Data Sheet

[Product Name]

BMS (Battery Management System)

[Model No.]

LT-BMS

[Release Note]

Edited By:	Ivan
Approved By:	River
Issue Date:	2011/10/19

LiFeTech Energy Inc.

No.30,Chung-Shan Rd., Tu-Cheng Dist.,
New Taipei City 23680,Taiwan
Tel: +886 2 2267 6733
Fax: +886 2 2267 2333

1. Scope

BMS supports 5~13 Series LiFePO4 Battery Cells of the LiFePO4 rechargeable batteries supplied by LiFeTech Energy Inc.

2. Model No. & Type

2.1 Model No.:	LT-BMS
2.2 BMS type:	Passive

3. Specification

3.1 Capability:

3.1.1 Each BMS supports cells in Series:	5~13
3.1.2 Max BMS in series:	16
3.1.3 Max total cells in series	16 x13=208
3.1.4 Max total voltage support	665.6V

3.2 OVP (Over Voltage Protection)

OVP on:	3.9V
OVP Release:	3.5V
OVP Delay Time:	1 Sec

(Values can be modified by special request)

3.3 UVP(Under Voltage Protection)

UVP On:	2.5V
UVP Release:	2.8V
UVP Delay Time:	1 Sec

(Values can be modified by special request)

3.4 OCP (Over Current Protection)

OCP On:	30~75A
OCP Release:	Remove Loader
OCP Delay Time:	8/16/32 Sec
Short Circuit:	60~150 A
Short Circuit Release:	Remove Loader
Short Circuit Delay Time:	96uSec

(OCP & SC functions are available only when Cut-Off board is installed)

3.5 OTP(Over Temperature Protection)

OTP On:	85℃
OTP Release:	60℃

(Values can be modified by special request)

3.6 Cell Balance current:	150mA
Cell Balance Voltage Start:	3.5V
Cell Balance Delta Voltage:	20mV

3.7 Mode of operation & Supply current

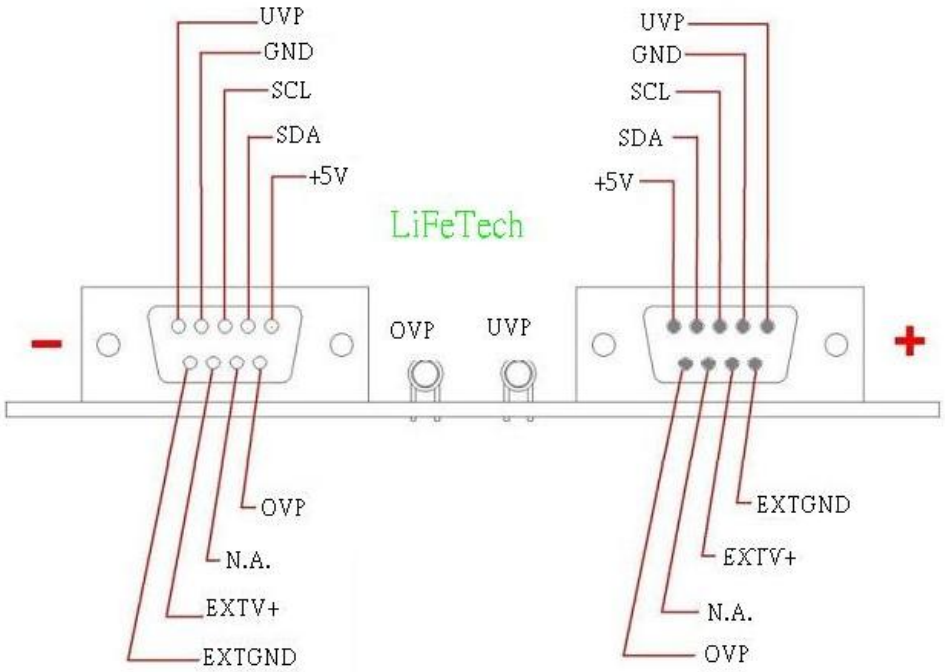
Standby Mode:	1mA
Sleeping Mode:	0.1mA

(When UVP is on, BMS will enter sleeping mode)

3.8 Environmental Temperature

Operation Temperature Range:	-40℃ ～ +90℃
Storage Temperature Range:	-50℃ ～ +150℃
10Sec Temperature for soldering:	+300℃

4. Pin Assignment:



4.1 +5V: input +5V to Pull High SDA & SCL. (Comm. power)
4.2 SDA: I2C Data
4.3 SCL: I2C Clock
4.4 GND: Comm. Ground
4.5 UVP - Normal: +5V/+10V，UVP on: 0V (Pair with EXTGND).
4.6 OVP - Normal: +5V/+10V，OVP on: 0V (Pair with EXTGND.)
4.7 EXTGND: External cut off ground.
4.8 EXTV+: External power input 5~12v.
4.9 N.A.: backup pin. No signal.

***Remark:**
OVP / UVP pin is used for external cut off requirement and only functioning when the EXTV+ pin has external power input. Without external power input, the UVP / OVP pin will not provide any voltage signal output.

5. Data Display

With Data Box & PC connected, Cells' voltage, current and temperature can be shown on PC or laptop.

6. Software Guide

6.1 Support I2C &RS232 Interface .
6.2 Slave Address 0x60~0x7E,
6.3 Voltage(V)= (VLow byte+VHigh byte*256)/6.5536
6.4 Temperature

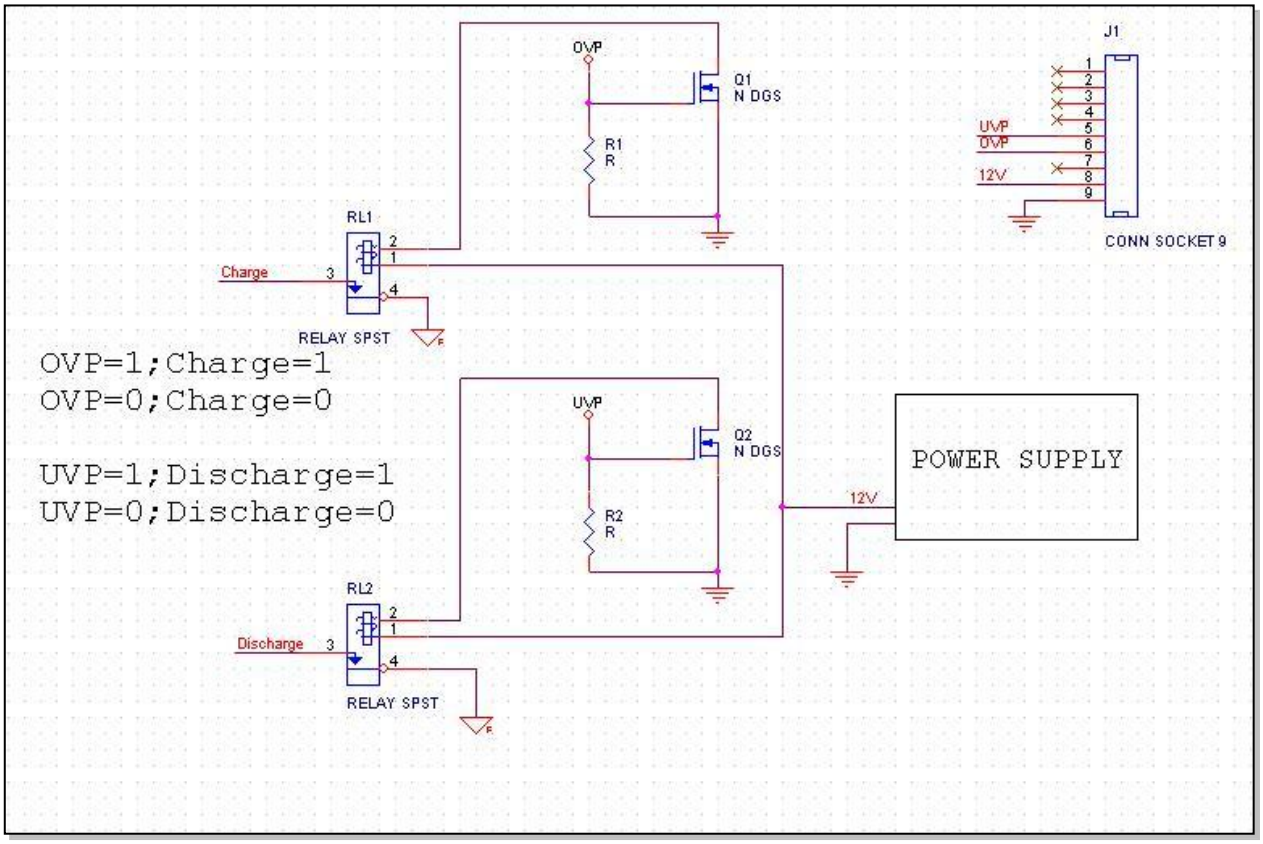
Vt= (float)(TLow byte+Thigh byte*256)/32768*2500;
Res=230000*Vt/(3300-Vt);
k=(1/298.15)-(log(10000/Res)/3435);
Temperature(C) =(1-273.15*k)/k;

6.5 Current

Current(A)=(CLow Byte+CHigh Byte*256)*0.25/32768/Rsense

Register address	function	Remark
0x1F	Battery Status	Bit0=OVP Bit1=UVP Bit2=OCP Bit3=SC Bit4=OTP
0x32	V1Low byte	
0x33	V1High byte	
0x34	V2Low byte	
0x35	V2High byte	
0x36	V3Low byte	
0x37	V3High byte	
0x38	V4Low byte	
0x39	V4High byte	
0x3A	V5Low byte	
0x3B	V5High byte	
0x3C	V6Low byte	
0x3D	V6High byte	
0x3E	V7Low byte	
0x3F	V7High byte	
0x40	V8Low byte	
0x41	V8High byte	
0x42	V9Low byte	
0x43	V9High byte	
0x44	V10Low byte	
0x45	V10High byte	
0x46	V11Low byte	
0x47	V11High byte	
0x48	V12Low byte	
0x49	V12High byte	
0x4A	V13Low byte	
0x4B	V13High byte	
0x4C	T1Low byte	
0x4D	T1High byte	
0x4E	T2Low byte	
0x4F	T2High byte	
0x54	C Low Byte	Need use Rsense
0x55	C High Byte	Need Use Rsense

7. Example of external cut off function.



8. I2C To RS232 Tool Code

RS232 Setting: (Baud Rate=19200 bps,Data bits=8,Stop Bits=1, Parity=NOPARITY)

8.1 SendCommandToI2CTool:

Example:Write 3 byte to I2CTool: 01 03 06 (Hex)

PCSend: (Length+2) Data1 Data2 Data3 checkSum

05 01 03 06 0E

I2CTool Return : 0E (read checkSum for verify I2CTool read command)

Checksum= Length+DATA1+...DATAn

0E = 05 +01+03+06

8.2 ToI2CToolToPC:

I2CTool: Write 3 byte to PC: 01 03 06 (Hex)

I2CTool: (Length+2) Data1 Data2 Data3 checkSum

05 01 03 06 0E

8.3 SetI2CChannel: (This Tool Is 1Rs232 To 8channel I2C,for our BMS24V MAX Cell Voltage detect=8*13*16=1664 ,8=I2C Channel, 13= Bms Max detect Voltage,16=slaveaddress.)

PCToI2CTool: 04 0B 00 0F

04=Length

0B=SetChannel Command

00=Set Channel 0 (0~7)

0F=04+0B+00 checksum

I2CTool Return : 0F checksum

Please set channel=0,for initial set.

8.4 I2CBusyorFree

PCToI2CTool: 03 12 15

03=Length

12=Command

15=Checksum

I2CToolReturn: 15Checksum

I2CToolReturn:03 80 83

03=Length

80 =complete (if return 81=Busy, please wait I2CTool,if return 82=fail,please resent command)

83=checksum

8.5 ReadI2C

PCToI2CTool: 06 0D 60 40 4 B7

06=Length

0D=ReadI2CCommand

60=SlaveAdd

40=address

4= readI2Creturn total data, read 4 data, (add40 ,add41,add42,add43) ,Max Total data=16 (0x10)

B7=06+0D+60+40+4checksum

I2CToolToPC=B7 checksum

I2CToolReturn: 06 01 02 03 04 10

06=Length

01=address40 data

02=address41 data

03=address42 data

03=address43 data

10=checksum

8.6 ReadIDN

PCToI2CTool: 03 03 06

03=length

03=ReadIDNCommand

06=checksum

I2CToolToPC=06 checksum

I2CToolToPC=Length R i v e r checksum

07 52 69 76 65 72 0F

9.C++Code reference

```
bool __fastcall TCtrlBox::ReadI2c(Byte SlaveAddress, Byte Address, int DataLength ,Byte ReBuf[256])
    {

    if( ( Address+DataLength >256 ) || (DataLength == 0 ) ||(DataLength >256 ) )
        return false;

    Byte TrBuf[30],Buf[30];
    int p=0;
    int PageNumber = (int)(DataLength/16);
    int ByteNumber = (int)(DataLength%16);

    //---------------- Page read (16 Byte each time) ---
    for (int i=0; i<PageNumber; i++)
        {
        TrBuf[0] = 6;
        TrBuf[1] = FUNC_READ_I2C;
        TrBuf[2] = SlaveAddress;
        TrBuf[3] = Address;
        TrBuf[4] = 16;
        if ( SendCommand(TrBuf) && GetReturnData(Buf) &&
(ProcessStatus==COMPLETED) )
            memcpy (&ReBuf[p],Buf,16);
          else
            return false;
        Address += (Byte)16;
        p+=16;
        }
    //---------------- Byte read ----------------------
    if( ByteNumber >0)
        {
```

```
        TrBuf[0] = 6;
        TrBuf[1] = FUNC_READ_I2C;
        TrBuf[2] = SlaveAddress;
        TrBuf[3] = Address;
        TrBuf[4] = (Byte)ByteNumber;
        if ( SendCommand(TrBuf) && GetReturnData(Buf) &&
(ProcessStatus==COMPLETED) )
            memcpy (&ReBuf[p],Buf,ByteNumber);
          else
            return false;
        }
    return true;
    }
```