monitoring select, 1 = enable, 0 = disable
7	SU_Req	Start up request is registered or active. Uses the Cmd98SUDef for start-up timer interval
8	SD_Req	Shut down request is active. Uses the value Cmd98SDDef for shutdown time interval
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	LowBattCap	Outputs cannot turn-on due to Low battery voltage or low battery capacity condition (read only)
15	ChecksumEn	Checksum protocol in effect on SerBus or PC/104 bus

25. 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 shutdown or start-up procedures.

Protocol: Read/Write Word
Range: 16-bit word - bit mapped - see Table 14.

Table 14 SDSUCauseCmd bit map

Bit#	Action	R/W	Bit Name	Cause of Start-Up request
0	SUreq = 1	R	ChFlags.SUreq	Power applied to HESC.
1	SUreq = 1	R	PowerSupplyStatusCmd (0x98)	PowerSupplyStatusCmd write (Bit 7 set)
2	SUreq = 1	R	IGN-SD	Received a start-up request from the IGN input.
3	SUreq = 1	R	PB-SD	Received a start-up request from the Pushbutton input.
4	SUreq = 1	R	RTC-SU	RTC ALM0 is set.
5	Not used			
6	Sureq = 1	R	Temp.SU	Start-up on temperature return to normal.
7	SUreq = 1	R/W	WD	HESC is working in Watchdog mode*.

Bit#	Action	R/W	Bit Name	Cause of Shutdown request
8	SDreq = 1	R	Main Power	Loss of main power.
9	SDreq = 1	R	PowerSupplyStatusCmd	PowerSupplyStatusCmd (0x98, Bit 8 set) or ShutDownCmd() (0x97).
10	SDreq = 1	R	IGN-SD	Received a shutdown request from the IGN input.
11	SDreq = 1	R	PB-SD	Received a shutdown request from the Pushbutton input.
12	SDreq = 1	R	RTC-SD	RTC ALM1 is set.
13	SDreq = 1	R	Battery Low	Battery low voltage or battery low capacity.
14	SDreq = 1	R	Temp.SD	Shutdown on temperature alarm.
15	SDreq = 1	R/W	WD	HESC is working in Watchdog mode*.

- The Host CPU can halt watchdog mode by clearing both bits 7 and 15 and enable watchdog mode by setting either or both bits 7 and 15.

26. I2ClowAlarmsCmd () (0x9A)

Description:
The Host uses this command to read which temperature sensors are reporting excessive low temperatures.

Purpose:
Allows the System Host to react to excessively cold temperatures.

Protocol: Read/Write Word

Range: 16-bit word - bit mapped – see Table 15.
- o Low byte temperature sensor alarm status.

Table 15 **I2ClowAlarmsCmd() (0x9A)**

Low byte of command I2ClowAlarmsCmd() (0x9A)							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TS-IC7	TS-IC6	TS-IC5	TS-IC4	TS-IC3	TS-IC2	TS-IC1	TS-IC0

- o High byte not used

27. I2ChighAlarmsCmd () (0x9B)

Description:
The Host uses this command to read which temperature sensors are reporting excessive high temperatures.

Purpose:
Allows the System Host to react to excessively high temperatures.

Protocol: Read/Write Word

Range:16-bit word - bit mapped – see Table 16.
- o Low byte temperature sensor alarm status.

Table 16 **I2ChighAlarmsCmd() (0x9B)**

Low byte of command I2ChighAlarmsCmd() (0x9B)							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TS-IC7	TS-IC6	TS-IC5	TS-IC4	TS-IC3	TS-IC2	TS-IC1	TS-IC0

- o High byte not used

28. V5Cmd () 0x9C, V12Cmd() 0x9D, V3Cmd() 0x9E

Description:
V5Cmd returns the 5V output voltage level. V12Cmd returns the 12V output voltage level. V3Cmd returns the 3.3V output voltage level. V5Cmd is support by the HESC-SERD, HESC104+, and V5SC units. V12Cmd() is support by the HESC-SERD and HESC104+ units. V3Cmd() is supported by the HESC104+ unit only.

Purpose:
These functions provide the Host power management system with an accurate measure of the various output voltages. The Host management system can use these voltage as part of a health monitoring program.

Protocol: Read Word

Range: 0 to 65,535 mV -- voltage measured at HESC terminals in mV increments.

29. ActiveEECmd() (0xA0)

Description:
The Host uses this command to set the address for reading/writing to EEprom. The HESC104 and HESC-SER have 256 bytes of EEprom memory. The HESC104+, HESC-SERD, and the V5SC have 1Kbyte of EEprom space, allowing for up to four different profiles to exist in the EEprom Memory. See paragraph 37 on ProfileBankCmd() for details on profile bank selection.

Purpose:
Allows setting the EEprom default change charging profiles and HESC setup. Address must be on even EEprom address location. The HESC masks off the low address bit to guarantee even address location. The high byte is used to select the auto increment mode.

Protocol: Read/Write Word

Range: Low byte = EEprom location 0 to 0xFF in each 256Byte profile block,
High byte: Bit-0; no auto increment = 0; auto increment = 1.

- See List of EEprom Setpoints and Configuration Variable

30. EEpromCmd () (0xA1)

Description:
The Host uses this command to read/write the data in the currently selected EEprom block (See paragraph 37 on ProfileBankCmd() for details on EEprom profile blocks). 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). Auto increment does not auto increment between EEprom profile blocks.

Purpose:
Allows the System Host to change HESC default EEprom profiles.

Protocol: Read/Write Word

Range: 16-bit word. See List of EEprom Setpoints and Configuration Variable

31. ActiveEEI2CCmd () (0xA2)

Description:
The HESC supports read/writes of up to seven 64Kbyte or three 128Kbyte I2C serial EEproms. The ActiveEEI2CCmd() enables a 128Kbyte block for data read/write using the EEpromI2CCmd(). Bit 7 of the low byte selects whether the EEpromI2CCmd() will read/write address or data values.

Note: The I2C devices must be enabled in the HESC "EEprom Setpoints and Configuration Variables" (configuration variable I2CDevEnDef) before the ActiveEEI2CCmd() and EEPromI2CCmd() will work.

Purpose:
Allows the System Host to store or retrieve data into non-volatile I2C memory.

Protocol: Read/Write Word

Range: Low byte:

- Bit 0-1 = 128Kbyte block to enable.
- Bit 2-5 = Reserved.
- Bit 7 = 0 EEpromI2CCmd() read/writes the active address of the I2C EEprom.
- Bit 7 = 1 EEpromI2CCmd() read/writes the data into the I2C EEprom.

Table 17 **ActiveEEI2CCmd() (0xA2)**

Low byte of command ActiveEEI2CCmd() (0xA2)							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Addr/Data Flag	Reserved	Reserved	Reserved	Reserved	Reserved	128Kbyte Block Select 1	128Kbyte Block Select 0

Description	Low two bits of command ActiveEEI2CCmd() (0xA2)	
	Bit 1	Bit 0
128Kbyte Block 1	0	0
128Kbyte Block 2	0	1
128Kbyte Block 3	1	0
128Kbyte Block 4	1	1

Range: High byte: no auto increment = 0, auto increment = 1.

32. EEPromI2CCmd () (0xA3)

Description:
The Host uses this command to read/write the address/data to the I2C EEprom. Command ActiveEEI2CCmd() Bit 7 determines whether the value read/written by command EEpromI2CCmd() is address or data. If set for data read/writes, the address in the I2C EEprom will increment before each read/write if the auto increment byte is set = 1 through command function ActiveEEI2CCmd() (0xA2). If set for address read/writes the address range is 0-65535 words where each word is two bytes.

Any read/writes at address 0x8000 to 0x800FF with the 128Kbyte Block Selection set for Block 4 will read/write to the EEprom memory on the RTC.

Purpose:
Allows the System Host to store or retrieve data into non-volatile I2C memory.

Protocol: Read/Write Word

Range: 16-bit word. See List of EEprom Setpoints and Configuration Variable

33. I2CLogPointerCmd () (0xA4)

Description:
The Host uses this command to read or write the sector number of the start of the last record. All sectors are 32 bytes long with the first sector starting from the bottom of the data logging memory. A value of zero indicates that no data logging data is available.

Purpose:
Allows the Host to find the latest logged data or reset the data logging to the start (or anywhere) in the memory.

Protocol: Read/Write Word

Input: unsigned int, 16-bit value. See List of EEprom Setpoints and Configuration Variable
Units: not applicable

34. I2CFcnCmd () (0xA5)

Description:
The Host uses this I2CFcnCmd to setup for an I2C function read or write that the HESC-UPS18 doesn't directly support. The I2CFcnDataCmd() is used to load the data into the HESC for writing with I2CFcnCmd(). The R/W flag is cleared by the HESC-UPS18 to indicate when the data returned by the I2CFcnCmd() is valid to read with I2CfcnDataCmd().

The I2C functions supported by I2CFcnCmd() are of the I2C/SMBus specified in Table 18 and Table 19.

Table 18 **: Write Function**

	I2CFcnCmd High Byte			I2CFcnCmd Low Byte		I2CFcnDataCmd Low Byte		I2CFcnDataCmd Low Byte		
S	7-bit Address	W	A	Command Function	A	Data Byte Low	A	Data Byte High	A	P

Table 19 **Read Function:**

	I2CFcnCmd High Byte			I2CFcnCmd Low Byte			I2CFcnCmd High Byte			I2CFcnDataCmd Low Byte		I2CFcnDataCmd Low Byte		
S	7-bit Address	W	A	Command Function	A	S	7-bit Address	R	A	Data Byte Low	A	Data Byte High	A*	P

- Where S = Start Condition, R/w = Read/write bit, A = Ack, A* = Nack, P = Stop Condition

Purpose:
Allows the System Host to set or retrieve data into from I2C devices or I2C registers not directly supported by the HESC-UPS18.

Protocol: Read/Write Word

Range: Low byte: Function command
High byte: See Table 20.

Table 20 **I2CFcnCmd() High byte:**

High Byte							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2C device address, 0 to 127							R/W Read = 1 Write = 0

35. I2CFcnDataCmd () (0xA6)

Description:
The Host uses this command to read/write the to be sent or is returned by the I2CFcnCmd(). During an I2CFcnCmd() read command, the returned data is valid after the R/W bit of I2CFcnCmd() is cleared by the HESC-UPS18.

Purpose:
Allows the System Host to set or retrieve data into from I2C devices or I2C registers not directly supported by the HESC-UPS18.

Protocol: Read/Write Word

Range: 16-bit word - 0 to 65,535

36. ProfileCmd () (0xA7)

Description:
The Host uses ProfileCmd() to save and restore HESC profiles to/from the main program Flash memory. HESC profiles include charging values, UPS counters and other operational settings that reside in the default EEprom memory.

Each HESC profile uses 256 bytes of EEprom memory. The HESC104 and HESC-SER have 256 bytes of EEprom memory. The HESC104+, HESC-SERD, and the V5SC have 1Kbyte of EEprom space, allowing for up to four different profiles to exist in the EEprom Memory. The HESC104+, HESC-SERD, and the V5SC have dual battery charging support and the first two profile blocks are used by the HESC-UPS18 firmware.

Purpose:
Allows the Host to easily restore an "archived" HESC profile. ProfileCmd() allows the Host CPU to save/restore customized profile versions. Jumpering both the SCL and SDA lines to ground and applying power for five seconds will restore the saved version of the profiles from Flash back to the EEprom memory.

Protocol: Read/Write Word

Range:

- Read operation: A Read results in the HESC reading the entire default EEprom memory and saving it to the HESC main program Flash memory. The return value from the HESC is zero.

- Write operation: A Write results in the HESC restoring all or selected profiles from the HESC main program Flash memory to the EEprom memory. For restoring only a
  - Low byte = 0; Source for restored profile is first Flash profile block. Destination is currently selected bank per ProfileBankCmd().
  - Low byte = 1; Source for restored profile is second Flash profile block. Destination is currently selected bank per ProfileBankCmd(). Not valid for HESC104 and HESC-SER products.
  - Low byte = 2; Source for restored profile is third Flash profile block. Destination is currently selected bank per ProfileBankCmd(). Not valid for HESC104 and HESC-SER products.
  - Low byte = 3; Source for restored profile is fourth Flash profile block. Destination is currently selected bank per ProfileBankCmd(). Not valid for HESC104 and HESC-SER products.
  - Low byte = 0xFF; Restore entire EEprom memory from main Flash memory.

  - High byte not used.

37. ProfileBankCmd() (0xA8)

Description:
The Host uses ProfileBankCmd() to select the EEprom bank (via low byte) that ActiveEEpromCmd(), EEpromCmd(), and ProfileCmd() work upon. The low byte is Read/Write capable. The HESC104+, HESC-SERD, and the V5SC have 1Kbyte of EEprom space, allowing for up to four different EEprom profiles to exist in the EEprom Memory. The high byte of ProfileBankCmd() is a read only field that allows the Host to determine which Profile bank is currently in use by the HESC for battery charging.

Protocol: Read/Write Word

Range: 16-bit word – bit mapped per Table 21.

Table 21 **ProfileBankCmd() (0xA8)**

Low byte of command ProfileBankCmd () (0xA8) – Read/Write							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	EEprom Block Select-1	EEprom Block Select-0

Description	Low two bits of command ProfileBankCmd() (0xA0)	
	Bit 1	Bit 0
Profile Block 1	0	0
Profile Block 2	0	1
Profile Block 3	1	0
Profile Block 4	1	1

High byte of command ProfileBankCmd () (0xA8) – Read Only							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	EEprom Block Select-1	EEprom Block Select-0

Description	Low two bits of command ProfileBankCmd () (0xA8)	
	Bit 1	Bit 0
Profile Block 1	0	0
Profile Block 2	0	1
Profile Block 3	1	0
Profile Block 4	1	1

38. RTCloCmd () (0xB0), RTChiCmd() (0xB1), [32-bit name is **RTC32]**

Description:

The Host uses these commands to read/write the time of the Real Time Clock (RTC) of the BAT104-RTC*. The HESC reads/writes the time from the RTC and converts it to/from a 32-bit word (RTC32). The 32-bit word is the number of seconds from 1 January 2001, 00:00. For communication purposes, RTC32 is split in two 16-bit words, RTCloCmd() , RTChiCmd(). RTCloCmd() is the low 16-bit word, and RTChiCmd() is the upper 16-bit word. All RTC32 commands must read/write the RTChiCmd() before the RTCloCmd().

Purpose:
Allows the System Host to read/write the time of the RTC on the BAT104-RTC.

Protocol: Read/Write Word

Range: 0 to 65535 for RTCloCmd() and RTChiCmd (0 to 4,294,967,295 seconds when combined into RTC32).

***Note:** The HESC-UPS18 supports the Xicor X1229 Real Time Clock controller. Other support battery modules are the BAT-NiMh-RTC and BAT-NiMh2-RTC.

39. RTCSUloCmd () (0xB2), RTCSUhiCmd() (0xB3), RTCSDloCmd () (0xB4), RTCSDhiCmd() (0xB5), [32-bit names are RTCSU32, RTCSD32]

Description:

The Host uses these commands to read/write the startup and shutdown time of the HESC. The startup and shutdown time is a 32-bit word (RTCSU32 and RTCSD32) representing the number of seconds from 1 January 2001, 00:00. For communication purposes, RTCSU32 is split in two 16-bit words, RTCSUloCmd() , RTCSUhiCmd(). Similarily, RTCSD32 is split in two 16-bit words, RTCSDloCmd() , RTCSDhiCmd(). RTCSUloCmd() and RTCSDloCmd() are the low 16-bit words, and RTCSUhiCmd() and RTCSDhiCmd() are the upper 16-bit words. All RTC32 commands must read/write the high 16-bit words before the low 16-bit words.

When the RTC time equals the RTCSU32, the HESC will register a start-up request (provided the HESC outputs are off, or a Shutdown is already registered). Also, when the RTC time equals the RTCSD32, the HESC will register a shutdown request (provided the HESC outputs are on). Battery packs, such as the BAT104-RTC*, have isolating Mosfets, so that the power consumption by the HESC can be reduced to zero. The RTC on these boards have their own alarm wake-up capability, so the battery packs can awaken itself at a preprogrammed time (RTCSU32) and re-energize the HESC.

Purpose:

Allows the System Host to read/write the RTC startup and shutdown time on the BAT104-RTC*.

Protocol: Read/Write Word

Range: 0 to 65535 for RTCSUloCmd() and RTCSUhiCmd (0 to 4,294,967,295 seconds when combined into RTCSU32). 0 to 65535 for RTCSDloCmd() and RTCSDhiCmd (0 to 4,294,967,295 seconds when combined into RTCSD32).

***Note:** The HESC-UPS18 supports the Xicor X1229 Real Time Clock controller. Other support battery modules are the BAT-NiMh-RTC and BAT-NiMh2-RTC.

Section 8 : 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 it's contents on power loss and has an endurance for a minimum of 100,000 cycles of erase/write cycles.

2. Variables and setpoints are stored with their least significant byte stored in the lower memory location, and the most significant byte in the high location.

Table 22 HESC Setpoints and Configuration Variables

EEprom start address Charge Cycle				Variable Size	Variable/Setpoint Name	Units	Range	Description
1	2	3	4					
0x00	0x20	0x40	0x60	Word	ChTermDef	Bit flags	N/A	Charge termination enable flags
0x02	0x22	0x42	0x62	Word	BattMaxCapDef	10mW	0 to 65535	Maximum capacity to charge battery
0x04	0x24	0x44	0x64	Word	Future2	N/A	0 to 65535	Reserved for future charge termination method.
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	Sec	0 to 65535	Maximum time allowed in Trickle charge mode
0x80, 0x81				Word	ChFlagsDef	Bit flags	N/A	Charger/power supply enable flags
0x82, 0x83				Word	LowVoltageDef	mV	0 to 65535	Minimum operating voltage
0x84, 0x85				Word	BattLowCapacityDef	10mW	0 to 65535	Minimum battery capacity allowed
0x86, 0x87				Word	MainPwrMaxDef	10mW	0 to 65535	Maximum input power allowed
0x88				Byte	MaxBusTimeDef	Timer Ticks	0 to 255	Maximum time before communications timeout
0x89				Byte	CHCycleMaxDef	Cycle	1 to 4	Defines how many charge cycles to use
0x8A, 0x8B				Word	BattTempMinDef	0.1K *	0 to 6553.5	Minimum battery charging temperature
0x8C, 0x8D				Word	BattTempMaxDef	0.1K *	0 to 6553.5	Maximum battery charging temperature
0x8E, 0x8F				Word	VminSUDef	mV	0 to 65535	Minimum input start-up voltage
0x90				Byte	ChTempSelectDef	Bit flags	0 to 8	I2C device to use for primary battery temp, if zero use Th
0x91				Byte	Ch2TempSelectDef	Bit flags	0 to 8	I2C device to use for secondary battery temp, if zero use Th2
0x92, 0x93				Word	I2CpollTimeDef	Sec	0 to 65535	Rate at which I2C devices are polled
0x94				Byte	I2CTsICenDef	Bit flags	N/A	Enables polling for selected I2C devices (TS-IC, TC-IC).
0x95				Byte	I2CDevEnDef	Bit flags	N/A	Enables polling for selected I2C EEproms devices.
0x96				Byte	BattSelDef	Bit flags	N/A	Selects multi-battery packs, if zero BE selects battery pack
0x97				Byte	I2CLogDef	Bit flags	N/A	Defines type of automatic data to be logged.
0x98, 0x99				Word	I2CLogTimeDef	Sec	0 to 65535	Rate at which data is logged to I2C EEproms
0x9A, 0x9B				Word	I2CTsAlmDef	Bit flags	N/A	Defines SD & SU action on temperature alarm.
0x9C, 0x9D				Word	I2CTsNormDef	Bit flags	N/A	Defines SD & SU action on return to normal temperature.
0x9E, 0x9F				Word	ChFlags_ExtDef	Bit flags	N/A	Extended Charger & HESC flags

EEprom start address	Variable Size	Variable/Setpoint Name	Units	Range	Description
0xA0	Word	I2Cconfig0	Bit flags	N/A	Defines I2C sensor 0 address, and operating modes
0xA2	Word	I2CSetPoint0	0.1K *	0 to 6553.5	I2C Sensor 0 low temp alarm setting
0xA4	Word	I2CHiLoAlarm0	0.1K *	0 to 6553.5	I2C Sensor 0 high temp alarm setting
0xA6	Word	I2Cconfig1	Bit flags	N/A	Defines I2C sensor 1 address, and operating modes
0xA8	Word	I2CSetPoint1	0.1K *	0 to 6553.5	I2C Sensor 1 low temp alarm setting
0xAA	Word	I2CHiLoAlarm1	0.1K *	0 to 6553.5	I2C Sensor 1 high temp alarm setting
0xAC	Word	I2Cconfig2	Bit flags	N/A	Defines I2C sensor 2 address, and operating modes
0xAE	Word	I2CSetPoint2	0.1K *	0 to 6553.5	I2C Sensor 2 low temp alarm setting
0xB0	Word	I2CHiLoAlarm2	0.1K *	0 to 6553.5	I2C Sensor 2 high temp alarm setting
0XB2	Word	I2Cconfig3	Bit flags	N/A	Defines I2C sensor 3 address, and operating modes
0XB4	Word	I2CSetPoint3	0.1K *	0 to 6553.5	I2C Sensor 3 low temp alarm setting
0XB6	Word	I2CHiLoAlarm3	0.1K *	0 to 6553.5	I2C Sensor 3 high temp alarm setting
0XB8	Word	I2Cconfig4	Bit flags	N/A	Defines I2C sensor 4 address, and operating modes
0xBA	Word	I2CSetPoint4	0.1K *	0 to 6553.5	I2C Sensor 4 low temp alarm setting
0xBC	Word	I2CHiLoAlarm4	0.1K *	0 to 6553.5	I2C Sensor 4 high temp alarm setting
0xBE	Word	I2Cconfig5	Bit flags	N/A	Defines I2C sensor 5 address, and operating modes
0xC0	Word	I2CSetPoint5	0.1K *	0 to 6553.5	I2C Sensor 5 low temp alarm setting
0xC2	Word	I2CHiLoAlarm5	0.1K *	0 to 6553.5	I2C Sensor 5 high temp alarm setting
0XC4	Word	I2Cconfig6	Bit flags	N/A	Defines I2C sensor 6 address, and operating modes
0xC6	Word	I2CSetPoint6	0.1K *	0 to 6553.5	I2C Sensor 6 low temp alarm setting
0xC8	Word	I2CHiLoAlarm6	0.1K *	0 to 6553.5	I2C Sensor 6 high temp alarm setting
0xCA	Word	I2Cconfig7	Bit flags	N/A	Defines I2C sensor 7 address, and operating modes
0xCC	Word	I2CsetPoint7	0.1K *	0 to 6553.5	I2C Sensor 7 low temp alarm setting
0xCE	Word	I2CHiLoAlarm7	0.1K *	0 to 6553.5	I2C Sensor 7 high temp alarm setting
0xD0	Word	PWRSDdebDef	Sec	0 to 65535	Debounce interval for main power loss
0xD2	Word	PWRSUdebDef	Sec	0 to 65535	Debounce interval for main power applied
0xD4	Word	PWRSDDef	Sec	0 to 65535	Shutdown delay for main power loss
0xD6	Word	PWRSUDef	Sec	0 to 65535	Start-Up delay for main power applied
0xD8	Word	IGNSDdebDef	Sec	0 to 65535	Debounce interval for IGN input turned off
0xDA	Word	IGNSUdebDef	Sec	0 to 65535	Debounce interval for IGN input turned on
0xDC	Word	IGNSDDef	Sec	0 to 65535	Shutdown delay for IGN input turned off
0XDE	Word	IGNSUDef	Sec	0 to 65535	Start-Up delay for IGN input turned on
0xE0	Word	PBSDDef	Sec	0 to 65535	Shutdown delay for Pushbutton off
0XE2	Word	PBSUDef	Sec	0 to 65535	Start-Up delay for Pushbutton on
0xE4	Word	BATTSDDef	Sec	0 to 65535	Shutdown delay for low battery voltage or capacity
0xE6	Word	Cmd98SDDef	Sec	0 to 65535	Shutdown delay for command PowerSupplyStatusCmd
0XE8	Word	Cmd98SUDef	Sec	0 to 65535	Start-Up delay for command PowerSupplyStatusCmd
0xEA	Word	TempSDDef	Sec	0 to 65535	Shutdown delay for alarm temperature condition.
0xEC	Word	TempSUDef	Sec	0 to 65535	Start-Up delay for return to normal temperature condition.
0xEE	Word	RTCSDDef	Sec	0 to 65535	Shutdown delay after RTC time reached.
0xF0	Word	RTCSDloDef	Sec	0 to 65535	Shutdown time in seconds from January 1, 2001, 00:00, lo word.
0xF2	Word	RTCSDhiDef	Sec	0 to 65535	Shutdown time in seconds from January 1, 2001, 00:00, hi word.
0xF4	Word	RTCSUloDef	Sec	0 to 65535	Start-Up time in seconds from January 1, 2001, 00:00, lo word.
0xF6	Word	RTCSUhiDef	Sec	0 to 65535	Start-Up time in seconds from January 1, 2001, 00:00, hi word.
0xF8	Word		mV	0 to 65535	
0xFA	Word	BattMaxCapDef	10mW	0 to 65535	Maximum Battery Capacity
0xFC	Word	BattRemCapDef	10mW	0 to 65535	Remaining Battery Capacity
0xFE	Word	BattTime2RechargeDef	Min	0 to 65535	Minutes from charge termination to start of recharge battery

B. SerBus EEprom Details

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

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

Purpose:
Allow the Host to configure the HESC to charge a particular battery type.

Range: 16-bit word - bit mapped – see Table 23.

Table 23 ChTerm bit map

	Bit#	Bit Name	Description
Default EEprom address 0x00, 0x20, 0x40, 0x60	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
Default EEprom address 0x01, 0x21, 0x41, 0x61	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	BattMaxCapEn	Charge termination method that terminates charging if the battery capacity is above the BattMaxCapDef. 1 = enable, 0 = disable
	13	Future	
	14	Future	
	15	Future	

2. BattMaxCapDef - EEprom Locations 0x02 0x22 0x42 0x62

Description:
When BattMaxCapEn is enabled, the HESC will terminate the current charge cycle of a standard battery pack when the battery capacity exceeds BattMaxCapDef 10mW. For SMBus batteries, the HESC terminate charging when the SMBus battery instructs it to stop charging. The SMBus protocol provides several methods the SMBus battery may use to terminate charging.

Purpose:
Prevent damage to batteries that can occur from excessive charging. Excessive charging can cause serious overheating, and possible venting.

Range: 0 to 65,535 x 10mWh – power in 10mWh increments.

Example: A 2100mA, 8.4V nominal cell = 1764 x 10mWh

3. Future - EEprom Locations 0x04 0x24 0x44 0x64

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

Description:
When BattVmaxEn is enabled, the HESC will terminate charging a standard battery pack when the battery voltage rises above BattVminDef. To enable BattVmaxDef refer ChTermDef.

Purpose:
Prevent damage to batteries that can occur from being overcharged.

Range: 0 to 65,535 mV – voltage in mV increments.

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

Description:
When BattVmaxTimeEn is enabled, the HESC will terminate charging a standard battery pack when the length of time the HESC has been charging since the last increase in battery voltage exceeds BattVmaxTimeDef minutes. To enable BattVmaxTimeDef refer to ChTermDef.

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.

Range: 0 to 65,535 minutes – timing interval in minute increments.

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

Description:
When BattVdeltaEn is enabled, the HESC will terminate charging a standard battery pack when the battery voltage has reduced BattVdeltaDef mV from the maximum peak voltage. To enable BattVdeltaDef refer to ChTermDef.

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.

Range: 0 to 65,535 mV – voltage in mV increments.

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

Description:
When TimeMaxEn is enabled, the HESC will terminate charging a standard battery pack when the length of time the HESC has been charging (this charge cycle) exceeds TimeMaxDef minutes. To enable TimeMaxDef refer to ChTermDef.

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

Range: 0 to 65,535 minutes – timing interval in minute increments.

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

Description:
When BattIminEn is enabled, the HESC will terminate charging a standard battery pack when the charge current is below BattIminDef.

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.

Range: 0 to 65,535 mA – Current in mA increments.

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

Description:
When BattImaxEn is enabled, the HESC will terminate charging a standard battery pack when the charge current exceeds BattImaxDef. To enable BattImaxDef refer to ChTermDef.

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

Range: 0 to 65,535 mA – Current in mA increments.

10. TimeTermEnDef - EEprom Locations 0x12, 0x32, 0x52, 0x72:

Description:
When TimeTermEn is enabled, TimeTermEnDef defines the minimum time before charge termination is allowed. Typically set for 2 to 5 minutes. To enable TimeTermEnDef refer to ChTermDef.

Purpose:
Prevent early charge termination due to battery chemistry settling into charge cycle.

Range: 0 to 65,535 minutes – timing interval in minute increments.

11. BattTempCompDef - EEprom Locations 0x14, 0x34, 0x54, 0x74:

Description:
When BattTempCompEn is enabled, the HESC will compensate the charging voltage for every °K above 298.2°K (25°C), the charging voltage is reduced by BattTempCompDef. To enable BattTempCompDef refer to ChTermDef.

Purpose:
Lead Acid batteries typically require a temperature compensation if operating in varying temperature ranges. Without temperature compensation, the electrolyte could be "boiled" away in high ambient temperatures, and not fully charged in low ambient temperatures.

Range: 0 to +65535 mV/°K – Temperature compensation in mV/°K increments.

Example: If BattTempCompDef = 15, and the battery temperature = 308.2°K, BattVDef (charging voltage setpoint) will be reduced by 150mV.

12. BattVDef - EEprom Locations 0x16, 0x36, 0x56, 0x76:

Description:
BattVDef defines the open circuit output voltage of the charger. If the charger is operating in "constant current" mode the output voltage will be less than BattVDef.

Purpose:
Sets the constant voltage charging setpoint.

Range: 0 to 65,535 mV – voltage in mV increments.

Example: The BAT104-NiMh typically has a BattVDef = 12,200 and BattIDef = 1000. Since NiMh batteries are usually charged in constant current mode, the charging voltage is set slightly higher than the maximum expected voltage (maximum expected is 11,200mV) across the battery terminals during charging. Charging current will remain at the one ampere level until the BAT104-NiMh is nearly fully charged, whereupon the charging current will begin to tamper off. At this point, charging is normally terminated.

13. BattIDef - EEprom Locations 0x18, 0x38, 0x58, 0x78:

Description:
BattIDef defines the "shourt circuit" output current of the charger. If the charger is operating in "constant voltage" mode the output current will be less than BattIDef.

Purpose:
Sets the constant current charging setpoint.

Range: 0 to 65,535 mA – current in mA increments.

Example: The BAT104-NiMh typically has a BattVDef = 12,200 and BattIDef = 1000. Since NiMh batteries are usually charged in constant current mode, the charging voltage is set slightly higher than the maximum expected voltage (maximum expected is 11,200mV) across the battery terminals during charging. Charging current will remain at the one-ampere level until the BAT104-NiMh is nearly fully charged, whereupon the charging current will begin to tamper off. At this point, charging is normally terminated.

14. BattTempRateDef - EEprom Locations 0x1A, 0x3A, 0x5A, 0x7A:

Description:
When BattTempRateEn is enabled, the HESC will terminate charging when the rate of battery temperature increases by more BattTempRateDef. To enable BattTempRateDef refer to ChTermDef.

Purpose:
Prevent damage to batteries that can occur from being overcharged. The rate of battery temperature increase is a primary charge termination for NiCd and NiMh batteries.

Range: 0 to +65,535 x 0.1°K/Min – Rate of temperature in 0.1°K/minute increments.

Example: Typically, BattTempRateDef is set for 5 x 0.1°K/Minute for NiCd and NiMh batteries.

15. BattTrickleDef - EEprom Locations 0x1C, 0x3C, 0x5C, 0x7C:

Description:
Used in the SMBus battery wakeup mode to assist in establishing initial communications with the SMBus battery. When in Trickle charge mode, BattTrickleDef sets the trickle charge current. Only the value is EEprom Location 0x1C is used for SMBus Trickle Charge current setting. (Future uses of the BattTrickleDef values for standard battery packs is being researched).

Purpose:
Used to establish communications with an SMBus battery.

Range: 0 to 65,535 mA – current in mA increments.

Example: A typical setting for waking an SMBus battery is 100mA. Some SMBus may require larger amounts of current.

16. BattTrickleTimeDef - EEprom Locations 0x1E, 0x3E, 0x5E, 0x7E:

Description:
BattTrickleTimeDef is automatically enabled/disabled by the HESC whenever main power is applied or a battery is "inserted" into an SMBus enabled (see ChFlagsDef, SMBactiveEn flag) system. The manual setting of BattTrickleTimeDef EEprom register has no current effect on the operation of the HESC. Future firmware developments may take advantage of this EEprom register setting.

17. ChFlagsDef - EEprom Locations 0x80, 0x81;

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 to the type of battery pack, and system it is installed into.

Range: 16-bit word - bit mapped – see Table 24.

Table 24 ChFlags bit map

	Bit#	Bit Name	Description
Default EEprom address 0x80	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. SMBus timing in effect when SMBactiveEn = 1.
	3	IgnHiOffEn	When IgnHiOff is set, the HESC will begin shutdown procedures when the IGN pin is high. If IgnHiOff is low, the HESC 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	SDserHiLo	Define polarity of SD (shut down) line on RS232 connector SDser = 0; SD RS232 input transitioning from +5V to -5V generates shut down request. SDser = 1; SD RS232 input transitioning from -5V to +5V generates shut down request.
	6	Then*	Thermistor monitoring select: 1 = enable, 0 = disable
	7	SUreq	Startup request when power applied when HESC is hard off**: 1 = off, 0 = on.
Default EEprom address 0x81	8	MultiBattEn*	Enables Multi battery pack charging: 1 = enable, 0 = disable.
	9	SDserdigEn	Enable the handshake SD line (not the RS232 communications).
	10	BattVchCalEn	Enable auto calibrate charging voltage output: 1 = Auto calibrate, 0 = no calibrate.
	11	BattIchCalEn	Enable auto calibrate charging current output: 1 = Auto calibrate, 0 = no calibrate.
	12	LEDDis	Disable LEDs to conserve power: 1 = disable = 1, 0 = enable.
	13	RTCen	Enable RTC (based on the Xircom X1226) 1 = enable, 0 = disable.
	14	WDmodeEn	Watchdog boot mode enable: 1 = enable, 0 = disabled.
	15	MasterModeEn	Master mode alert enable: 1 = enable, 0 = off.

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

18. BattLowVoltageDef – EEprom Locations 0x82, 0x83

Description:
When the battery voltage drops below BattLowVoltageDef a Shutdown is requested if the power supply outputs are energized. The shut down will occur in BATTSDDef seconds. Setting BattLowVoltageDef to zero prevents any possible shut down due to low battery voltage.

Purpose:
Allow time for the Operating System to shut down before the battery is fully discharged or damaged.

Range: 0 to 65,535 mV – voltage in mV increments.

19. BattLowCapacityDef – EEprom Locations 0x84, 0x85

Description:
When the battery capacity drops below BattLowCapDef a Shutdown is requested if the power supply outputs are energized. The shut down will occur in BATTSDDef seconds. Setting BattLowCapDef to zero prevents any possible shut down due to low battery capacity.

Purpose:
Allow time for the Operating System to shut down before the battery is fully discharged or damaged.

Range: 0 to 65,535 x 10mWh – power in 10mWh increments.

20. MainPwrMaxDef – EEprom Locations 0x86, 0x87

Description:
When the main power exceeds the set limit, the maximum battery charging current is reduced. A setting of zero disables this function.

Purpose:
To prevent overloading of the input supply. Typically, this input supply might be a limited power source such as a wall "cube".

Range: 0 to 65,535 x 10mWh – power in 10mWh increments.

21. MaxBusTimeDef – EEprom Locations 0x88

Description:
Resets the HESC communication (SerBus for RS232 connected HESCs, and PC/104 bus for PC/104 bus connected HESCs) in case of communication error resulting from incomplete communications transaction. Set to value = 0 disables bus timer operation. Normal value for bus timer operation is 255.

Purpose:
Reset HESC communications.

Range: 0 to 255 – no units.

22. CHCycleMaxDef – EEprom Locations 0x89

Description:
CHCycleMaxDef is the number of charge cycles the HESC charger uses in charging the standard (non-SMBus) battery. The HESC uses the charging termination defines from EEprom registers 00 to 7E.

Purpose:
Defines the number of charge cycles to use.

Range: 1 to 4 – no units.

23. BattTempMinDef - EEprom Locations 0x8A:

Description:
When BattTempMinEn is enabled, the HESC 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.

Range: 0 to 65,535 x 0.1°K -- cell temperature in tenth degree Kelvin (0.1°K) increments

24. BattTempMaxDef - EEprom Locations 0x8C:

Description:
When BattTempMaxEn is enabled, the HESC 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.

Range: 0 to 65,535 x 0.1°K -- cell temperature in tenth degree Kelvin (0.1°K) increments

25. BattVminDef - EEprom Locations 0x8E:

Description:
When BattVminEn is enabled, the HESC 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.

Range: 0 to 65,535 mV – voltage in mV increments.

26. ChTempSelectDef (0x90), Ch2TempSelectDef (0x92):

Description:
ChTempSelect selects a temperature sensor for the charging control of the primary battery. Ch2TempSelect selects a temperature sensor for the charging control of the secondary battery (for HESC product with dual battery capability). The selected sensor is used to report the temperature for BattTempCmd(). Please refer to Section 9 for configuration of digital temperature devices such as the TS-ICs, and the digital temperature sensors used on the Tri-M battery boards. Note: The temperature for the active battery pack is reported by BattTempCmd().

Table 25 **Temperature device selection values:**

Description	Value:
Thermister	0
Temperature Device 0	1
Temperature Device 1	2
Temperature Device 2	3
Temperature Device 3	4
Temperature Device 4	5
Temperature Device 5	6
Temperature Device 6	7
Temperature Device 7	8

Purpose:
Indicates what I2C devices are on the I2C/SMBus to use for the main temperature sensor for battery monitoring and charging.

Range: 0 to 8 – no units.

27. I2CpollTimeDef - EEprom Locations 0x92, 0x93:

Description:
Polling timer interval for TS-IC temperature sensors, and RTC support. Typical polling rates are 4 to 10 seconds, although it is possible to use larger or smaller values. Too small a value and the HESC might not be able to complete all the requested polling before the start of the next poll. Too large and the temperature response for charge termination is impeded.

Purpose:
Allows customizing of the scanning of the I2C devices.

Range: 0 to 65,535 seconds.

28. I2CTsICenDef - EEprom Locations 0x94:

Description:
I2CTsICenDef is a set of bit flags that indicate what I2C TS-IC or TC-IC devices are connected to the HESC. Up to eight TS-ICs or IC-ICs can be used. Each bit enables a TS-IC or TC-IC. Refer to "Section 8: Temperature Sensor, TS-IC and Controller, TC-IC" for details on setting up TS-IC and TC-IC devices. Each device is enabled for polling by select respective bit enable.

Table 26 **TS-IC and TC-IC device enable settings:**

Default EEprom address 0x94							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Device 7	Device 6	Device 5	Device 4	Device 3	Device 2	Device 1	Device 0

Purpose:
Indicates what I2C devices are on the I2C/SMBus connected to the HESC.

29. I2CDevEnDef - EEprom Locations 0x95:

Description:
64Kbtye or 128Kbyte External EEproms can be added to HESCs via the I2C/SMBus. The Host CPU can use this memory for automatic datalogging or for Host CPU data storage. A maximum of seven 64Kbyte EEproms or three 128Kbyte EEproms can be added. This memory is defined in 64Kbytes blocks for automatic datalogging or for Host data storage. Bits 0 to 2 of I2CDevEnDef define what 64Kbyte blocks are for Host CPU data storage and bits 4 to 6 of I2CDevEnDef define the total of the 64Kbyte EEprom blocks. The Host CPU is free to use the memory defined by bits 0 to 2 in any manner it wishes. Memory above bits 0 to 2 is reserved for automatic datalogging. Automatic datalogging starts at the bottom of its memory and fills the memory until it reaches the top. Once the automatic datalogging memory is full, automatic datalogging starts overwriting data from the bottom again.

Supported I2C EEproms includes the 64Kbyte devices (compatible with Microchip 24LC512) and 128Kbyte devices (compatible with ST Microelectonics M24M01). The 64Kbyte devices typically have address strapping inputs (A2, A1 & A0), while the 128Kbyte devices have address strapping inputs (E2, E1 or A2, A1). Each serial EEProm must have their address strapping set to create a continuous storage range starting from the lowest setting.

Table 27 list the bit settings for the memory. Select the bit settings for the total I2C EEprom memory in the system (not including EEprom memory in the RTC) from bits 4-6. Select the bit settings for memory to be used for Host data storage from bits 0-2. Or the two values are a bit wise level to create an 8-bit value. This 8-bit value is combined with I2CtsICenDef (EEprom location 0x94) to create a 16-bit word used to read or write from the HESC.

Table 27 I2CdevEnDef bit flags:

Description	64Kbyte	128Kbyte	Reserved	Total 64Kbytes define	Reserved	6Kbytes Host CPU data storage

Total size of I2C memory	Device#	Device#	Bit7	Bit 6	Bit 5	Bit 4	Bit3	Bit 2	Bit 1	Bit 0
No I2C Memory	0	0	N/A	0	0	0	N/A	0	0	0
64K bytes	1	1	N/A	0	0	1	N/A	0	0	1
128K bytes	2	1	N/A	0	1	0	N/A	0	1	0
192K bytes	3	2	N/A	0	1	1	N/A	0	1	1
256K bytes	4	2	N/A	1	0	0	N/A	1	0	0
320K bytes	5	3	N/A	1	0	1	N/A	1	0	1
384K bytes	6	3	N/A	1	1	0	N/A	1	1	0
448K bytes	7	-	N/A	1	1	1	N/A	1	1	1

Note:

- Total size of memory does not include memory on RTC (512 bytes).
- Maximum seven 64Kbyte devices or three 128Kbytes can be connected.
- Bits 3 & 7 are don't cares.

Purpose:
Defines usage of the External I2C/SMBus Memory.

30. I2CTsAlmDef - EEprom Locations 0x9A, 0x9B:

Description:
I2CTsAlmDef is a set of bit flags that indicate what Start-up or Shutdown action to take on a temperature alarm condition. A pair of bits defines the action for each TS-IC or TC-IC. Refer to "Section 8: Temperature Sensor, TS-IC and Controller, TC-IC" for details on setting up TS-IC and TC-IC devices. Each device must be enabled for polling by selecting respective bit enable of I2CdevEnDef (0x95).

The HESC will use the TempSDDef interval for an SD temperature event and the TempSUDef interval for a SU temperature event. When both an SD & SU event is requested, the HESC will first shutdown and then restart. The SD or SU event will be initiated on the detection of the first temperature alarm condition. Subsequent

Table 28 **I2CTsAlmDef SD & SU action definitions.**

Action on Alarm condition	Device7	Device6	Device5	Device4	Device3	Device2	Device1	Device0	Device7	Device6	Device5	Device4	Device3	Device2	Device1	Device0
	Default EEprom address 0x9B								Default EEprom address 0x9A							
	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
No Action	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Request SD	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
Request SU	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
Request SD & SU	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Purpose:
Defines actions required when an I2C temperature sensor detects a temperature out of range.

31. I2CTsNormDef - EEprom Locations 0x9C, 0x9D:

Description:
I2CTsNormDef is a set of bit flags that indicate what Start-up or Shutdown action to take on a return to normal temperature condition. A pair of bits defines the action for each TS-IC or TC-IC. Refer to "Section 8: Temperature Sensor, TS-IC and Controller, TC-IC" for details on setting up TS-IC and TC-IC devices. Each device must be enabled for polling by selecting respective bit enable of I2CdevEnDef (0x95).

The HESC will use the TempSDDef interval for an SD temperature event and the TempSUDef interval for a SU temperature event. When both an SD & SU event is requested, the HESC will first shutdown and then restart.

Table 29 **I2CTsAlmDef SD & SU action definitions.**

Action on Alarm return to normal condition	Device7	Device6	Device5	Device4	Device3	Device2	Device1	Device0	Device7	Device6	Device5	Device4	Device3	Device2	Device1	Device0
	Default EEprom address 0x9D								Default EEprom address 0x9C							
	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
No Action	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Request SD	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
Request SU	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
Request SD & SU	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Purpose:
Defines actions required when an I2C temperature sensor detects a return to normal temperature range.

32. ChFlags_ExtDef – EEprom Locations 0x9E, 0x9F:

Description:
Timer Tick Duration - For applications requiring a smaller time resolution than one second, the timer tick can be changed to 0.1 second from a one second interval. When set for 0.1 second tick resolution, all UPS interval times are changed to a 0.1 second minimum to a 6,535.5 maximum duration (maximum 1.8 hour duration).

LowBatt-Lockout – When set, prevents the main outputs turning on until battery capacity is greater than minimum or battery voltage is above minimum setting.

Table 30 ChFlags_Ext bit map

	Bit#	Bit Name	Description
Default EEprom address 0x9E	0	Timer Tick Duration	Length of basic timer tick: 0 = 1 second, 1 = 0.1 second.
	1	LowBatt-Lockout	Prevent startup of main output on low battery capacity or low battery voltage.
	2	BATTSUen	BATTSUen = 1: Enable startup on battery voltage above minimum startup battery voltage.
	3	RTCSUBattIsolate	RTCSUBattIsolate = 1: Prevent battery isolation unless valid RTC startup time is set (any startup time in future).
	2..7	Reserved	Bit-4 to bit7 reserved.
Default EEprom address 0x9F	8..15	Reserved	Bit-8 to bit15 reserved.

Purpose:
ChFlags_Ext defines the extended charger and HESC Flags.

33. PWRSDdebDef (0xD0), PWRSUdebDef (0xD2), IGNSDdebDef (0xD8), IGNSUdebDef (0xDA)

Description:
Defines the timer interval for debouncing signal. Debounce timers include:

- Main power loss.
- Main power restored.
- Ignition switch activated.
- Ignition switch de-activated

Purpose: Prevents unnecessary start-up or shutdown of the Host CPU.

Range: 0 to 65535 seconds. Setting a value equal to zero does **NOT** disable the interval timer.

34. PWRSDDef (0xD4), IGNSDDef (0xDC), PBSDDef (0xE0), BATTSDDef (0xE4), Cmd98SDDef (0xE6), TempSDDef (0xEA)

Description:
Defines the timer interval for initiating a Shutdown. Initiating a Shutdown in the HESC results in the Host CPU being notified of an impending shutdown of the HESC outputs. Possible shutdown causes are:

- Main power loss.
- Ignition switch activated.
- Momentary push-button switch depressed.
- Host CPU, either SER-Bus command or serial port handshake lines.
- Operating temperature out of range.

Purpose: Shutdown timers allow time to close files, and properly stop the operating system.

Range: 0 to 65535 seconds. Setting a value equal to zero disables the timer.

35. PWRSUDef (0xD6), IGNSUDef (0xDE), PBSUDef (0xE2), Cmd98SUDef (0xE8), TempSUDef (0xEC)

Description:
Defines the timer interval for initiating a Start-Up. Possible shutdown causes are:

- Return of Main power.
- Ignition switch de-activated.
- Momentary push-button switch depressed.
- Host CPU, either SER-Bus command or serial port handshake lines.
- Operating temperature return to normal.

Purpose: Start-up timers allow the delayed start of the Host CPU to conserve energy.

Range: 0 to 65535 seconds. Setting a value equal to zero disables the timer.

36. RTCSDDef(0xEE)

Description:
Defines the timer interval for initiating a Shutdown due to a RTCSD time event.

Note:
The RTCSDDef() shutdown interval is initiated only after the "RTC Shutdown Event" (RTCSDE32Def). Initiating a Shutdown results in the Host CPU being notified of an impending shutdown of the HESC outputs.

Range: 0 to 65535 seconds. Setting RTCSDDef equal to zero disables the timer interval and no shutdown will occur due to a RTC Shutdown Event.

Purpose: Shutdown timer allows time to close files, and properly stop the operating system.

37. RTCSDEloDef(0xF0), RTCSDEhiDef(0xF2), [32-bit name is RTCSDE32Def]

Description:

Defines the "RTC Shutdown Event" time in seconds from January 1, 2001, 00:00 hours, using the RTC support provided on the BAT104-RTC. The RTCSDEloDef() and RTCSDEhiDef() are combined to create a 32-bit word RTCSDE32Def that allows for a programmed Shutdown Event to any second up to year 2137.

Note:
After the RTC Shutdown Event time is elapsed, the HESC uses the interval time in the RTCSDDef() to initiate the Shutdown. Initiating a Shutdown in the HESC results in the Host CPU being notified of an impending shutdown of the HESC outputs. If the RTCSDDef() is set to zero, no shutdown will occur due to a RTC time setting.

Range: 0 to 65535 for RTCSDEloDef and RTCSDEhiDef (0 to 4,294,967,295 seconds when combined into RTCSDE32Def).

Purpose:
Allows the HESC to turn off its outputs based on a "real time" instead of a time interval. Use of a real time clock controller can provide a more accurate stop of the system.

38. RTCSUloDef(0xF4), RTCSUhiDef(0xF6) , [32-bit name is RTCSUE32Def]

Description:
Defines the "RTC Start-up Event" time in seconds from January 1, 2001, 00:00 hours, using the RTC support provided on the BAT104-RTC. The RTCSUEloDef() and RTCSUEhiDef() are combined to create a 32-bit word RTCSUE32Def (0 to 4,294,967,295 seconds) that allows for a programmed Start-up Event to any second up to year 2137.

Note:
After the RTC Start-up Event time is elapsed, the HESC initiates a Start-Up request resulting in an immediate turn-on of the outputs. No Start-Up interval timer is required (unlike the RTCSDE32Def).

Range: 0 to 65535 for RTCSUEloDef and RTCSUEhiDef (0 to 4,294,967,295 seconds when combined into RTCSUE32Def).

Purpose:
Allows the HESC to turn on its outputs based on a "real time" instead of a time interval. Use of a real time clock controller can provide a more accurate start of the system.

39. BattMaxCapDef – EEprom Locations 0xFA, 0xFB

Description:
For standard battery packs (non-SMBus), BattMaxCapDef is required to be set to the capacity of the battery. BattMaxCapDef is used to provide a limit to the value BattRemCapCmd().

Purpose:
To assist in tracking remaining battery capacity. Useful to prevent runaway capacity values as a result of trickle charging.

Range: 0 to 65,535 x 10mWh –capacity in 10mWh increments.

40. BattRemCapDef – EEprom Locations 0xFC, 0xFD

Description:
For standard battery packs (non-SMBus), BattRemCapDef is used to store the value value BattRemCapCmd() into EEprom just before the HESC isolates itself from a battery pack equipped with isolating Mosfet controlled by the BE output signal.

Purpose:
To assist in tracking remaining battery capacity. The HESC loads the value stored in BattRemCapDef when "cold-started" (ie: when power is applied to an HESC that has no main or battery power connected).

Range: 0 to 65,535 x 10mWh –capacity in 10mWh increments.

41. BattTime2RechargeDef - EEprom Locations 0xFE, 0xFF:

Description:
Minutes from last charge termination to start of a recharge battery cycle.

Purpose:
Prevents batteries from self-discharging if HESC has terminated charging for a long time period.

Range: 0 to 65,535 x minutes – delay time in minute increments.

Section 9 : Temperature Sensor, TS-IC and Controller, TC-IC

The HESC can support up to a total of eight TS-IC sensors (TCN75 compatible), or four TS2-IC sensors (LM92 compatible), or four 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 > (65535 - I2ChiLoAlarm#(high byte)), then High alarm setting = 65535;

For TS-IC: High alarm value. Alarm range = 0 to 65,535.

Table 31 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. 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.
1	Cmd bit 1	
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 (TC-IS): For TC-IC set bit to 1, for TS-IC set bit to 0

Note: The "#" represent the number (0 to 7) of the TC-IC or TS-IC device.

Section 10 : Battery Charging Algorithms and Strategies

A. 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 32 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 33 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

B. Dual Battery Support – HESC-SERD, HESC104+, and V5SC

The HESC-SERD, HESC104+ and V5SC have special hardware and the HESC-UPS18 provides firmware support for dual battery operation. Batteries such as the BAT-NiMh2, and BAT-NiMh2-RTC are compatible with these supplies and can provide dual battery operation.

Because of chemical limitations inherent in charging NiMh batteries, the maximum capacity of a battery that can be safely charged by these supplies is eight ampere-hour using a 0.5C charge rate. Charge rates between trickle charge (C/20) and C/2 should be avoided. Batteries can be safely paralleled if assembled correctly and with batteries of the same age and condition. Three AA size NiMh batteries paralleled into a battery pack will achieve a capacity of 6.3 ampere-hours. Larger 18650 NiMh cells can supply up to 4.5 ampere-hours and two can be safely paralleled and charged by the HESC chargers.

The dual charging support HESC-SERD, HESC104+ provide for maximum backup capacity of sixteen ampere-hours (two eight ampere-hour battery packs). Where “hot-swapping” battery packs is required, the charging support of the V5SC allows an external battery to be replaced while the on-board batteries of the V5SC provide the “power bridging” support.

Section 11 : HESC-UPS18 FIRMWARE UPDATE PROCEDURE

NOTE: The HESC profile should always be updated before HESC-UPS18 firmware is loaded if the HESC did not have HESC-UPS18 firmware already loaded. This will load known values into the EEprom and prevent a situation where a default value of 0xFFFF for a start up timer will prevent the HESC outputs from turning on. Should the situation arise where the outputs can't be turned on because of an incorrect value in the HESC EEprom, the HESC can be "hot wired" by connecting +5 volts from another source to the HESC +5 volt output. This will not cause any harm or damage to the HESC.

****

To update the EEPROM profile:

****

1. Connect serial cable between Serial Port connector indicated on the HESC-SER(D) heat sink and host CPU COM port or plug Host CPU board onto HESC104 PC/104 connector.
2. Apply main power to the Main Input of HESC.
3. Start SCU 3.0.15 utility on host.
4. Select INIT-DEBUG SETUP from menu and select AUTO_SELECT; press OK button after HESC is initialized.
5. Select FILE-OPEN from menu and select the file.
6. Select EEPROM from menu and select UPDATE ALL.
7. Select WRITE to update the profile.
8. Close SCU after profile is updated.
9. Remove main power.

****

To update the firmware:

****

1. Connect a wire between SDA, Th(1) and COM from the External Battery connector to put the power supply in firmware upgrade mode.
2. Apply power to the Main Input of HESC.
3. Start SCU 3.0.15 utility on host.
4. Select FILE-OPEN from menu and select the firmware file.
5. Select INIT-DEBUG SETUP and select the following:
    a. Access Type: Select correct port type and address.
    b. Protocol: Select "Bus Timer and No Checksum".
    c. Firmware Type: Select "HESC-UPS18".
6. Press OK button after HESC is initialized.
7. Select INIT-FW UPDATE and press the START/STOP UPDATE button.
8. Read the info from the dialog box, wait for at least 8 seconds and then press OK button.
9. Remove the wire connecting SDA, Th(1) and COM 'AFTER' firmware update is started.
10. Close SCU after update is completed..
11. Remove main power.
12. Wait for 30 seconds after power down.
13. Apply main power again.

Note that once you have the Input Power applied and the jumper wire SDA, Th to COM installed you should always wait for at least 8 seconds before attempting to unplug the Main power, as this could result in non-reversible non-operating power supply condition!!!

Section 12 : Implementing a smart “Watchdog Timer” on the Host CPU

A smart watchdog timer can be implemented using a one of the HESC-UPS18 commands. If the Host CPU fails to “tickle” the HESC within the watchdog interval, the HESC will turn off the outputs to the Host CPU, then turn the outputs back on resulting in a “cold boot”. The watchdog interval, as well as the interval when the outputs are turned off can be set.

The following steps must be taken to implement the watchdog function.

1. Set the default command shut down time interval in the EEProm profile to a value larger than the required watchdog interval. This interval is in seconds and can have a value up to 65535 seconds and can be set with the SCU.exe utility from Tri-M Engineering.

2. Set the default command start up time interval in the EEProm profile for the Host CPU to remain unpowered before starting up again. This interval is in seconds and can have a value up to 65535 seconds and can be set with the SCU.exe utility from Tri-M Engineering.

3. The Host CPU must continue to issue the command SDSUCauseCmd() (0x99) with bits 7 and/or 15 set to tickle the HESC at a rate greater than the watchdog interval for normal operation to continue.

Note: The Host CPU can halt watchdog mode with SDSUCauseCmd() by clearing both bits 7 and 15. Setting bits 7 and/or 15 at any time with SDSUCauseCmd() will restart the watchdog mode.

Section 13 : HESC-UPS LED Operation

1. The HESC-UPS firmware toggles the LED-HB (CPU heart-beat) and the LED-CH (Charger indication) LEDs displaying the current status and operation of the HESC. (Note: The HESC104 does not have an LED-CH LED.)

2. During normal operation (non-firmware upload), the LED-HB toggles continuously. The frequency of the toggling is 1Hz when not in a start-up or shutdown mode. In a Shutdown mode the toggling frequency is 0.5Hz and for a Start-up mode the frequency is 2Hz (Table1).

3. The LED-CH is off when not charging and when charging, the LED-CH toggles. The frequency of the toggling is according to the start-up and shutdown modes (Table 2). When charging is terminated, the LED-CH turns steady on.

4. When both LED-HB and LED-CH are toggling, they toggle opposite to each other.

5. When a standard battery pack is detected as a low, the LEDs will toggle at a 4 Hz rate. The battery Low Level Condition can be disabled by setting the default EEprom register BattLowVoltageDef and BattLowCapacityDef to zero. There is no method of overriding the Low Level Condition when issued by an SMBus battery (other than romving the battery).

Description		LED-HB
Battery Level Condition	**HESC operating mode**	**Toggle Rate**
Low Battery	N/A	Toggle @ 4 Hz
Battery Okay	Normal mode	Toggle @ 1Hz
Battery Okay	Shutdown mode	Toggle @ 0.5Hz
Battery Okay	Start-up mode	Toggle @ 2Hz

Table 34 Heartbeat LED toggle rates

Description		LED-CH
Battery Level Condition	**HESC operating mode**	**Toggle Rate**
Low Battery	N/A	Toggle @ 4 Hz
Battery Okay	HESC is not in charge mode	Off
Battery Okay	HESC is charging in Normal mode	Toggle @ 1Hz
Battery Okay	HESC is charging in Shutdown mode	Toggle @ 0.5Hz
Battery Okay	HESC is charging in Start-up mode	Toggle @ 2Hz
Battery Okay	HESC has terminated charging	Steady on

Table 35 Charger LED Operation

Section 14 : Adding an External Real Time Clock to a HESC

New in the HESC-UPS18 firmware is support for an external Real Time Clock (RTC) controller (Xicor X1226). The X1226 uses the I2C/SMBus 2-wire signalling that comprises of a clock signal (SCL) and a data signal (SDA). The only connections required between the HESC and the X1226 are the SCL, SDA, and GND. These signals are provided on the HESC connector CN5 and CN7 as below:

Table 36 **Location of Interface Signals on HESC**

Signals on HESC	HESC CN5	HESC CN7
SCL	6	5
SDA	5	4
GND	2&4	2

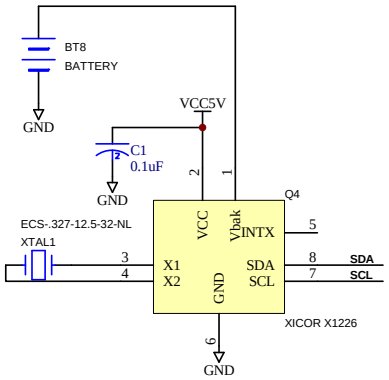


Figure 2, Xicor X1226 Wiring

In addition to the ability to read/write the time in the RTC, the HESC can program the Start-Up time and Shutdown time in the RTC. When the Start-Up setting matches the RTC time, the INTX (X1226 pin 5) goes active (low). The INTX output is an “open drain” output and can be used to enable an isolating mosfet to awaken a hibernating battery pack. Figure 2 use Q1B to wake initially turn on the main isolating Mosfet Q3. After Q3 supplies power from battery BT1 to the HESC, VCC5V will be generated and the HESC will assert Q1A thus keeping the battery enabled. Battery BT2 is a primary Lithium cell to maintain the RTC operation when main power is not available. A jumper applied across CN2 pin 1 & 2 will also wake up the battery pack.

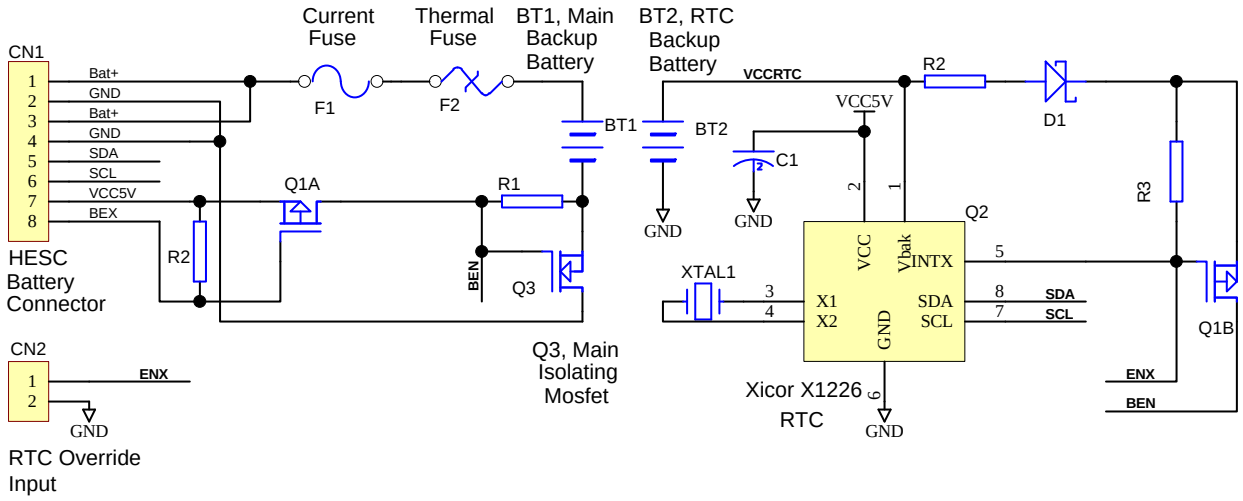


Figure 3, RTC with Alarm Wake-Up function

Section 15 CRC-8 Firmware Implementation

The HESC-UPS18 supports a CRC-8 for the PC/104 and the RS232 SerBus. CRC values are calculated upon a message viewed as a bit-stream. The computed CRC is the remainder of the CRC division between the message bit-stream and a polynomial. The polynomial used in the CRC-8 calculations is: X8+X2+X1+X0. The CRC-8 will detect all single bit errors, all odd number of bit errors, any burst error less than 8-bits long, and most of other type of errors.

Using a 256 x 8 table, a very simple algorithm to calculate the CRC-8 can be performed of a multi-byte message.

The algorithm is as follows:

- Initialize the CRC register to 0.
- XOR the ADDR, CMD and two bytes of data to create the CRC value for both the Read and Write command.
- Use this value as the index to the table to obtain the CRC value up to that point.
- Continue until all bytes have passed through the process.
- The last byte retrieved from the table is the final CRC value.

CRC-UPS18.C can be downloaded from the Tri-M Engineering WEB support site at:
"http://engineering.tri-m.com/products/engineering/files/firmware/HESC-UPS18/CRC-UPS18.zip".

```
// ---------------------------------------------------------------------------------------------------------------------------------------------
//    Table lookup method for generating a CRC-8 value on a byte-by-byte basis.
//    At the start of each start of each message or datalogging, the CRC value must be set to zero.
//    After each byte is received or written, the data value and the current CRC value is passed to
//    the Crc8 subroutine for processing.  The Crc8 will return the calculated CRC-8 value to that
//    point.  The CRC value returned after the last byte is processed is the final CRC-8 value.
//---------------------------------------------------------------------------------------------------------------------------------------------
int8 Crc8(int8 data2process, int8 CRCvalue)
{
  int8 const table[256] =
  {
      0x00,0x07,0x0E,0x09,0x1C,0x1B,0x12,0x15,0x38,0x3F,0x36,0x31,0x24,0x23,0x2A,0x2D,
      0x70,0x77,0x7E,0x79,0x6C,0x6B,0x62,0x65,0x48,0x4F,0x46,0x41,0x54,0x53,0x5A,0x5D,
      0xE0,0xE7,0xEE,0xE9,0xFC,0xFB,0xF2,0xF5,0xD8,0xDF,0xD6,0xD1,0xC4,0xC3,0xCA,0xCD,
      0x90,0x97,0x9E,0x99,0x8C,0x8B,0x82,0x85,0xA8,0xAF,0xA6,0xA1,0xB4,0xB3,0xBA,0xBD,
      0xC7,0xC0,0xC9,0xCE,0xDB,0xDC,0xD5,0xD2,0xFF,0xF8,0xF1,0xF6,0xE3,0xE4,0xED,0xEA,
      0xB7,0xB0,0xB9,0xBE,0xAB,0xAC,0xA5,0xA2,0x8F,0x88,0x81,0x86,0x93,0x94,0x9D,0x9A,
      0x27,0x20,0x29,0x2E,0x3B,0x3C,0x35,0x32,0x1F,0x18,0x11,0x16,0x03,0x04,0x0D,0x0A,
      0x57,0x50,0x59,0x5E,0x4B,0x4C,0x45,0x42,0x6F,0x68,0x61,0x66,0x73,0x74,0x7D,0x7A,
      0x89,0x8E,0x87,0x80,0x95,0x92,0x9B,0x9C,0xB1,0xB6,0xBF,0xB8,0xAD,0xAA,0xA3,0xA4,
      0xF9,0xFE,0xF7,0xF0,0xE5,0xE2,0xEB,0xEC,0xC1,0xC6,0xCF,0xC8,0xDD,0xDA,0xD3,0xD4,
      0x69,0x6E,0x67,0x60,0x75,0x72,0x7B,0x7C,0x51,0x56,0x5F,0x58,0x4D,0x4A,0x43,0x44,
      0x19,0x1E,0x17,0x10,0x05,0x02,0x0B,0x0C,0x21,0x26,0x2F,0x28,0x3D,0x3A,0x33,0x34,
      0x4E,0x49,0x40,0x47,0x52,0x55,0x5C,0x5B,0x76,0x71,0x78,0x7F,0x6A,0x6D,0x64,0x63,
      0x3E,0x39,0x30,0x37,0x22,0x25,0x2C,0x2B,0x06,0x01,0x08,0x0F,0x1A,0x1D,0x14,0x13,
      0xAE,0xA9,0xA0,0xA7,0xB2,0xB5,0xBC,0xBB,0x96,0x91,0x98,0x9F,0x8A,0x8D,0x84,0x83,
      0xDE,0xD9,0xD0,0xD7,0xC2,0xC5,0xCC,0xCB,0xE6,0xE1,0xE8,0xEF,0xFA,0xFD,0xF4,0xF3
   };
  CRCvalue = table[CRCvalue ^ data2process];

  return CRCvalue;
} // end Crc8
```

Example code snippets demonstrating Read and Write checksum calculations.

```
// ---------------------------------------------------------------------------------------------------------------------------------------------
//    verifyread_Chksum is used to verify the checksum during a Read command.
//    The ADDR and CMD are sent by the Host to the HESC.  After receiving DATALOW and DATAHIGH (as well as
//    ACKs) sent by the HESC, the checksum verifyread_Chksum() procedure validates not only the returned
//    DATALOW and DATAHIGH, but the requested command request.
// ---------------------------------------------------------------------------------------------------------------------------------------------
    int1 verifyread_Chksum(void)
    {
        CRC8val = 0;                                // Initialize as required by Crc8
        CRC8val = Crc8(ADDR, CRC8value);            // Compute CRC8 value for the ADDR
        CRC8val = Crc8(CMD, CRC8value);             // Compute CRC8 value for the ADDR
        CRC8val = Crc8(DATALOW, CRC8value);         // Compute CRC8 value for the ADDR
        CRC8val = Crc8(DATAHIGH, CRC8value);        // Compute CRC8 value for the ADDR

        if (CRC8val == CHKSUM)                      // Does received checksum match computed value?
            COMstatus = true;                       // Received data is okay.
        else COMstatus= false;                      // Signal Failure.

        return COMstatus;
    } // end verifyread_Chksum

// ---------------------------------------------------------------------------------------------------------------------------------------------
//    calc_Chksum is used to generate the checksum for a Write command.
//    Procedure calc_Chksum() calculates the checksum of ADDR, CMD, DATALOW, DATAHIGH.   Afterwards,
//    ADDR, CMD, DATALOW, DATAHIGH and CHKSUM can be sent by the Host to the HESC.
// ---------------------------------------------------------------------------------------------------------------------------------------------
    void calc_Chksum(void)
    {
        CRC8val = 0;    // Initialize
        CRC8val = Crc8(ADDR, CRC8value);            // Compute CRC8 value for the ADDR
        CRC8val = Crc8(CMD, CRC8value);             // Compute CRC8 value for the ADDR
        CRC8val = Crc8(DATALOW, CRC8value);         // Compute CRC8 value for the ADDR
        CRC8val = Crc8(DATAHIGH, CRC8value);        // Compute CRC8 value for the ADDR

        CHKSUM = CRC8value;                         // Calculated checksum
    } // end calc_Chksum
```