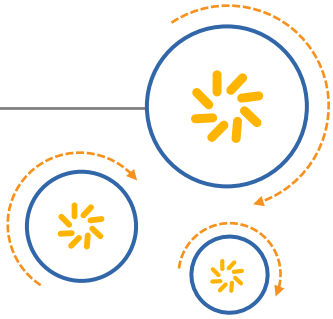
Qualcomm Technologies, Inc.



PM8941 Battery Monitoring System (BMS)

Hardware/Software Application Note

80-NE399-12 Rev. B

May 7, 2015

Questions or comments: https://support.cdmatech.com/

Qualcomm Technologies, Inc.
5775 Morehouse Drive
San Diego, CA 92121
U.S.A.

Revision history

Revision	Date	Description
A	December 2012	Initial release
B	May 2015	▪ Changed the document title from *PM8941 Battery Monitoring System (BMS) Hardware/Software Document* to *PM8941 Battery Monitoring System (BMS) Hardware/Software Application Note* ▪ Table 1-2 Reference documents: Removed the document *PM8841 and PM8941Training Slides* (80-NA555-21), as it is now obsolete ▪ Section 3.2.2.1 Battery removed or inserted after boot: Added information about the 0xB0 register reset

Contents

Figures

Tables

1 Overview

1.1 Summary

The battery monitoring system (BMS) is comprised of hardware and software components. The hardware module provides the necessary functions to monitor the battery capacity by gathering data from the current analog-to-digital converter (IADC) and the voltage analog-to-digital converter (VADC). The algorithm in this module is designed to work autonomously. All of the necessary information is stored in registers and can be requested via software.

This software provides the ability to configure the BMS hardware, collect necessary data, and calculate the battery state-of-charge (SoC).

This document describes the BMS as part of the PM8941 module.

1.2 Terms and acronyms

Table 1-1 lists terms and acronyms used throughout this document.

Table 1-1 Terms and acronyms

Acronym	Description
AFE	Analog front-end
AMUX	Analog multiplexer
ATE	Automated test equipment
BMS	Battery monitoring system
CC	Coulomb counter
DVT	Design verification testing
FCC	Full charge capacity
FSM™	Finite state machine
GPIO	General purpose input/output
IADC	Current analog to digital converter
LDO	Low drop-out (voltage regulator)
LSB	Least significant byte
MBG	Master band gap
MPP	Multipurpose pin
MSB	Most significant byte
MSM™	Mobile Station Modem
NA	Not applicable
OCV	Open circuit voltage

Acronym	Description
PA	Power amplifier
PC	Percentage charge
PMIC	Power management IC
RC	Remaining charge
Acronym	Description
RTC	Real-time clock
RUC	Remaining usable charge
SBL	Secondary boot loader
SoC	State-of-charge
SPMI	Single-wire interface
UUC	Unusable charge
USB	Universal serial bus
VADC	Voltage analog to digital converter
VI	Verification/integration
XO	Crystal oscillator

1.3 Reference documents

Table 1-2 lists reference documents applicable to the PM8941 device.

Table 1-2 Reference documents

Number	Document	DCN
1	*PM8941 Device Specification*	80-NA555-1
2	*PM8941 Device Revision Guide*	80-NA555-4
3	*PM8841 and PM8941 Power Management Design Guidelines*	80-NA555-5

1.4 Changes between PM8921 and PM8941

The BMS module was modified from the prior design in PM8921 and includes several new key features:

- Internal R_{sense} via BATFET – The BATFET gate control is designed to maintain a fixed Rds(on) so that the IADC can reliably use BATFET as a sensing element.
- Optional external R_{sense} – If customers require an external sense resistor, the design has a multiplexer that can select either the internal or external sensing element.
- Improved poweron open circuit voltage (OCV) performance – The master band gap (MBG) settling time has been reduced. Additionally, control registers in state S7 (poweron state) have been added to control both the delay and the number of averages.
- Configurable OCV interrupt thresholds – The ability to interrupt only on OCVs reaching a specific threshold has been added. This feature assists with the interrupt driven SoC calculations.

- Coulomb counter configuration – Additional controls have been added to the main Coulomb counter. Limits can be set to avoid OCV updates in the flat portion of the battery curve. The ability to select whether the Coulomb counter resets when an OCV occurs allows continuity of the Coulomb count in sleep/standby modes.
- Software Coulomb counter – A software-dedicated shadow Coulomb counter has been added. This Coulomb counter does not reset with OCV updates and is completely under software control.
- Charger interface – Additional signaling with the charger has been added to improve voice core processing (VCP) capability.

1.5 PM8941 issue list

Refer to the *PM8941 Device Revision Guide* (80-NA555-4) for all BMS-related issues.

2 BMS functional description

2.1 Architecture

Figure 2-1 shows the basic functional block diagram for the entire BMS. The BMS controller is the only dedicated BMS hardware. The IADC uses the internal BATFET as the sensing element to measure current. The BATFET resistance is maintained at the trimmed value when closed. The IADC reads the voltage across the BATFET, and the software converts the voltage reading to current using the BATFET trim value for resistance. As an alternate option, an external sense resistor can be used when necessary. The BMS controller is the primary client for IADC data, but other clients can request measurements. The VADC is used by the BMS to measure battery voltage. The applications processor can read stored values for OCV and the Coulomb count from the BMS controller, as needed, to calculate the SoC.

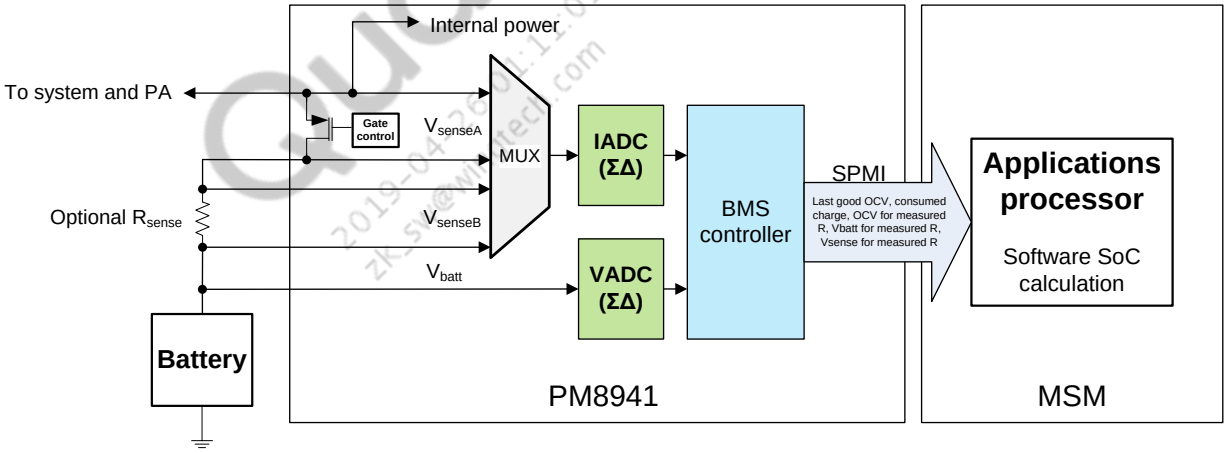


Figure 2-1 BMS block diagram

2.2 BMS controller

The BMS controller is intended to add a layer of control and intelligence over the IADC and VADC. The ADCs contain the analog portion and the immediate digital control (clock generation, decimation filter, switched capacitor control, etc.). The BMS controller issues conversion requests that initiate the IADC and VADC measurements and the logic to determine necessary data and set the proper timing for an optimally efficient and accurate SoC determination. It also independently identifies and stores certain key values for increased SoC accuracy without requiring software interference.

Figure 2-1 shows the BMS controller's role with respect to the remaining components involved in SoC determination and battery monitoring. The two key values in SoC determination are the battery voltage V_{batt} and the voltage across the sense resistor – which corresponds to the current through the battery – V_{sense}. The BMS controller instructs the VADC and the IADC to produce digitized V_{batt} and V_{sense}, as needed to determine the SoC. The BMS controller presents its results as raw data to the SoC approximation software located on a processor.

A voltage representing the battery temperature (V_{th}) is also collected for SoC determination, but this is not managed by the BMS controller.

The module features include:

- Automated Coulomb counting
- Intelligent determination of settled battery voltage for reliable battery OCV measurements
- Coulomb counter updates from the battery OCV to reduce an integrated error
- Measurement of raw data for internal battery resistance estimation
- User programmability with separate settings for active and standby modes:
  - Measurement frequency (0 ms delay to 16.5 s delay)
  - Samples per averaged measurement (4 to 512 samples)
  - Conditions for state changes (thresholds and durations)
- Fully internal finite state machine (FSM)
- Override mode available for software-driven operation

With these features, the raw data necessary for SoC calculation is continuously provided to the software without the need for additional software interaction. The end-of-line SoC software only needs to poll the outputs of the BMS controller when an update is requested from the client; no other hardware-software interaction is required.

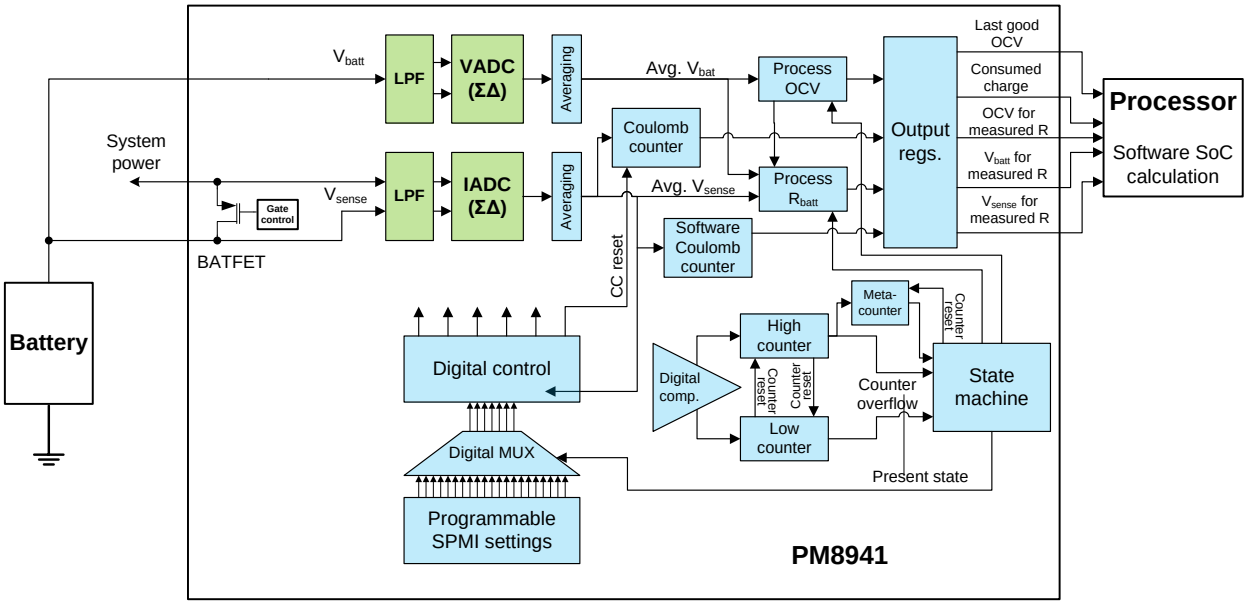


Figure 2-2 BMS block diagram (BMS controller shown in blue)

Figure 2-2 shows the BMS controller functional block diagram, with AFE and external blocks added for reference.

When the BMS controller instructs the IADC to perform a V_{sense} measurement, the associated averaging block accumulates and averages the ADC's results. This average V_{sense} is compared to a programmable threshold, and either the corresponding high or low counter is incremented. When either counter reaches an individually programmed overflow value, a signal is sent to the state machine of the BMS controller, both counters are reset to zero, and the state is updated, if necessary. (The meta-counter counts overflows of the high counter, but otherwise performs identically to the high and low counters.)

The appropriate configuration values defining a given state are forwarded to the digital control functional block from the programmable single-wire interface (SPMI) registers using a digital multiplexer with the present state serving as the select signal. The digital control block triggers the sampling of the averaging and high/low blocks, manages the delay between the measurements, controls ADC activation, resets the Coulomb counter when necessary, and decodes the threshold data from the register configuration values.

A primary Coulomb counter is implemented as a simple accumulator that updates the stored value on each clock tick, which is independent of the BMS state or ADC settings. The process OCV and process R_{batt} functional blocks perform additional comparisons of V_{batt} and V_{sense} measurements and forward the appropriate raw measurements to the output registers. The output read-only SPMI registers store the necessary raw data for a SoC approximation from the end-of-line SoC software. (See Section 2.2.7 for SoC calculation details.)

A secondary Coulomb counter has been added to maintain the Coulomb count independent of the OCV updates. This Coulomb counter is intended for software use and control.

2.2.1 FSM and state transitions

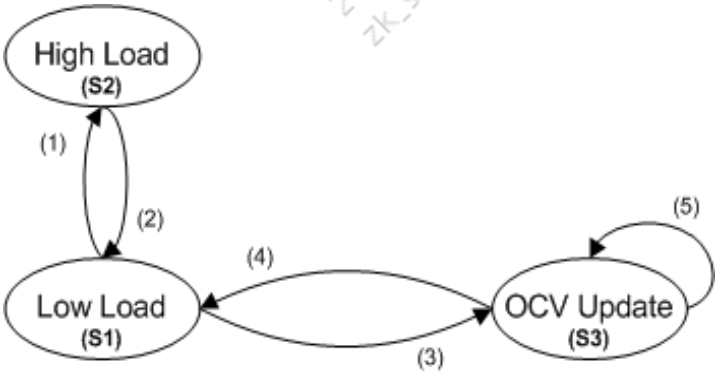


Figure 2-3 FSM and state transitions diagram

The BMS controller maintains a simple FSM with four states and eight arcs.

- Low load state (S1):
  - Reduces BMS power consumption by decreasing the measurement frequency during phone standby
- High load state (S2):
  - Increases the Coulomb counting accuracy with frequent measurements during active phone use

- OCV update (S3):
  - Requests simultaneous V_{batt} and V_{sense} measurements with a large number of samples
  - The process OCV block evaluates V_{batt}, and if indicated, updates the last good OCV output and resets the Coulomb counter to zero.
  - The process R_{bat} block updates OCV for the measured R, if indicated.

The state transitions are only triggered by a counter overflow. In a given state, both high and low counters operate simultaneously. The first counter to overflow triggers a state transition. As an example, if in S1, a transition to either S2 or S3 is determined by which counter overflows first. Transitioning from S1 to S2 requires eight high-current counts. Transitioning from S1 to S3 requires 128 low-current counts. If eight high-current counts occur in any order, prior to 128 low current counts, the state transitions from S1 to S2. A state transition does not require successive high or low counts. As counter overflows and V_{sense} thresholds are programmable, the state transitions can be altered to suit the specific system needs.

- Transition (1) [**S1** to **S2**] – Triggered by a high counter
  - A small threshold on the high counter allows for a quick response to an active load.
- Transition (2) [**S2** to **S1**] – Triggered by a low counter
  - A moderate threshold on the low counter reduces the chance of an incorrect drop to standby.
- Transition (3) [**S1** to **S3**] – Triggered by a low counter
  - A large threshold on the low counter sets delay before the OCV update is first attempted.
- Transition (4) [**S3** to **S1]** – Triggered by a low counter OR meta-counter
  - The low counter threshold = 1.
  - The OCV measurement has either completed successfully or failed (after repeated high measurements), so the state returns to standby.
- Transition (5) [**S3** to **S3**] – Triggered by a high counter
  - The high counter threshold = 1.
  - The OCV measurement may have occurred during modem current spike; retry the measurement.

Timing diagrams of the primary two states show when measurements are executed during operation. Figure 2-4 shows the timing diagram for the low load state. Figure 2-5 shows the timing diagram for the high load state.

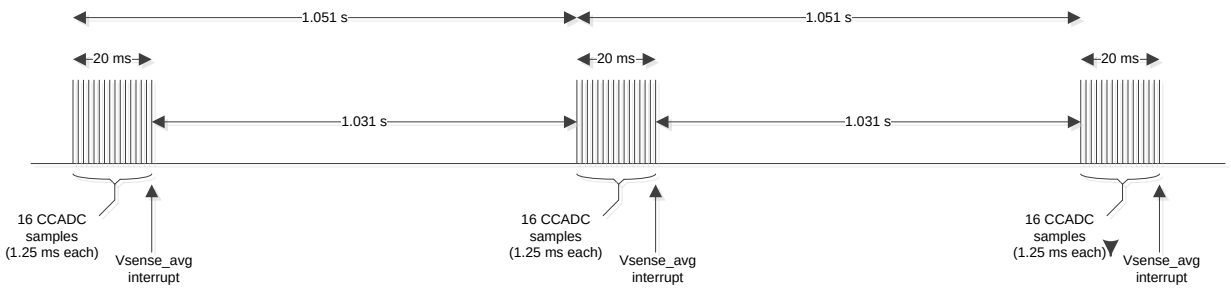


Figure 2-4 Low load state (S1) BMS timing diagram

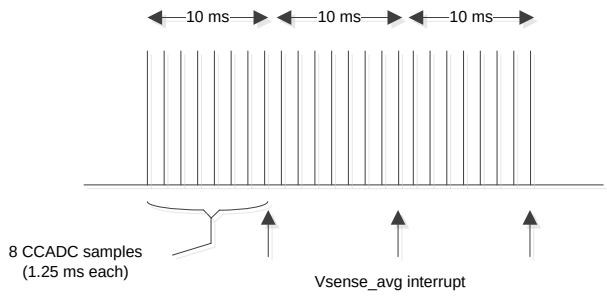


Figure 2-5 High load state (S2) BMS timing diagram

Figure 2-6 shows the OCV state (S3) BMS timing diagram.

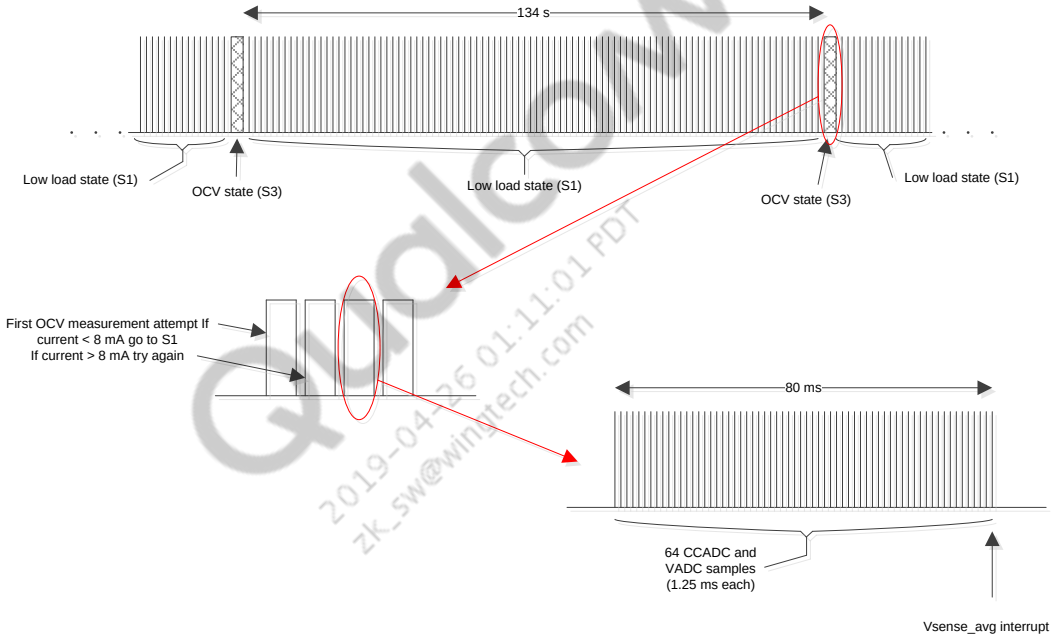


Figure 2-6 OCV state (S3) BMS timing diagram

Table 2-1 lists the times for transitioning between states and the approximate power consumption.

Table 2-1 BMS state definition, timing, and current

BMS state definition		
State definition	**Name**	**Description**
S1	Low load state	This is the primary state for standby/sleep. The BMS performs a current measurement every 1.0508 s and is active 1.9% of the time. The BMS stays in this state until 8 current measurements of > 50 mA or < 0 mA (charging) occur.
S2	High load state	This is the primary state for an active system. The BMS performs a current measurement every 9.824 ms but is operating 100% of the time. The BMS stays in this state until 32 current measurements of < 45 mA occur or when current is negative (charging).
S3	OCV state	The BMS measures the battery OCV in this state. This occurs approximately every 134.6 s if the battery current remains below 50 mA. A valid OCV measurement requires the current to be less than 8 mA at the time the battery voltage is measured. A qualifying good measurement requires two successive measurements to be within 800 µV to ensure the battery is settled.

State transition	Sample delays (s)	Sample average	Conversion times (s)	Count required	Transition times (s)	Count threshold requirement
S1 to S3	1.03215	16	0.01389	128	133.893	Current 0–50 mA to get into S3, and < 8 mA during OCV measurement
S1 to S2	1.03215	16	0.01389	8	8.368	Current > 50 mA
S2 to S1	0	8	0.00694	32	0.222	Current < 45 mA
S3 to S1	–	64	0.05555	4	0.222	The state transitions in less than four counts if a qualifying measurement occurs sooner (current < 8 mA)

State	Sample delays	Sample average	Conversion times	Conversion rates	On duty cycle (%)	Average current (mA)	Comments
S1	1.03125	16	0.01389	1.04514	1.3%	0.009	Low load state
S2	0	8	0.00694	0.00694	100%	0.700	High load state

2.2.2 OCV state probability

Obtaining periodic OCV measurements is critical to the accuracy of the BMS solution. To understand the probability of getting into the OCV update state (S3) in standby mode, a detailed study was conducted. Bursts of standby current for Bluetooth, WLAN, WAN, and GPS were simulated with a system in standby for 36 hours (of battery time), randomly shifting the phase of the bursts listed in Table 2-2 every 60 minutes.

Table 2-2 Feature burst currents in standby

Feature	Time between bursts	Burst duration	Burst current
Bluetooth standby	1280 ms	15 ms	15.0 mA
WLAN standby	100 ms	5 ms	25.0 mA
WAN standby	2500 ms	40 ms	150.0 mA
GPS tracking	1000 ms	230 ms	40.0 mA
Background	–	–	0.5 µA

The output from the simulation is the number of successful OCV measurements and good OCV BMS updates each hour. Figure 2-7 shows the results.

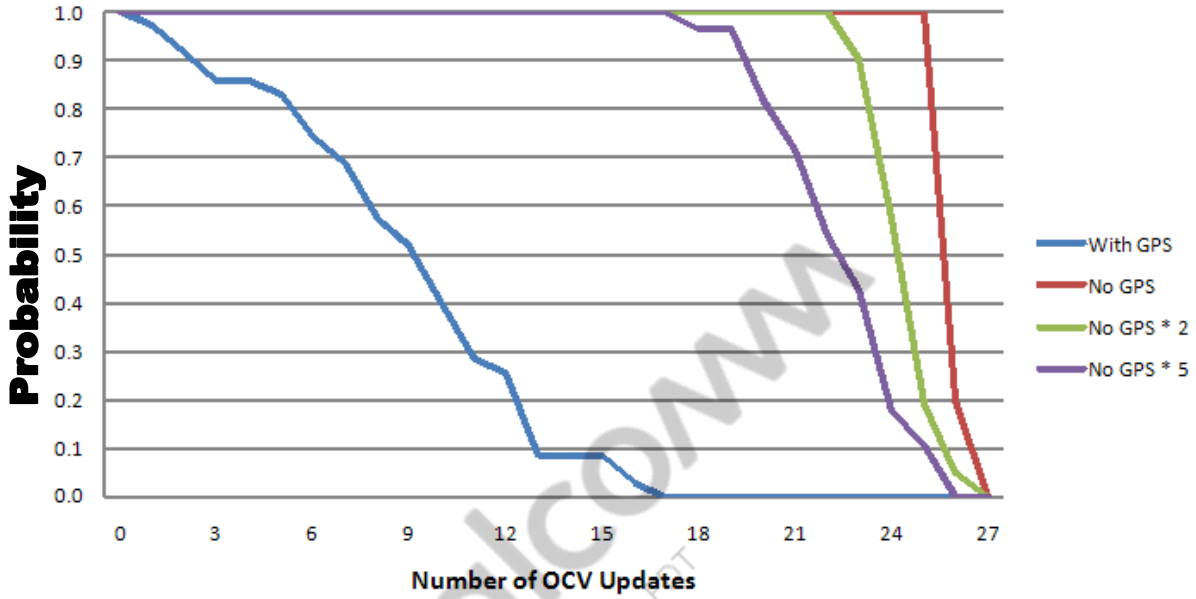


Figure 2-7 Cumulative probability of good OCV updates/hour

Figure 2-7 shows that when the maximum number of randomly positioned current bursts are generated from subsystems operating in standby mode, a large number of good OCV measurements can be obtained in non-GPS mode. When GPS is operating, the probability is reduced, but there is still a 90% probability that three good OCV updates each hour will occur.

2.2.3 Coulomb counter details

The Coulomb counter accumulates readings from the CCADC. It is a 2's compliment counter, which is centered at 0x0_0000_0000. The Coulomb counter compensates for the polarity of the voltage in the sense resistor. It counts up when a charge is removed from the battery (BMS_CSP is negative relative to BMS_CSN pin) and counts down when the battery is being charged (BMS_CSP is positive relative to BMS_CSN pin). BMS_CSN must always be connected to the positive side of battery, and BMS_CSP must be connected to the system load (Vph_pwr if an internal BATFET is used for sensing element or PMIC Vbat pin if an external sensing resistor is used). These pins must never be reversed.

The Coulomb counter value is 36 bits, stored in 5 bytes. In the PM8941 2.0 module, the Coulomb counter resolution is 5.42535 µV. The Coulomb counter accumulates the last stored IADC measurement using an internal clock set to 56 counts of the 32 kHz clock. This results in an accumulation rate of either 1.7092 ms if a 19.2 MHz source is used to generate 32 KHz or 1.7090 ms if a 32 KHz crystal is used. If there is no new CCADC measurement, the Coulomb counter continues to use the last stored measurement for accumulation. Since the BATFET is the primary sensing element, when the BATFET is open, a short is placed across the IADC, so that readings yield a measurement of 0 µV. The BMS has an option to accumulate with or without removing the CCADC offset. The system defaults to removing the offset, which means the value of the IADC offset trim register is subtracted from the measurement value prior to being accumulated. As an example, assume a fixed 1 A is going through the BATFET with an Rds(on)

of 10 mΩ. This creates an IADC measurement value of 10 mV. The Coulomb counter increments by 1843 decimal bits (= 10 mV/5.42535 µV) each 1.7092 ms.

Before the Coulomb counter value can be used, it must compensate for gain, as described in Section 3.1.1. To use the Coulomb counter value in system operation, it is necessary to convert the reading into a usable value. Use the following equation to convert the Coulomb counter value and the sense resistor value to mAh for PM8941 2.0 modules:

$$C\frac{C}{R_{sense}}\,(\text{mAh}) = CC(\text{decimal}) \times \frac{5.42535\ \mu\text{V} \times 0.0017092\ \text{s}}{3600\ \text{s/hr} \times \text{Rsense\_m}\Omega}$$

The Coulomb counter is designed with a programmable threshold. If the threshold is crossed in either direction, the bms_cc_thr_int is asserted. There are two operation modes where the Coulomb counter threshold can be used. The historical mode uses it as a notification that the Coulomb count is reaching the end of the range or is beyond the battery capacity. At this point, the software should read the value, store the value, and then reset the Coulomb counter (see Section 3.2.2.4). The counter saturates if it reaches the limit in either direction (0x7_ffff_ffff or 0x8_0000_0000). Table 2-3 provides the maximum battery size that can theoretically be supported with the Coulomb counter without saturation.

Table 2-3 Coulomb counter maximum battery size vs. sense resistor

PM8941 2.0 module		
Resistor (Ω)	**CC LSB (µV)**	**Maximum battery size (Ah)**
0.010	5.42535	8.73
0.015	5.42535	5.82
0.020	5.42535	4.37
0.025	5.42535	3.49

The Coulomb counter has threshold settings to generate an interrupt when the threshold is crossed. The default value for the threshold setting is full scale (0x7_ffff_ffff), so for the threshold to have meaning, it should be set to a realistic value. For the PM8941 2.0 module, the Coulomb counter size is large enough to avoid saturation, and the threshold should never be exceeded. The value should be set 20% above the battery size to capture an error condition. The set value can be found by selecting the maximum battery size for a given sense resistor value and scaling the hex value of 0x7fff_ffff for the desired value.

$$CC_{threshold}(hex) = \left(0x7_{FFFF_{FFFF}}\right) \times \frac{(1.2 \times battery\_size)}{(max\_battery\_size)}$$

The second mode of using the Coulomb counter threshold is to create interrupts at specific points. Typically, this would be used in an interrupt driven SoC operational mode. In this case, the threshold would be set to the next desired SoC threshold. Section 3.4.6 contains the interrupt driven SoC configuration details.

2.2.4 Recommended settings

The BMS controller is programmable to meet a variety of system needs. Table 2-4 shows the recommended settings for the BMS state machine timing control using a system and battery with the following characteristics:

- Average current during standby = 5 mA
- R_{sense} = 10–25 mΩ
- Battery voltage low-pass time constant = 2 minutes

Table 2-4 Recommended values for programmable settings

Threshold	Register name	Register value	LSB	Default value	Recommended value				Comments
					10 mΩ	15 mΩ	20 mΩ	25 mΩ	
OCV tolerance	TOL_CTL	0x44 bits<7:4>	400 µV	0x2 (800 µV)	Default	Default	Default	Default	Use a default value for the PM8941 2.0 module.
I_{bat} tolerance	TOL_CTL	0x44 bits<3:0>	22 µV	0x3 (66 µV)	Default	Default	Default	Default	This allows a near-zero load current, measured as slightly negative due to noise, to still be identified as a low positive current.
S1 sample delay	S1_DELAY_CTL	0x5A bits<3:0>	–	0x0B (1s)	Default	Default	Default	Default	Reduced power consumption during standby
S2 sample delay	S2_DELAY_CTL	0x5B bits<3:0>	–	0x00 (0 ms)	Default	Default	Default	Default	Constant monitoring for increased accuracy during high load
S6 sample delay	S6_DELAY_CTL	0x5C bits<3:0>	–	0x00 (0 ms)	Default	Default	Default	Default	Override sample delay set to match S2 delay.
S7 sample delay	S7_DELAY_CTL	0x5D bits<3:0>	–	0x07 (64.5 ms)	Default	Default	Default	Default	PON OCV delay to allow MBG settling.
S1 number samples	S1_SAMP_AVG_CTL	0x60 bits<2:0>	–	0x02 (16)	Default	Default	Default	Default	–
S2 number samples	S2_SAMP_AVG_CTL	0x61 bits<2:0>	–	0x01 (8)	Default	Default	Default	Default	–
S3 number samples	S3_SAMP_AVG_CTL	0x62 bits<2:0>	–	0x04 (64)	Default	Default	Default	Default	–
S4 number samples	S4_SAMP_AVG_CTL	0x63 bits<2:0>	–	0x04 (64)	Default	Default	Default	Default	–
S6 number samples	S6_SAMP_AVG_CTL	0x64 bits<3:0>	–	0x00 (1)	Default	Default	Default	Default	–
S7 number samples	S7_SAMP_AVG_CTL	0x65 bits<2:0>	–	0x04 (64)	Default	Default	Default	Default	–
S1 low counter	S1_CNT_CTL	0x6A bits<2:0>	–	0x6 (128)	Default	Default	Default	Default	128 × 1 sec = 2.2 minutes before OCV measurement is attempted (see note 3)
S1 high counter	S1_CNT_CTL	0x6A bits<6:4>	–	0x3 (8)	Default	Default	Default	Default	8 × 1 sec = 8 second maximum delay before transition to S2
S2 low counter	S2_CNT_CTL1	0x6C bits<2:0>	–	0x5 (32)	Default	Default	Default	Default	–
S2 high counter	S2_CNT_CTL1	0x6C bits<6:4>	–	0x6 (128)	Default	Default	Default	Default	–

Threshold	Register name	Register value	LSB	Default value	Recommended value				Comments
					10 mΩ	15 mΩ	20 mΩ	25 mΩ	
S2 meta counter	S2_CNT_CTL2	0x6D bits<2:0>	–	0x7 (256)	Default	Default	Default	Default	–
S3low counter	–	–	–	1	Default	Default	Default	Default	Not configurable
S3 high counter	–	–	–	1	Default	Default	Default	Default	Not configurable
S3 meta counter	S3_CNT_CTL	0x6F bits<2:0>		0x2 (4)	Default	Default	Default	Default	Number of OCV retries before giving up
S1 V_{sense}	S1_VSENSE_THR_CTL	0x73	11 µV	0x2E (506 µV)	Default	0x45	0x5C	0x73	50 mA settings for PM8941 2.0 module (default set for 10 mΩ)
S2 V_{sense}	S2_VSENSE_THR_CTL	0x75	11 µV	0x29 (451 µV)	Default	0x39	0x52	0x66	45 mA settings for PM8941 2.0 module (default set for 10 mΩ)
S3 V_{sense}	S3_VSENSE_THR_CTL	0x77	11 µV	0x08 (88 µV)	Default	0x0C	0x10	0x14	8 mA settings for PM8941 2.0 module (default set for 10 mΩ) (see note 1)
R_{sense}	IADC1_xxx_TRIM2_ADC_ FULLSCALE1	0xF4	0.01562 5 mΩ	10 mΩ	–	–	–	–	The BATFET Rds(on) value is stored in registers 0xF4 for each peripheral. The value stored is an offset from 10 mΩ, where the first bit is the sign, and the remaining bits have an LSB of 0.015625 mΩ. If using an external R_{sense}, the resistance value must be stored in the board file. The usable range is 5 mΩ to 25 mΩ.
V_{cutoff}	–	–	–	3.4 V	Customer value	Customer value	Customer value	Customer value	The loaded voltage at a point where the battery is considered empty. The default value is 3.4 V. This value is stored in the board file.
Rconn	–	–	–	–	–	–	–	–	This is the additional PCB resistance between the battery terminals and the VADC monitoring points. This value includes resistance in both leads of the battery. This value is stored in the board file.
FCC	–	–	–	1500 mAh	Customer value	Customer value	Customer value	Customer value	The full charge capacity of the battery. This depends on specific battery the customer selects.

Threshold	Register name	Register value	LSB	Default value	Recommended value				Comments
					10 mΩ	15 mΩ	20 mΩ	25 mΩ	
Lookup()	–	–	–	–	Customer value	Customer value	Customer value	Customer value	A function that returns an approximate percentage charge when provided a voltage. This depends on the specific battery the customer selects. Section 3.1.5 provides lookup table details, and Figure 2-8 shows a graphic representation.
SoC match %	–	–	–	100%	Default	Default	Default	Default	The SoC threshold for guiding use of poweron OCV measurement. If the shutdown SoC is with the matching percentage of the new SoC, then the old SoC is used at boot. This value is stored in the board file.
Trickle charge offset	–	–	–	0.033 V	Default	Default	Default	Default	The voltage to subtract from the initial OCV battery measurement if trickle charging has been on (= $I_{trkl} \times R_{bat2}$, where I_{trkl} = 325 mA and R_{bat2} = 100 mΩ)

Notes:

1. In **S3**, the OCV measurement is repeated until a measurement with V_{sense} below this threshold, or **S3** meta counter is triggered. Decreasing this threshold increases the accuracy of the OCV measurements, but decreases the frequency of the valid OCV updates. If the threshold is set too far below the background, standby current and OCV measurements do not occur.
2. The **S1** low counter should be set so that the product of the counter times the sample delay is approximately the battery voltage time constant. The OCV and Coulomb counter are not updated until the process OCV algorithm verifies the OCV measurement. (See Section 2.2.8.)
3. The **S2** meta counter should be set so that the product of **S2** meta counter, **S2** high counter, and the total sample time (which is sample delay + sample duration × **S2** number samples) is at least twice the battery voltage time constant.

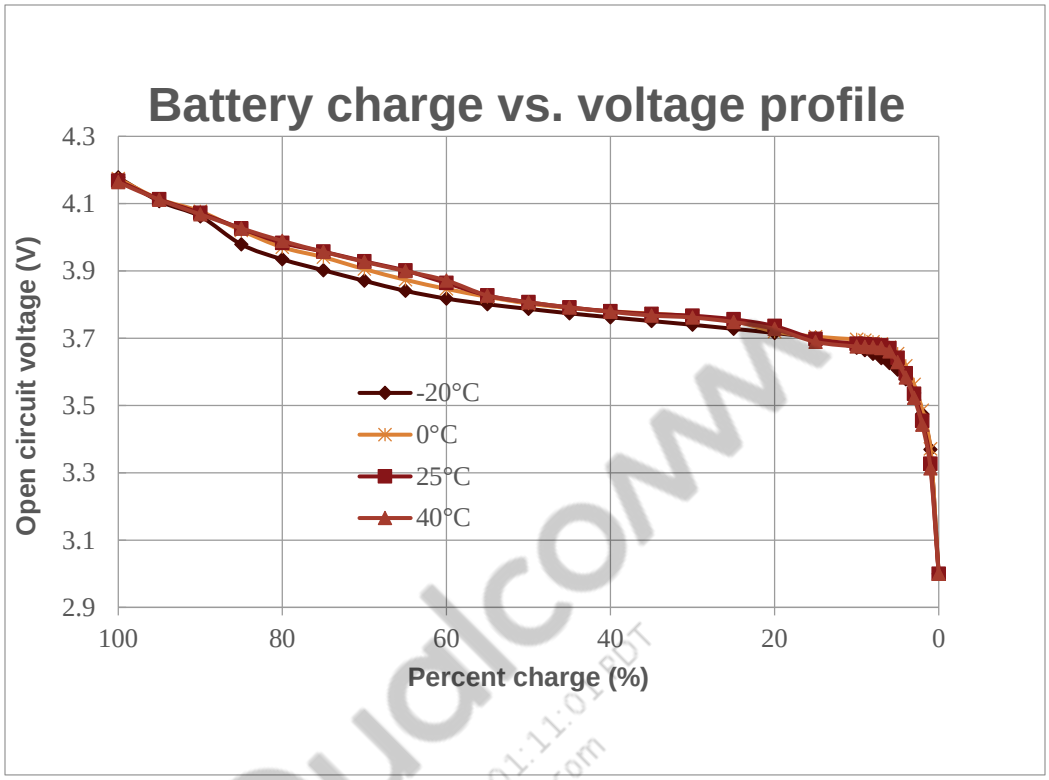


Figure 2-8 Percentage charge graph example

2.2.5 BMS function descriptions

The BMS hardware functions are described in this section.

2.2.5.1 BMS interrupts

Table 2-5 lists and defines the specific BMS interrupts. PM8941 has a standardized interrupt configuration.

Table 2-5 BMS interrupts

Register bit	Interrupt name	Interrupt definition and use
7	VSENSE_FOR_R_INT	V_{sense} for the resistance measurement completed interrupt
6	VSENSE_AVG_INT	V_{sense} sample averaging measurement completed interrupt
5	SW_CC_THR_INT	▪ Software's shadow Coulomb accumulator greater than the threshold interrupt ▪ Signal HIGH when the sample average > threshold, LOW otherwise
4	OCV_THR_INT	This interrupts software when OCV has gone below the threshold setting.
3	CHARGE_BEGIN_INT	This interrupts software when the charging begins or ends.
2	GOOD_OCV_INT	▪ Last good open circuit voltage Vbat sample measurement completed interrupt ▪ One 32 kHz clock cycle wide HIGH signal

Register bit	Interrupt name	Interrupt definition and use
1	OCV_FOR_R_INT	▪ Open circuit voltage for resistance measurement completed interrupt ▪ One 32 kHz clock cycle wide HIGH signal
0	CC_THR_INT	▪ Coulomb accumulator greater than the threshold interrupt. ▪ Signal HIGH when sample average > threshold, LOW otherwise

2.2.5.2 BMS mode control

The BMS can operate in several modes. The autonomous mode is the normal mode of operation. The BMS state machine is fully operational in this mode, and states are transitioned based on the system performance. The override mode creates a set sampling time for the BMS current readings and for the simultaneous battery voltage and current measurements if the EN_VBAT (register 0x40 bit<0>) bit is set. In the override mode, the BMS does not transition through the various states of operation. The default timing for samples is set to the S1 values. The sampling continues at this specified rate until the override mode is exited, or the sampling rate is changed for S1. PM8941 has the ability to configure whether the voltage collapse protection (VCP) override is enabled by setting the EN_VCP_OVERRIDE bit (register 0x40 bit<0>). Setting this bit causes the BMS state machine to move into S2 when a VCP event occurs. Table 2-6 defines the BMS mode control bits.

Table 2-6 BMS1_MODE_CTL (register 0x40)

Register bit	Interrupt name	Interrupt definition and use
7	BMS_OVERRIDE_MODE_EN	Selects the BMS controller operating mode: ▪ 0 – Autonomous ▪ 1 – Override
6	EN_VCP_OVERRIDE	Enables the transition to override mode when the vcp_bms_override input signal is asserted
5:1	RESERVED_BITS5_1	–
0	EN_VBAT	Enables the Vbat measurements in override mode

2.2.5.3 BMS Coulomb counter control

The BMS has two Coulomb counters available for system use. The main Coulomb counter is intended to be the primary source for BMS data when calculating SoC. This Coulomb counter is reset when an OCV update occurs, unless the CC_MANUAL_RESET bit is set (register 0x42 bit<1>). The software shadow Coulomb counter can be used for specific functions that need the Coulomb counter to operate under complete software control. Both Coulomb counters can be manually reset. The registers in Table 2-7 and Table 2-8 describe how to control the Coulomb counters. Note that the HOLD_OREG_DATA bit disables the processed data register updates for all BMS registers to permit reading multiple register values without them changing during the reads.

Table 2-7 BMS1_CC_DATA_CTL (register 0x42)

Register bit	Interrupt name	Interrupt definition and use
7:2	RESERVED_BITS7_2	–
1	CC_MANUAL_RESET	Forces a manual reset of the Coulomb counter: ▪ Reset automatically by the OCV ▪ Reset only by a register write
0	HOLD_OREG_DATA	▪ 0 – Enables the processed data register updates ▪ 1 – Disables the processed data register updates

Table 2-8 BMS1_CC_CLEAR_CTL (register 0x43)

Register bit	Interrupt name	Interrupt definition and use
7:2	CLEAR_CC	▪ 0 – Allows the Coulomb counter operation ▪ 1 – Clears the Coulomb counter
1	CLEAR_SW_CC	▪ 0 – Allows the Coulomb counter operation ▪ 1 – Clears the Coulomb counter
0	RESERVED_BITS5_0	–

As described in Section 2.2.3, both Coulomb counters have thresholds that can be configured. The primary Coulomb counter threshold registers are 0x7A (LSB) to 0x7E (MSB). The software shadow Coulomb counter threshold registers are 0xA0 (LSB) to 0xA4 (MSB).

2.2.5.4 BMS enable

The BMS is enabled by setting the BMS_EN bit. Table 2-9 provides the contents of the BMS_EN_CTL1 register.

Table 2-9 BMS1_EN_CTL1 (register 0x46)

Register bit	Interrupt name	Interrupt definition and use
7	BMS_EN	Enables the BMS module. When HIGH, the BMS is enabled for autonomous or override operations.
6:0	RESERVED_BITS6_0	–

2.2.5.5 BMS OCV limit operation

There are several new OCV features available in PM8941. This feature enables selecting the range of use for OCV updates. In flat portions of the battery profile curve, there can be as little as 1 mV change for each % of SoC change. When battery profiles are this flat, the Coulomb counter can be used through the flat portion of the curve with better accuracy than an OCV update. The OCV settings for the high side and low side are each 2 bytes. No OCV updates are used between the high and low settings when this feature is enabled. Note that the OCV values used in the limit settings are raw values from the ADC. The values used must be reverse calibrated to factor in the offset and gain error from the raw measurement. It is recommended that the high and low limits are set to where the OCV points battery profile are showing less than 3 mV change per each SoC %. This range changes for each unique battery. Figure 2-8 shows that the typical range of flatness between the 45% and 25% SoC points, which maps to the 3.792–3.756 V OCV values for this battery. Some batteries may never fall under the 3 mV/SoC % level.

Table 2-10 BMS1_OCV_USE_LOW_LIMIT_THR0 (register 0x48)

Register bit	Interrupt name	Interrupt definition and use
7:0	OCV_USE_LOW_LIMIT_THR_7_0	▪ Use this LSB of the lower voltage range when the OCV limit is enabled. ▪ If enabled, the OCV use is limited to outside the lower and upper limits.

Table 2-11 BMS1_OCV_USE_LOW_LIMIT_THR1 (register 0x49)

Register bit	Interrupt name	Interrupt definition and use
7:0	OCV_USE_LOW_LIMIT_THR_15_8	▪ Use this MSB of the lower voltage range when the OCV limit is enabled. ▪ If enabled, the OCV use is limited to outside the lower and upper limits.

Table 2-12 BMS1_OCV_USE_HIGH_LIMIT_THR0 (register 0x4A)

Register bit	Interrupt name	Interrupt definition and use
7:0	OCV_USE_HIGH_LIMIT_THR_7_0	▪ Use this LSB of the upper voltage range when the OCV limit is enabled. ▪ If enabled, the OCV use is limited to outside the lower and upper limits.

Table 2-13 BMS1_OCV_USE_HIGH_LIMIT_THR1 (register 0x4B)

Register bit	Interrupt name	Interrupt definition and use
7:0	OCV_USE_HIGH_LIMIT_THR_15_8	▪ Use this MSB of the upper voltage range when the OCV limit is enabled. ▪ If enabled, the OCV use is limited to outside the lower and upper limits.

Table 2-14 BMS1_OCV_USE_LIMIT_CTL (register 0x4C)

Register bit	Interrupt name	Interrupt definition and use
7	OCV_USE_LIMIT_EN	▪ 0 – Good OCV updates can occur at any battery voltage. ▪ 1 – Good OCV updates can only occur when the battery voltage is outside the specified range.
6:0	RESERVED_BITS6_0	–

When operating in the interrupt-driven SoC mode, PM8941 can set an OCV threshold to issue an interrupt at a specific SoC level that corresponds to the OCV value. When going to sleep/standby, an interrupt is generated when this OCV is reached. Note that the OCV values in the threshold setting are raw values from the ADC and must be reverse calibrated to factor in the offset and gain error in the raw measurement. Table 2-9 lists the contents of the BMS_EN_CTL1 register.

Table 2-15 BMS1_OCV_THR0 (register 0x50)

Register bit	Interrupt name	Interrupt definition and use
7:0	OCV_THR_7_0	This is the LSB of the threshold setting for the OCV_THR signal, which asserts when an OCV measurement is below the selected settable threshold.

Table 2-16 BMS1_OCV_THR1 (register 0x51)

Register bit	Interrupt name	Interrupt definition and use
7:0	OCV_THR_15_8	This is the MSB of the threshold setting for the OCV_THR signal, which asserts when an OCV measurement is below the selected settable threshold.

Table 2-17 BMS1_OCV_THR_CTL (register 0x53)

Register bit	Interrupt name	Interrupt definition and use
7:0	OCV_THR_EN	▪ This enables the OCV_THR interrupt signal, which asserts when an OCV measurement is below the selected settable threshold. ▪ This signal is intended for use when the software requires interrupts based on the SoC levels.

2.2.5.6 BMS state configuration

The BMS FSM has a large number of configuration registers available to modify BMS performance. Table 2-4 lists these registers, the default, and the recommended values.

2.2.6 BMS data descriptions

BMS data is stored in the registers described in Table 2-18. The sources of data are either the BMS FSM or BMS driver software. Formulas using the following values are included in Section 2.2.7. Table 2-18 provides a description of each BMS register value, its naming convention, and when it is updated.

Table 2-18 BMS data registers

Register value	Data description	Naming convention	Updated
0x80 (LSB) 0x81 (MSB)	OCV for resistance measurement	$V_{B\text{-}R2}$ in SoC equation	Only at the first good OCV measurement after taking V_{sense}/V_{batt} for R_{batt} ($V_{S\text{-}R}/V_{B\text{-}R1}$)
0x8A (LSB) to 0x8E (MSB)	▪ Primary Coulomb counter value ▪ This is a 36-bit value, so only bits<3:0> are used in MSB.	*CC* MSB in SoC equation	Updates every 1.6785 ms
0x90 (LSB) 0x91 (MSB)	Last good OCV	*OCV* in SoC equation	Updates at ~4.3 minutes in low load
0x94 (LSB) 0x95 (MSB)	V_{sense} PON	Not applicable – this value is only used to check current during PON OCV.	Record during PON

Register value	Data description	Naming convention	Updated
0x98 (LSB) 0x99 (MSB)	V_{sense} average	▪ *V_{sense}* in common discussion ▪ Not used in the SoC equation	Updates at every BMS measurement (1 Hz in low load, ~100 Hz in high load)
0x9E (LSB) 0x9F (MSB)	Vbatt average	▪ *Vbatt* in common discussion ▪ Not used in the SoC equation	Updates any time Vbatt is measured, i.e., S3
0xA8 (LSB) to 0xAC (MSB)	▪ Software shadow Coulomb counter value ▪ This is a 36-bit value, so only bits<3:0> are used in MSB.	Not applicable – use is up to the software	Updates every 1.6785 ms
0xB0	▪ The software stores data in this register to determine if the battery has been removed at PON. ▪ This register is dVdd_rb.	BMS_DATA_REG_0	At boot
0xB1	▪ The software stores shutdown SoC data in this register for use at PON. ▪ This register is xVdd_rb.	BMS_DATA_REG_1	SoC is written to this register at shut down.
0xB1 (LSB) 0xB2 (MSB)	▪ The software stores the shutdown average current for use at PON. ▪ These registers are xVdd_rb.	BMS_DATA_REG_2 and BMS_DATA_REG_3	The average system current is written to these registers at shutdown.

A detailed explanation of the BMS output values follows:

- Accumulated charge – [CC]
  - The primary Coulomb counter functional block accumulates the charge, either added or removed from the battery since the last OCV update and Coulomb counter reset.
  - The data comes in 5 bytes, LSB is 5.42535 µV for the PM8941 2.0 module, accumulation rate is 1.6785 ms.
- Last good OCV for SoC – [OCV]
  - When the process OCV block validates an open-circuit voltage measurement, this value is updated and the primary Coulomb counter is reset.
  - This OCV value provides a baseline state of charge that is adjusted by the above accumulated charge value.
  - Data is in 2 bytes, LSB is 292.969 µV (includes V_{bat} scaling in AMUX).
- V_{batt} for resistance measurement – [$V_{B\text{-}R1}$]
  - This measurement is recorded by the process R_{batt} block when the battery current remains high for a sufficient duration.
  - Subsequent measurements replace the stored measurement.
  - This measurement calculates the voltage drop across the battery internal resistance.
  - Data is in 2 bytes, LSB is 292.969 µV (includes V_{bat} scaling in AMUX).

- V_{sense} for resistance measurement – [V_{S-R}]
  - This measurement is recorded by the process R_{batt} block simultaneously with V_{B-R1} for resistance measurement.
  - Subsequent measurements replace the stored measurement.
  - This is a measurement of a high current that calculates the battery internal resistance.
  - Data is in 2 bytes, LSB is 5.42535 µV for the PM8941 2.0 module.
- OCV for resistance measurement – [V_{B_R2}]
  - This measurement is recorded by the process R_{batt} block, triggered by the first validated OCV to occur after the above two high-current measurements.
  - When this value is not zero, the necessary data to calculate battery internal resistance is ready and valid. This value is zero when resistance calculation data has not yet been captured or is in progress.
  - This measurement is calculates the voltage drop across the battery internal resistance.
  - Data is in 2 bytes, LSB is 292.969 µV (includes V_{bat} scaling in AMUX).

The VSENSE_FOR_RBATT (V_{S-R}) is always positive since the BMS controller removes negative values during charging. Readings of the VSENSE_AVG can be negative since current can be flowing into or out of the battery. Similarly, the full 36-bit Coulomb counter value (CC_MSB and CC_LSB) can be positive or negative. Both are handled in 2's complement format. The battery voltages measured (OCV_FOR_RBATT, VBATT_AVG, LAST_GOOD_OCV_VALUE, and VBATT_FOR_RBATT) should never be negative, even though they are handled in 2's complement format. A negative value for any battery voltage reading indicates an error condition.

2.2.6.1 Converting current readings from voltage to current

The IADC driver normally provides current values upon request. Since the BMS raw V_{sense} values may be read, it may be necessary to convert the readings to current. In the default condition, the BMS and IADC provide resolution of 5.42535 µV per bit. To convert the reading to current, complete the following steps:

- Obtain the IADC reading.
- Calibrate the IADC reading for gain, and if needed, for offset.
- Convert the IADC reading to current using the following equation:

$$Current\ (mA) = \frac{IADC(decimal) \times 5.42535\ \mu V}{Rsense\_m\Omega}$$

NOTE: The equation must be modified with the actual sense resistor value. When using Rds(on) of the BATFET for R_{sense}, the nominal R_{sense} value is stored in the trim register 0xF4 (IADC_BMS_TRIM2_ADC_FULLSCALE1). The value stored is an offset from 10 mΩ, where the first bit is the sign, and the remaining bits have an LSB of 0.015625 mΩ. As an example, if the register value is 0x95, the R_{sense} is 9.6719 mΩ (= 10 mΩ - 21 bits × 0.015625 mΩ/bit).

2.2.7 Calculation of SoC metrics

The following equations outline how to calculate battery metrics from the outputs of the BMS controller. To calculate these metrics, other battery values must be provided to the software performing these calculations. These values are either system-specific or battery-specific and should be stored in nonvolatile memory and loaded when the SoC algorithm is initialized.

- R_{sense}: The value (in Ω) of the sense resistor, which produces V_{sense}. Usable range is 5 mΩ to 25 mΩ. When using Rds(on) of the BATFET for R_{sense}, the nominal R_{sense} value is stored in the trim register 0xF4 (IADC_BMS_TRIM2_ADC_FULLSCALE1). The value stored is an offset from 10 mΩ, where the first bit is the sign, and the remaining bits have an LSB of 0.015625 mΩ. As an example, if the register value is 0x95, the R_{sense} is 9.6719 mΩ (= 10 mΩ - 21 bits × 0.015625 mΩ/bit).
- I_{avg}: The average system current over the last several minutes.
- V_{cutoff}: The loaded voltage at which the customer desires the phone to shut down. The value is set by the customer and stored in the board file. Typical values are 3.2–3.4 V.
- FCC: The full charge capacity of the battery, i.e., the total charge which can be extracted from a battery in an ideal circuit after a full charge cycle. This value can be programmed or learned through monitoring of charging cycles.
- Lookup(): A function that returns an approximate percentage charge (PC) remaining when provided a voltage. A PC vs. OCV table or piecewise linear (or higher order) approximation specific to the battery must be provided. This table is stored in the battery profile.

Battery internal resistance R_{batt} is calculated as follows:

$$R_{batt} = \frac{\boldsymbol{V_{B-R2}} - \boldsymbol{V_{B-R1}}}{\boldsymbol{V_{S-R}}} \times R_{sense}$$

Unusable charge (UUC) calculates the stored charge that is inaccessible due to the voltage drop across R_{batt} pushing V_{batt} to the system cutoff voltage, V_{cutoff}. To minimize the variation in UUC, the value for R_{batt} in the calculation is the value at the approximate point where SoC = UUC. Finding the exact value requires an iteration near the point of interest. It is calculated as follows:

$$UUC = FCC \times Lookup(R_{batt} \times I_{avg} + V_{cutoff})$$

NOTE: When comparing BMS SoC to the theoretical SoC values, UUC should be set to 0. This is primarily a test mode of operation.

If UUC does not match what is used in the theoretical SoC, there is a difference in SoC results.

The remaining charge (RC) indicates the amount of charge left in the battery. The RC value is calculated by multiplying the battery percentage charge (PC) with the battery FCC. The PC is found using the last good OCV in the PC vs. OCV lookup table.

PC = Lookup(***OCV***)

RC = PC × FCC

Calculating the remaining usable charge (RUC) allows for time-to-empty calculations to be performed. RUC is calculated by subtracting the accumulated charge and unusable charge from the remaining charge indicated by the last good OCV measurement.

$$RUC = RC - \frac{\boldsymbol{CC}}{R_{sense}} - UUC$$

Finally, the state of charge (SoC) is a percentage that reports the remaining usable capacity on a scale from 100–0 %. This metric is most useful for reports to the end user.

$$SoC = \frac{RUC}{FCC - UUC}$$

2.2.8 OCV processing algorithm

The algorithm to determine when the open-circuit battery voltage (OCV) has settled sufficiently to be valid follows the logic set forth below.

- The first time an OCV measurement is submitted to the algorithm, the value is stored in a temporary register (OCV_temp).
- The next time an OCV measurement is submitted (OCV_in), it is compared to the stored OCV.
  - If the new OCV is within a window centered on the old OCV and plus or minus a programmable error threshold,
    (OCV_in < OCV_temp + error_OCV) && (OCV_in > OCV_temp - error_OCV)
  - Then
    - The OCV is marked as settled, meaning that future measurements skip the above comparison and goes straight to the following steps.
    - The new OCV (OCV_in) is stored as the last good OCV value, which can be accessed via the SPMI interface.
    - The Coulomb counter is reset to zero.
  - Else
    - The temporary OCV is overwritten by the new OCV (OCV_temp ← OCV_in).
- Finally, whenever a transition from S1 to S2 occurs, the temporary OCV is set to zero, and the OCV is unmarked as settled.

2.2.9 HOLD_OREG_DATA field

The BMS controller can keep data from getting updated in its output registers. Gathering data requires at least 2 bytes, which must be done in sequential reads. To avoid data registers changing in the middle of a read, the BMS can stop the output registers from being updated. By setting bit <0> of the BMS1_CC_DATA_CTL register (0 × 42), the register data is no longer updated. This field should be used when data is read from the BMS registers. Before reading data, bit <0> should be set to 1. More importantly, ensure bit <0> is changed back to 0 when the read is finished. If not, old data may get stuck in the registers.

3 BMS software functions

3.1 Calibration and stored data

To accurately execute SoC calculations, the BMS must have the IADC values calibrated for offset and gain, the sense resistor value must be known to within 1%, and the battery discharge profile must be stored in memory.

3.1.1 Current value calibration

The BMS stores V_{sense} and Coulomb count values. These values are from the IADC with the offset removed, but still contain a gain error. The IADC driver executes periodic calibration for offset and gain of the IADC. By default, the BMS values have the offset removed. The offset removed is the value calibrated and stored by the IADC driver into the IADC USER peripheral. When the BMS driver is using either V_{sense} measurements or Coulomb counter measurements, they must be compensated for gain error. The BMS driver can request the gain calibration through the IADC driver. This should be done every time the BMS driver is using V_{sense} or Coulomb counter data in a calculation.

The method for gain error compensations is described below:

$$D_{OUT\_Final} = D_{OUT} \cdot \left(\frac{D_{GAIN\_Ideal}}{D_{GAIN} - Offset} \right)$$

where:

D_{OUT_Final} = Calibrated output (can be either V_{sense} or Coulomb count value)

D_{OUT} = Measurement data not yet calibrated for gain, but with offset already removed

D_{GAIN} = Gain calibration value

D_{GAIN_Ideal} = Ideal data value (17.8571 mV)

Offset = Offset calibration value (IADC driver removes this value.)

3.1.2 Voltage value calibration

The BMS stores OCV and VBAT measurements. These measurements are raw, un-calibrated values from the VADC, and are not compensated for offset or gain error. All raw data from the VADC is centered at 0x6000 (i.e., 0x6000 = 0 V), so this intrinsic offset must be removed from all measurements. The VADC LSB value is 97.656 µV. It should be noted, however, that all battery measurements are scaled by one-third when routed through the AMUX. This results in each VADC LSB representing a three-time increase in value for battery measurements, but this should not be applied until after calibration. The calibration can be done either in decimal bits or voltage, but all units must be the same. All battery measurements must be calibrated using the following steps:

1. Gather the battery measurement, remove the intrinsic offset (0x6000), and convert it to voltage using the 97.656 µV/bit scale factor. Store the measurement value while gathering the calibration data.
2. From the VADC driver API, obtain a 1.25 V calibration reading.
3. From the VADC driver API, obtain a 0.625 V calibration reading.
4. Execute the calibration using the three values collected, with the following equation:

$$V_{in} = \frac{D_{OUT} - D_{0.625}}{D_{1.25} - D_{0.625}} \cdot 0.625\ \text{V} + 0.625\ \text{V}$$

5. Scale the results to correct the voltage by multiplying the above value by 3.

3.1.3 Sense resistor value

There are two options for the sense resistor in PM8941 devices. Either the BATFET Rds(on) or an external sense resistor can be used. If the BATFET Rds(on) is used, the nominal value is written in trim register 0xF4 (IADC_BMS_TRIM2_ADC_FULLSCALE1). The value stored is an offset from 10 mΩ, where the first bit is the sign, and the remaining bits have an LSB of 0.015625 mΩ. As an example, if the register value is 0x95, the R_{sense} is 9.6719 mΩ (= 10 mΩ - 21 bits × 0.015625 mΩ/bit). If an external sense resistor is used, the value is between 5–25 mΩ and should be stored in the board file. The software must provide the ability to use either value based on which sense resistor element is selected. If a customer uses resistors of a 1% tolerance or better, the nominal value is stored. When > 1% tolerance resistors are used, or the customer desires increased accuracy, a measured value must be stored. The stored value must hold the resistor value to the hundredths of a mΩ (i.e., 10.04 mΩ or 0.01004 Ω).

The default configuration is for the BMS to use the BATFET Rds(on) value for the current sensing element. If customers want to improve overall BMS accuracy or attach loads directly to the battery without going through the BATFET, then they may opt for using an external sense resistor. To properly configure the BMS for an external sense resistor, the SBL code must write the value 0x01 to the IADC1_BMS_ADC_CH_SEL_CTL register (0x48) early in the boot process since the internal BATFET is used until this register gets set.

3.1.4 Battery charge cycles

The software must track and store the number of battery charge cycles for a given battery. Additionally, the software must have a configuration option to select how and when the charge cycle value is reset. Some customers require the value to reset when the battery is removed, other customers never want the value reset unless forced, and some customers want charge cycles automatically allocated to one of several similar batteries, based on matching the battery resistance to specific batteries. The charge cycles should be logged as discharge cycles since many customers use multiple batteries that are externally charged. Since the primary Coulomb counter resets when OCV updates occur, using the primary Coulomb counter to log discharge cycles would be difficult. The software shadow Coulomb counter is the perfect solution for this feature and should be used to determine battery discharge cycles. A charge or discharge cycle is when the SoC has completed a 100% charge in any number of partial charges or discharges. Table 3-1 is an example of how the number of charge cycles are calculated for four separate charges.

Table 3-1 Charge cycle example

Charge	Starting SoC	Ending SoC	SoC change
1	22%	74%	52%
2	35%	100%	65%
3	55%	85%	30%
4	17%	95%	78%
		Total SoC change =	225%
		Charge cycles =	2.25

3.1.5 Battery parameters

Table 3-2 contains battery parameters that should be generated by the customer for each battery used with the BMS. Some values are mandatory, and others can be learned. Each is described in more detail in subsequent sections.

Table 3-2 Battery parameter summary

Battery parameter	Description
Battery ID	Either the resistor value (translated to voltage) or the digital ID of the battery.
Percentage charge profile	The percentage charge profile maps the remaining charge, in terms of percentage, to the OCV measurements. The profile must be entered. This is critical to accurate SoC reporting.
Percentage charge profile vs. charge cycles	This provides charge cycle degradation for the percentage charge profile. This input must be entered. While it is possible to gather data during charge and discharge cycles, the data is gathered sporadically over non-ideal conditions.
Percentage charge profile vs. temperature	This provides temperature offsets for the percentage charge profile. This input must be entered. While it is possible to gather data during charge and discharge cycles, the data is gathered sporadically over non-ideal conditions.
FCC	The battery FCC should be entered to provide a good starting point for the system. This value can be learned through charge cycles.

Battery parameter	Description
FCC vs. charge cycles	This provides charge cycle degradation for the FCC. While advantageous to have the data entered, it can be learned.
FCC vs. temperature	This provides temperature offsets for the FCC. While advantageous to have the data entered, it can be learned.
Battery resistance	The battery resistance tables must be provided as a function of SoC. While the BMS can learn the battery resistance during operation, the constraints for valid measurements make it difficult to obtain.
Battery resistance vs. charge cycles	This provides charge cycle degradation for battery resistance. While it is advantageous to have the data entered, it can be learned through normal BMS operation.
Battery resistance vs. temperature	This provides temperature offsets for battery resistance. Battery resistance vs. temperature must be provided as a function of SoC. While the BMS can learn the battery resistance during operation, the constraints for valid measurements make it difficult to obtain.

3.1.5.1 Battery discharge profile

The ability to map battery voltage to SoC based on the battery's discharge profile is fundamental to determining SoC. The software must have a mechanism to load unique battery discharge profiles in nonvolatile memory, and then use this profile when calculating SoC. Customers have unique discharge profiles and may want to change the stored profile.

It is critical for the battery discharge profiles to generate after charging the battery in the same way it will be charged in actual use. The charge termination voltage, charge termination current, and minimum voltage must be the same for profiling and actual use.

The profile is used as a lookup function that returns an approximate RC percentage when provided a voltage. An RC vs. OCV table or piecewise linear (or higher order) approximation specific to the battery must be stored. Ideally, temperature is input to this table. If temperature is not part of the original data, the software should have a mechanism to learn it during the charging cycles and fill in the temperature dependency in the lookup table (LUT). Table 3-3 and Table 3-4 show examples of the discharge profile as functions of temperature and charge cycles.

Table 3-3 Battery percentage charge over temperature LUT example

	%	Temperature (ºC)							
		-30	-20	-10	0	10	25	40	60
OCV at percent charge	100	3.65	3.79	3.92	4.00	4.08	4.14	4.15	4.15
	95	3.58	3.68	3.84	3.93	4.00	4.07	4.08	4.08
	90	3.53	3.59	3.78	3.87	3.95	4.02	4.02	4.02
	85	3.48	3.53	3.70	3.82	3.89	3.96	3.97	3.98
	80	3.42	3.49	3.65	3.78	3.85	3.91	3.92	3.92
	75	3.37	3.44	3.57	3.72	3.80	3.86	3.86	3.87
	70	3.35	3.41	3.54	3.68	3.75	3.81	3.81	3.82
	65	3.33	3.39	3.50	3.64	3.71	3.76	3.76	3.76
	60	3.31	3.36	3.47	3.60	3.66	3.71	3.70	3.71
	55	3.29	3.34	3.43	3.56	3.62	3.66	3.65	3.66
	50	3.27	3.31	3.40	3.52	3.57	3.61	3.60	3.61
	45	3.25	3.28	3.37	3.48	3.52	3.56	3.55	3.55
	40	3.23	3.26	3.33	3.44	3.48	3.51	3.50	3.50
	35	3.21	3.23	3.30	3.40	3.43	3.46	3.44	3.45
	30	3.19	3.21	3.26	3.36	3.39	3.41	3.39	3.39
	25	3.17	3.18	3.23	3.32	3.34	3.36	3.34	3.34
	20	3.15	3.15	3.20	3.28	3.29	3.31	3.29	3.29
	15	3.13	3.13	3.16	3.24	3.25	3.26	3.24	3.23
	10	3.11	3.10	3.13	3.20	3.20	3.21	3.18	3.18
	9	3.10	3.09	3.11	3.19	3.19	3.19	3.17	3.16
	8	3.09	3.08	3.10	3.17	3.17	3.18	3.15	3.15
	7	3.07	3.07	3.09	3.16	3.15	3.16	3.13	3.13
	6	3.06	3.06	3.07	3.14	3.14	3.14	3.12	3.11
	5	3.05	3.05	3.06	3.13	3.12	3.13	3.10	3.09
	4	3.04	3.04	3.04	3.11	3.11	3.11	3.08	3.07
	3	3.03	3.03	3.03	3.10	3.09	3.09	3.07	3.06
	2	3.01	3.01	3.02	3.08	3.07	3.08	3.05	3.04
	1	3.00	3.00	3.00	3.07	3.06	3.06	3.03	3.02
	0	3.00	3.00	3.00	3.00	3.00	3.00	3.00	3.00

Table 3-4 Battery percentage charge over charge cycles LUT example

	%	Charge cycles – aging scale factor				
		100	200	300	400	500
OCV scale factor at percent charge	100	0.99	0.95	0.92	0.88	0.87
	95	0.99	0.95	0.92	0.88	0.87
	90	0.99	0.95	0.92	0.88	0.87
	85	0.99	0.95	0.92	0.88	0.87
	80	0.99	0.95	0.92	0.88	0.87
	75	0.99	0.95	0.92	0.88	0.87
	70	0.99	0.95	0.92	0.88	0.87
	65	0.99	0.95	0.92	0.88	0.87
	60	0.99	0.95	0.92	0.88	0.87
	55	0.99	0.95	0.92	0.88	0.87
	50	0.99	0.95	0.92	0.88	0.87
	45	0.99	0.95	0.92	0.88	0.87
	40	0.99	0.95	0.92	0.88	0.87
	35	0.99	0.95	0.92	0.88	0.87
	30	0.99	0.95	0.92	0.88	0.87
	25	0.99	0.95	0.92	0.88	0.87
	20	0.99	0.95	0.92	0.88	0.87
	15	0.99	0.95	0.92	0.88	0.87
	10	0.99	0.95	0.92	0.88	0.87
	9	0.99	0.95	0.92	0.88	0.87
	8	0.99	0.95	0.92	0.88	0.87
	7	0.99	0.95	0.92	0.88	0.87
	6	0.99	0.95	0.92	0.88	0.87
	5	0.99	0.95	0.92	0.88	0.87
	4	0.99	0.95	0.92	0.88	0.87
	3	0.99	0.95	0.92	0.88	0.87
	2	0.99	0.95	0.92	0.88	0.87
	1	0.99	0.95	0.92	0.88	0.87
	0	0.99	0.95	0.92	0.88	0.87

3.1.5.2 FCC

The FCC provides necessary scaling to use Coulomb counting as part of the SoC calculation. Customers must provide a starting value for the FCC, so the BMS can immediately provide accurate SoC values. Since the BMS must compensate the SoC solution for temperature variations, preloading FCC vs. temperature values is critical for accurate results. Examples of the FCC as functions of temperature and charge cycles are shown in Table 3-5 and Table 3-6.

It is critical for the battery FCC to be generated after the battery is charged in the same way as it will be charged in actual use. The charge termination voltage and charge termination current must be the same for determining FCC and actual use.

Table 3-5 FCC over temperature LUT example

	Temperature (ºC)							
	-30	**-20**	**-10**	**0**	**10**	**25**	**40**	**60**
FCC (mAh)	730	780	850	880	940	1000	1020	1040

Table 3-6 FCC over charge cycles LUT example

	Charge cycles – aging scale factor				
	100	**200**	**300**	**400**	**500**
FCC scale factor	0.9	0.84	0.81	0.78	0.74

3.1.5.3 Battery resistance

The internal resistance of the battery is a key parameter in determining the UUC portion of the FCC and in error correction. Similar to the FCC value, the customer must provide starting values for the battery resistance so the BMS can immediately provide accurate SoC values. Since the BMS must compensate the SoC solution for temperature variations, preloading battery resistance vs. temperature is essential in providing accurate SoC calculations to the user beforehand. The battery resistance must be measured by the BMS driver in a real-time fashion. The battery resistance measurements must be accurate enough to create proper error correction and terminate the battery at the proper voltage. Additionally, battery resistance measurements should be accurate enough to use as an individual signature for situations where two similar batteries are being exchanged by a single user. The goal is to use battery resistance to properly allocate charge cycles and FCC learned values to a specific battery.

3.1.5.4 Multiple batteries

Customers may have phones that can use multiple types of batteries. In this situation, software must be able to store all battery information for each potential battery. Reading the battery ID determines which battery is in use. If the customer's batteries do not have a battery ID, then there must be an initialization parameter that the software reads to determine which battery profile is used.

3.2 Powerup

This section describes several BMS powerup considerations.

3.2.1 Powerup configuration of BMS

At powerup, the BMS may need to have some configuration changes. This section will be updated as new requirements are discovered.

3.2.1.1 Using external R_{sense}

The default configuration is for the BMS to use the BATFET Rds(on) values for current sensing element. For 2.0 parts, the external R_{sense} must be used. Going forward, if customers want to improve overall BMS accuracy or attach loads directly to the battery without going through the BATFET, then they may opt for using an external sense resistor. To properly configure the BMS for an external sense resistor, the SBL code must write the value 0x01 to the IADC1_BMS_ADC_CH_SEL_CTL register (0x48). This must be done early in the boot process since the internal BATFET is used until this register is set.

3.2.2 SoC at poweron

During shutdown, software must calculate and store the SoC value and average current in nonvolatile memory (see Section 2.2.6 regarding registers 0xB0 to 0xB3). At poweron, the shutdown SoC is essential to indicate where the system was prior to powerdown. At poweron, software must execute several housekeeping activities to determine if the BMS poweron OCV measurement should be used, or whether the stored SoC should be used. When a battery has high levels of current drawn, or has been charged, the OCV takes time to settle, resulting in an inaccurate OCV measurement. Additionally, a different or externally charged battery may have been inserted. The software must handle the following situations.

3.2.2.1 Battery removed or inserted after boot

There are three locations for gathering battery presence and removal status. The SMBB_BAT_IF peripheral status register name SMBB_BAT_IF_BAT_PRES_STATUS (0x1208) bits 7 and 6 provide battery presence information. If BAT_PRES (bit 7) is 0, the battery was removed when the system was either active or in sleep/standby. If BAT_REMOVED_OFFMODE (bit 6) is set, the battery was removed while the system was off. Register 0xB0 is a dVdd_rb register, intended to determine if the battery has been removed during power cycling. Note that register 0xB0 can reset its content if VPH_PWR (or battery voltage) drops below 2.1 V, since it is a dVdd_rb register. The software should read this register at boot, and then write some data to this register. When the read does not match the write, the software should recognize that the battery has been removed. This register requires the battery to be removed for up to six seconds before being reset, so it should be used in conjunction with the BAT_REMOVED_OFFMODE, described above, to determine if the battery was removed during the power cycle. If the battery has been removed during the last power cycle, the stored SoC should not be used. Additionally, if the software cannot determine if the battery is different, it must determine if any learned battery data is valid anymore. Ideally, the user can be queried if the battery is new or old. If the battery is new, learned data should be abandoned. If the user cannot be queried, the software must have a customer setting to select whether to use or discard the historical data.

Battery removed

Follow these steps when the battery has been removed:

1. After PON, check the charger BAT_PRES, charger BAT_REMOVED_OFFMODE, and BMS 0xB0 register status.
2. If the battery has been removed, do not use the stored SoC information.
3. Use the software configuration setting to determine if the learned data for the battery should be kept or deleted.
4. Calculate the SoC.

Battery inserted after booting

If the battery has been inserted after booting, follow these steps:

1. After PON, check the charger BAT_PRES status.
2. If the battery is present, run as normal.
3. If battery is not present, or if any time after booting a battery is removed, set SoC = 0%, and keep it at 0% until the battery is inserted.
4. Force BATFET open.
5. Set any necessary flags that indicate the battery is not inserted.
6. Once the battery is inserted, measure the simultaneous voltage and current using a five-sample average and ensuring that the average battery current is less than 8 mA.
7. Use the average voltage measurement as PON OCV.
8. Reset the Coulomb counter.
9. Allow BATFET to be open.
10. Calculate the SoC.

3.2.2.2 Using shutdown SoC

PM8941 2.0 has three registers dedicated to storing the shutdown values for SoC and average current, and one register for determining if the battery was removed prior to boot (see Section 2.2.6). As mentioned in Section 3.2.2.1, register 0xB0 is a dVdd_rb register, used to determine if the battery has been removed during power cycling. When the phone power is cycled without the battery being removed, customers require that the same SoC value is provided by the BMS. If a battery has high levels of current drawn or has been charged just prior to power cycling, the PON OCV will be inaccurate. To obtain a true OCV, the battery takes time to settle. In these situations, if the battery has not been removed, but the power was cycled, the first reported SoC should match the shutdown SoC. Some customers require that the shutdown SoC only be used if within a certain range of the SoC generated from the poweron OCV. This range is the SoC match % value that is stored in the board file. The default value for the SoC match % is set to 100%, so any stored value of shutdown SoC is used at boot if the battery has not been removed. Customers can alter this value. The risk of changing the value is that cycling power after heavy discharge while the battery is in the very flat portion of the battery discharge curve causes up to a 40% difference between the shutdown SoC and PON SoC.

Follow these steps:

1. Determine if the battery was removed (see Section 3.2.2.1).
2. If not, collect the shutdown SoC, shutdown average current, and SoC match % (board file).
3. Calculate the PON SoC.
4. Check if |PON SoC – shutdown SoC| < SoC match %.
5. If yes, then use the shutdown SoC as starting point.
6. When using the shutdown SoC, use the shutdown current to calculate UUC, then calculate RC and use the reverse table lookup to obtain the OCV.

3.2.2.3 Using estimated OCV

At poweron, if a shutdown SoC is available and the battery has not been removed, then the shutdown SoC is used as the starting point. If the shutdown SoC is not valid, then the software must determine if three events occurred to make the PON OCV inaccurate. If the battery was trickle charged prior to poweron, if the battery was being charged during PON, or if the reset event was a warm boot, then the poweron OCV measurement will not be accurate. The software must create an OCV by taking simultaneous voltage and current readings, and then creating an OCV.

Follow these steps:

1. Use the BMS ADC peripheral to obtain the simultaneous voltage and current measurements.
2. Read the values for V_{bat} and V_{sense}.
3. Apply calibration to both values.
4. Calculate $R_{bat\_eff} = R_{bat} + R_{PCB}$, where R_{bat} is the lookup value found using SOCrbatt and R_{PCB} is the measured board resistance. SOCrbatt is the last reported SoC value modified by setting UUC = 0. R_{PCB} depends on each PCB layout and varies for every single use. The actual value must be measured and stored in the board file.
5. Convert V_{sense} to current value (I_{bat}).
6. Create $OCV_{est} = V_{bat} + I_{bat} \times R_{bat\_eff}$.
7. Reset the Coulomb counter.
8. Calculate the SoC using OCV_{est} as the starting point.

3.2.2.4 Poweron steps to calculate initial SoC

Follow these specific poweron steps:

1. Read and calibrate the PON OCV V_{bat} and V_{sense} values.
2. Read the battery ID and map it to the correct battery profile.
3. Compensate the OCV Vbat value for IR drop using V_{sense} and R_{bat} values.
4. Verify that the value is between 3.0 and 4.35 V. If not, construct and OCV by obtaining simultaneous V&I readings and the table value for battery resistance.

5. Determine if the battery was trickle charged or if a warm reset occurred prior to boot. If so, follow the instructions in Section 3.2.2.3. Calculate the PON OCV-based SoC using the following equations:
   a. $SoC = \frac{RUC}{FCC-UUC}$
   b. $RUC = RC - \frac{CC}{R_{sense}} - UUC$
   c. RC = FCC × Lookup(OCV)
   d. $UUC = FCC \times Lookup(R_{batt} \times I_{avg} + V_{cutoff})$

NOTE: FCC = fully charged capacity (a known or learned constant)

6. Read the SoC match % from the board file.
7. Determine if the battery was removed. If so, use PON OCV-based SoC. If not, use the SoC match % number to determine if the shutdown SoC or PON OCV SoC was used.

3.3 Software configuration of hardware

The BMS must be configured for use each time power is cycled. There are large numbers of counter thresholds and sample averages that can be configured to optimize performance. For most BMS register settings, the default values should be used. There are several register values that should be configured. Settings for all registers are summarized in Table 2-4 and in the following paragraphs. Many register values depend on the sense resistor value.

3.3.1 V_{sense} register values

All V_{sense} threshold register values depend on the specific value of the sense resistor. Table 2-4 provides the settings for each resistor value. For PM8941 2.0, the default register values should be used when the sensing element is the BATFET Rds(on).

3.3.2 Coulomb counter threshold

See Section 2.2.3 for detailed information. The default value for the threshold setting is full scale (0x7_ffff_ffff), so for the threshold to have meaning, it should be set to a realistic value. For the PM8941 2.0 module, the Coulomb counter size is large enough to avoid saturation, and the threshold should never be exceeded. The value should be set 20% above the battery size to capture an error condition.

3.4 Software functions

Software plays a large role in generating an accurate SoC. The following sections describe software function details.

3.4.1 Interrupt handling

The BMS has eight signals that are routed to the interrupt module. It is important to understand that the BMS was designed so that software never needs to wake-up to read interrupts. All the data can be collected on an as-needed-basis when software needs to execute an SoC calculation. With this in mind, for normal operation, all listed interrupts should be masked. Table 3-7 describes how each should be handled by software.

Table 3-7 BMS interrupts

Bit	INT name	Function	Software action required
7	VSENSE_FOR_R_INT	▪ V_{S-R} for resistance measurement completed interrupt ▪ One 32 kHz clock cycle wide HIGH signal	The software does not need to monitor this interrupt, since BMS_OCV_FOR_R can be used to determine when new values for both V_{S-R} and V_{B-R} values are available.
6	VSENSE_AVG_INT	▪ V_{sense} sample averaging measurement completed interrupt ▪ One 32 kHz clock cycle wide HIGH signal	The software does not need to monitor this interrupt. If knowing when a new V_{sense} sample has arrived is required, then the software can enable the interrupt.
5	SW_CC_THR_INT	▪ Software shadow Coulomb counter greater than or less than threshold interrupt ▪ Signal HIGH when either value > maximum threshold or value < minimum threshold, LOW otherwise	The software can use this Coulomb counter and threshold setting for any desired function. The software can set this threshold for an interrupt-based SoC calculation.
4	OCV_THR_INT	This interrupts the software when OCV has gone below the threshold setting.	The software can set this threshold for an interrupt-based SoC calculation.
3	CHARGE_BEGIN_INT	This interrupts the software when charging begins or ends.	Software can use this interrupt to know when charging has started and execute necessary housekeeping for charge cycle tracking.
2	GOOD_OCV_INT	▪ Last good open circuit voltage V_{bat} sample measurement completed interrupt ▪ One 32 kHz clock cycle wide HIGH signal	This indicates a new good OCV value is available. If awake, the software can read the value and store it; if not awake, there is no urgency to read it.
1	OCV_FOR_R_INT	▪ Open circuit voltage for resistance measurement completed interrupt ▪ One 32 kHz clock cycle wide HIGH signal.	This indicates that a new set of OCV for resistance, V_{sense} for R and V_{bat} for R values are available. If awake, the software can read the value and store it; if not awake, there is no urgency to read it.
0	CC_THR_INT	▪ Coulomb accumulator greater than or less than threshold interrupt ▪ Signal HIGH when either value > maximum threshold or value, minimum threshold, LOW otherwise	The software can use this Coulomb counter and threshold setting for any desired function. The software can set this threshold for an interrupt-based SoC calculation.

3.4.2 SoC calculation

3.4.2.1 Raw SoC value

See Section 2.2.7 for the SoC calculation description.

3.4.2.2 SoC calculation timing

New SoC calculations should be calculated on a fixed timer (currently at 20 second intervals), not when a SoC request comes from the HLOS. Customers can have extremely short durations between SoC requests. Error corrections on the SoC should only be done on a fixed timer to ensure the SoC linearity is maintained. The software must include a layer that keeps the SoC calculation timing independent of the SoC requests from the HLOS.

3.4.2.3 UUC calculation

The basic equation for UUC is $UUC = FCC \times Lookup(R_{batt\_term} \times I_{avg} + V_{cutoff})$.

When calculating UUC, the R_{batt_term} value should not be the value at the current location in the SoC, but should be the termination Rbatt, or the battery resistance at the point where the UUC value maps to the SoC curve. The easiest way to determine termination Rbat is to find the crossover value for the UUC voltage (= $R_{batt} \times I_{avg} + V_{cutoff}$) and OCV when adding Rbat values for different SoC points. Table 2-1 provides an example using Vcutoff = 3.4 V and Iavg = 1 A. Note that Iavg is the average current over three minutes. The value for Iavg is created using 16 samples of 20 second Coulomb counter difference values, which are converted to current. To avoid UUC jumps when going from charging to discharging, the 20 second values for Iavg are set to 300 mA when charging. When Iavg is going through large changes, the software must limit UUC changes to 1% every 20 seconds.

Table 3-8 Calculation of termination Rbatt example

SoC	Rbat (from table)	OCV (from table)	Vuuc (calculated with Rbat from table)	OCV-Vuuc	Comments
30	0.1539	3761	3553.9	207.1	–
25	0.16074	3727	3560.74	166.26	–
20	0.19836	3696	3598.36	97.64	–
15	0.24282	3669	3642.82	26.18	Linearly interpolate Rbat between these two values.
10	0.25479	3593	3654.79	-61.79	
9	0.27018	3567	3670.18	-103.18	–
8	0.28557	3540	3685.57	-145.57	–
7	0.30267	3514	3702.67	-188.67	–
6	0.32148	3474	3721.48	-247.48	–
5	0.34884	3437	3748.84	-311.84	–
4	0.3933	3397	3793.3	-396.3	–
3	0.47367	3347	3873.67	-526.67	–
2	0.62586	3280	4025.86	-745.86	–
1	2.60775	3183	6007.75	-2824.75	–

3.4.2.4 SoC smoothing

All SoC-related values (PC, RC, UUC, etc.) should be calculated and maintained with at least a three-decimal place resolution to permit updated values and error corrections with smooth changes. Maintaining these values in 1% resolution did not meet customer requirements for step size variation. During normal operation, the reported SoC should never have dramatic, short-term jumps in either a positive or negative direction due to OCV updates. If an OCV update causes an increase in SoC without the system being charged, then software should continue reporting the old SoC until the current SoC falls below the historical value. The reported SoC should never increase without the battery being charged. Some customers would like a configuration option as to whether SoC can increase with significant changes in phone temperature.

One hard constraint that Qualcomm Technologies, Inc. (QTI) places on SoC is that it cannot increase without a charger plugged in. With temperature variation (i.e., cold to hot), this can cause SoC to be held artificially low. If a charger is then plugged in, catching up to the actual SoC without showing jumps is required. The current method is to try and catch up within 60 seconds per % SoC. In other words, if the catch-up amount is 4%, use four minutes. Using an internal timer (CTS), the reported SoC uses the scaled_soc value when the timer is running.

Scaled_soc = FLOOR[((60 × ERROR_% - cts) × last_soc + cts × new_soc)/(60 × ERROR_%)]

last_soc = scaled_soc

Note that at the end-of-charge, forcing SoC = 100% and setting the OCV value to maximum is required.

3.4.2.5 SoC approaching charging termination

During charging, the SoC calculation relies on CC values since no OCV update occurs. It is important for the SoC to be at 100% when the charger terminates charging. To ensure the calculated SoC is in alignment with the level of charge, the software must begin reading the IADC current when the charge termination voltage is reached (CV charging). The software must then linearly interpolate between the current SoC and 100% SoC-based on the IADC current relative to the termination current. In other words, if the charge termination current is 100 mA, the IADC current is 300 mA, and the SoC is 90%, the SoC had 10% to climb while the current drops 200 mA. This indicates the SoC should increase 1% for each 20 mA current reduction while in CV charging. Figure 3-1 shows a typical charge cycle.

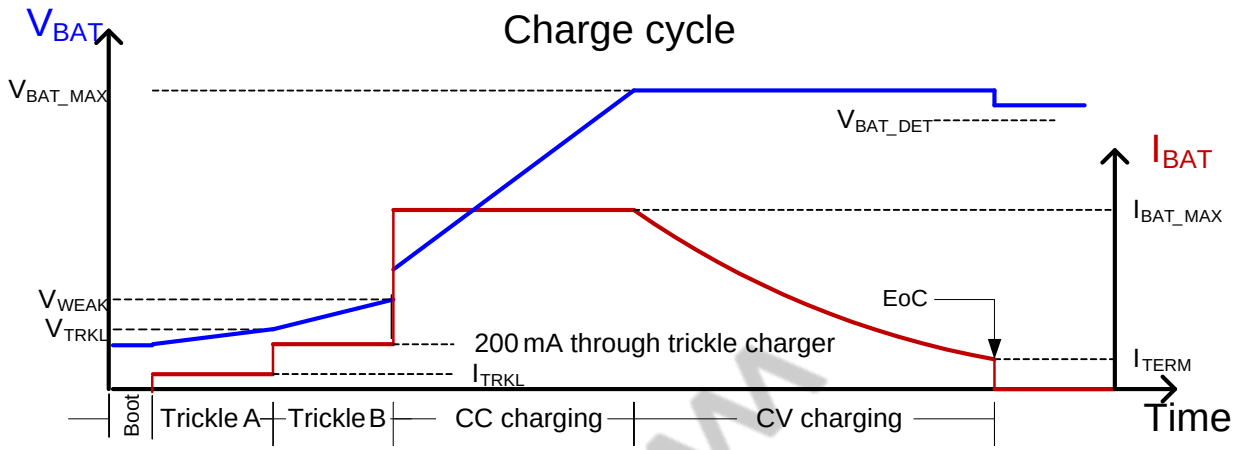


Figure 3-1 Charge cycle

3.4.3 Temperature compensation

The BMS software must read the battery temperature and compensate for battery changes. Changes to the PC vs. voltage profile, the FCC, and the internal resistance must have compensation for temperature. The software must accommodate data that is either loaded as part of the overall battery profile or is part of the learning capability. The specific data for how temperature changes battery performance is unique for each battery. The general trends, however, are consistent across battery technology types. The battery percentage charge and FCC lookup tables contain temperature data for customers concerned about generating an accurate SoC value. Customers require flexibility in the amount of temperature points used for generating data. The software must handle tables containing anywhere from one to ten columns of temperature data and interpolate the PC and FCC from each table for any usable temperature and OCV result.

The battery internal resistance also has temperature dependence. Figure 3-2 shows how battery resistance varies with temperature and SoC for a typical 1500 mA battery. This data must initially be stored in the battery profile table. During system operation, updated values are learned by the software. Updated values are normalized and stored as the 25°C and 100% point.

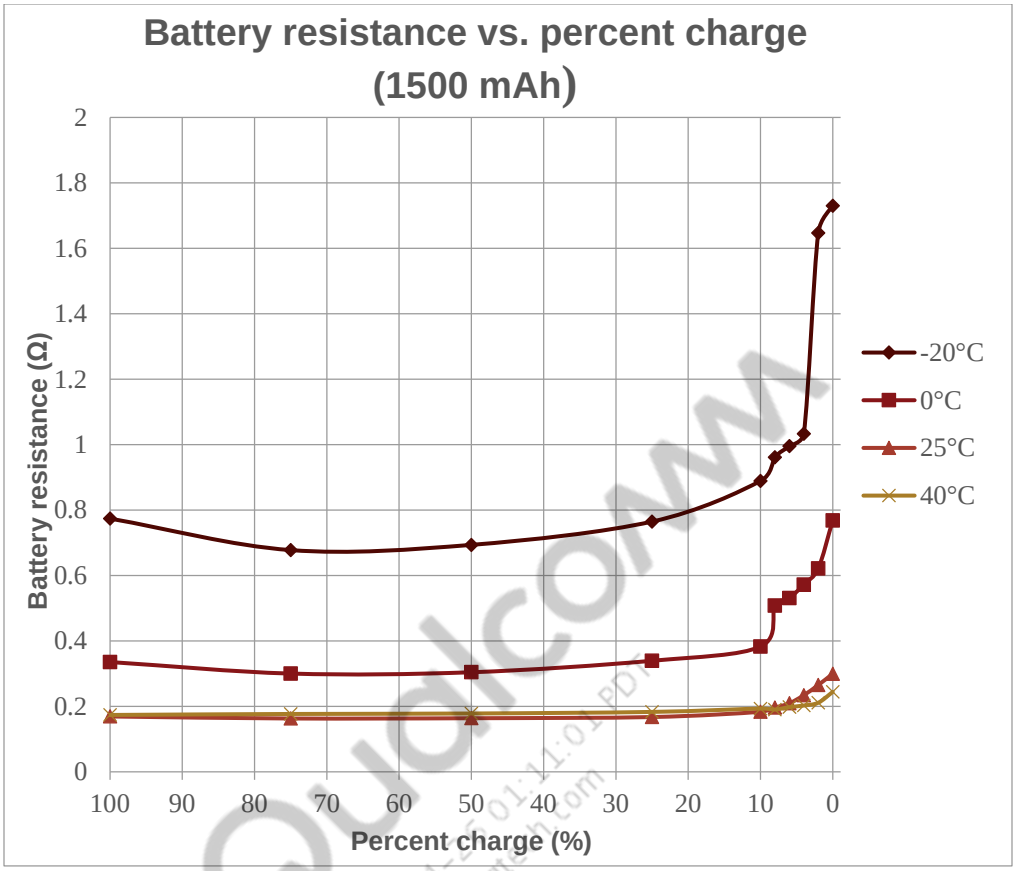


Figure 3-2 Battery resistance vs. temperature

3.4.4 Learning capability

Battery aging, battery charging cycles, and ambient temperature change the battery characteristics for SoC calculation. Ideally, the vendor provides the battery data for these changes; however, the tables may have to be filled over time. The software has to manage data entry, storage, and provide the use of lookup tables.

3.4.5 Soft reset preparation

In some situations, the software needs to issue a soft reset to the PM8941. When this occurs, most PM8941 registers are reset to the default. Prior to issuing a soft reset, the software must store the latest available values of SoC and average current. These values are lost after a soft reset is issued and are necessary to continue calculating an accurate SoC value after coming out of the soft reset.

3.4.6 Interrupt-based SoC calculation

Customers may want an interrupt-based SoC instead of having to poll for SoC. The PM8941 BMS can be configured for interrupt driven SoC operation using a combination of Coulomb count thresholds and OCV thresholds. When the system is in operational mode, the Coulomb count interrupts are used. When the system is in sleep/standby, the OCV threshold interrupt is used in conjunction with the Coulomb counter threshold interrupt. In sleep/standby mode or low load state, an OCV threshold may or may not be crossed prior a Coulomb counter threshold for a given SoC value. Both thresholds must be set.

3.4.6.1 Coulomb counter based SoC interrupt

In operational mode or high load state, the software must calculate the current SoC, determine the SoC for the desired interrupt, calculate the Coulomb counter value that would change the current SoC to the interrupt SoC, and then set the Coulomb counter threshold at that value. Once the Coulomb counter crosses the set threshold, an in interrupt occurs, signifying that the desired SoC threshold has been crossed. At this point, the interrupt should be cleared, and the next Coulomb counter threshold should be calculated and configured.

NOTE: Either the primary Coulomb counter or the software shadow Coulomb counter can be used for this purpose.

3.4.6.2 OCV-based SoC interrupt

In sleep/standby mode or low load state, an OCV threshold may or may not be crossed prior a Coulomb counter threshold for a given SoC value. Both thresholds must be set. PM8941 has the ability to disable Coulomb counter resets when an OCV occurs, so software can operate 100% on Coulomb counter values, if necessary. The only downside to this approach is that OCV values outside of the flat portion of the battery curve produces better results. This is why PM8941 provides the ability to avoid OCV updates when inside the window of the flat portion of the battery profile. Section 2.2.5.5 describes how to configure this function.

The simplest plan to implement an interrupt-based OCV is to use the software shadow Coulomb counter with the OCV thresholds. This method avoids Coulomb counter resets, independent of the settings. The software can set the shadow Coulomb counter and the OCV threshold to the desired SoC threshold and then handle the first interrupt. Once the interrupt occurs, both values must be set to the next SoC based thresholds.

3.4.7 SoC error correction algorithm

3.4.7.1 Goal

The intent of this function is to improve SoC error at the end of battery life. The phone system should never go into UVLO due to SoC error. This algorithm uses battery voltage and current to compensate for SoC error, shut down the phone at the appropriate point, and keep the system out of UVLO.

3.4.7.2 Problem

The current system design does not have a method to compensate for PON OCV error in the initial SoC estimate and accumulated Coulomb counter error. When the PON OCV is in the very flat portion of the battery curve or the battery is not well represented by the curve, the initial SoC value can be off. If the system is in a high load state until phone termination, an update OCV to cannot occur to correct the error. Additionally, batteries may have two time constants, with the second being very long. The plot shown in Figure 3-3 has a relatively short 40-second time constant and a 23-minute RC time constant.

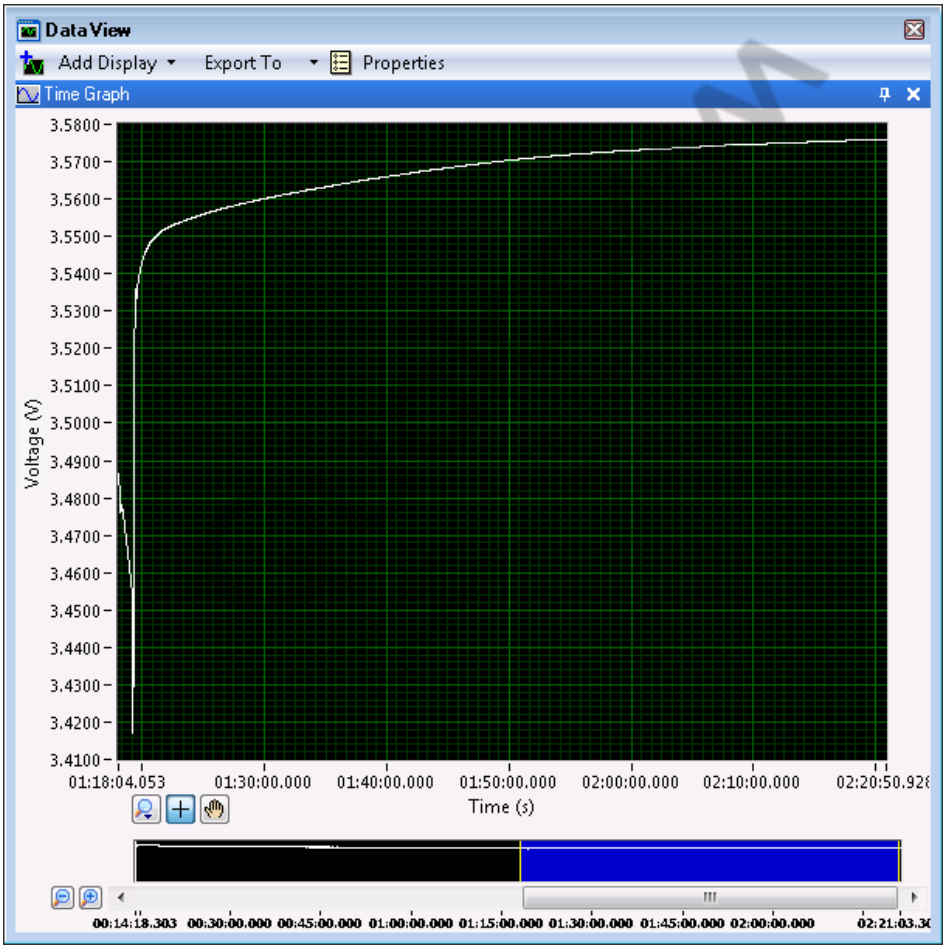


Figure 3-3 Battery relaxation vs. time

3.4.7.3 Solution

The SoC calculation must use simultaneous battery voltage and current measurements and the estimated battery resistance to create an estimated OCV level. Use an IIR filter, weighted by the SoC level, to reduce the error between the current OCV and estimated OCV. In this situation, Rbat is R1+ R2 from the battery table, plus an incremental 60 mΩ that is tapered in at the end to compensate for the R3 resistance that is not characterized in the current battery profiles.

3.4.7.4 Steps

Follow these steps:

1. After a boot, keep track of the time.
2. At the 20 second timer, read the change in Coulomb count (ΔCC_mAh) and change in time (Δtime_s), if not exactly 20 s, since the last SoC calculation.
3. Create an average current (Ibat_avg) using a moving average of last 16 samples, where a sample (I) is calculated as ΔCC_mAh × 3600 s/h/(Δtime_s), each time SoC is calculated. The first time, however, when Δtime_s = 0, use the instantaneous current. When charging, use 300 mA as the sample current to keep UUC at a nominal value.

$$\text{Ibat\_avg} = \frac{\sum_0^{n-1} I}{n}$$

   where n is a maximum of 16.
4. Calculate the SoC using UUC shown below:

$$\text{UUC} = FCC \times Lookup(R_{batt\_term} \times I_{bat\_avg} + V_{Cut\_off})$$

Where FCC is a lookup value, V_{cut_off} is the customer setting for loaded shutdown voltage (typically 3.2–3.4 V). R_{batt_term} should not be the value at the current location in the SoC, but should be the termination R_{batt}, or the battery resistance at the point where the UUC value maps to the SoC curve. The easiest way to determine the termination R_{batt} is to find the crossover value for the UUC voltage (= $R_{batt} \times I_{avg} + V_{cutoff}$) and OCV when using values R_{batt} for different SoC points. An example is provided in Section 3.4.2.3. Note that Iavg is the average current over three minutes. The value for Iavg is created using 16 samples of 20 second Coulomb counter difference values, which are converted to current. To avoid UUC jumps when going from charging to discharging, the 20 sec values for Iavg are set to 300 mA when charging. When Iavg is going through large changes, the software must limit the UUC changes to 1% every 20 seconds.).

5. Use the BMS ADC peripheral to obtain simultaneous V and I measurements
6. Read the values for V_{bat} and V_{sense}
7. Apply calibration to both values.
8. Calculate $R_{bat\_eff} = R_{bat} + R_{PCB}$, where R_{bat} is the lookup value found using SOCrbatt and R_{PCB} is the measured board resistance. SOCrbatt is the last reported SoC value modified by setting UUC = 0. R_{PCB} depends on each PCB layout and varies for every single use. The actual value must be measured and stored in the board file.
9. Convert V_{sense} to the current value (I_{bat}).
10. Create $OCV_{est} = V_{bat} + I_{bat} \times R_{bat\_eff}$.
11. Map OCV_{est} to SoC_{est}.
12. Set N = MIN(200, SoC + SoC_{est} + Last_SoC_{est})).
13. ΔOCV = M × (SoC_{est} - SoC)/(N × 1000), where M = slope of the battery profile curve near the OCV point.

NOTE: 1000 scales OCV to mV if SoC is in % (i.e., 1% change in SoC should equal 1 mV change in OCV).

14. Limit ΔOCV to a maximum value, where ΔOCV_max_uV = Ibat_avg_mA × 1 mΩ. As an example, if Ibat_avg_mA = 500 mA, then ΔOCV_max_uV = 500 μV.
15. If both SoC and SoC_{est} are outside the range of 25–45%, set OCV = OCV + ΔOCV.
16. Calculate the SoC using the updated OCV value.
17. If SoC = 0 and SoC_{est} > 0, then SoC = 1 (avoids premature shutdown).

When the next SoC is requested, repeat the previous steps, but use the updated OCV as the starting point, until the hardware generates a good OCV update.

3.4.8 Charge termination voltage

While charge termination is a charger software function, the BMS relies on it for accuracy. It is mentioned in this document as a reminder.

3.4.8.1 Problem

The BMS depends upon the battery being charged to 4.20 V as the 100% point. Historically, the charger's internal battery voltage measurement determined the charge voltage setting. The internal measurement has accuracy and IR drop limitations.

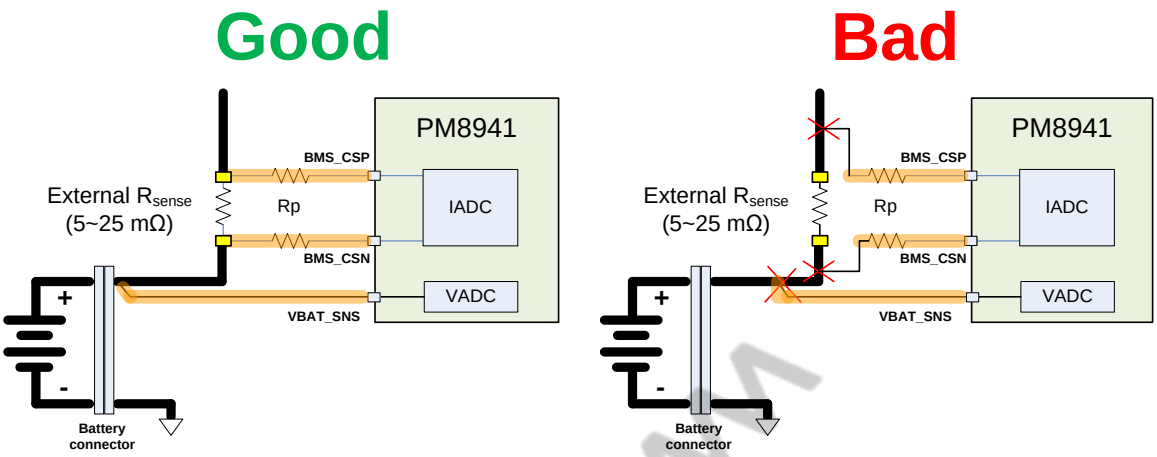
3.4.8.2 Solution

To improve the accuracy of the battery charge voltage, the charger uses the VADC battery voltage measured using the VBAT_SNS input. This is the remote sensed voltage of the battery, which removes the IR drop between the PMIC and battery. Additionally, it uses the same voltage reference as the BMS, providing common values.

4 BMS hardware considerations

4.1 PCB layout

There are several important layout considerations for the PM8941 BMS. If using an external current sensing element, the resistor should be placed as close as possible to the PMIC. This minimizes the trace length of the sensing leads. Figure 4-1 shows the layout recommendation for an external R_{sense} component. Rp is the parasitic resistance associated with routing and packaging. This resistance shall be kept less than 1 Ω. The routing and packaging associated with BMS_CSP and BMS_CSN shall be kept the same and close together to minimize thermal differences. Additionally, the BMS_CSP and BMS_CSN pins must connect directly to the sense resistor pad, to avoid any additional trace resistance being added to the sense resistor. Using a four-terminal resistor helps ensure that the sense resistor pick-off points to the CCADC are done properly. Any additional current carrying trace resistance length added between the sense resistor and BSM_CSP and BMS_CSN pick-off points directly adds to the sense resistor value and results in an uncompensated error source. The VBAT_SNS lead should be routed as close to the battery connector as possible. Any additional trace resistance between the sensing point and the battery creates an error.



BMS trace routing rules:

- BMS_CSN must connect to the positive terminal of the battery. Do not change the polarity.
- Route BMS_CSP and N traces directly to the R_{sense} pad without sharing any power traces.
- Route BMS_CSP and N traces close together, with common length and common resistance.
- Route VBAT_SNS trace as close to the battery connector as possible.
- Keep the trace resistance (Rp) less than 1 Ω.
- Maintain similar thermal coupling in BMS_CSP and N traces.
- Keep all traces away from high-power traces and devices.
- Place the sense resistor as close to the PMIC as possible without adding unnecessary trace resistances to battery leads.

Figure 4-1 BMS layout recommendation

4.2 Sense resistor

If not using the internal BATFET as a current-sensing element, the sense resistor is critical to BMS performance and is one of the main error sources in SoC accuracy. This section defines the important component selection parameters.

Table 4-1 Sense resistor parameters

Parameter	Minimum value	Maximum value	Considerations
Value	5 mΩ	25 mΩ	▪ A larger value is better for measurement accuracy (generates a larger voltage). Table 4-2 shows the relative accuracy for different R_{sense} values. ▪ A larger value dissipates more power. Table 4-2 provides the amount of energy consumed for different R_{sense} values.
Tolerance	–	1%	▪ An upper limit of 1% is factored into the system model. ▪ The lower limit is bounded by cost. ▪ If tolerance is greater than 1%, to meet the system performance specifications, the resistor needs to be accurately measured and the value must be stored.
Power rating	0.25 W	–	A system running at 2 A, with multiple GSM slots at 2 A, dissipates 0.225 W using a 25 mΩ value.
Temperature coefficient	–	±100 ppm/ºC	At ±100 ppm/°C, temperature variation is a significant error source.

Table 4-2 Accuracy and energy consumed vs. sense resistor value

R_{sense} (mΩ)	BMS SoC accuracy [1]	Energy consumption [2]
25	±2.3%	0.10%
20	±3.0%	0.08%
15	±3.4%	0.06%
10	±4.3%	0.04%

Notes:

1. BMS SoC accuracy is based on 12-hour MATLAB simulations of normal-to-heavy phone use.
2. Energy consumption is percent of FCC, based on a 150 mA continuous current and 1500 mAh battery.