BQ27542-G1

Technical Reference Manual

Literature Number: SLUUB65B
MAY 2015 – REVISED DECEMBER 2022

Table of Contents

List of Figures

List of Tables

About This Manual

This document is a detailed Technical Reference Manual (TRM) for using and configuring the BQ27542-G1 battery fuel gauge. This TRM is intended to complement, but not supersede, the information in the *BQ27542-G1 Single Cell Li-Ion Battery Fuel Gauge Data Sheet*.

Formatting Conventions Used in This Document

Notational Conventions

Information Type	Formatting Convention	Example
Commands	*Italics* with parentheses and no breaking spaces	*RemainingCapacity()* command
Data Flash (DF)	*Italics*, **bold**, and breaking spaces	***Design Capacity*** data
Register bits and flags	Brackets and *italics*	*[TDA]* bit
Data Flash bits	Brackets, *italics*, and **bold**	***[LED1]*** bit
Modes and states	ALL CAPITALS	UNSEALED mode

Related Documentation from Texas Instruments

To obtain a copy of updated related documentation, go to www.ti.com.

1. *BQ27542-G1 Single Cell Li-Ion Battery Fuel Gauge with Integrated Protection Data Sheet*)
2. *Theory and Implementation of Impedance Track™ Battery Fuel-Gauging Algorithm in BQ2750x Family Application Report*
3. *How to Generate Golden Image for Single-Cell Impedance Track™ Devices Application Report*
4. *BQ27742EVM Single Cell Impedance Track™ Technology Evaluation Module User's Guide*

Trademarks

Impedance Track™ are trademarks of Texas Instruments.
All trademarks are the property of their respective owners.

This page intentionally left blank.

Chapter 1
Introduction

The BQ27542-G1 fuel gauge accurately predicts the battery capacity and other operational characteristics of a single Li-based rechargeable cell. It can be interrogated by a system processor to provide cell information, such as state-of-charge (SOC), time-to-empty (TTE), and time-to-full (TTF).

Information is accessed through a series of commands, called *Standard Commands*. Further capabilities are provided by the additional *Extended Commands* set. Both sets of commands, indicated by the general format *Command()*, read and write information within the control and status registers, as well as its data flash (DF) locations. Commands are sent from system to fuel gauge using the serial communications engine, and can be executed during application development, pack manufacture, or end-equipment operation.

Cell information is stored in the non-volatile flash memory. Many of these data flash locations are accessible during application development. They cannot, generally, be accessed directly during end-equipment operation. Access to these locations is achieved by using the companion evaluation software, through individual commands, or through a sequence of data-flash-access commands. To access a desired data flash location, the correct data flash subclass and offset must be known.

The fuel gauge provides 64 bytes of user-programmable data flash memory, partitioned into two 32-byte blocks: ***Manufacturer Info Block A*** and ***Manufacturer Info Block B***. This data space is accessed through a data flash interface. For specifics on accessing the data flash, see Section 11.1, *Manufacturer Information Blocks*.

The key to the high-accuracy fuel gauging prediction is the Texas Instruments proprietary Impedance Track™ algorithm. This algorithm uses cell measurements, characteristics, and properties to create state-of-charge predictions that can achieve less than 1% error across a wide variety of operating conditions and over the lifetime of the battery.

The fuel gauge measures charge and discharge activity by monitoring the voltage across a small-value series sense resistor (5-mΩ to 20-mΩ typical) located between the CELL+ and PACK+ terminals of the battery pack. When a cell is attached to the fuel gauge and Impedance Track™ is enabled, cell impedance is calculated based on the selected load profile, open-circuit voltage (OCV) at present depth-of-discharge (DOD), and measured cell voltage under load. In addition, the maximum chemical capacity (Qmax) of the cell is updated after the fuel gauge records two qualified OCV measurements (taken when the battery pack is in a well-relaxed state) and the accumulated charge between them is large enough. An update of these parameters allows the fuel gauge to maintain accurate capacity prediction over the life of the battery despite increasing impedance and chemical capacity loss due to Li-Ion aging effects.

External temperature sensing is supported with the use of a high-accuracy negative temperature coefficient (NTC) thermistor with R25 = 10 kΩ ± 1% and B25/85 = 3435 kΩ ± 1% (for example, Semitec 103AT-2) for measurement. The fuel gauge can also be configured to use its internal temperature sensor. The fuel gauge monitors cell temperature to accurately compensate open-circuit voltage and resistance values used in remaining capacity simulations as well as provide overtemperature protection for the cell and temperature-dependent charging parameter (for example, JEITA charging profile) reporting to the host system.

To minimize power consumption, the fuel gauge has three different power modes: NORMAL, SLEEP, and FULLSLEEP. The fuel gauge passes automatically between these modes, depending upon the occurrence of specific events, though a system processor can initiate entry into some of these modes directly. More details can be found in Section 3.1, *Power Modes*.

This page intentionally left blank.

Chapter 2

Basic Measurement System

2.1 Introduction

The BQ27542-G1 gauge contains an integrating analog-to-digital converter (ADC) for current measurement, and a second order delta-sigma ADC for individual cell and temperature measurements.

2.2 Current and Coulomb Counting

The integrating delta-sigma ADC in the BQ27542-G1 measures the charge/discharge flow of the battery by measuring the voltage drop across a small-value sense resistor between the SRP and SRN pins. The 15-bit integrating ADC measures bipolar signals from –0.125 V to 0.125 V with 10-µV resolution. The gauge reports charge activity when VSR = V(SRP) – V(SRN) is positive, and discharge activity when VSR = V(SRP) – V(SRN) is negative. The BQ27542-G1 continuously monitors the measured current and integrates the digital signal over time using an internal counter. The data reported through the *Current()* can also have a deadband applied to it. This removes any noise or offset that has not been calibrated out from being reported as real current.

2.3 Voltage

The BQ27542-G1 updates the cell voltage at 1-s intervals. The BQ27542-G1 uses this information for both gauging and individual cell fault functions. The internal 14-bit ADC of the BQ27542-G1 measures the cell voltage value, which is then scaled and translated into unit mV.

2.4 Temperature

The fuel gauge measures battery temperature via the TS input to supply battery temperature status information to the fuel gauging algorithm and charger-control sections of the gauge. Alternatively, the gauge can also measure internal temperature via its on-chip temperature sensor if the ***[TEMPS]*** bit of ***Pack Configuration*** is cleared.

Regardless of which sensor is used for measurement, a system processor can request the current battery temperature by reading the *Temperature()* register (see Section 15.1, *Standard Data Commands*, for specific information).

The thermistor circuit requires the use of an external negative temperature coefficient (NTC) thermistor with R25 = 10 kΩ ± 1% and B25/85 = 3435 kΩ ± 1% (such as Semitec 103AT-2) that connects between the REG25 and TS pins.

This page intentionally left blank.

3.1 Introduction

The fuel gauge has three power modes: NORMAL, SLEEP, and FULLSLEEP. In NORMAL mode, the fuel gauge is fully powered and continually refreshes its dataset every 1 second. In SLEEP mode, the fuel gauge CPU is halted and frequency of data measurement is reduced to 20-second intervals for increased power savings when the system is in a standby state. In FULLSLEEP mode, the fuel gauge disables its high frequency oscillator (HFO) for highest operating power savings. The relationship between these modes is shown in Figure 3-1. Details are described in Section 3.1.1 through Section 3.1.3.

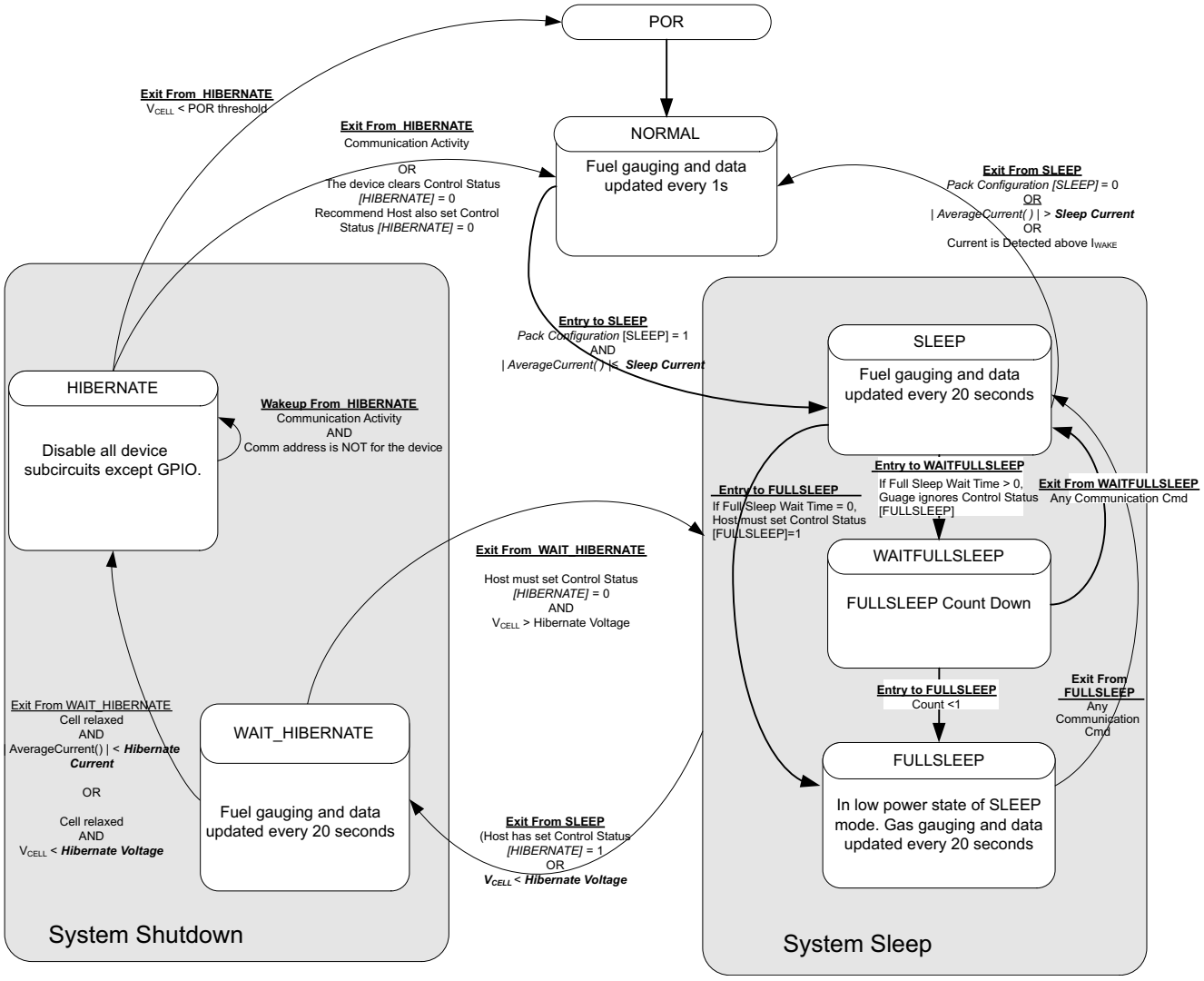


Figure 3-1. Power Mode Diagram—System Sleep

3.1.1 NORMAL Mode

NORMAL mode is the fuel gauge's standard operational mode where *Voltage()*, *AverageCurrent()*, and *Temperature()* measurements are taken and the full interface dataset is updated. Because the gauge consumes the most power in NORMAL mode, the Impedance Track algorithm minimizes the time the fuel gauge remains in this mode.

3.1.2 SLEEP Mode

When *AverageCurrent()* is less than ***Sleep Current*** or greater than (–)***Sleep Current***, the gauge enters SLEEP mode if the feature is enabled by setting the ***Pack Configuration [SLEEP]*** bit. This setting should be below any normal application currents.

The configuration options for SLEEP mode are in the following data flash.

Table 3-1. Sleep Current

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
68	Power	2	Sleep Current	I2	0	100	15	mA

3.1.3 FULLSLEEP Mode

In FULLSLEEP mode, the fuel gauge also takes data measurements and updates its data set every 20 seconds, but disables its HFO for additional power savings. As a result, the fuel gauge may hold the serial communication lines low for as long as 4 ms when transitioning out of FULLSLEEP mode to allow sufficient time for its HFO frequency to stabilize.

If ***FS Wait*** time > 0, then FULLSLEEP mode is entered automatically when the fuel gauge is in SLEEP mode and its counter increments from 0 to ***FS Wait*** time. Manual entry into FULLSLEEP mode can be commanded from the host if ***FS Wait*** time = 0 and the *SET_FULLSLEEP* command is issued, which sets the *CONTROL_STATUS [FULLSLEEP]* bit and immediately transitions the fuel gauge into this mode.

FS Wait provides the time to wait for the fuel gauge to go from SLEEP mode to FULLSLEEP mode. When the ***FS Wait*** value is 0, the gauge waits for the *SET_FULLSLEEP* subcommand, once the gauge receives this command while in SLEEP mode, it immediately goes to FULLSLEEP mode. If ***FS Wait*** is non-zero, the gauge switches to FULLSLEEP from SLEEP, once the timer expires. During the wait time, *SET_FULLSLEEP* subcommand is ignored. Note that when the gauge is in FULLSLEEP mode, any communication with the gauge triggers it to get out of FULLSLEEP mode, and the ***FS Wait*** counter is reset. If FULLSLEEP state is exited due to any other condition, the ***FS Wait*** counter is not reset. The best way to check the mode of the gauge is to monitor the drawn current out of the gauge.

The fuel gauge exits FULLSLEEP mode if any of the following conditions are detected:

- Any communication command OR
- *AverageCurrent()* > ***Sleep Current*** OR
- |***Current***| > I_{WAKE} through R_{SENSE} is detected when the I_{WAKE} comparator is enabled.

The configuration options for FULLSLEEP mode are in the following data flash.

Table 3-2. Full Sleep Wait Time

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
68	Power	13	FS Wait	U1	0	255	0	s

3.1.4 HIBERNATE Mode

HIBERNATE mode should be used for long-term pack storage or when the host system needs to enter a low-power state, and minimal gauge power consumption is required. This mode is ideal when the host is set to its own HIBERNATE, SHUTDOWN, or OFF mode. The gauge waits to enter HIBERNATE mode until it has

taken a valid OCV measurement (cell relaxed) and the magnitude of the average cell current has fallen below ***Hibernate Current***. The fuel gauge enters HIBERNATE mode if the following conditions are met:

1. *[HIBERNATE]* bit of the CONTROL_STATUS register is set OR
2. Cell voltage < ***Hibernate V***

The gauge will remain in HIBERNATE mode until any communication activity appears on the communication lines and the address is for BQ27542-G1. In addition, the SE pin SHUTDOWN mode function is supported only when the fuel gauge enters HIBERNATE due to low cell voltage.

If the *AverageCurrent()* is > ***Hibernate Current***, then the device exits from HIBERNATE mode. When the gauge wakes up from HIBERNATE mode, the [HIBERNATE] bit of the CONTROL_STATUS register is cleared. The host is required to set the bit to allow the gauge to re-enter HIBERNATE mode if desired.

Because the fuel gauge is dormant in HIBERNATE mode, the battery should not be charged or discharged in this mode, because any changes in battery charge status will not be measured. If necessary, the host equipment can draw a small current (generally infrequent and less than 1 mA, for purposes of low-level monitoring and updating); however, the corresponding charge drawn from the battery will not be logged by the gauge. Once the gauge exits to NORMAL mode, the IT algorithm will take about 3 seconds to re-establish the correct battery capacity and measurements, regardless of the total charge drawn in HIBERNATE mode. During this period of reestablishment, the gauge reports values previously calculated prior to entering HIBERNATE mode. The host can identify exit from HIBERNATE mode by checking if *Voltage()* < ***Hibernate Voltage*** or [HIBERNATE] bit is cleared by the gauge.

If a charger is attached, the host should immediately take the fuel gauge out of HIBERNATE mode before beginning to charge the battery. Charging the battery in HIBERNATE mode will result in a notable gauging error that will take several hours to correct. It is also recommended to minimize discharge current during exit from HIBERNATE.

The configuration options for HIBERNATE mode are in the following data flash.

Table 3-3. Hibernate Current/Voltage

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
68	Power	9	Hibernate I	I2	-32768	32767	8	mA
68	Power	11	Hibernate V	I2	0	32767	2550	mV

3.2 Power Control

3.2.1 Reset Functions

When the fuel gauge detects a software reset by sending *Control() [RESET]* subcommand, it resets the firmware and increments the reset counter. This counter is accessible by issuing the command *Control()* function with the *RESET_DATA* subcommand.

3.2.2 Wake-Up Comparator

The wake-up comparator indicates a change in cell current while the fuel gauge is in SLEEP modes. ***Pack Configuration*** uses bits ***[RSNS1, RSNS0]*** to set the sense resistor selection. ***Pack Configuration*** also uses the ***[IWAKE]*** bit to select one of two possible voltage threshold ranges for the given sense resistor selection. An internal interrupt is generated when the threshold is breached in either charge or discharge directions. Setting both ***[RSNS1]*** and ***[RSNS0]*** to 0 disables this feature.

Table 3-4. I_{WAKE} Threshold Settings

IWAKE	RSNS1	RSNS0	Vth(SRP-SRN)
0	0	0	Disabled
1	0	0	Disabled
0	0	1	1.0 mV or –1.0 mV
1	0	1	2.2 mV or –2.2 mV

Table 3-4. I_{WAKE} Threshold Settings (continued)

IWAKE	RSNS1	RSNS0	Vth(SRP-SRN)
0	1	0	2.2 mV or –2.2 mV
1	1	0	4.6 mV or –4.6 mV
0	1	1	4.6 mV or –4.6 mV
1	1	1	9.8 mV or –9.8 mV

3.2.3 Flash Updates

Data flash can only be updated if *Voltage()* ≥ ***Flash Update OK Voltage***. It is critical that data flash is not updated when the battery voltage is too low. Data flash programming takes much more current than normal operation of the gauge, and with a depleted battery, this current can cause the battery voltage to drop dramatically, forcing the gauge into reset before completing a data flash write. The effects of an incomplete data flash write can corrupt the memory, resulting in unpredictable and extremely undesirable results. The voltage setting in ***Flash Update OK Voltage*** prevents any writes to the data flash below this value. If a charger is detected, then ***Flash Update OK Voltage*** is ignored.

Ensure that ***Flash Update OK Voltage*** is set to a voltage where the battery has plenty of capacity to support data flash writes but below any normal battery operation conditions.

Table 3-5. Flash Update OK Voltage

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
68	Power	0	Flash Update OK Voltage	I2	0	5000	2800	mV

Chapter 4
Device Configuration Registers

4.1 Introduction

The BQ27542-G1 device has four configuration registers to use to control the device modes. These registers also contain control bits to enable or disable Impedance Track features.

4.2 Registers Subclass

4.2.1 Pack Configuration Register

Some pin configurations and algorithm settings are configured via the ***Pack Configuration*** data flash register, as indicated in Table 4-1. This register is programmed and read via the methods described in Section 16.2.1, *Accessing the Data Flash*. The register is located at subclass = 64, offset = 0.

Table 4-1. Pack Configuration Bit Definition

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
High Byte	RSVD	INTPOL	INTSEL	HOST_IE	RSOC HOLD99	IWAKE	RSNS1	RSNS0
	0	0	1	0	1	0	0	1
	0x29							
Low Byte	GNDSEL	RFACTSTEP	SLEEP	RMFCC	RSOC HOLD1	RMHOLD 100	RMHOLD0	TEMPS
	0	1	1	1	1	1	1	1
	0x7F							

High Byte

RSVD = Bit 7 is reserved. Must be 0.

INTPOL = Polarity for Interrupt Pin. (See Section 5.3, *INTERRUPT Mode.*)

INTSEL = Interrupt Pin Select: 0 = SE pin, 1 = HDQ Pin

HOST_IE = Flag Interrupt Enable:

1 = Interrupt from *Flags()* is enabled.

0 = Interrupt from *Flags()* is disabled.

RSOCHOLD99 = The fuel gauge will prevent *StateofCharge()* from reporting 100% until *Flags()[FC]* is set. Set to 1 to enable.

IWAKE, RSNS1, RSNS0 = These bits configure the current wake function. (See Section 3.2.2, *Wake-Up Comparator.*)

Low Byte

GNDSEL = The ADC ground select control. V_{SS} (pins C1 and C2) is selected as ground reference when the bit is clear. Pin A1 is selected when the bit is set.

RFACTSTEP = Enables Ra step up/down to Max/Min Res Factor before disabling Ra updates.

SLEEP = The fuel gauge can enter SLEEP, if operating conditions allow. True when set. (See Section 3.1.2, *SLEEP Mode*.)

RMFCC = RM is updated with the value from FCC, on valid charge termination. True when set. (See Section 7.5, *Full Charge Termination Detection*.)

RSOCHOLD1 = The fuel gauge will prevent *StateofCharge()* from reporting 0% until *Voltage()* is less than or equal to ***Terminate Voltage***. Set to 1 to enable.

RMHOLD100 = The fuel gauge will hold *StateofCharge()* at 100% while in an overcharge condition and not decrement until the charge surplus is equalized. Set to 1 to enable.

RMHOLD0 = The fuel gauge will hold *StateofCharge()* at 0% while in an overdischarge condition and not decrement until the charge deficit is equalized. Set to 1 to enable.

TEMPS = Selects external thermistor for *Temperature()* measurements. True when set. (See Section 2.4, *Temperature*.)

4.2.2 Pack Configuration B Register

Some pin configurations and algorithm settings are configured via the ***Pack Configuration B*** data flash register, as indicated in Table 4-2. This register is programmed and read via the methods described in Section 16.2.1, *Accessing the Data Flash*. The register is located at subclass = 64, offset = 2.

Table 4-2. Pack Configuration B Bit Definition

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ChgDoDEoC	SE_TDD	SimCtrl	SE_ISD	RSVD	LFPRelax	DoDWT	FConvEn
1	0	0	0	0	1	1	1
0x87							

ChgDoDEoC = Enables DoD at EoC recalculation during charging only. True when set. Default setting is recommended.

SE_TDD = Enables Tab Disconnect Detection. True when set. (See Section 5.6.2, *Tab Disconnect Detection*.)

SimCtrl = Dynamic Simulation of Voltage Consistency:

0 = Dynamic Simulation Step Enabled

1 = Voltage Consistency Enabled

SE_ISD = Enables Internal Short Detection. True when set.

RSVD = Bit 3 is reserved. Must be 0.

LFPRelax = Enables $LiFePO_4$ long RELAXATION mode. True when set.

DoDWT = Enables DoD weighting feature of gauging algorithm. This feature can improve accuracy during relaxation in a flat portion of the voltage profile, especially when using $LiFePO_4$ chemistry. True when set.

FConvEn = Enables fast convergence algorithm. Default setting is recommended. (See Section 6.7, *Fast Resistance Scaling*.)

4.2.3 Pack Configuration C Register

Some algorithm settings are configured via the ***Pack Configuration C*** data flash register, as indicated in Table 4-3. This register is programmed and read via the methods described in Section 16.2.1, *Accessing the Data Flash*. The register is located at subclass = 64, offset = 3.

Table 4-3. Pack Configuration C Bit Definition

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FastQmax	FConvTempEn	RIxSmEn	SmoothEn	SleepWkChg	RSVD	RSVD	BTP_EN
1	0	1	1	1	0	0	1
0xB9							

FastQmax = Fast Qmax feature is enabled.

FConvTempEn = Thermal modeling is enabled while in FAST RESISTANCE SCALING mode. Set to 1 to enable. Default of 0 is recommended.

RlxSmEn = SOC smoothing is enabled while in battery RELAXATION state. Set to 1 to enable.

SmoothEn = Enables SOC smoothing algorithm. True when set. (See Section 6.8, *StateOfCharge() Smoothing*.)

SleepWkChg = Enables compensation for the passed charge missed when waking from SLEEP mode.

RSVD = Bits 1 and 2 are reserved. Must be 0.

BTP_EN = BTP interrupts are enabled on the HDQ pin. When enabled, all other interrupts are disabled. Set to 1 to enable.

4.2.4 Pack Configuration D

Table 4-4. Pack Configuration D Bit Definition

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RSVD	SE_POL	SE_PU	SE_EN	SMRLXSYNC	PREDICT AMB	IMAXRESRVEN	IMAXEN
0	0	1	1	0	1	1	1
0x57							

RSVD = Bit 7 is reserved. Must be 0.

SE_POL = Pullup enable for SE pin. True when set (push pull).

SE_PU = Polarity bit for SE pin. SE is active high when set (makes SE high when gauge is ready for shutdown).

SE_EN = 0 = Shutdown feature is disabled.

1 = Shutdown feature is enabled.

SMRLXSYNC= SOC smoothing in relax will immediately equalize differences in true SOC vs reported *StateofCharge()* instead of gradually converging capacity (RM and FCC) differences over time.

PREDICTAMB = 0 = Disables ambient temperature adaptability for gauging.

1 = Enables ambient temperature adaptability for gauging.

IMAXRESRVEN = Enables usage of ***Reserve Capacity*** in the *Imax()* calculation

IMAXEN = Enables maximum allowed discharge reporting in *Imax()*

This page intentionally left blank.

Chapter 5

System Control Function

5.1 Introduction

The fuel gauge provides system control functions, which enable the fuel gauge to enter SHUTDOWN mode to power-off with the assistance of an external circuit, or provide interrupt function to the system. Table 5-1 shows the configurations for SE and HDQ pins.

Table 5-1. SE and HDQ Pin Functions

[INTSEL]	COMMUNICATION MODE	SE PIN FUNCTION	HDQ PIN FUNCTION
0 (default)	I^2C	INTERRUPT Mode(1)	Not Used
	HDQ		HDQ Mode(2)
1	I^2C	SHUTDOWN Mode	INTERRUPT Mode
	HDQ		HDQ Mode(2)

(1) [SE_EN] bit in ***Pack Configuration*** can be enabled to use *[SE]* and *[SHUTDWN]* bits in *CONTROL_STATUS()* function. The SE pin shutdown function is disabled.
(2) HDQ pin is used for communication and HDQ Host Interrupt Feature is available.

5.2 SHUTDOWN Mode

In SHUTDOWN mode, the SE pin is used to signal external circuit to power-off the fuel gauge. This feature is useful to shut down the fuel gauge in a deeply discharged battery to protect the battery. By default, the SHUTDOWN mode is in NORMAL state. By sending the SET_SHUTDOWN subcommand or setting the ***[SE_EN]*** bit in ***Pack Configuration*** register, the *[SHUTDWN]* bit is set and enables the SHUTDOWN feature. When this feature is enabled and *[INTSEL]* is set, the SE pin can be in NORMAL state or SHUTDOWN state. The SHUTDOWN state can be entered in HIBERNATE mode (only if HIBERNATE mode is enabled due to low cell voltage), all other power modes will default SE pin to NORMAL state. Table 5-2 shows the SE pin state in NORMAL or SHUTDOWN mode. The CLEAR_SHUTDOWN subcommand or clearing ***[SE_EN]*** bit in the ***Pack Configuration*** register can be used to disable SHUTDOWN mode.

The SE pin will be high impedance at power-on reset (POR), the *[SE_POL]* does not affect the state of SE pin at POR. Also, *[SE_PU]* configuration changes will only take effect after POR. In addition, the *[INTSEL]* only controls the behavior of the SE pin; it does not affect the function of *[SE]* and *[SHUTDWN]* bits.

Table 5-2. SE Pin State

		SHUTDOWN Mode *[INTSEL]* = *1* and (*[SE_EN] or [SHUTDOWN]* = 1)	
[SE_PU]	*[SE_POL]*	NORMAL State	SHUTDOWN State
0	0	High Impedance	0
0	1	0	High Impedance
1	0	1	0
1	1	0	1

5.3 INTERRUPT Mode

By using the INTERRUPT mode, the system can be interrupted based on detected fault conditions as specified in Table 5-5. The SE or HDQ pin can be selected as the interrupt pin by configuring the ***[INTSEL]*** bit .In addition, the pin polarity and pullup (SE pin only) can be configured according to the system needs, as described in Table 5-3 or Table 5-4.

Table 5-3. SE Pin in INTERRUPT Mode ([INTSEL] = 0)

[SE_PU]	*[INTPOL]*	INTERRUPT CLEAR	INTERRUPT SET
0	0	High Impedance	0
0	1	0	High Impedance
1	0	1	0
1	1	0	1

Table 5-4. HDQ Pin in INTERRUPT Mode ([INTSEL] = 1)

[INTPOL]	INTERRUPT CLEAR	INTERRUPT SET
0	High Impedance	0
1	0	High Impedance

Table 5-5. Interrupt Trigger Events

Interrupt Condition	Flags() Status Bit	Enable Condition	Comment
SOC1 Set	*[SOC1]*	Always	This interrupt is raised when the *[SOC1]* flag is set.
Battery High	*[BATHI]*	Always	This interrupt is raised when the *[BATHI]* flag is set.
Battery Low	*[BATLOW]*	Always	This interrupt is raised when the *[BATLOW]* flag is set.
Over-Temperature Charge	*[OTC]*	***OT Chg Time*** ≠ 0	This interrupt is raised when the *[OTC]* flag is set.
Over-Temperature Discharge	*[OTD]*	***OT Dsg Time*** ≠ 0	This interrupt is raised when the *[OTD]* flag is set.
Internal Short Detection	*[ISD]*	***[SE_ISD]*** = 1 in ***Pack Configuration B***	This interrupt is raised when the *[ISD]* flag is set.
Tab Disconnect Detection	*[TDD]*	***[SE_TDD]*** = 1 in ***Pack Configuration B***	This interrupt is raised when the *[TDD]* flag is set.
Imax	*[IMAX]*	***[IMAXEN]*** = 1 in ***Pack Configuration D***	This interrupt is raised when the *[IMAX]* flag is set.
Battery Trip Point (BTP)	*[SOC1]*	***[BTP_EN]*** = 1 in ***Pack Configuration C***. The BTP interrupt supersedes all other interrupt sources, which are unavailable when BTP is active.	This interrupt is raised when *RemainingCapacity()* ≤ *BTPSOC1Set()* or *RemainingCapacity()* ≥ *BTPSOC1Clear()* during battery discharge or charge, respectively. The interrupt remains asserted until new values are written to both the *BTPSOC1Set()* and *BTPSOC1Clear()* registers.

5.4 Low Capacity

The fuel gauge has two flags available in the *Flags()* register that warn when the SOC of the battery has fallen to critically low levels. The two flags and their setting/clearing mechanisms are shown below:

Table 5-6. SOC1 Flags

Flags()[SOC1]	CONDITION
1	*RemainingCapacity()* < ***SOC1 Set Threshold***
0	*RemainingCapacity()* > ***SOC1 Clear Threshold***

SOC1 Set Threshold is typically used as an initial low *StateOfCharge()* warning. If ***SOC1 Set Threshold*** is set to (–)1, then the *[SOC1]* bit becomes inoperative. ***SOC1 Set Threshold*** is normally set to 10% of ***Design Capacity***. ***SOC1 Clear Threshold*** is normally set to 5% above the ***SOC1 Set Threshold***; that is, 15% of ***Design Capacity***.

Table 5-7. SOCF Flags

Flags()[SOCF]	CONDITION
1	*RemainingCapacity()* < ***SOCF Set Threshold***
0	*RemainingCapacity()* > ***SOCF Clear Threshold***

SOCF Set Threshold is typically used as a final low *StateOfCharge()* warning. If ***SOCF Set Threshold*** is set to (–)1, then the *[SOCF]* bit becomes inoperative.

SOCF Set Threshold is normally set to 2% of ***Design Capacity***. ***SOCF Clear Threshold*** is normally set to 3% above the ***SOCF Set Threshold***, which is 5% of ***Design Capacity***.

The *[SOC1]* flag can be configured to control an interrupt pin (SE or HDQ) by enabling INTERRUPT mode. See *INTERRUPT Mode* for details.

Table 5-8. SOC1/SOCF Set and Clear Threshold

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
49	Discharge	0	SOC1 Set Threshold	U2	0	65535	150	mAh
		2	SOC1 Clear Threshold	U2	0	65535	175	mAh
		4	SOCF Set Threshold	U2	0	65535	75	mAh
		6	SOCF Clear Threshold	U2	0	65535	100	mAh

5.5 Battery Level

The fuel gauge can indicate when battery voltage has fallen below or risen above predefined thresholds.

Table 5-9. Battery Level Conditions, High

[BATHI]	CONDITION
1	*Voltage()* > ***BH Set Volt Threshold*** for ***BH Volt Time***
0	*Voltage()* ≤ ***BH Clear Volt Threshold***

The device must not be in SLEEP mode. It is recommended that the ***BH Set Volt Threshold*** is configured higher than the ***BH Clear Volt Threshold*** to provide proper voltage hysteresis.

Table 5-10. Battery Level Conditions, Low

[BATLO]	CONDITION
1	*Voltage()* < ***BL Set Volt Threshold*** for ***BL Set Volt Time***
0	*Voltage()* ≥ ***BL Clear Volt Threshold***

The device must not be in SLEEP mode. It is recommended that the ***BL Set Volt Threshold*** is configured lower than the ***BL Clear Volt threshold*** to provide proper voltage hysteresis.

Both the *[BATHI]* and *[BATLOW]* flags can be configured to control an interrupt pin (SE or HDQ) by enabling INTERRUPT mode. See *INTERRUPT Mode* for details.

Table 5-11. Battery Low Set Voltage Threshold, Time, and Clear

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
49	Discharge	8	BL Set Volt Threshold	I2	0	5000	2500	mV
		10	BL Set Volt Time	U1	0	60	2	s
		11	BL Clear Volt Threshold	I2	0	5000	2600	mV

Table 5-12. Battery High Set Voltage Threshold, Time, and Clear

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
49	Discharge	13	BH Set Volt Threshold	I2	0	5000	4500	mV
		15	BH Volt Time	U1	0	60	2	s
		16	BH Clear Volt Threshold	I2	0	5000	4400	mV

5.6 Safety Conditions

5.6.1 Overtemperature Fault Conditions During Charge/Discharge

The fuel gauge can indicate detection of overtemperature in charge, overtemperature in discharge, internal short, and tab disconnect events. To enable overtemperature interrupts, the following conditions are required:

1. ***OT Chg Time*** and ***OT Dsg Time*** must be configured to non-zero values AND
2. ***Pack Configuration [HOST_IE]*** = 1.

To enable internal short and tab disconnect interrupts:

- ***Pack Configuration B [SE_ISD]*** and ***Pack Configuration B [SE_TDD]*** must be set.
- During Charge (***Current*** > ***Chg Current Threshold***)
- *Flags() [OTC]* = 1 when *Temperature()* > ***OT Chg for OT Chg Time***

If the OTC condition clears prior to the expiration of the ***OT Chg Time*** timer, then the *Flags() [OTC*] bit is not set. This setting depends on the environment temperature and the battery specification. This parameter needs to be set based on the temperature limits mentioned in the battery specification.

OT Chg Time is a buffer time allotted for overtemperature in the charge direction condition. The timer starts every time that *Temperature()* is greater than ***OT Chg*** and during a charge. When the timer expires, the fuel gauge forces an *[OTC]* in *Flags()*. Setting the ***OT Chg Time*** to 0 disables this function. Default is set to 2 seconds, sufficient for most applications. Temperature is normally a slow-varying condition that does not need high-speed triggering. It must be set long enough to prevent false triggering of the *Flags() [OTC]* bit, but short enough to prevent damage to the battery pack.

To recover from an ***OT Chg*** fault, *Temperature()* < ***OT Chg Recovery***.

This is the only recovery method for an ***OT Chg*** fault. The default is 50°C, which is 5°C lower than ***OT Chg***.

During discharge (***Current*** > (–)***Dsg Current Threshold***)

Flags() [OTD] = 1 when *Temperature()* > ***OT Dsg for OT Dsg Time***

If the OTC condition clears prior to the expiration of the ***OT Dsg Time*** timer, then the *Flags() [OTD]* bit is not set. This setting depends on the environment temperature and the battery specification. Verify that the battery specification allows temperatures up to this setting while discharging, and verify that these setting are sufficient for the application temperature. The default is 60°C, which is sufficient for most Li- Ion applications. The default ***OT Dsg*** is higher than the default ***OT Chg*** because Li-Ion can handle a higher temperature in the discharge direction than in the charge direction.

OT Dsg Time is a buffer time allotted for overtemperature in the discharge direction condition. The timer starts every time that *Temperature()* measured is greater than ***OT Dsg*** during a discharge. When the timer expires, then the fuel gauge forces the *Flags() [OTD]* bit to be set. Setting the ***OT Dsg Time*** to 0 disables this feature. This is normally set to 2 seconds, which is sufficient for most applications. Temperature is normally a slow-acting condition that does not need high-speed triggering. Set ***OT Dsg Time*** long enough to prevent false triggering of the *[OTD]* bit in *Flags()*, but short enough to prevent damage to the battery pack.

To recover from a ***OT Dsg Fault***, *Temperature()* < ***OT Dsg Recovery***.

This is the only recovery method for an ***OT Dsg*** fault. The default is 55°C, which is 5°C lower than ***OT Dsg***.

Table 5-13. Charging Overtemperature Threshold, Delay Time, and Recovery

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
2	Safety	0	OT Chg	I2	0	1200	550	0.1°C
		2	OT Chg Time	U1	0	60	5	s
		3	OT Chg Recovery	I2	0	1200	500	0.1°C

Table 5-14. Discharging Overtemperature Threshold, Delay Time, and Recovery

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
2	Safety	5	OT Dsg	I2	0	1200	600	0.1°C
		7	OT Dsg Time	U1	0	60	5	s
		8	OT Dsg Recovery	I2	0	1200	550	0.1°C

5.6.2 Tab Disconnect Detection

A tab disconnect condition is detected and the *SafetyStatus()[TDD]* flag set when the ratio of current *StateofHealth()* to previous *StateofHealth()* is less than ***TDD SOH percent***. An interrupt can be configured to trigger on the (SE or HDQ) pin when a tab disconnect condition is detected. Enabling/disabling the Tab Disconnect Detection (TDD) feature is controlled via the ***[SE_TDD]*** bit in ***Pack Configuration B***.

Table 5-15. TDD State Of Health Percent

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
48	Data	18	TDD SOH Percent	U1	0	100	80	Percent

The *[TDD]* flag can be configured to control an interrupt pin (SE or HDQ) by enabling INTERRUPT mode. See *INTERRUPT Mode* for details.

5.6.3 ISD Faults

The fuel gauge can indicate detection of slow developing shorts, or micro shorts, within the battery if the ***[SE_ISD]*** bit in ***Pack Configuration B*** is set. The gauge compares the self-discharge current calculated based on RELAXATION mode to the *AverageCurrent()* measured in the system. The self-discharge rate is measured at 1-hour intervals. When battery *SelfDischargeCurrent()* is less than the predefined ***–Design Capacity/ISD Current*** threshold, the *[ISD]* of *Flags()* is set high. The *[ISD]* of *Flags()* can be configured to control interrupt pin (HDQ or SE) by enabling the INTERRUPT mode.

It is recommended to test this feature and tune ***ISD Current*** to obtain the optimal value to avoid both false positives and false negatives.

To help avoid false positives, a filtering the amount of change allowed in *SelfDischargeCurrent()* register is possible. A large value of ***ISD I Filter*** restricts large fluctuations in the value of *SelfDischargeCurrent()* if the most recent current value read by the gauge is significantly different from the previous readings. A small value of ***ISD I Filter*** allows the value of *SelfDischargeCurrent()* to update to a value that is closer to the most recent value read by the gauge. The *SelfDischargeCurrent()* is most accurate when the load on the gauge has been completely removed. The ***Min ISD Time*** parameter defines the amount of time(hrs) the gauge needs to wait after the initial DOD measurement is made in RELAXATION mode before comparing *SelfDischargeCurrent()* to the computed threshold.

Table 5-16. ISD Current, I Filter, and Min ISD Time

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
48	Data	19	ISD Current	U2	1	32767	10	Hour Rate
48	Data	21	ISD I Filter	U1	0	255	127	Count

Table 5-16. ISD Current, I Filter, and Min ISD Time (continued)

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
48	Data	22	Min ISD Time	U1	0	255	7	Hour

The *[ISD]* flag can be configured to control an interrupt pin (SE or HDQ) by enabling INTERRUPT mode. See Section 5.3, *INTERRUPT Mode*, for details.

5.7 Battery Trip Point Interrupt Function

To provide increased flexibility for capacity-based interrupts to the host, the fuel gauge incorporates a Battery Trip Point (BTP) function that allows the system to dynamically update the traditional ***SOC1 Set Threshold*** and ***SOC1 Clear Threshold*** at runtime using the *BTPSOC1Set()* and *BTPSOC1Clear()* standard commands. These thresholds are used to trigger an interrupt on the HDQ pin whenever the set or clear thresholds are crossed following an update to the *BTPSOC1Set()* and *BTPSOC1Clear()* values. Configuration of the interrupt polarity and enable/disable of the feature is provided via the ***Pack Configuration [INTPOL]*** and ***Pack Configuration C [BTP_EN]*** bits, respectively, while initialization values for the interrupt set and clear thresholds are programmed in ***SOC1 Set Threshold*** and ***SOC1 Clear Threshold*** as normal.

Note

Enabling of the BTP feature automatically disables all other interrupt sources on the HDQ pin, so care must be taken in configuring the fuel gauge for each particular end application, especially if non-BTP interrupts such as overtemperature, internal short detection, Tab Disconnect Detection, and battery low and high indications are required in the system.

When BTP is enabled, the fuel gauge continuously compares *RemainingCapacity()* with the values programmed in *BTPSOC1Set()* and *BTPSOC1Clear()* to determine whether or not it has crossed below the set or above the clear threshold. Once a threshold is crossed, additional conditions are verified to guard against an unintended interrupt trigger. For the BTP set threshold, the direction of current flow is checked to confirm that a discharge event is occurring. If true, the *Flags()[SOC1]* bit is set to 1 and an interrupt asserts on the HDQ pin. For the BTP clear threshold, the device again checks the direction of current flow to ensure that a charge event is occurring. Afterwards, an internal variable is examined to determine whether or not a change in the state of *Flags()[SOC1]* has already occurred due to a prior clear threshold crossing. If true, no change is made and a new interrupt will not fire; however, it is implied that a pre-existing interrupt will still be asserted. If false, the current state of *Flags()[SOC1]* is flipped to its opposite value and an interrupt subsequently triggered on the HDQ pin. In this way, the correct behavior is guaranteed in cases where the host updates the BTP set and clear thresholds diligently based on HDQ interrupts, but also when there is a failure to update the thresholds. If, at any time, new values are written to either *BTPSOC1Set()* or *BTPSOC1Clear()*, then the *[SOC1]* flag automatically reinitializes to 0 and the HDQ pin de-asserts to its default state. The entire functional flow of the BTP feature is illustrated in Figure 5-1, *BTP Algorithm Flow*.

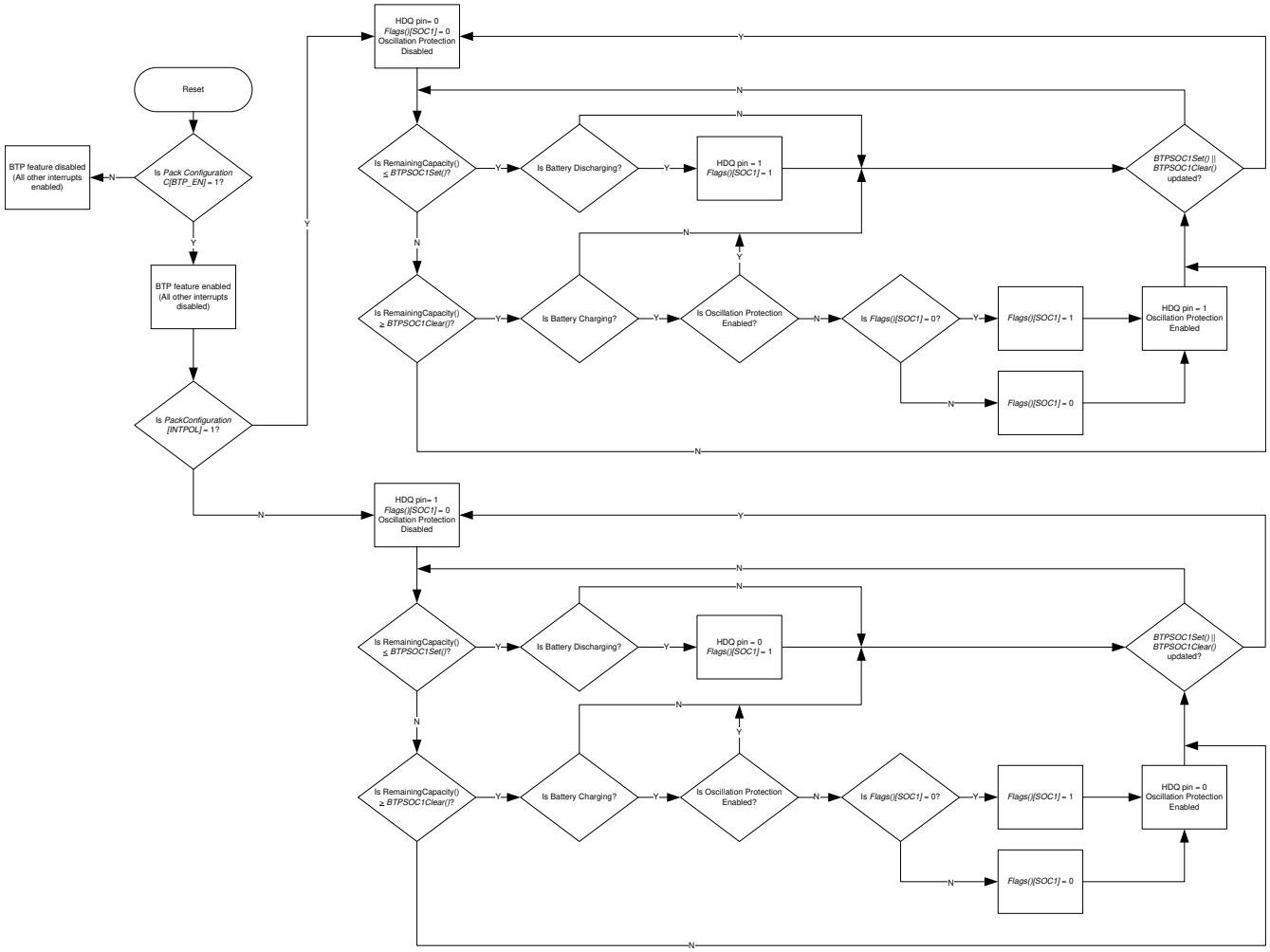


Figure 5-1. BTP Algorithm Flow

In normal usage, the BTP thresholds are continuously updated by the host system at predetermined increments, each time reinitializing the *Flags()[SOC1]* bit to 0 and waiting for the crossing of the next threshold to trigger a new interrupt. If the thresholds are always updated after each interrupt, then it is implied that the crossing of a set or clear threshold always triggers a new interrupt. This is highlighted below in Figure 5-2, *BTP Configuration with Multiple Thresholds*.

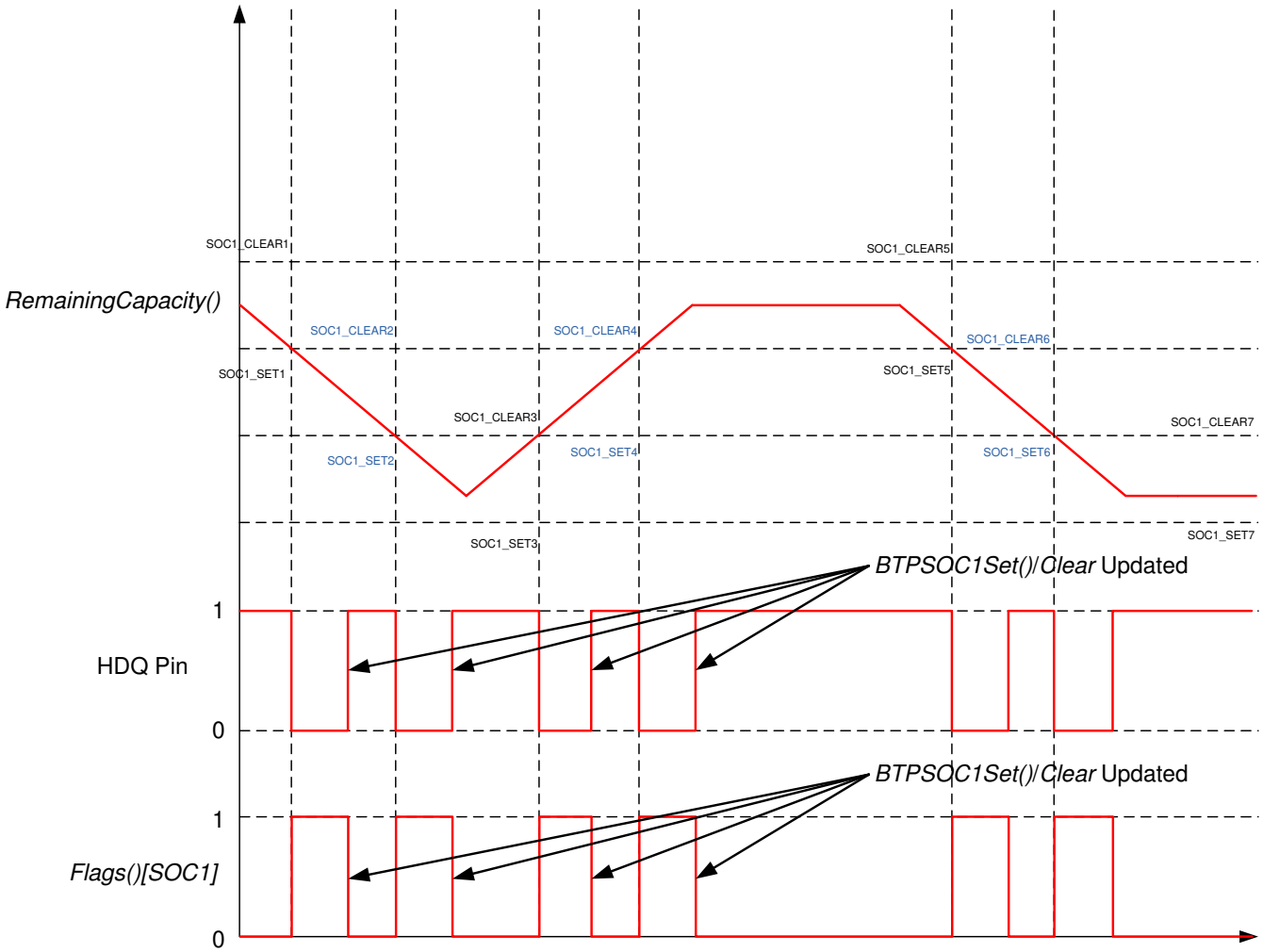


Figure 5-2. BTP Configuration with Multiple Thresholds

However, it is possible that the host may fail to write new thresholds or experience a significant delay in attempting to do so. In this case, there could be an occurrence where the clear threshold is crossed after an interrupt due to a prior set threshold crossing. Thus, the *[SOC1]* bit would experience a change but a new interrupt would not be triggered on HDQ. Thus, continued crossings without updates to *BTPSOC1Set()* or *BTPSOC1Clear()* will only result in changes to *Flags()[SOC1]*. Figure 5-3, *BTP Configuration with Shared Thresholds*, shows the case where identical thresholds are written to *BTPSOC1Set()* or *BTPSOC1Clear()*. Figure 5-4, *BTP Configuration with Separate Thresholds*, shows the alternate case where unique thresholds are written to *BTPSOC1Set()* or *BTPSOC1Clear()*.

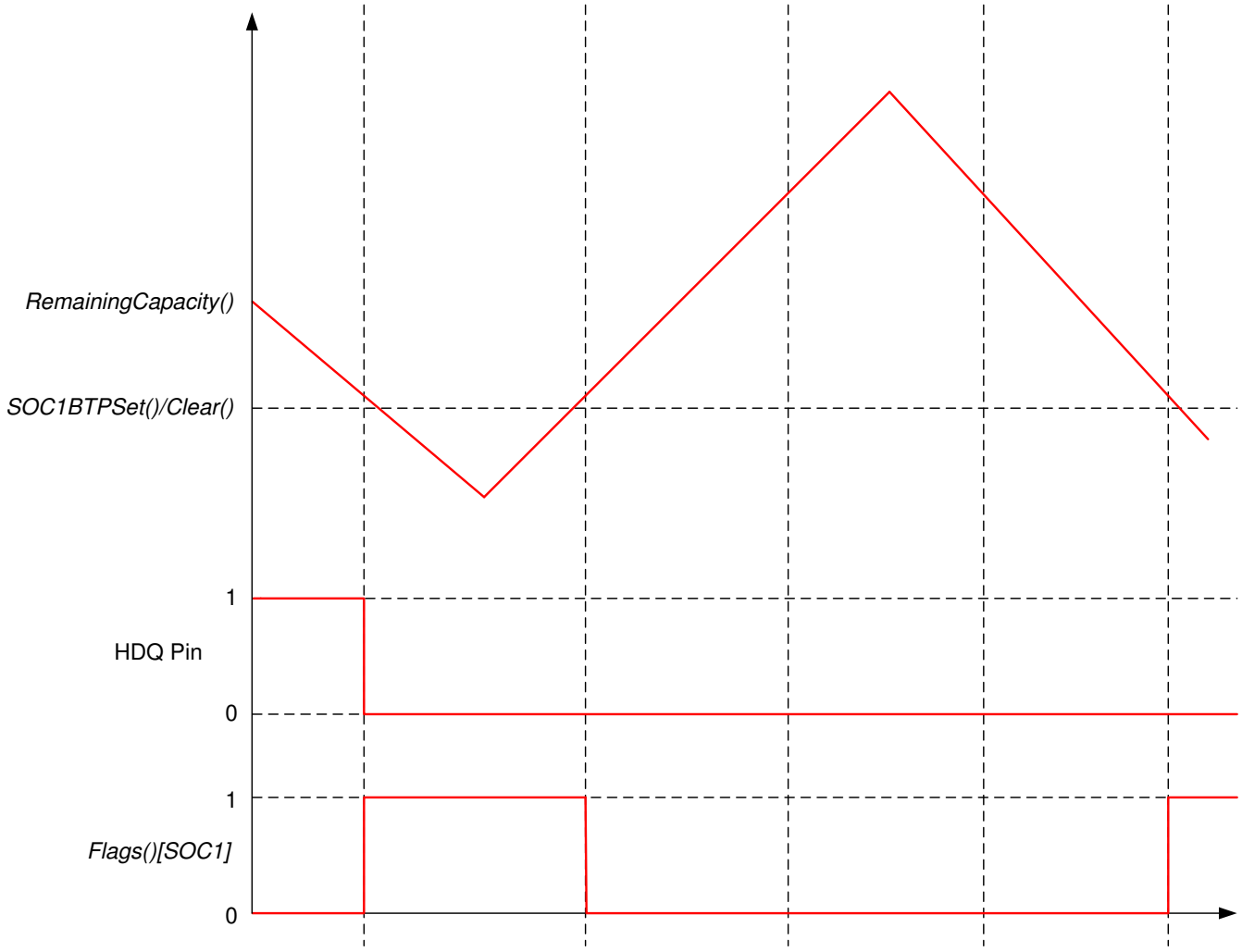


Figure 5-3. BTP Configuration with Shared Thresholds

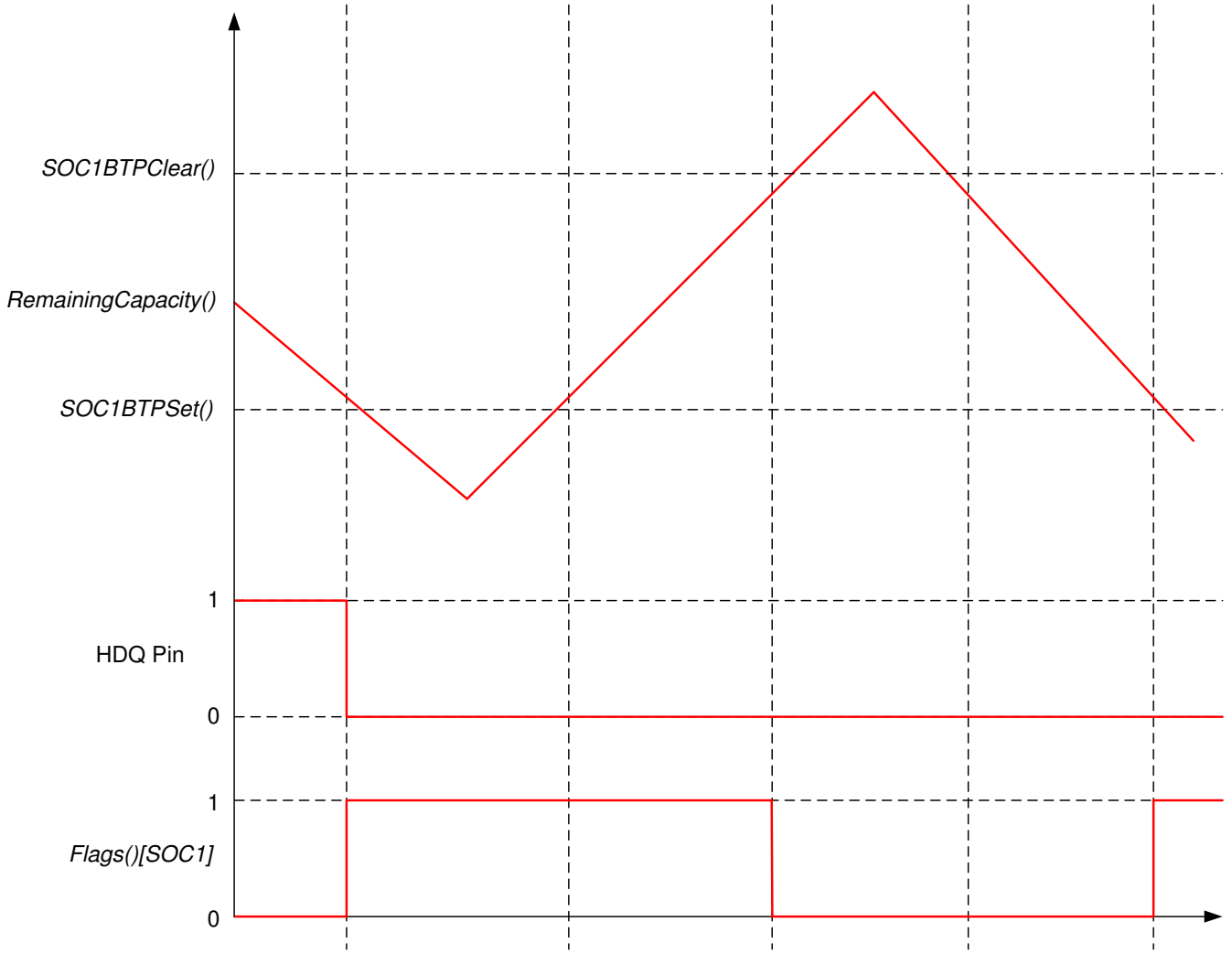


Figure 5-4. BTP Configuration with Separate Thresholds

Chapter 6

Impedance Track Fuel Gauging

6.1 Introduction

The BQ27542-G1 fuel gauge measures the cell voltage, current, and temperature to determine battery SOC based on outputs from the Impedance Track™ algorithm (see the *Theory and Implementation of Impedance Track Battery Fuel-Gauging Algorithm Application Report* [SLUA450] for more information). The fuel gauge monitors charge and discharge activity by sensing the voltage across a small-value resistor (5-mΩ to 20-mΩ typical) between the SRN and SRP pins and in series with the cell. By integrating charge passing through the battery, the SOC is accurately adjusted during charge or discharge operation. The total chemical capacity of the battery (Qmax) is found by comparing states of charge before and after applying a discharge that results in charge passed of at least 37% of ***Design Capacity***.

The initial ***Qmax Cell 0*** and ***Design Capacity*** are set based on values from the applicable cell manufacturers' data sheet multiplied by the number of parallel cells. When a system load is applied, the impedance of the cell is calculated from the OCV, the measured voltage under load, and the discharge current or power as configured in ***Load Select***. The ***Load Select*** parameter can be set to various options such as average discharge current from the last cycle (***Avg I Last Run***), present average discharge current, 14s-filtered *AverageCurrent()*, ***Design Capacity***/5 discharge rate, a user-defined discharge rate written to *AtRate()*, or the value programmed in ***User Rate-mA***. The ***Load Mode*** parameter extends this further by supporting the same options with respect to power instead of current.

During battery discharge, the fuel gauge simulates an iterative discharge based on the selected load profile and sums the resulting charge it takes to reach ***Terminate Voltage***, the charge passed since the last OCV measurement, and the starting charge passed that is required to reach the last OCV point to determine *FullChargeCapacity()*, which is the total capacity that can be extracted from a fully charged battery based on the present load profile and temperature. The predicted capacity left in the battery, or *RemainingCapacity()*, is simply the summed charge from present SOC to ***Terminate Voltage*** as determined by the simulation result. The fuel gauge uses the battery impedance and OCV profiles, Qmax, and present SOC to achieve this accurate prediction.

During battery charge, the fuel gauge coulomb counts up from where the last discharge ended and stores the battery voltage once charge termination is detected (***V at Chg Term***) to compute the depth-of-discharge (DOD) at end of charge (DOD@EOC). This provides a more accurate 100% SOC reference point for deriving starting charge passed (Qstart) since most systems do not charge the battery to absolutely full. In addition to *RemainingCapacity()* and *FullChargeCapacity()*, the fuel gauge also reports uncompensated (that is, <C/20) versions of capacity in *NominalAvailableCapacity()* and *FullAvailableCapacity()*, respectively. At relaxation entry (|*AverageCurrent()*| < ***Quit Current*** for ***Chg Relax Time*** or ***Dsg Relax Time***, depending on previous state), the fuel gauge waits 60 seconds before beginning to take OCV measurements to check that the battery is in a well-relaxed state (dV/dt < 1 µV/s) and to update the reported SOC every hour (although accumulated charge can result in change to SOC between these updates).

6.1.1 System Design Parameters

6.1.1.1 Design Voltage

Design Voltage is the nominal voltage of the pack as specified by the battery vendor. The value should be set based on the battery specification.

6.1.1.2 Cycle Count

Cycle count records the number of cycles the battery has experienced. One cycle occurs when accumulated discharge ≥ ***CC Threshold***. The value is reported in *CycleCount()*.

6.1.1.3 Cycle Count Threshold

This value increments *CycleCount()*. When the gauge accumulates enough discharge capacity equal to ***CC Threshold***, then it increments *CycleCount()* by 1. This discharge capacity does not have to be consecutive. The internal register that accumulates the discharge is not cleared at any time except when the internal accumulating register equals the ***CC Threshold***, and increments *CycleCount()*.

This is normally set to about 90% of the ***Design Capacity***.

6.1.1.4 Design Capacity

This is the original chemical capacity of the pack as specified by the battery vendor. This is used in the Impedance Track algorithm in remaining and full charge capacity (RM and FCC) calculations. The value should be set based on the battery specification.

6.1.1.5 Design Energy

Design Energy is similar to ***Design Capacity*** but represented in energy units.

Design Energy = ***Design Capacity*** × Design Voltage

The actual unit of this parameter is dependent on ***Design Energy Scale***.

6.1.1.6 State of Health Load I

StateOfHealth() is calculated using the ratio of *FullChargeCapacity()* (FCC) and *DesignCapacity()*. The FCC used in the SOH calculation is simulated using a fixed temperature (25ºC) and load (defined by ***SOH Load I***). The FCC value used is not necessarily the same as the *FullChargeCapacity()* data RAM register since the value reported in data RAM register changes based on current system load and temperature.

The default is –400 mA. It is recommended to set this value to a typical system current.

6.1.1.7 Design Energy Scale

Design Energy Scale selects the scale and units of a set of data flash parameters. The value of ***Design Energy Scale*** can be either 1 or 10. For battery capacities larger than 6 Ahr, ***Design Energy Scale*** = 10 is recommended.

Table 6-1. Data Flash Parameter Unit/Scale Based on Design Energy Scale

Data Flash	Design Energy Scale = 1 (default)	Design Energy Scale = 10
Design Energy	mWh	cWh
Reserve Capacity (mWh)	mWh	cWh
Avg Power Last Run	mW	cW
User Rate-Pwr	mWh	cWh
T Rise	No Scale	Scaled by ×10

6.1.1.8 System Design Parameters DF

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
48	Data	0	Design Voltage	I2	2000	5000	3800	mV
		8	Cycle Count	U2	0	65535	0	Count
		10	CC Threshold	I2	100	32767	900	mAh
		12	Design Capacity	I2	0	14500	1000	mAh
		14	Design Energy	I2	0	32767	3800	mWh
		16	SOH Load I	I2	–32767	0	–400	mA
		23	Design Energy Scale	U1	1	10	1	Number

6.2 Gauge FW Operation Modes

6.2.1 CHARGE Mode

CHARGE mode is entered when *AverageCurrent()* > ***Chg Current Threshold***. During CHARGE mode, ***FLAGS[DSG]*** bit is cleared.

Table 6-2. Charge Detection Threshold

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
81	Current Thresholds	2	Chg Current Threshold	I2	0	2000	75	mA

6.2.2 RELAXATION Mode

RELAXATION mode is entered either from CHARGE mode or DISCHARGE mode based on the relationship between *AverageCurrent()* and ***Quit Current***. ***Quit Current*** sets a current threshold to determine when the fuel gauge goes into RELAXATION mode from CHARGE or DISCHARGE mode. Either of the following criteria must be met to enter RELAXATION mode:

1. *AverageCurrent()* is **less than** (–)***Quit Current*** and then goes within (±)***Quit Current*** for ***Dsg Relax Time***.
2. *AverageCurrent()* is **greater than** ***Quit Current*** and then goes within (±)***Quit Current*** for ***Chg Relax Time***.

During RELAXATION mode, ***FLAGS[DSG]*** bit is set.

The ***Chg Relax Time*** is used in the function to determine when to go into RELAXATION mode after charge current ceases. When *AverageCurrent()* is greater than ***Quit Current*** and then goes within (±)***Quit Current***, the ***Chg Relax Time*** timer is initiated. If the current stays within (±)***Quit Current*** until the ***Chg Relax Time*** timer expires, then the fuel gauge goes into RELAXATION mode.

The ***Dsg Relax Time*** is used in the function to determine when to go into RELAXATION mode after discharge current ceases. When *AverageCurrent()* is less than (–)***Quit Current*** and then goes within (±)***Quit Current***, the ***Dsg Relax Time*** timer is initiated. If the current stays within (±)***Quit Current*** until the ***Dsg Relax Time*** timer expires, then the fuel gauge goes into RELAXATION mode.

After 30 minutes in RELAXATION mode, the fuel gauge starts checking if the dV/dt < 1 µV/s requirement for OCV readings is satisfied. When the battery relaxes sufficiently to satisfy this criterion, the fuel gauge takes an OCV reading for updating Qmax. These updates are used by the Impedance Track algorithm.

It is critical that the battery voltage be relaxed during OCV readings to get the most accurate results. The quit current threshold must not be higher than ***Design Capacity***/20 when attempting to go into RELAXATION mode; however, it should not be so low as to prevent going into RELAXATION mode due to noise. The current threshold that the ***Quit Current*** parameter sets should always be less than the magnitude of the current threshold the ***Chg Current Threshold*** sets and less than the magnitude of the current threshold the ***Dsg Current Threshold*** sets.

Table 6-3. Quit Current, Discharge, and Charge Relax Time

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
81	Current Thresholds	4	Quit Current	I2	0	1000	40	mA
		6	Dsg Relax Time	U2	0	65535	60	s
		8	Chg Relax Time	U1	0	255	60	s

6.2.3 DISCHARGE Mode

DISCHARGE mode is entered when *AverageCurrent()* ≤ –***Dsg Current Threshold***.

Note

Current is negative while discharging.

Dsg Current Threshold should be set low enough to be below any normal application load current but high enough to prevent noise or drift from affecting the measurement (note that ***Dsg Current Threshold*** is a positive value). During DISCHARGE mode, ***FLAGS[DSG]*** bit is set.

Table 6-4. Discharge Detection Threshold

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
81	Current Thresholds	0	Dsg Current Threshold	I2	0	2000	60	mA

6.3 Current/Power Profiles

For better accuracy and performance, the fuel gauge can be configured with the system or application loading conditions. This helps the gauge to better adapt the algorithm based on the use case.

When a system load is applied, the impedance of the cell is calculated from the OCV, the measured voltage under load, and the discharge current or power as configured in the ***Load Select*** parameter. ***Load Select*** defines the type of power or current model to be used to compute load-compensated capacity in the Impedance Track algorithm.

For simulations during RELAXATION, CHARGE, and at the start of DISCHARGE (since a new average has not been gathered), it would use data flash values for simulations. Once the discharge lasts 500 seconds, the gauge re-simulates using the new running average. Thereafter, during discharge, it uses the continuous running average for any subsequent simulations. Re-simulations can also be triggered by a temperature change, which can cause an updated simulation to occur earlier than 500 seconds into a discharge. In this case, this simulation would use the present running average.

The ***Load Mode*** parameter extends this further by supporting the same options with respect to power instead of current. ***Load Mode*** selects either the constant-current or constant-power model for the Impedance Track algorithm as used in ***Load Select***. When ***Load Mode*** is 0, the constant current model is used (default). When ***Load Mode*** is 1, the constant-power model is used. The CONTROL_STATUS [LDMD] bit reflects the status of ***Load Mode***.

If ***Load Mode*** = 0 (constant-current model), then the options presented in Table 6-5.

Table 6-5. Constant-Current Model Used When SHUTDOWN State = 0

Load Select Value	Current Model Used
0	Average discharge current from previous cycle: There is an internal register that records the average discharge current through each entire discharge cycle. The previous average is stored in this register.
1 (default)	Present average discharge current: This is the average discharge current from the beginning of this discharge cycle until present time.
2	Average current: based off the *AverageCurrent()*
3	Current: based off of a low-pass-filtered version of *AverageCurrent()* (τ = 14 s)

Table 6-5. Constant-Current Model Used When SHUTDOWN State = 0 (continued)

Load Select Value	Current Model Used
4	Design capacity/5: C Rate based off of Design Capacity /5 or a C/5 rate in mA
5	Use the value specified by *AtRate()*
6	Use the value in ***User_Rate-mA.*** This gives a completely user-configurable method.

If ***SHUTDOWN state*** = 1 (constant-power model) then the following options are available:

Table 6-6. Constant-Power Model Used When SHUTDOWN State = 1

Load Select Value	Power Model Used
0	Average discharge power from previous cycle: There is an internal register that records the average discharge power through each entire discharge cycle. The previous average is stored in this register.
1	Present average discharge power: This is the average discharge power from the beginning of this discharge cycle until present time.
2	Average current × voltage: based off the *AverageCurrent()* and *Voltage()*.
3	Current × voltage: based off of a low-pass-filtered version of *AverageCurrent()* (τ = 14 s) and *Voltage()*
4	Design energy/5: C Rate based off of Design Energy /5 or a C/5 rate in mW or cW
5	Use the value specified by *AtRate()*
6	Use the value in User_Rate-Pwr. This gives a completely user-configurable method.

The fuel gauge logs the *AverageCurrent()* averaged from the beginning to the end of each discharge. It stores this average current from the previous discharge period in ***Avg I Last Run*** provided that the previous discharge lasted at least 500 seconds.

Note

It is recommended that users set ***Avg I Last Run*** to typical values for their system to correctly initialize predictions.

The fuel gauge logs the power averaged from the beginning to the end of each discharge. It stores this average power from the previous discharge period in ***Avg P Last Run*** provided the previous discharge lasted at least 500 seconds. To get a correct average power reading, the fuel gauge continuously multiplies instantaneous current with *Voltage()* to get power. It then logs this data to derive the average power.

Note

It is recommended that users set ***Avg P Last Run*** to typical values for their system to correctly initialize predictions.

In situations wherein the available load options do not match the application, custom current or power profiles can also be provided to the gauge. ***User Rate-mA*** can be used to indicate a current load that the algorithm will use for the *RemainingCapacity()* computation in the Impedance Track algorithm.

Note

This parameter is used only when ***Load Select*** = 6 and ***SHUTDOWN state*** = 0. An example application that requires this register is one that has increased predefined current at the end of discharge. With this application, it is logical to adjust the rate compensation to this period because the IR drop during this end period is affected the moment ***Terminate Voltage*** is reached.

User Rate-Pwr can be used to indicate a power load that the algorithm will use for the *RemainingCapacity()* computation in the Impedance Track algorithm.

Note

This parameter is used only when ***Load Select*** = 6 and ***SHUTDOWN state*** = 1. An example application that requires this register is one that has increased predefined power at the end of discharge. With this application, it is logical to adjust the rate compensation to this period because the IR drop during this end period is affected the moment ***Terminate Voltage*** is reached. The actual unit of this parameter is dependent on ***Design Energy Scale***.

The gauge also provides maximum and minimum limits for current/power used in IT simulations. These are the ***Max Sim Rate*** and ***Min Sim Rate*** parameters. They are a function of ***Design Capacity*** or ***Design Energy***.

Table 6-7. SHUTDOWN state Limits

SHUTDOWN state	Limit
0	***–Design Capacity***/***Max Sim Rate*** ≤ Simulation Current ≤ ***–Design Capacity***/***Min Sim Rate***
1	***–Design Energy***/***Max Sim Rate*** ≤ Simulation Power ≤ ***–Design Energy***/***Min Sim Rate***

In situations where a controlled shutdown after 0% *RemainingCapacity()* is reached, the ***Reserve Cap-mAh*** can be used that determines how much actual remaining capacity exists after reaching 0% *RemainingCapacity()*, before ***Terminate Voltage*** is reached when SHUTDOWN state = 0 is selected. A loaded rate or no-load rate of compensation can be selected for *Reserve Cap* by setting the ***[RESCAP]*** bit in the ***Pack Configuration*** data flash register. This is a specialized function to allow time for a controlled shutdown after 0 *RemainingCapacity()* is reached.

The voltage must dip below ***Terminate Voltage*** for at least this many seconds before *RemainingCapacity()* and *StateOfCharge()* will be forced to 0.

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
80	IT Cfg	0	Load Select	U1	0	6	1	Number
		1	SHUTDOWN state	U1	0	1	1	Number
		64	Terminate Voltage	I2	2800	3700	3000	mV
		73	User Rate-mA	I2	0	32767	0	mA
		75	User Rate-Pwr	I2	0	32767	0	cW
		77	Reserve Cap-mAh	I2	0	14500	0	mAh
		88	Max Sim Rate	U1	0	255	1	HourRate
		89	Min Sim Rate	U1	0	255	20	HourRate
		118	TermV Valid t	U1	0	255	2	s
82	State	5	Avg I Last Run	I2	–32768	0	–299	mA
		7	Avg P Last Run	I2	–32768	0	–1131	mA

6.3.1 Load Select

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
80	IT Cfg	0	Load Select	U1	0	6	1	Number

Load Select defines the type of power or current model to be used to compute load-compensated capacity in the Impedance Track algorithm. By default, ***Load Select*** is set to 1, which means the IT algorithm uses a running average of the current discharge period. Once the discharge stops, the algorithm stores the average in data flash as the ***Avg I Last Run*** and ***Avg P Last Run*** variables. For simulations during RELAXATION, CHARGE, and at the start of DISCHARGE (since a new average has not been gathered), it would use data flash values for simulations. Once the discharge lasts 500 seconds, the gauge re-simulates using the new running average. Thereafter, during discharge, it uses the continuous running average for any subsequent simulations.

Re-simulations can also be triggered by a temperature change, which can cause an updated simulation to occur earlier than 500 seconds into a discharge. In this case, this simulation would use the present running average.

If ***SHUTDOWN state*** = 0 (constant-current model), then the options presented in Table 6-8 are available.

Table 6-8. Constant-Current Model Used When SHUTDOWN State = 0

Load Select Value	Current Model Used
0	Average discharge current from previous cycle: There is an internal register that records the average discharge current through each entire discharge cycle. The previous average is stored in this register.
1 (default)	Present average discharge current: This is the average discharge current from the beginning of this discharge cycle until present time.
2	Average current: based off the *AverageCurrent()*
3	Current: based off of a low-pass-filtered version of *AverageCurrent()* (τ = 14 s)
4	Design capacity/5: C Rate based off of Design Capacity /5 or a C/5 rate in mA.
5	Use the value specified by *AtRate()*
6	Use the value in ***User_Rate-mA.*** This gives a completely user-configurable method.

If ***SHUTDOWN state*** = 1 (constant-power model) then the following options are available:

Table 6-9. Constant-Power Model Used When SHUTDOWN State = 1

Load Select Value	Power Model Used
0	Average discharge power from previous cycle: There is an internal register that records the average discharge power through each entire discharge cycle. The previous average is stored in this register.
1	Present average discharge power: This is the average discharge power from the beginning of this discharge cycle until present time.
2	Average current × voltage: based off the *AverageCurrent()* and *Voltage()*.
3	Current × voltage: based off of a low-pass-filtered version of *AverageCurrent()* (τ = 14 s) and *Voltage()*
4	Design energy/5: C Rate based off of Design Energy /5 or a C/5 rate in mW or cW.
5	Use the value specified by *AtRate()*
6	Use the value in User_Rate-Pwr. This gives a completely user-configurable method.

6.3.2 Thermal Rise Factor

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
82	State	11	T Rise	I2	0	32767	50	Num

This is the thermal rise factor that is used in the single time constant heating-cooling thermal modeling. If set to 0, this feature is disabled and simulations in the IT algorithm will not account for self-heating of the battery cell. Larger values of ***T Rise*** lead to higher temperature rise estimates for the IT simulation.

6.3.3 Thermal Time Constant

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
82	State	13	T Time Constant	I2	0	32767	1000	Num

This is the thermal time constant that is used in single time constant heating-cooling thermal modeling. The default setting can be used, or it can be modified to improve low-temperature accuracy if testing shows the model does not match the actual performance.

T Time Constant defaults to 1000. This is sufficient for many applications. However, it can be modified if better predictive accuracy at low temperatures is desired.

6.4 Qmax Update

The Impedance Track algorithm defines Qmax as the maximum chemical capacity of a cell measured in milliampere hours (mAh). As the battery ages, Qmax can reduce. For the Impedance Track algorithm to

account for the capacity fade, it is crucial to update Qmax periodically. For a proper Qmax update, the following conditions must be met:

1. Two OCV measurements separated by a capacity change of the cell.
2. Between the two OCV measurements, 37% of ***Design Capacity*** must pass, whether from discharging or charging.
3. The time between the OCV measurements must not be too long. The time depends on battery capacity, sense resistor size, and deltaQ. Typically, an average of 16 mA must have passed between the two OCV measurements. It is not a continuous requirement, but deltaQ/time must be > 16 mA.
4. The temperature at the two OCV readings must be within the range of 10°C–40°C.
5. The RDIS flag must be 0.

If ***VOK*** is set while the gauge detects either charging or discharging, then a Qmax update at the next relax is possible. If at entry to Relax ***VOK*** is set and then clears when OCVTAKEN sets, then there has been a successful Qmax measurement.

Note

VOK is cleared until a charge/discharge cycle is started.

If the dV/dt condition is not met and the fuel gauge continues to reside in RELAXATION mode, then a forced OCV measurement and subsequent DOD computation occurs after 5 hours have elapsed; however, a Qmax update is still subject to the temperature, voltage, and minimum passed charge requirements before an update can occur.

If *AverageCurrent()* > ***Deadband*** is detected during the OCV measurement, then IR correction is used to compensate the value prior to using it to compute a new DOD. The value programmed in ***Max IR Correct*** determines the maximum allowed correction voltage based on detected charge current. If discharge current is detected instead, then no cap is applied. ***Max IR Correct*** only applies to OCV lookup after wakeup with detected charge current when gauge needs to establish capacity baseline, but the current is already flowing. If current is flowing during a voltage measurement that is used for finding initial DOD, IR correction eliminates the effect of the IR drop across the cell impedance and obtain true OCV. ***Max IR Correct*** is the maximum value of IR correction that is used. It is to avoid artifacts due to very high resistance at low DOD values during charge.

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
81	Current Thresholds	9	Max IR Correct	I2	0	1000	400	mV

The time between the first OCV point and second OCV points should not exceed CC offset error accumulation limit.

Disqualification time is calculated as:

Disqualification time = ***Design Capacity*** × ***Qmax Capacity Err*** ÷ (offset current × 100)

where offset current = CC Deadband ÷ sense resistor.

For the value of CC deadband = 10 µV and sense resistor 10 mΩ, offset current is 1 mA.

So if ***Design Capacity*** is 1000 mA, disqv time = 1000 mA × 1.5 ÷ (1 × 100) = 15 hrs.

Several parameters are used by the fuel gauge to prevent large swings in Qmax in exceptional circumstances. Before updating Qmax, the fuel gauge checks that the New Raw Qmax is close to Present Qmax by using the below check:

abs(Present Qmax – New Raw Qmax)/Present Qmax ≤ ***Max Qmax Change***

If this condition is not satisfied, the fuel gauge will not update Qmax. If this condition is satisfied, then the fuel gauge will calculate the new filtered Qmax value and update it in the data flash subject to the following bounds. To prevent large, sudden changes in Qmax:

If Abs(Present Qmax – New Filtered Qmax)/***Design Capacity*** > ***Qmax Max Delta %***

New Filtered Qmax is capped to:

Present Qmax ± ***Design Capacity*** × ***Qmax Max Delta %*** ÷ 100

A further check is made to make sure that the new filtered Qmax does not go beyond a pre-defined upper limit during the lifetime of a pack:

New Filtered Qmax > ***Upper Bound Qmax*** × ***Design Capacity*** ÷ 100

Then the new Qmax value written to the data flash will be capped to:

Upper Bound Qmax × ***Design Capacity*** ÷ 100

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
80	IT Cfg	61	Qmax Capacity Err	U1	0	100	15	0.10%
		62	Max Qmax Change	U1	0	255	30	%
		96	Qmax Max Delta %	U1	0	100	5	mAh
		97	Qmax Bound %	U1	0	255	130	mAh

6.4.1 Charge Hysteresis Voltage Shift

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
80	IT Cfg	106	Charge Hys V Shift	I2	0	2000	40	mV

Charge Hys V Shift is a flash parameter that helps the gauge to avoid Qmax update in the flat region after a charge to avoid OCV hysteresis effects. If OCV (in mV) < Flat region upper bound (typically ~3800 mV) + ***Charge Hys V Shift***, then Qmax update is not allowed.

It is recommended to keep this value at the default setting of 40 mV.

6.5 Fast Qmax Update

In certain applications, it can be difficult to achieve traditional Qmax updates for systems with noisy standby currents that prohibit the dV/dt requirement from being met and whose total time in relax never reaches 5 hours, or for systems that rarely enter RELAXATION at all. The Fast Qmax feature provides a complement to traditional Qmax updates to more reliably account for aging effects in such systems by allowing Qmax updates to be achieved with only one or no qualified DOD points from a battery RELAXATION state. The feature is enabled via the ***Pack Configuration C [FastQmax]*** bit and uses several conditional checks to determine when fast Qmax-specific DOD point collection is allowed to start and when these are qualified for use in a fast Qmax update.

Note

The Fast Qmax Update algorithm is not used during a learning cycle if ***Update Status*** ≠ 2.

The algorithm begins taking new discharge-based fast Qmax DOD points every 30 seconds once the following conditions are detected:

- DOD > ***Design Capacity*** or Voltage < ***Terminate Voltage*** + ***Fast Qmax Start Volt Delta, and***
- Current ≤ ***Design Capacity***/***Fast Qmax Current Threshold***

The algorithm qualifies and saves the discharge-based fast Qmax DOD point at the end of discharge when the following conditions are met:

- DOD > ***Fast Qmax End DOD%*** or Voltage ≤ ***Terminate Voltage*** + ***Fast Qmax Volt Buffer***, and
- Number of Fast Qmax measurements ≥ 3

The algorithm also qualifies and saves a charge-based fast Qmax DOD point 50 mV when full charge termination is achieved (that is, *Flags()[FC]* is set).

Note

The ***Pack Configuration C [FastQmax]*** bit only controls enable/disable of DOD point collection in the near end of discharge region. DOD points at end of charge are always recorded for fast Qmax purposes.

As a result, it is possible to get fast Qmax updates with DOD points from relaxation, end of discharge, and end of charge. However, a fast Qmax update will not happen immediately upon end of charge, and will only be enforced if an attempt to record a new OCV has failed during battery relaxation following a full charge event. A Fast Qmax update is disqualified if:

1. The intended DOD points were captured outside of the allowed 10°C to 40°C temperature range OR
2. Passed charge accumulated between them is less than 37% × ***Design Capacity***.

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
80	IT Cfg	39	Fast Qmax Start DOD %	U1	0	100	92	%
		41	Fast Qmax Start Volt Delta	I2	0	4200	200	mV
		43	Fast Qmax Current Threshold	U2	0	1000	4	HourRate
		40	Fast Qmax End DOD %	U1	0	100	96	%

6.6 Resistance Update

Resistance measurements are conducted over the course of a discharge cycle and are constantly stored in RAM. The actual resistance value in data flash is updated if certain conditions are met. Resistance update qualifications are:

1. The applied discharge current must exceed ***Design Capacity***/10 rate OR The IR drop between open-circuit voltage and the measured voltage under load must be > ***Res V Drop***.
2. Discharge time > 500 seconds.
3. RUP_DIS must be 0 and IT must be enabled.

Res V Drop is used during battery discharge to qualify sufficient conditions for measuring and storing resistance values. It is useful in applications with low-rate discharge or frequent cold temperature usage that typically have trouble achieving consistent resistance updates. Even with low current, the voltage drop requirement can still be met if enough cell resistance is evident.

The Resistance values that are calculated are subject to additional checks to prevent sudden changes or drops.

This is a weighting factor that takes a certain percentage of the previous Ra table value and the remaining percentage comes from the newest calculated Ra value. This is to prevent resistances in the Ra table from changing quickly. ***Ra Filter*** is a filter constant used to calculate the filtered Ra value that is stored into data flash from the old Ra value.

Ra = (Ra_old × ***Ra Filter*** + Ra_new × (1000 – ***Ra Filter***)) ÷ 1000

It is normally set to 800 (80% previous Ra value plus 20% learned Ra value to form new Ra value).

During the update of Ra values a filtering process is performed to eliminate unexpected fluctuations in the updated Ra values. ***Ra Max Delta*** limits the change in Ra values to an absolute magnitude per Ra update. This value should be set to 15% of the Ra[4] value. The value needs to be manually adjusted after a chemistry change.

After this filter has been applied, there is a final check to make sure that the new resistances satisfy both ***Max Res Factor*** and ***Min Res Factor***.

Max Res Factor is the maximum allowable cumulative percentage (ratio) increase for impedance values stored in the Ra table (over 15 gridpoint updates).

For Ra_new > Ra_old,

New Ra = min(Ra_new, Ra_old × ***Max Res Factor*** ÷ 10)

The default setting is 15. The algorithm divides the value of this parameter by 10. The upper bound is determined by multiplying (***Max Res Factor***/10) by the impedance value stored in the Ra table. Therefore, a value of 15 indicates resistance can only change by 50% from the current resistance value in the positive direction.

Min Res Factor is the maximum allowable cumulative percentage (ratio) decrease for impedance values stored in the Ra table (over 15 gridpoint updates).

For Ra_new < Ra_old

New Ra = max(Ra_new, Ra_old × ***Min Res Factor*** ÷ 10)

The default setting is 5. The algorithm divides the value of this parameter by 10. The lower bound is determined by multiplying (***Min Res Factor***/10) by the impedance value stored in the Ra table. Therefore, a value of 5 indicates resistance can only change by 50% from the current resistance value n the negative direction.

Transient modelling is used during resistance measurements. ***Res Relax Time*** or resistance relaxation time represents the time it takes for the internal resistance to be fully saturated. The resistance increases from 0 to the final value determined by the Ra table, as defined by the exponent with time constant ***Res Relax Time*** during discharge simulation. This way, the gauge will not simulate immediate large IR drops when it calculates the instantaneous voltage from the battery under load. The default value is 500 seconds, which is sufficient for most applications.

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
80	IT Cfg	17	Max Res Factor	U1	0	255	15	num
		18	Min Res Factor	U1	0	255	5	num
		20	Ra Filter	U2	0	1000	800	num
		22	Res V Drop	I2	0	32767	50	mV
		69	ResRelax Time	U2	0	65535	500	s
		90	Ra Max Delta	I2	0	32767	54	mΩ

6.7 Fast Resistance Scaling

At low temperatures and very high rate discharges, SOC convergence to 0 using the interpolated resistances can lead to higher errors. Fast Resistance Scaling provides improved SOC convergence to 0% (that is, ***Terminate Voltage***) by scaling resistance values used in capacity prediction simulations instead of using interpolated resistance table values as-is. The algorithm becomes active once one of the following is true:

- *StateofCharge()* ≤ ***Fast Scale Start SOC*** OR
- *Voltage()* < (***Terminate Voltage*** + ***Term V Delta***)

It then begins scaling resistance values every 30 seconds based on the ratio of most recent measured resistance (R_{new}) to stored resistance (R_{old}) at the present SOC. This allows the predicted remaining capacity to gradually converge to the cell's empty point and avoids potential for SOC jumps to empty near the end of discharge. The minimum and maximum scaling factors that can be employed in the Fast Resistance Scaling algorithm are stored in ***Min Res Scale*** and ***Max Res Scale***, respectively, where a value of 1000 corresponds to 1x and 200 corresponds to 0.2x. For most applications, the default value of ***Term V Delta*** and ***Fast Scale Start SOC*** are recommended. Further, it is typically best to keep (***Terminate Voltage*** + ***Term V Delta***) below 3.6 V for most battery applications. The feature is enabled via the ***[FConvEn]*** bit in ***Pack Configuration B***.

Thermal modeling is designed to estimate cell heating based on current flow through the battery and compensates the predicted *RemainingCapacity()* based on the true cell temperature output from the model.

However, it is possible to overestimate cell self-heating in particular cases and this could result in overestimation of *RemainingCapacity()*. As a result, thermal modeling is disabled by default in the ***Fast Resistance Scaling*** region but can be enabled for a given application using the ***Pack Configuration C [FConvTempEn]*** configuration bit.

For better flexibility, users can configure an independent load profile for use during Fast Resistance Scaling, using the ***Fast Scale Load Select*** parameter. This parameter can be set to any value supported by the standard ***Load Select*** and is useful for systems that exhibit significant load changes near the end of discharge, enabling the gauge to better predict remaining SOC in such cases. The default value for ***Fast Scale Load Select*** is set to 3 (14 s average of the current/power). This makes it more responsive to changes in load near empty and to help it converge better to 0%. This helps in cases where the discharge was at a relatively light load during most of the discharge, but the load increases dramatically near the end.

If typical load behavior is consistent throughout the entirety of the battery discharge curve, then the feature can be disabled by setting ***Fast Scaling Load Select*** to the same value as ***Load Select***.

Ra Scaling is reset under the following conditions:

In CHARGE mode when:

Voltage > ***Terminate Voltage*** + ***Term V Delta*** AND SOC > ***Fast Scale Start SOC***

- When negative scale is computed on initialization.
- After OCV measurement when T > ***RaScl Ocv Rst Temp Thresh***.

RaScl Ocv Rst Temp Thresh determines the temperature threshold at which the scaling factor used in FAST RESISTANCE SCALING mode is reset if a new open-circuit voltage measurement is captured.

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
80	IT Cfg	66	Term V Delta	I2	0	4200	200	mV
		100	Max Res Scale	U2	0	32767	5000	Num
		102	Min Res Scale	U2	0	32767	200	Num
		104	Fast Scale Start SOC	U1	0	100	10	%
		105	Fast Scale Load Select	U1	0	6	3	Number
		108	RaScl OCV Rst Temp Thresh	U1	0	127	15	°C

6.8 StateOfCharge() Smoothing

6.8.1 SOC Smoothing in Charge/Discharge

It is common for sudden changes in operating conditions such as temperature and discharge load to cause drastic but legitimate changes in the amount of capacity that can be extracted from a given Li-Ion cell or pack. These changes are typically perceived as jumps in reported SOC and can sometimes be alarming to end-equipment users who may not understand how environmental conditions impact available battery capacity. SOC smoothing solves this by gradually equalizing the difference in reported SOC vs "true" SOC over the present cycle. The method for accomplishing this differs, depending on whether or not the current cycle is a charge or discharge. During discharge, the algorithm adds or removes delta charge (deltaQ) from the present coulomb count to accelerate or decelerate change in *RemainingCapacity()* until it is able to converge to the true value by the time the battery reaches empty. During charge, *FilteredFCC()* is modified to account for deltas in the true and reported versions of Remaining Capacity, as well as the true and reported versions of Full Charge Capacity. Since *FilteredFCC()* is continuously modified to ensure SOC convergence in charge, it is not a real determinant of the total available battery capacity and *UnfilteredFCC()* should instead be referred to for this purpose. The ***[SmoothEn]*** bit in ***Pack Configuration C*** determines whether unfiltered or filtered values are mapped to *RemainingCapacity()*, *FullChargeCapacity()*, and *StateofCharge()* for reporting to the system host, as shown in the table below.

Pack Configuration C [SmoothEn]	*RemainingCapacity()*	*FullChargeCapacity()*	*StateOfCharge()*
0	*UnfilteredRM()*	*UnfilteredFCC()*	*UnfilteredRM()*/*UnfilteredFCC()*
1	*FilteredRM()*	*FilteredFCC()*	*FilteredRM()*/*FilteredFCC()*

6.8.2 SOC Smoothing in Relaxation

In RELAXATION state, temperature changes and applied currents below ***Quit Current*** can still trigger changes in the reported *StateofCharge()*. Similarly, other scenarios can cause differences in true vs reported *RemainingCapacity()* and *FullChargeCapacity()* when entering a battery RELAXATION state. To enable convergence between true and reported SOC values, a similar smoothing algorithm is also supported during cell relaxation.

Smoothing during RELAXATION mode can be configured by the SMSYEN and RlxSMen bits in the Pack Configuration Registers.

Table 6-10. SOC Smoothing in Relaxation Descriptions

SMSYNEN	*RlxSmEn*	Description
0	0	No SOC Smoothing during RELAXATION
0	1	SOC Smoothing enabled during RELAXATION mode, Convergence over time
1	0	Reserved
1	1	SOC Smoothing with enabled during RELAXATION mode, Instant Convergence

The amount of time, the *RelativeStateOfCharge()* is smoothed during RELAXATION is configured using the ***Relax Smooth Time*** parameter.

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
80	IT Cfg	114	Relax Smooth Time	U2	1	65535	1000	s

6.8.3 SOC Smoothing in Overcharge and Overdischarge Conditions

In cases where the cell is in an overcharged or overdischarged state, *FilteredRM()* will begin decrementing or incrementing immediately when a load or charger is applied, respectively. However, this behavior can be overridden to hold *FilteredRM()* at full charge (100%) or empty (0%) until the charge surplus (overcharged state) or charge deficit (overdischarged state) is equalized, at which point it is then allowed to change. The hold at full in overcharged state and hold at empty in overdischarged state options can be enabled via the ***Pack Configuration [RMHOLD100]*** and ***Pack Configuration [RMHOLD0]*** bits, respectively.

6.8.4 StateofCharge() Hold at 99%

The *StateofCharge()* hold at 99% feature prohibits the fuel gauge from reporting 100% until full charge termination is detected and the *Flags() [FC]* bit is set. It is enabled using the ***[RSOCHOLD99]*** bit in Pack Configuration.

6.8.5 StateofCharge() Hold at 1%

The *StateofCharge()* hold at 1% feature prohibits the fuel gauge from reporting 0% until ***Terminate Voltage*** is reached. It is enabled using the ***[RSOCHOLD1]*** bit in Pack Configuration.

6.9 Additional Impedance Track Gauging Features

6.9.1 Trace and Downstream Resistance Compensation

Prediction accuracy for remaining capacity simulations can be further improved in systems with excessive trace lengths between cell and fuel gauge or fuel gauge and system point of load by setting a nominal value in ***Trace Resistance*** or ***Downstream Resistance***, respectively. The fuel gauge adds the ***Trace Resistance*** and

Downstream Resistance to the cell resistance values used in capacity simulations to obtain a more realistic voltage drop under load in simulated discharges when faced with nontrivial trace parasitics in a given pack design. Likewise, trace resistance is removed from any resistance measurements made during discharge prior to storing in data flash.

Trace Resistance is the nominal resistance between the cell and the coulomb counter measurement point in a given application. Flex cabling and long copper traces on the PCB itself can contribute to this resistance and inject error into the SOC prediction. The fuel gauge will offset cell resistance with this value to improve *RemainingCapacity()* estimation.

Downstream Resistance is the nominal resistance between the coulomb counter measurement point and the system voltage node in a given application. Long copper traces on the PCB itself can contribute to this resistance and inject error into the SOC prediction. The fuel gauge will offset cell resistance with this value to improve *RemainingCapacity()* estimation.

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
80	IT Cfg	92	Trace Resistance	I2	0	32767	0	mΩ
		94	Downstream Resistance	I2	0	32767	0	mΩ

6.9.2 Imax Calculation

For systems that require intelligent load throttling at various points of operation, the Imax feature helps to determine how much load can be applied for ***Max Current Pulse Duration*** without causing an instant drop to ***Terminate Voltage***.

Imax() can be enabled by setting ***Pack Configuration D [IMAXEN]***. An interrupt can be triggered on the (SE or HDQ) pin if *Imax()* changes by more than ***Max Current Interrupt Step***.

Reserve Capacity is factored into the *Imax()* calculation if ***Pack Configuration D [IMAXRESRVEN]*** = 1.

Max Allowed Current is the worst-case current pulse that the system expects to impose on the battery for ***Max Current Pulse Duration***. ***Max Current Pulse Duration*** specifies the longest time the ***Max Allowed Current*** is expected to be applied in a given system.

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
80	IT Cfg	109	Max Allowed Current	I2	0	32767	8500	mA
		111	Max Current Pulse Duration	U1	0	255	10	s
		112	Max Current Interrupt Step	I2	–32768	32767	500	mA

6.9.3 Predict Outside Temp Time

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
80	IT Cfg	116	Predict Outside Temp Time	U2	0	65535	2000	s

Predict Outside Temp Time determines the wait time before the algorithm starts to predict the ambient temperature during charge/discharge.

6.9.4 State Subclass

6.9.4.1 Qmax Cell 0

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
82	State	0	Qmax Cell 0	I2	0	14500	1000	mAh

Qmax contains the maximum chemical capacity of the cell profiles, and is determined by comparing states of charge before and after applying the load with the amount of charge passed. They also correspond to capacity at low rate of discharge, such as C/20 rate. For high accuracy, this value is periodically updated by the gauge during operation. Based on the battery cell capacity information, the initial value of the chemical capacity should be entered in Qmax filed. The Impedance Track algorithm updates this value and maintains it.

Before an optimization cycle is run, set this value to the battery cell data sheet capacity. After the optimization cycle is run and for creation of the golden settings, set it to the learned value. The default is 1000 mAh.

6.9.4.2 Update Status

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
82	State	2	Update Status	H1	0x0	0x6	0x0	num

Since this is a pack-side gauge, the ***Update Status*** register can be represented by the bits below:

x	x	x	x	x	Bit 2	Bit 1	Bit 0

Three bits in this register are important:

- Bit 2 (0x04) indicates whether the Impedance Track algorithm is enabled.
- Bit 1 (0x02) indicates that the fuel gauge learned optimized values for Qmax and the Ra tables during a learning cycle.
- Bit 0 (0x01) indicates that the fuel gauge learned an initial value for Qmax after the charging portion of a learning cycle.

6.9.5 OCV Table Class

6.9.5.1 OCVa Table Subclass

6.9.5.1.1 Chemistry Identification

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
83	OCVa Table	0	ChemID	H2	0x00	0xFFFF	0x0354	flags

The ***ChemID*** determines the type of chemistry which is programmed on the gauge. Changing this value by replacing it in data flash has no effect on what is programmed on to the gauge. To obtain a new chemistry you must go through an actual chemistry tool. For the fuel gauge, this can be done using the BQCONFIG tool.

It defaults to 0128 when you program the default flash image, which can be obtained from the Texas Instruments website.

6.9.6 Ra Table Class

This data is automatically updated during device operation. Do not make changes except for reading the values from another pre-learned pack for creating *Golden Image Files*. Profiles have format *Cell0 R_a* M where M is the number indicating the state of charge to which the value corresponds.

Cell0 R_a flag

xCell0 R_a flag

Each subclass (R_a0 and R_a0x) in the Ra Table class is a separate profile of resistance values normalized at 0 degrees for the cell in a design. The cell has two profiles. They are denoted by the x or absence of the x at the end of the subclass title:

R_a0 or **R_a0x**.

The purpose for two profiles for the cell is to ensure that at any given time at least one profile is enabled and is being used while attempts can be made to update the alternate profile without interference. Having two profiles

also helps reduce stress on the flash memory. At the beginning of each of the two subclasses (profiles) is a flag called ***Cell0 R_a flag*** or ***xCell0 R_a flag***. This flag is a status flag that indicates the validity of the table data associated with this flag and whether this particular table is enabled or disabled.

Each flag has two bytes:

1. The least-significant byte (LSB) indicates whether the table is currently enabled or disabled. It has the following options:
   a. 0x00: means the table had a resistance update in the past; however, it is not the currently enabled table for the cell. (The alternate table for the cell must be enabled at this time.)
   b. 0xFF: This means that the values in this table are default values. These table resistance values have never been updated, and this table is not the currently enabled table for the cell. (The alternate table for the indicated cell must be enabled at this time.)
   c. 0x55: This means that this table is enabled for the indicated cell. (The alternate table must be disabled at this time.)
2. The most-significant byte (MSB) indicates the status of the data in this particular table. The possible values for this byte are:
   a. 0x00: The data associated with this flag has a resistance update and the *Qmax Pack* is updated.
   b. 0x05: The resistance data associated with this flag is updated and the pack is no longer discharging (this is prior to a *Qmax Pack* update).
   c. 0x55: The resistance data associated with this flag is updated and the pack is still discharging. (Qmax update attempt not possible until discharging stops.)
   d. 0xFF: The resistance data associated with this flag is all default data.

This data is used by the fuel gauge to determine which tables need updating and which tables are being used for the Impedance Track algorithm.

This data is used by the Impedance Track algorithm. The only reason this data is displayed and accessible is to allow the resistance data on golden image files to be updated. This description of the ***xCell0 R_a flags*** are intended for information purposes only. It is not intended to give a detailed functional description for the resistance algorithms.

Cell0 R_a0 – Cell0 R_a14,

xCell0 R_a0 – xCell0 R_a14,

The ***Ra Table*** class has 15 values for each R_a subclass. Each of these values represents a resistance value normalized at 0°C for the associated *Qmax Pack*-based SOC grid point as found by the following rules:

For ***Cell0 R_aM*** where:

1. If 0 ≤ M ≤ 7: The data is the resistance normalized at 0° for: SOC = 100% – (M × 11.1%).
2. If 8 ≤ M ≤ 14: The data is the resistance normalized at 0° for:
   SOC = 100% – [77.7% + (M – 7) × 3.3%)].

This gives a profile of resistance throughout the entire SOC profile of the battery cells concentrating more on the values closer to 0% where resistance quickly increases.

SOC, as stated in this description is based on *Qmax Pack*. It is not derived as a function of SOC. These resistance profiles are used by the fuel gauge for the Impedance Track algorithm. The only reason this data is displayed and accessible is to allow the resistance data on golden image files to be updated. This resistance profile description is for information purposes only. It is not intended to give a detailed functional description for the resistance algorithms. It is important to note that this data is in mΩ units and is normalized to 25°C. The following are useful observations to note with this data throughout the application development cycle:

- Watch for negative values in the **Ra Table** class. Negative numbers in profiles should never be anywhere in this class.

Watch for smooth consistent transitions from one profile grid point value to the next throughout each profile. As the fuel gauge does resistance profile updates, these values should be roughly consistent from one learned update to another without huge jumps in consecutive grid points.

6.9.6.1 Ra0 Subclass

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
88	Ra0	0	Ra flag	H2	0xFF55	0xFF55	0xFF55	
		2	Ra 0	I2	0	32767	272	2^{-10} Ω
		4	Ra 1	I2	0	32767	316	2^{-10} Ω
		6	Ra 2	I2	0	32767	374	2^{-10} Ω
		8	Ra 3	I2	0	32767	507	2^{-10} Ω
		10	Ra 4	I2	0	32767	360	2^{-10} Ω
		12	Ra 5	I2	0	32767	330	2^{-10} Ω
		14	Ra 6	I2	0	32767	389	2^{-10} Ω
		16	Ra 7	I2	0	32767	345	2^{-10} Ω
		18	Ra 8	I2	0	32767	352	2^{-10} Ω
		20	Ra 9	I2	0	32767	367	2^{-10} Ω
		22	Ra 10	I2	0	32767	374	2^{-10} Ω
		24	Ra 11	I2	0	32767	397	2^{-10} Ω
		26	Ra 12	I2	0	32767	455	2^{-10} Ω
		28	Ra 13	I2	0	32767	808	2^{-10} Ω
		30	Ra 14	I2	0	32767	1182	2^{-10} Ω

6.9.6.2 Ra0x Subclass

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
89	Ra0x	0	xRa flag	H2	0xFFFF	0xFFFF	0xFFFF	
		2	Ra 0	I2	0	32767	272	2^{-10} Ω
		4	Ra 1	I2	0	32767	316	2^{-10} Ω
		6	Ra 2	I2	0	32767	374	2^{-10} Ω
		8	Ra 3	I2	0	32767	507	2^{-10} Ω
		10	Ra 4	I2	0	32767	360	2^{-10} Ω
		12	Ra 5	I2	0	32767	330	2^{-10} Ω
		14	Ra 6	I2	0	32767	389	2^{-10} Ω
		16	Ra 7	I2	0	32767	345	2^{-10} Ω
		18	Ra 8	I2	0	32767	352	2^{-10} Ω
		20	Ra 9	I2	0	32767	367	2^{-10} Ω
		22	Ra 10	I2	0	32767	374	2^{-10} Ω
		24	Ra 11	I2	0	32767	397	2^{-10} Ω
		26	Ra 12	I2	0	32767	455	2^{-10} Ω
		28	Ra 13	I2	0	32767	808	2^{-10} Ω
		30	Ra 14	I2	0	32767	1182	2^{-10} Ω

Chapter 7

Charging Features

7.1 Introduction

The BQ27542-G1 device contains various charging features that help the host to better control a charger. Many status flags are generated to indicate fault conditions that the host can use to control the charger operation effectively. The gauge also provides support for the JEITA charging algorithm.

7.2 Charge Suspend

If *Temperature()* < T1 Temp or > T5 Temp during active charging, a charge suspend condition is indicated by setting the *Flags()[CHG_SUS]* bit to 1 and clearing *ChargingCurrent()* and *ChargingVoltage()* to 0.

7.3 Charge Inhibit

If *Temperature()* < T1 Temp or > T4 Temp without active charging, a charge inhibit condition is indicated by setting the *Flags()[CHG_INH]* bit to 1 and clearing *ChargingCurrent()* and *ChargingVoltage()* to 0.

7.4 JEITA Charging Profile

The fuel gauge provides full support for the JEITA charging algorithm, which employs separate constant-current constant-voltage (CCCV) charging parameters, depending on the measured*Temperature()*. The allowable charging range is divided into four regions defined by ***T1 Temp***, ***T2 Temp***, ***T3 Temp***, ***T4 Temp***, and ***T5 Temp***, each with its own dedicated *ChargingCurrent()* and *ChargingVoltage()* values.

- If *Temperature()* < ***T1 Temp***, *ChargingCurrent()* and *ChargingVoltage()* are set to 0.
- If ***T1 Temp*** ≤*Temperature()* ≤ ***T2 Temp***, ***T1-T2 Chg Current*** and ***T1-T2 Chg Voltage*** are reported.
- If ***T2 Temp*** <*Temperature()* ≤ ***T3 Temp***, ***T2-T3 Chg Current*** and ***T2-T3 Chg Voltage*** are reported.
- If ***T3 Temp*** <*Temperature()* ≤ ***T4 Temp***, ***T3-T4 Chg Current*** and ***T3-T4 Chg Voltage*** are reported.
- If ***T4 Temp*** <*Temperature()* ≤ ***T5 Temp***, ***T4-T5 Chg Current*** and ***T4-T5 Chg Voltage*** are reported.
- If *Temperature()* > ***T5 Temp***, *ChargingCurrent()* and *ChargingVoltage()* are set to 0.

Figure 7-1, *JEITA Charging Current Profile*, and Figure 7-2, *JEITA Charging Voltage Profile*, provide a visual depiction of the JEITA charging algorithm.

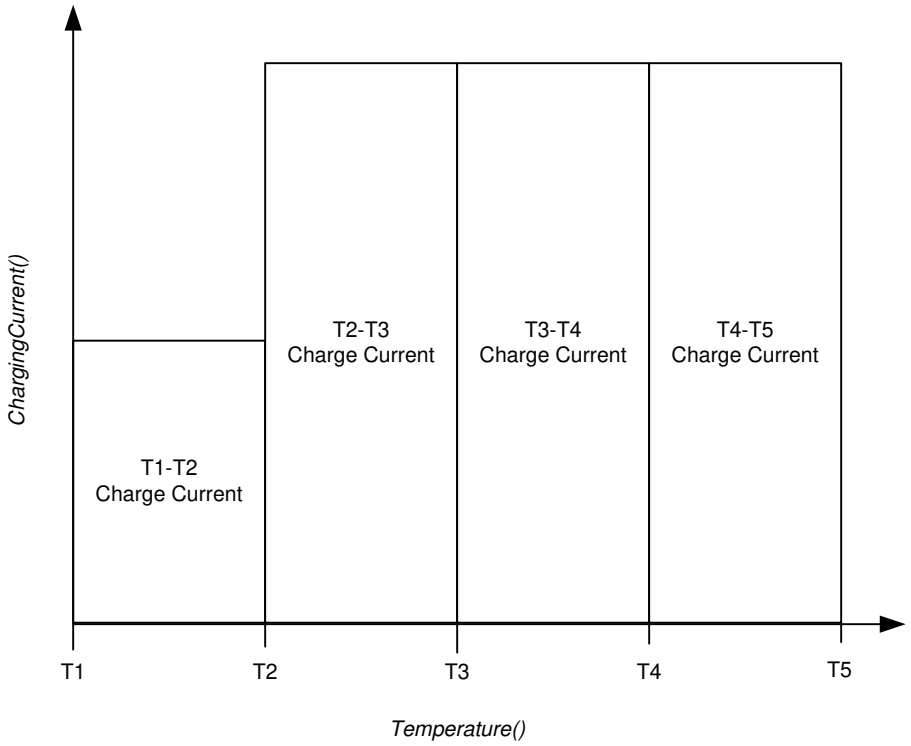


Figure 7-1. JEITA Charging Current Profile

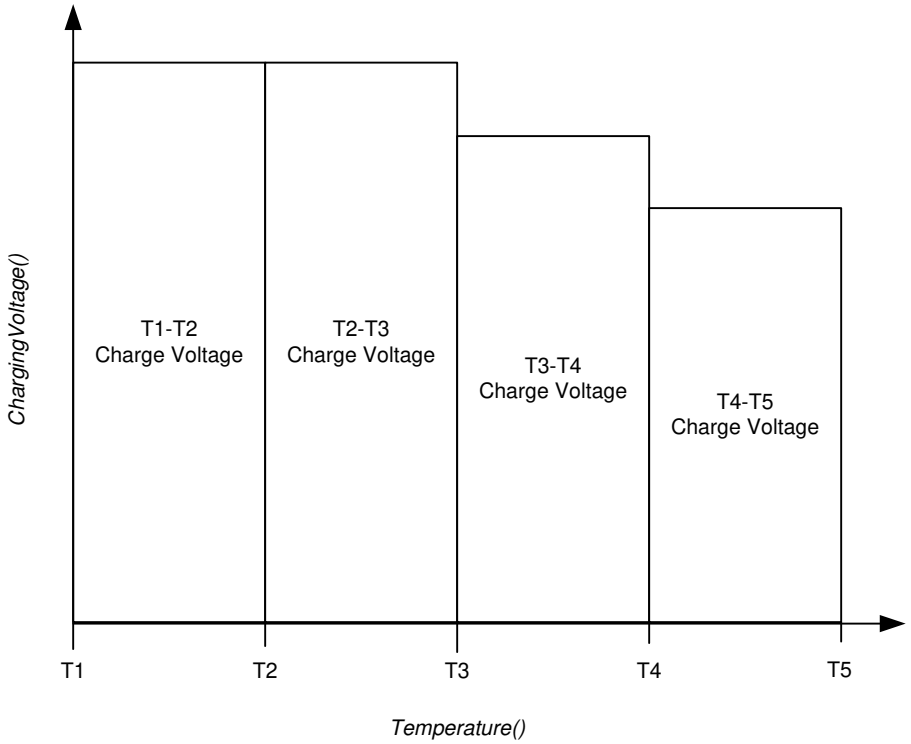


Figure 7-2. JEITA Charging Voltage Profile

Temperature hysteresis (***Temp Hys***) is also applied to movement between various ranges to prevent charging parameter oscillation when *Temperature()* continuously changes by a few degrees right on the edge of a temperature boundary. When moving from cooler to warmer temperatures, positive hysteresis is applied to the ***T1 Temp*** and ***T2 Temp*** thresholds. On the contrary, when moving from warmer to cooler temperatures, negative

hysteresis is applied to the ***T3 Temp***, ***T4 Temp***, and ***T5 Temp*** thresholds. To convert the four-range JEITA profile to a classic, notebook-style three-range version, simply set T4 Temp = T3 Temp.

Figure 7-3, *Temperature Hysteresis for Charging Current*, and Figure 7-4, *Temperature Hysteresis for Charging Voltage*, illustrate how temperature hysteresis is applied depending on transition direction.

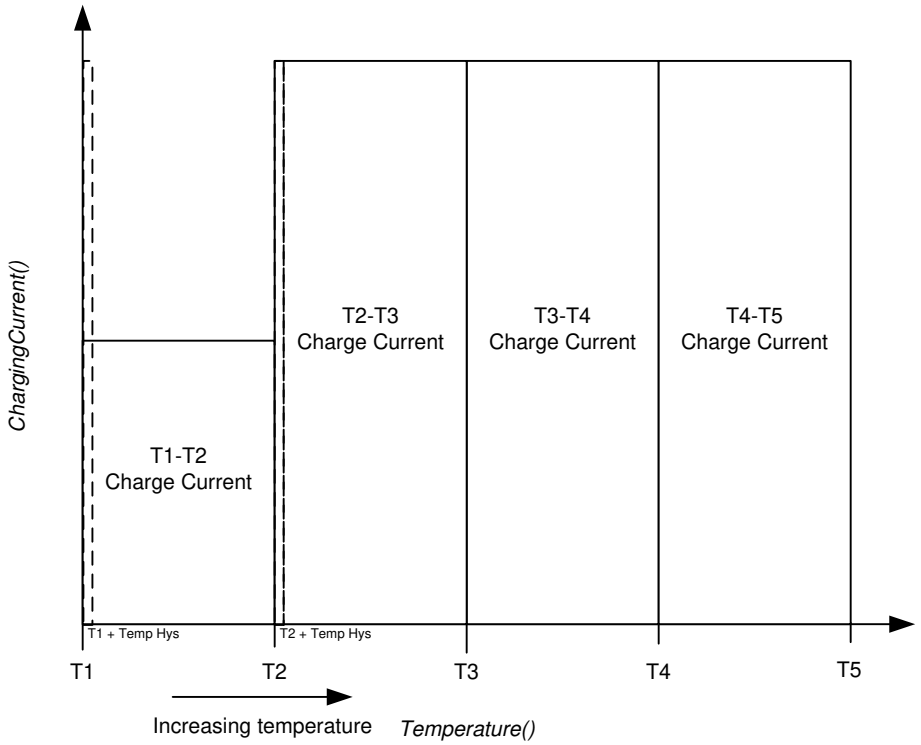


Figure 7-3. Temperature Hysteresis for Charging Current

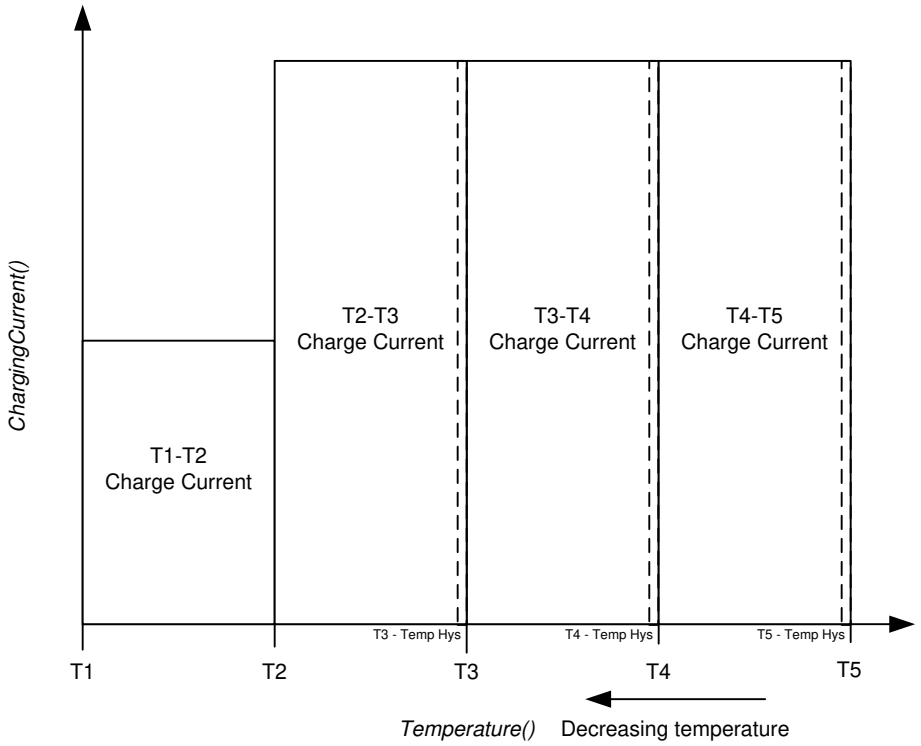


Figure 7-4. Temperature Hysteresis for Charging Voltage

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
39	JEITA	0	T1 Temp	I1	–128	127	0	°C
		1	T2 Temp	I1	–128	127	10	°C
		2	T3 Temp	I1	–128	127	45	°C
		3	T4 Temp	I1	–128	127	50	°C
		4	T5 Temp	I1	–128	127	60	°C
		5	Temp Hys	I1	–128	127	1	°C
		6	T1-T2 Chg Voltage	I2	0	4600	4350	mV
		8	T2-T3 Chg Voltage	I2	0	4600	4350	mV
		10	T3-T4 Chg Voltage	I2	0	4600	4300	mV
		12	T4-T5 Chg Voltage	I2	0	4600	4250	mV
		14	T1-T2 Chg Current	U1	0	100	50	Percent
		15	T2-T3 Chg Current	U1	0	100	80	Percent
		16	T3-T4 Chg Current	U1	0	100	80	Percent
		17	T4-T5 Chg Current	U1	0	100	80	Percent

7.5 Full Charge Termination Detection

Full charge termination is detected on the basis of voltage-, current-, and capacity-based conditions or SOC level, depending on the setting configured in ***FC Set %***.

Table 7-1. Full Charge Termination Detection

FC Set %	Termination Criteria
–1	FC bit is set based on charge termination detection with current/voltage.
Any other value	FC bit is set based on *StateofCharge()*.

1. *Voltage()* ≥ ***Charging Voltage – Taper Voltage*** AND
2. During two consecutive periods of ***Current Taper Window***, the *AverageCurrent()* is < ***Taper Current*** AND
3. During two consecutive periods of ***Current Taper Window***, the accumulated change in capacity > 0.25 mAh.

AverageCurrent() is not used for the qualification because its time constant is not the same as the ***Current Taper Window***. Two current qualifications are done to prevent false current taper qualifications. False primary charge terminations happen with pulse charging and with random starting and stopping of the charge current. This is particularly critical at the beginning or end of the qualification period.

Note

It is important to note that as the ***Current Taper Window*** value is increased, the current range in the second requirement for primary charge termination is lowered. If the ***Current Taper Window*** is increased, then the current used to integrate to the ***Min Taper Capacity*** is decreased and this threshold becomes more sensitive.

Once full charge termination conditions are met, the *Flags()[FC]* bit is set to indicate charge termination to the host. Additionally, if ***Pack Configuration [RMFCC]*** = 1, then *RemainingCapacity()* is set equal to *FullChargeCapacity()* upon full charge termination. The fuel gauge exits charge termination and associated flags are cleared when SOC decreases below ***FC Clear %***. ***FC Clear %*** sets a *StateOfCharge()* percentage threshold at which the *Flags() [FC]* bit is cleared.

The gauge also records voltage at charge termination and stores it in ***V at Chg Term***. It is used by the gauge to learn the depth of discharge (DoD) of a full battery for a given system. This is updated by the gauge after every charge termination to account for variations between systems and different temperatures.

V at Chg Term defaults to 4200 mV but can be initialized to the nominal charging voltage of the system.

Note

FC Set % and ***FC Clear %*** only affects the *Flags() [FC]* bit, which does not affect the charge termination process.

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
34	Charge	0	Charging Voltage	I2	4000	5000	4350	mV
36	Charge Termination	0	Taper Current	I2	0	1000	100	mA
		2	Min Taper Capacity	I2	0	1000	25	mAh
		4	Taper Voltage	I2	0	1000	100	mV
		6	Current Taper Window	U1	0	60	40	s
		9	FC Set %	I1	–1	100	–1	Percent
		10	FC Clear %	I1	–1	100	98	Percent
82	State	3	V at Chg Term	I2	0	5000	4350	mV

The CHG bit in the Flags register is used to indicate when charging is complete.

[CHG] bit is cleared:

- At taper termination if ***TCA Set %*** is –1.
- When *StateOfCharge()* ≥ ***TCA Set %*** and if ***TCA Set %*** is not –1.
- If *Flags() [OTC]* or *[CHG_INH]* is set.

[CHG] bit is set:

- When any of the conditions for *[CHG]* bit to be cleared does not exist and *StateOfCharge()* ≤ ***TCA Clear %***.

Note

TCA Set % and ***TCA Clear %*** only affect the *Flags() [CHG]* bit, but does not affect the charge termination process or the gauging function.

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
36	Charge Termination	7	TCA Set %	I1	–1	100	–1	Percent
		8	TCA Clear %	I1	–1	100	98	Percent

7.6 Pulse Loads

To handle pulse load conditions, the gauge uses additional parameters. ***Delta Voltage*** is the maximum difference of *Voltage()* during short load spikes and normal load, so the Impedance Track algorithm can calculate remaining capacity for pulse loads. The ***Delta Voltage*** value is automatically updated by the gauge during operation as voltage spikes are detected. It can be initialized to a higher value if large spikes are typical for the system. Allowable values are limited by ***Max DeltaV*** and ***Min DeltaV***.

Max DeltaV is the maximum ***Delta Voltage*** that is saved during discharge and ***Min DeltaV*** is the minimum ***Delta Voltage*** that is saved during discharge.

During the IT simulations, the target voltage of the empty battery is (***Terminate Voltage*** + ***Delta Voltage***). This feature allows ***Terminate Voltage*** to be set at the minimum operating voltage of the system with assurance that the 0% point will be reached at a sufficiently high voltage to prevent voltage spikes from crashing the system, while still extracting maximum run time from the battery when spikes are small.

The ***DeltaV Max Delta*** variable provides a limit on how far ***Delta Voltage*** grows or shrinks on one grid update (in mV).

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
80	IT Cfg	84	Max DeltaV	I2	0	32767	200	mV
		86	Min DeltaV	I2	0	32767	0	mV
		98	DeltaV Max Delta	U2	0	65535	10	mV
82	State	9	Delta Voltage	I2	0	32767	2	mV

7.7 Terminate Voltage Valid Time

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
80	IT Cfg	118	TermV Valid t	U1	0	255	2	s

The voltage must dip below Terminate Voltage for at least this many seconds before *RemainingCapacity()* and *StateOfCharge()* will be forced to 0.

7.8 Charge Termination Subclass

7.8.1 DOD at EOC Delta Temperature

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
36	Charge Termination	11	DODatEOC Delta T	I2	0	1000	50	0.1°C

DODatEOC Delta T represents the temperature change threshold to update Q_{start} and *RemainingCapacity()* due to temperature changes. During relaxation and at the start of charging, the remaining capacity is calculated as *RemainingCapacity()* = *FullChargeCapacity()* – Q_{start}. As temperature decreases, Q_{start} can become much smaller than that of the old *FullChargeCapacity()* value, resulting in overestimation of *RemainingCapacity()*. To improve accuracy, *FullChargeCapacity()* is updated whenever the temperature change since the last *FullChargeCapacity()* update is greater than ***DODatEOC Delta T*** × 0.1°C.

The default value is 50.

Note

The units are a tenth of a °C, which means a value of 50 corresponds to 5°C.

This page intentionally left blank.

Chapter 8

Lifetime Data Logging Features

8.1 Introduction

The Lifetime Data logging function helps with the development and diagnosis of the fuel gauge.

8.2 Lifetime Data Logging Parameters

The *IT_ENABLE* subcommand needs to be enabled (command 0x0021) for lifetime data logging functions to be active. The fuel gauge logs the lifetime data as specified in the ***Lifetime Data*** and ***Lifetime Temp Samples*** data flash subclasses. The data log recordings are controlled by the ***Lifetime Resolution*** data flash subclass.

The Lifetime Data Logging can be started by setting the *IT_ENABLE* subcommand and setting the Update Time register to a non-zero value.

Once the Lifetime Data Logging function is enabled, the measured values are compared to what is already stored in the data flash. If the measured value is higher than the maximum or lower than the minimum value stored in the data flash by more than the Resolution set for at least one parameter, the entire ***Data Flash Lifetime*** Registers are updated after at least ***LTUpdateTime***.

LTUpdateTime sets the minimum update time between DF writes. When a new maximum or minimum is detected, an LT Update window of [update time] seconds is enabled and the DF writes occur at the end of this window. Any additional maximum or minimum value detected within this window is also updated. The first new maximum or minimum value detected after this window triggers the next LT Update window.

Internal to the fuel gauge, there exists a RAM maximum/minimum table in addition to the DF maximum/minimum table. The RAM table is updated independent of the resolution parameters. The DF table is updated only if at least one of the RAM parameters exceeds the DF value by more than resolution associated with it. When DF is updated, the entire RAM table is written to DF. Consequently, it is possible to see a new maximum/minimum value for a certain parameter even if the value of this parameter never exceeds the maximum or minimum value stored in the data flash for this parameter value by the resolution amount.

The Life Time Data Logging of one or more parameters can be reset or restarted by writing new default (or starting) values to the corresponding data flash registers through SEALED or UNSEALED access as described below. However, when using UNSEALED access, new values take effect only if the device is reset within ***LT Update Time*** after the DF is loaded with new values.

The logged data in ***Lifetime Data*** subclass (subclass ID = 59) can be read and written in both SEALED and UNSEALED modes. However, in SEALED mode, access to this subclass is using a process identical to accessing ***Manufacturer Info Block B***. The *DataFlashBlock()* command code is 4. See Section 11.1, *Manufacturer Information Blocks*, for details of this sequence.

The subclasses ***Lifetime Resolution*** (subclass ID = 66) and ***Lifetime Temp Samples*** (subclass ID = 59) that contain settings for lifetime data logging can be configured only in UNSEALED mode using the regular DF access method.

The Lifetime resolution registers contain the parameters that set the limits related to how much a data parameter must exceed the previously logged maximum/minimum value to be updated in the lifetime log. For example, V must exceed MaxV by more than Voltage Resolution to update MaxV in the data flash.

8.3 Feature Access

8.4 Lifetime Data Subclass, Lifetime Resolution Subclass

The ***Lifetime Data*** subclass contains black box data that records various data over the life of the pack. This data can be very useful for performing failure analysis on the returned packs. ***Lifetime Data*** is enabled if the *CONTROL_STATUS [QEN]* bit is 1. The *[QEN]* bit is set by sending the *IT_ENABLE* subcommand.

The lifetime update for the values below is throttled to not happen more than once per 60 seconds to avoid data flash wear out. The frequency of the updates will naturally slow down once the pack updates the minimum and maximum values over several packs:

- ***Lifetime Max Temp***: Maximum temperature observed by the gauge. It is initialized to 300. The unit is 0.1°C.
- ***Lifetime Min Temp***: Minimum temperature observed by the gauge. It is initialized to 200. The unit is 0.1°C.
- ***Lifetime Max Pack Voltage***: Maximum battery voltage observed by the gauge. It is initialized to 3200. The unit is mV.
- ***Lifetime Min Pack Voltage***: Minimum battery voltage observed by the gauge. It is initialized to 4200. The unit is mV.
- ***Lifetime Max Chg Current***: Maximum charge current observed by the gauge. It is initialized to 0. The unit is mA.
- ***Lifetime Max Dsg Current***: Maximum discharge current observed by the gauge. It is initialized to 0. The unit is mA.
- ***LT Flash Cnt***: Lifetime flash page update counter keeps track of total number of updates. It is initialized to 0. The unit is counts.

8.4.1 Maximum Temperature, Minimum Temperature, Temperature Resolution

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
59	Lifetime Data	0	Lifetime Max Temp	I2	–600	1400	0	0.1°C
		2	Lifetime Min Temp	I2	–600	1400	500	0.1°C
66	Lifetime Resolution	0	LT Temp Res	U1	0	255	10	Num

The ***Lifetime Max Temp*** value is updated if one of the following conditions is met:

- *Temperature()* – ***Lifetime Max Temp*** > ***LT Temp Res***
- *Temperature()* > ***Lifetime Max Temp*** and any other lifetime value is updated.

The ***Lifetime Min Temp*** value is updated if one of the following conditions is met:

- ***Lifetime Min Temp*** – *Temperature()* > ***LT Temp Res***
- *Temperature()* < ***Lifetime Min Temp*** and any other lifetime value is updated.

8.4.2 Maximum Pack Voltage, Minimum Pack Voltage, Voltage Resolution

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
59	Lifetime Data	4	Lifetime Max Pack Voltage	I2	0	32767	2800	mV
		6	Lifetime Min Pack Voltage	I2	0	32767	5000	mV
66	Lifetime Resolution	1	LT V Res	U1	0	255	25	Num

The ***Lifetime Max Pack Voltage*** value is updated if one of the following conditions is met:

- *Voltage()* – ***Lifetime Max Pack Voltage*** > ***LT V Res***
- *Voltage()* > ***Lifetime Max Pack Voltage*** and any other lifetime value is updated.

The ***Lifetime Min Pack Voltage*** value is updated if one of the following conditions is met:

- ***Lifetime Min Pack Voltage*** – *Voltage()* > ***LT V Res***
- *Voltage()* < ***Lifetime Min Pack Voltage*** and any other lifetime value is updated.

8.4.3 Maximum Charge Current, Maximum Discharge Current, Current Resolution

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
59	Lifetime Data	8	Lifetime Max Chg Current	I2	–32767	32767	0	mA
		10	Lifetime Max Dsg Current	I2	–32767	32767	0	mA
66	Lifetime Resolution	2	LT Cur Res	U1	0	255	100	Num

The ***Lifetime Max Chg Current*** value is updated if one of the following conditions is met:

- *Current()* – ***Lifetime Max Chg Current*** > ***LT Cur Res***
- *Current()* > ***Lifetime Max Chg Current*** and any other lifetime value is updated.

The ***Lifetime Max Dsg Current*** value is updated if one of the following conditions is met:

- ***Lifetime Max Dsg Current*** – *Current()* > ***LT Cur Res***
- ***Lifetime Max Dsg Current*** > *Current()* and any other lifetime value is updated.

Note

During discharge, current is negative.

8.5 Lifetime Resolution Subclass

8.5.1 Lifetime Update Time

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
66	Lifetime Resolution	3	LT Update Time	U2	0	65535	60	Num

This parameter sets the minimum time between data flash writes to update the Lifetime Parameters. The default for this register is 60.

8.6 Lifetime Temp Samples Subclass

8.6.1 Flash Write Count

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
59	Lifetime Temp Samples	12	LT Flash Cnt	U2	0	32767	0	Count

LT Flash Cnt tracks the number of lifetime data flash updates.

This page intentionally left blank.

Chapter 9

Authentication

9.1 Introduction

The fuel gauge supports a SHA-1-based authentication protocol that allows a host to securely verify battery pack authenticity. Sending a 160-bit random challenge initiates the authentication process wherein the fuel gauge computes a response digest using a double SHA-1 transform. The transmitted challenge is appended to a secret 128-bit authentication key and run through the transform. Afterwards, the resulting hash is then re-appended to the same key and a second hash is computed, resulting in the final 160-bit digest that is returned to the host. The host reproduces the same digest calculation on its side, using the shared key, and compares to the one read from the fuel gauge. If they match, the authentication process is successful.

9.2 Key Programming (Data Flash Key)

By default, the fuel gauge contains a default plain-text authentication key of 0x0123456789ABCDEFFEDCBA9876543210. This default key is intended for development purposes. It must be changed to a secret key and the part immediately SEALED, before putting a pack into operation. Once written, a new plain-text key cannot be read again from the fuel gauge while in SEALED mode.

To change the pre-programmed authentication key from its default value, follow the steps below:

1. Unseal the fuel gauge.
2. Send the *Authenticate()* command.
3. Write 0x00 to *BlockDataControl()* to enable the authentication data commands.
4. Send the *DataFlashClass()* command with 112 (0x70) to set the ***Security*** class.
5. Up to 32 bytes of data can be read directly from the *BlockData()* (0x40 through 0x5F) and the authentication key is located at 0x48 (0x40 + 0x08 offset) to 0x57 (0x40 + 0x17 offset).
6. Write the new authentication key to the corresponding locations (0x48 through 0x57) using the *BlockData()* command.
7. Compute the correct checksum for the whole block (0x40 to 0x5F). The checksum is (255 – x) where x is the 8-bit summation of the *BlockData()* (0x40 through 0x5F) on a byte-by-byte basis.
8. Send the *BlockDataChecksum()* (0x60) with the computed checksum from the previous step.
9. Seal the fuel gauge.

9.3 Key Programming (Secure Memory Key)

As the name suggests, the secure-memory authentication key is stored in the secure memory of the fuel gauge. If a secure-memory key has been established, that key can be used for authentication challenges ONLY if the data flash key is all 0s.

For example, ***Authen Key3*** = ***Authen Key2*** = ***Authen Key1*** = ***Authen Key0*** = 0x00000000.

9.4 Executing an Authentication Query

To execute an authentication query in UNSEALED mode, follow the steps below:

1. Host must first write 0x01 to the *BlockDataControl()* command to enable the authentication data commands. If in SEALED mode, 0x00 must be written to *DataFlashBlock()*.
2. Host writes a 20-byte authentication challenge to the *Authenticate()* address locations (0x40 through 0x53).
3. Host writes a valid checksum for the challenge to *AuthenticateChecksum()*.

The fuel gauge uses the challenge, in conjunction with the programmed authentication key, in its SHA-1 computations. After completion, the resulting digest is stored in the *Authenticate()* register, overwriting the preexisting challenge. The host must wait at least 45 ms to read the resulting digest. The host may then read this response and compare it against the result created by its own parallel computation.

9.5 Codes Subclass

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
112	Codes	0	Sealed to Unsealed	H4	0x00	0xFFFF FFFF	0x3672 0414	
		4	Unsealed to Full	H4	0x00	0xFFFF FFFF	0xFFFF FFFF	
		8	Authen Key3	H4	0x00	0xFFFF FFFF	0x0123 4567	
		12	Authen Key2	H4	0x00	0xFFFF FFFF	0x89AB CDEF	
		16	Authen Key1	H4	0x00	0xFFFF FFFF	0xFEDC BA98	
		20	Authen Key0	H4	0x00	0xFFFF FFFF	0x7654 3210	

9.5.1 Sealed to Unsealed

This register contains the security code to transition the device from SEALED mode to UNSEALED mode. The default code is set to 0x36720414.

9.5.2 Unsealed to Full Access

This register contains the security code to transition the device from UNSEALED mode to FULL ACCESS mode. The default code is set to 0xFFFFFFFF.

9.5.3 Authentication Keys

In this register, store the SHA-1 authentication key to allow a system to authenticate the battery pack. The default key is set to 0x0123456789ABCDEFFEDCBA9876543210.

Chapter 10
Communications

10.1 HDQ Single-Pin Serial Interface

The HDQ interface is an asynchronous return-to-one protocol where a processor sends the command code to the fuel gauge. With HDQ, the least significant bit (LSB) of a data byte (command) or word (data) is transmitted first. Note that the DATA signal on pin 12 is open-drain and requires an external pullup resistor. The 8-bit command code consists of two fields: the 7-bit HDQ command code (bits 0:6) and the 1-bit RW field (MSB bit 7). The RW field directs the fuel gauge either to:

- Store the next 8 or 16 bits of data to a specified register, or
- Output 8 bits of data from the specified register.

The HDQ peripheral can transmit and receive data as either an HDQ master or slave.

HDQ serial communication is normally initiated by the host processor sending a break command to the fuel gauge. A break is detected when the DATA pin is driven to a logic-low state for a time $t_{(B)}$ or greater. The DATA pin should then be returned to its normal ready high logic state for a time $t_{(BR)}$. The fuel gauge is now ready to receive information from the host processor.

The fuel gauge is shipped in the I^2C mode. TI provides tools to enable the HDQ peripheral. The *HDQ Communication Basics Application Report* (SLUA408) provides details of HDQ communication basics, including an alternative method to use a standard two-wire UART for single-wire HDQ communication.

If users develop their own tools to program the DFI and need to set the device to HDQ mode, the following steps are required. After writing the DFI but before sending the commands to exit ROM mode, send the following commands:

1. I^2C Command 0x00: Byte 0x16
2. I^2C Command 0x04: Byte 0x05
3. I^2C Command 0x64: Byte 0x1B
4. I^2C Command 0x65: Byte 0x00

10.2 HDQ Host Interruption Feature

The default fuel gauge behaves as an HDQ slave-only device when HDQ mode is enabled. If the HDQ interrupt function is enabled, the fuel gauge is capable of mastering and also communicating to an HDQ device. There is no mechanism for negotiating what is to function as the HDQ master and care must be taken to avoid message collisions. The interrupt is signaled to the host processor with the fuel gauge mastering an HDQ message. This message is a fixed message that will be used to signal the interrupt condition. The message itself is 0x80 (slave write to register 0x00) with no data byte being sent as the command is not intended to convey any status of the interrupt condition. The HDQ interrupt function is disabled by default and needs to be enabled by command and ***Pack Configuration [HOST_IE]*** should be set to 1.

When the SET_HDQINTEN subcommand is received, the fuel gauge will detect any of the interrupt conditions as specified in Table 5-5 and assert the interrupt at 1-second intervals until the CLEAR_HDQINTEN command is received or the count of ***HDQHostIntrTries*** has lapsed.

The number of tries for interrupting the host is determined by the data flash parameter named ***HDQHostIntrTries***.

10.2.1 Low Battery Capacity

This feature will work identically to SOC1. It will use the same data flash entries as SOC1 and will trigger interrupts as long as SOC1 = 1 and *HDQIntEN* = 1.

10.2.2 Temperature

This feature will trigger an interrupt based on the OTC (Over-Temperature in Charge) or OTD (Over-Temperature in Discharge) condition being met. It uses the same data flash entries as OTC or OTD and will trigger interrupts as long as either the OTD or OTC condition is met and *HDQIntEN* = 1.

10.3 I²C Interface

The fuel gauge supports the standard I²C read, incremental read, one-byte write quick read, and functions. The 7-bit device address (ADDR) is the most significant 7 bits of the hex address and is fixed as 1010101, or 0x55. The 8-bit device address is therefore 0xAA or 0xAB for write or read, respectively.

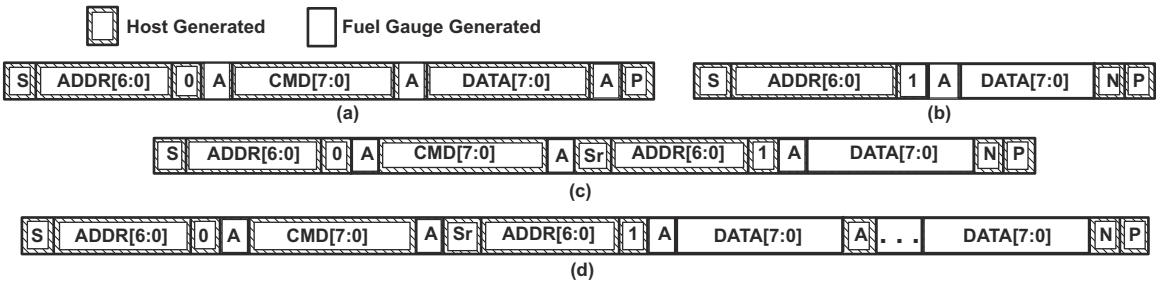


Figure 10-1. Supported I²C Formats

The quick read returns data at the address indicated by the address pointer. The address pointer, a register internal to the I²C communication engine, increments whenever data is acknowledged by the fuel gauge or the I²C master. Quick writes function in the same manner and are a convenient means of sending multiple bytes to consecutive command locations (such as two-byte commands that require two bytes of data).

Attempt to write a read-only address (NACK after data sent by master):

Attempt to read an address above 0x7F (NACK command):

Attempt at incremental writes (NACK all extra data bytes sent):

Incremental read at the maximum allowed read address:

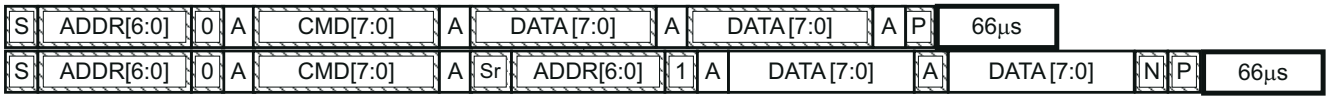
The I²C engine releases both SDA and SCL if the I²C bus is held low for $t_{(BUSERR)}$. If the fuel gauge was holding the lines, releasing them frees the master to drive the lines. If an external condition is holding either of the lines low, the I²C engine enters the low-power SLEEP mode.

10.3.1 I²C Time Out

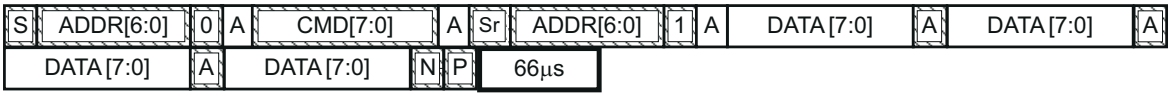
The I²C engine releases both SDA and SCL if the I²C bus is held low for about 2 seconds. If the fuel gauge was holding the lines, releasing them frees the master to drive the lines.

10.3.2 I²C Command Waiting Time

To make sure the correct results of a command with the 400-kHz I²C operation, a proper waiting time must be added between issuing command and reading results. For subcommands, the following diagram shows the waiting time required between issuing the control command the reading the status with the exception of the checksum command. A 100-ms waiting time is required between the checksum command and reading result. For read-write standard commands, a minimum of 2 seconds is required to get the result updated. For read-only standard commands, there is no waiting time required, but the host must not issue all standard commands more than two times per second. Otherwise, the gauge could result in a reset issue due to the expiration of the watchdog timer.



Waiting time between control subcommand and reading results



Waiting time between continuous reading results

10.3.3 I²C Clock Stretching

I²C clock stretches can occur during all modes of fuel gauge operation. In the SLEEP mode, a short clock stretch occurs on all I²C traffic as the device must wake-up to process the packet. In NORMAL and SLEEP modes, clock stretching only occurs for packets addressed for the fuel gauge. The timing of stretches varies as interactions between the communicating host and the gauge are asynchronous. The I²C clock stretches may occur after start bits, the ACK/NAK bit and first data bit transmit on a host read cycle. The majority of clock stretch periods are small (≤ 4 ms) as the I²C interface peripheral and CPU firmware perform normal data flow control. However, less frequent but more significant clock stretch periods may occur when data flash (DF) is being written by the CPU to update the resistance (Ra) tables and other DF parameters such as Qmax. Due to the organization of DF, updates need to be written in data blocks consisting of multiple data bytes.

An Ra table update requires erasing a single page of DF, programming the updated Ra table and a flag. The potential I²C clock stretching time is 24-ms maximum. This includes 20-ms page erase and 2-ms row programming time (×2 rows). The Ra table updates occur during the discharge cycle and at up to 15 resistance grid points that occur during the discharge cycle.

A DF block write typically requires a maximum of 72 ms. This includes copying data to a temporary buffer and updating DF. This temporary buffer mechanism protects from a power failure during a DF update. The first part of the update requires 20 ms to erase the copy buffer page, 6 ms to write the data into the copy buffer and the program progress indicator (2 ms for each individual write). The second part of the update is writing to the DF and requires 44-ms DF block update time. This includes a 20-ms each page erase for two pages and 2-ms each row write for two rows.

In the event that a previous DF write was interrupted by a power failure or reset during the DF write, an additional 44-ms maximum DF restore time is required to recover the data from a previously interrupted DF write. In this power failure recovery case, the total I²C clock stretching is 116-ms maximum.

Another case where I²C clock stretches is at the end of discharge. The update to the last discharge data goes through the DF block update twice because two pages are used for the data storage. The clock stretching in this case is 144-ms maximum. This occurs if there has been an Ra table update during the discharge.

This page intentionally left blank.

Chapter 11

Manufacturer Information

11.1 Manufacturer Information Blocks

The fuel gauge contains 64 bytes of user-programmable data flash storage: ***Manufacturer Info Block A*** and ***Manufacturer Info Block B***. The method for accessing these memory locations is slightly different, depending on whether the device is in UNSEALED or SEALED modes.

When in UNSEALED mode and when 0x00 has been written to *BlockDataControl()*, accessing the Manufacturer Info Blocks is identical to accessing general data flash locations. First, a *DataFlashClass()* command sets the subclass, then a *DataFlashBlock()* command sets the offset for the first data flash address within the subclass. The *BlockData()* command codes contain the referenced data flash data. When writing the data flash, a checksum is expected to be received by *BlockDataChecksum()*. Only when the checksum is received and verified is the data actually written to data flash.

As an example, the data flash location for ***Manufacturer Info Block B*** is defined as having a Subclass = 58 and an Offset = 32 through 63 (32-byte block). The specification of Class = System Data is not needed to address ***Manufacturer Info Block B***, but is used instead for grouping purposes when viewing data flash info in the evaluation software.

When in SEALED mode or when *BlockDataControl()* does not contain 0x00, data flash is no longer available in the manner used in UNSEALED mode. Rather than issuing subclass information, a designated Manufacturer Information Block is selected with the *DataFlashBlock()* command. Issuing a 0x01 or 0x02 with this command causes the corresponding information block (A or B, respectively) to be transferred to the command space 0x40 through 0x5F for editing or reading by the system. Upon successful writing of checksum information to *BlockDataChecksum()*, the modified block is returned to data flash.

11.2 Manufacturer Information Subclass

11.2.1 Block A and Block B

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
58	Manufacturer Info	0 through 31	Block A 0 through 31	H1	0x00	0xFF	0x00	
		32 through 63	Block B 0 through 31	H1	0x00	0xFF	0x00	

Note

Manufacturer Info Block A is read-only when in SEALED mode.

This page intentionally left blank.

12.1 Introduction

The gauge provides data flash areas to store ***Manufacturer Data*** parameters relevant to the system/board in use.

12.2 Manufacturer Data Subclass

12.2.1 Pack Lot Code

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
56	Manufacturer Data	0	Pack Lot Code	H2	0x00	0xFFFF	0x00	hex

The pack manufacturer can use this location to store the pack lot code.

12.2.2 PCB Lot Code

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
56	Manufacturer Data	2	PCB Lot Code	H2	0x00	0xFFFF	0x00	hex

The pack manufacturer can use this location to store the PCB lot code.

12.2.3 Firmware Version

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
56	Manufacturer Data	4	Firmware Version	H2	0x00	0xFFFF	0x00	hex

The pack manufacturer can use this location to store a firmware version number for their system or pack. This value is user-defined and is not related to the gauge's *Control(FW_VERSION)*.

12.2.4 Hardware Revision

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
56	Manufacturer Data	6	Hardware Revision	H2	0x00	0xFFFF	0x00	hex

The pack manufacturer can use this location to store a hardware version number for their system or pack. This value is user-defined and is not related to the gauge's *Control(HW_VERSION)*.

12.2.5 Cell Revision

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
56	Manufacturer Data	8	Cell Revision	H2	0x00	0xFFFF	0x00	hex

The pack manufacturer can use this location to store the version of their cell.

12.2.6 Data Flash Configuration Version

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
56	Manufacturer Data	10	DF Config Version	H2	0x00	0xFFFF	0x00	hex

The pack manufacturer can use this location to store the data flash configuration version. Version control of DFI files used in production is recommended.

Chapter 13
Integrity Data or Checksum

13.1 Introduction

To prevent false writes to data flash while storing configuration parameters or chemistry information, all writes to flash are controlled with checksums.

13.2 Data Flash Checksum

The contents in the DF are protected by three checksum parameters:

- ***All Data Flash Checksum*** validates all parameters that are not pack-specific.
- ***Static Data Flash Checksum*** validates all parameters that are static.
- ***Static Chem Data Flash Checksum*** validates chemistry-specific data. This checksum is executed with the IT_ENABLE subcommand. If the checksum operation fails, the Impedance Track algorithm is disabled.

Each of these checksums is a 16-bit unsigned integer sum of each byte in the data flash. The sum is calculated on a byte-by-byte basis. The most significant bit of the checksum is masked yielding a 15-bit checksum. This checksum is compared with the value generated by the command 0x1A. The checksum execution takes approximately 5 ms and during this time, the fuel gauge does not communicate.

For each of the above checksums, certain data flash locations are excluded in the calculation of the checksum.

Table 13-1 shows the data flash locations that are excluded for the ***All Data Flash Checksum*** computation.

Table 13-1. All Data Flash Checksum Exclusions

Class	Subclass ID	Subclass	Comment
Configuration	56	Manufacturer Data	Pack Lot Code, PCB Lot Code
Configuration	57	Integrity Data	Reset Counter – Full (private) Reset Counter – Watchdog (private)
Configuration	57	Integrity Data	All DF Checksum
System Data	58	Manufacturer Info	Block A Block B
Calibration	104	Data	CC Gain CC Delta CC Offset Board offset Int Temp Offset Ext Temp Offset Pack V offset

Table 13-2 shows the data flash locations that are included for the ***Static Chem Data Flash Checksum*** computation.

Table 13-2. All Chemistry Data Checksum Inclusions

Class	Subclass ID	Subclass	Comment
OCV Table	83	OCV Table	ChemID (public) OCVa Table (private)
OCVb Table	84	OCVb Table	OCVb Table (private)
Rb_Hi Table	85	Rb_Hi Table	Rb Hi Table (private)

Table 13-2. All Chemistry Data Checksum Inclusions (continued)

Class	Subclass ID	Subclass	Comment
Rb_Lo Table	108	Rb_Lo Table	Rb Lo Table (private)
Gas Gauging	80	IT Cfg	Q Invalid Max V Q Invalid Min V

Table 13-3 shows the data flash locations that are excluded for the ***Static Data Flash Checksum*** computation.

Table 13-3. All Static Data Flash Checksum Exclusions

Class	Subclass ID	Subclass	Comment
Configuration	48	Data	Cycle Count
Configuration	56	Manufacturer Data	Pack Lot Code, PCB Lot Code
Configuration	57	Integrity Data	Reset Counter – Full (private) Reset Counter – Watchdog (private)
Configuration	57	Integrity Data	All DF Checksum
Configuration	57	Integrity Data	Static DF Checksum
System Data	58	Manufacturer Info	Block A Block B
LT Data	59	Lifetime Data	All Lifetime Data
LT Data	59	Lifetime Temp Samples	All Lifetime Temp Samples
Gas Gauging	82	State	Qmax Cell 0 Cycle Count Update_Status V at Chg Term Avg I Last Run Avg P Last Run Delta Voltage
Ra Tables	88	Data	Ra Table
Ra Tables	89	Data	Rax Table
Calibration	104	Data	CC Gain CC Delta CC Offset Board offset Int Temp Offset Ext Temp Offset Pack V offset

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
57	Integrity Data	6	All DF Checksum	H2	0x00	0x7FFF	0x00	Number
	Integrity Data	8	Static Chem DF Checksum	H2	0x00	0x7FFF	0x7C23	Number
	Integrity Data	10	Static DF Checksum	H2	0x00	0x7FFF	0x00	Number

Chapter 14
Calibration

14.1 Introduction

The device has integrated routines that support calibration of current, voltage, and temperature readings.

Most values do not require user modification. They are only modified by the calibration commands in CALIBRATION mode. For calibration using a host system, see Chapter 17, *Factory Calibration*.

14.2 Current Calibration

CC Gain and ***CC Delta*** are two parameters used to calibrate current measurement readings.

CC Gain is the gain factor for calibrating Sense Resistor, Trace, and internal coulomb counter (integrating ADC delta sigma) errors. It is used in the algorithm that reports charge and discharge in and out of the battery through the *RemainingCapacity()* register. The difference between ***CC Gain*** and ***CC Delta*** is that the algorithm that reports *AverageCurrent()* cancels out the time base because *AverageCurrent()* does not have a time component (it reports in mA) and ***CC Delta*** requires a time base for reporting *RemainingCapacity()* (it reports in mAh). The default values are based off readings on TI provided EVMs and will need to be changed based on customers' boards.

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
104	Data	0	CC Gain	F4	1.00E–01	4.00E+01	0.9536	Number
104	Data	4	CC Delta	F4	2.98E+04	1.19E+06	1119000	Number

14.3 Offset Calibration

The gauge measurement readings are subject to offset errors, which can be calibrated during production. There are five offset parameters that can be modified, based on board/device parasitics.

14.3.1 Coulomb Counter Offset/Board Offset

CC Offset and ***Board Offset*** are two parameters used for calibrating the offset of the internal coulomb counter, board layout, sense resistor, copper traces, and other offsets from the coulomb counter readings.

CC Offset is the calibration value that primarily corrects for the offset error of the coulomb counter circuitry. ***Board Offset*** is used to calibrate everything the ***CC Offset*** does not calibrate. This includes board layout, sense resistor, copper trace, and other offsets that are external to the chip. The simplified ground circuit design in the fuel gauge requires a separate ***Board Offset*** for each tested device.

To minimize external influences when doing ***CC Offset*** calibration by automatic ***CC Offset*** calibration or ***CC Offset*** calibration function in CALIBRATION mode, an internal short is placed across the SRP and SRN pins inside the fuel gauge. ***CC Offset*** is a correction for small noise and errors; therefore, to maximize accuracy, it takes about 20 seconds to calibrate the offset. Because it is impractical to do a 20-s offset during production, there are two different methods for calibrating ***CC Offset***.

- The first method is to calibrate ***CC Offset*** by putting the fuel gauge in CALIBRATION mode and initiating the ***CC Offset*** function as part of the entire calibration suite. This is a short calibration that is not as accurate as the second method mentioned below. Its primary purpose is to calibrate ***CC Offset*** enough so that it does not affect any other coulomb counter calibrations. This is only intended as a temporary calibration, because the

automatic calibration is done the first time the I2C data and clock are low for more than 20 seconds, which is a much more accurate calibration.
- During NORMAL gas gauge operation when the I2C clock and data lines are low for more than 5 seconds and *AverageCurrent()* is less than ***Sleep Current*** in mA, then an automatic ***CC Offset*** calibration is performed. This takes approximately 16 seconds and is much more accurate than the method in CALIBRATION mode.

The fuel gauge provides an autocalibration feature that measures the voltage offset error across SRP and SRN from time-to-time as operating conditions change. It subtracts the resulting offset error from normal sense resistor voltage, VSR, for maximum measurement accuracy. Autocalibration of the CC begins on entry to SLEEP mode, except if *Temperature()* is ≤ 5°C or *Temperature()* ≥ 45°C, but will not occur more than once per every 10 hours.

The fuel gauge also performs autocalibration offset calibration any time the following conditions are detected:

1. The condition of *AverageCurrent()* ≤ 100 mA, and
2. A voltage change since last offset calibration ≥ 256 mV or a temperature change since the last offset calibration is greater than 8°C for ≥ 60 seconds.

Capacity and current measurements continue at the last measured rate during the offset calibration when these measurements cannot be performed. If the battery voltage drops more than 32 mV during the offset calibration, the load current has likely increased considerably, and the offset calibration is stopped.

14.3.2 Internal/External Temperature Offset

Int Temp Offset is used to calibrate offset errors in the measurement of the reported *Temperature()* if the internal temperature sensor is used. The gain of the internal temperature sensor is accurate enough that a calibration for gain is not required. ***Ext Temp Offset*** is for calibrating the offset of the thermistor connected to the TS1 pin of the fuel gauge as reported by *Temperature()*. The gain of the thermistor is accurate enough that a calibration for gain is not required.

14.3.3 Pack Voltage Offset

Pack V Offset is a calibration value that is used to correct for any offset relating to the analog-to-digital converter (ADC) cell voltage measurement.

14.3.4

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
104	Data	8	CC Offset	I2	–32768	32767	1432	mA
104	Data	10	Board Offset	I1	–128	127	88	µA
104	Data	11	Int Temp Offset	I1	–128	127	0	Number
		12	Ext Temp Offset	I1	–128	127	0	Number
104	Data	13	Pack V Offset	I1	–128	127	0	Number

14.4 Current Measurement Noise Filtering

Multiple parameters are available to filter out noise when measuring current using the coulomb counter.

14.4.1 Filter

The *AverageCurrent()* reading is a running average of the current samples that are taken by the coulomb counter. The ***Filter*** specifies the number of samples that are used to generate the final reading.

14.4.2 Deadband

Deadband creates a filter window to the reported *AverageCurrent()* register where the current is reported as 0. Any negative current above this value or any positive current below this value is displayed as 0. This defaults to 5 mA. Only a few reasons may require changing this value:

1. If the fuel gauge is not calibrated.

2. ***Board Offset*** has not been characterized.
3. If the PCB layout has issues that cause inconsistent board offsets from board to board.
4. An extra noisy environment along with reason 3.

14.4.3 CC Deadband

CC Deadband creates a filter window below which the measured coulomb count is not accumulated. Any coulomb count below this value will be thrown away. This parameter defaults to 34 × 294 nV based on a default sense resistor value of 5 mΩ. It should be scaled based on any changes to the sense resistor value in a given design per CC Deadband × (R_{old}/R_{new}).

14.4.4

Subclass ID	Subclass	Offset	Name	Data Type	Value			Unit
					Min	Max	Default	
107	Current	0	Filter	U1	0	255	239	Number
107	Current	1	Deadband	U1	0	255	5	mA
107	Current	2	CC Deadband	U1	0	255	34	294 nV

This page intentionally left blank.

15.1 Standard Data Commands

The BQ27542-G1 fuel gauge uses a series of 2-byte standard commands to enable system reading and writing of battery information. Each standard command has an associated command-code pair, as indicated in Table 15-1. Each protocol has specific means to access the data at each Command Code. Data RAM is updated and read by the gauge only once per second. Standard commands are accessible in the NORMAL operation mode.

Table 15-1. Standard Commands

Command Name	Command Code	Unit	SEALED Access
Control()	0x00 and 0x01	—	RW
AtRate()	0x02 and 0x03	mA	RW
UnfilteredSOC()	0x04 and 0x05	%	R
Temperature()	0x06 and 0x07	0.1°K	R
Voltage()	0x08 and 0x09	mV	R
Flags()	0x0A and 0x0B	—	R
NomAvailableCapacity()	0x0C and 0x0D	mAh	R
FullAvailableCapacity()	0x0E and 0x0F	mAh	R
RemainingCapacity()	0x10 and 0x11	mAh	R
FullChargeCapacity()	0x12 and 0x13	mAh	R
AverageCurrent()	0x14 and 0x15	mA	R
TimeToEmpty()	0x16 and 0x17	min	R
FulllChargeCapacityFiltered()	0x18 and 0x19	mAh	R
SafetyStatus()	0x1A and 0x1B	—	R
FullChargeCapacityUnfiltered()	0x1C and 0x1D	mAh	R
Imax()	0x1E and 0x1F	mA	R
RemainingCapacityUnflitered()	0x20 and 0x21	mAh	R
RemainingCapacityFiltered()	0x22 and 0x23	mAh	R
BTPSOC1Set()	0x24 and 0x25	mAh	RW
BTPSOC1Clear()	0x26 and 0x27	mAh	RW
InternalTemperature()	0x28 and 0x29	0.1°K	R
CycleCount()	0x2A and 0x2B	Counts	R
StateofCharge()	0x2C and 0x2D	%	R
StateofHealth()	0x2E and 0x2F	%/num	R
ChargingVoltage()	0x30 and 0x31	mV	R
ChargingCurrent)	0x32 and 0x33	mA	R
PassedCharge()	0x34 and 0x35	mAh	R
DOD0()	0x36 and 0x37	hex	R
SelfDischargeCurrent()	0x38 and 0x39	mA	R

15.1.1 Control(): 0x00 and 0x01

Issuing a *Control()* command requires a subsequent 2-byte subcommand. These additional bytes specify the particular control function desired. The *Control()* command allows the system to control specific features of the fuel gauge during normal operation and additional features when the fuel gauge is in different access modes, as described in Table 15-2.

Table 15-2. Control() Subcommands

Subcommand Name	Subcommand Code	SEALED Access	Description
CONTROL_STATUS	0x0000	Yes	Reports the status of DF Checksum, Impedance Track, and so on
DEVICE_TYPE	0x0001	Yes	Reports the device type of 0x0542 (indicating BQ27542-G1)
FW_VERSION	0x0002	Yes	Reports the firmware version on the device type
HW_VERSION	0x0003	Yes	Reports the hardware version on the device type
RESET_DATA	0x0005	Yes	Returns reset data
PREV_MACWRITE	0x0007	Yes	Returns previous *Control()* subcommand code
CHEM_ID	0x0008	Yes	Reports the chemical identifier of the Impedance Track configuration
BOARD_OFFSET	0x0009	No	Forces the device to measure and store the board offset
CC_OFFSET	0x000A	No	Forces the device to measure the CC offset
DF_VERSION	0x000C	Yes	Reports the data flash version of the device
SET_FULLSLEEP	0x0010	Yes	Sets the *CONTROL_STATUS[FULLSLEEP]* bit to 1
SET_HIBERNATE	0x0011	Yes	Forces *CONTROL_STATUS[HIBERNATEn]* to 1
CLEAR_HIBERNATE	0x0012	Yes	Forces *CONTROL_STATUS[HIBERNATE]* to 0
SET_SHUTDOWN	0x0013	Yes	Sets the *CONTROL_STATUS[SHUTDWN]* bit to 1
CLEAR_SHUTDOWN	0x0014	Yes	Clears the *CONTROL_STATUS[SHUTDWN]* bit to 1
SET_HDQINTEN	0x0015	Yes	Forces *CONTROL_STATUS[HDQIntEn]* to 1
CLEAR_HDQINTEN	0x0016	Yes	Forces *CONTROL_STATUS[HDQIntEn]* to 0
STATIC_CHEM_CHKSUM	0x0017	Yes	Calculates chemistry checksum.
ALL_DF_CHKSUM	0x0018	Yes	Reports checksum for all data flash excluding device specific variables
STATIC_DF_CHKSUM	0x0019	Yes	Reports checksum for static data flash excluding device specific variables
SYNC_SMOOTH	0x001E	Yes	Synchronizes *RemCapSmooth()* and *FCCSmooth()* with *RemCapTrue()* and *FCCTrue()*
SEALED	0x0020	No	Places the fuel gauge in SEALED access mode
IT_ENABLE	0x0021	No	Enables the Impedance Track algorithm
IMAX_INT_CLEAR	0x0023	Yes	Clears an Imax interrupt that is currently asserted on the (SE or HDQ) pin
CAL_ENABLE	0x002D	No	Toggle CALIBRATION mode
RESET	0x0041	No	Forces a full reset of the fuel gauge
EXIT_CAL	0x0080	No	Exit CALIBRATION mode
ENTER_CAL	0x0081	No	Enter CALIBRATION mode
OFFSET_CAL	0x0082	No	Reports internal CC offset in CALIBRATION mode

15.1.1.1 CONTROL_STATUS: 0x0000

Instructs the fuel gauge to return status information to control addresses 0x00 and 0x01. The status word includes the following information.

Table 15-3. CONTROL_STATUS Flags

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
High Byte	SE_EN	FAS	SS	CALMODE	CCA	BCA	QMAXUP DATE	HOSTIE
Low Byte	SHUTDN_EN	HIBERNATE	FULLSLEEP	SLEEP	LDMD	RUP_DIS	VOK	QEN

High Byte

SE_EN = Status bit indicating if shutdown is active.

FAS = Status bit indicating the fuel gauge is in FULL ACCESS SEALED state. Active when set (no data flash access).

SS = Status bit indicating the fuel gauge is in the SEALED state. Active when set (no ROM access).

CALMODE = Status bit indicating the calibration function is active. True when set. Default is 0.

CCA = Status bit indicating the Coulomb Counter Calibration routine is active. The CCA routine takes place approximately 1 minute after the initialization and periodically as gauging conditions change. Active when set.

BCA = Status bit indicating the Board Calibration routine is active. Active when set.

QMAXUPDATE = Status bit toggling every time there is a QMAX update.

HOSTIE = Status bit indicating SET_HDQINTEN subcommand has been received and armed the HDQ Host interrupt.

Low Byte

SHUTDN_EN = Control bit indicating that the *SET_SHUTDOWN* subcommand has been sent and signals an external shutdown of the fuel gauge when conditions permit. See Section 5.2, *SHUTDOWN Mode*.

HIBERNATE = Status bit indicating a request for entry into HIBERNATE from SLEEP mode has been issued. True when set. Default is 0.

FULLSLEEP = Status bit indicating the fuel gauge is in FULLSLEEP mode. True when set. The state can be detected by monitoring the power used by the fuel gauge because any communication automatically clears it.

SLEEP = Status bit indicating the fuel gauge is in SLEEP mode. True when set.

LDMD = Status bit indicating the Impedance Track algorithm is using *constant-power* model. True when set. Default is 0 (*constant-current* model).

RUP_DIS = Status bit indicating the Ra table updates are disabled. True when set.

VOK = Status bit indicating cell voltages are OK for Qmax updates. True when set.

QEN = Status bit indicating the Qmax updates are enabled. True when set.

15.1.1.2 DEVICE_TYPE: 0x0001

Instructs the fuel gauge to return the device type to addresses 0x00 and 0x01. The BQ27542-G1 device type returns 0x0542.

15.1.1.3 FW_VERSION: 0x0002

Instructs the fuel gauge to return the firmware version to addresses 0x00 and 0x01. The BQ27542-G1 firmware version returns 0x0201.

15.1.1.4 HW_VERSION: 0x0003

Instructs the fuel gauge to return the hardware version to addresses 0x00 and 0x01. For BQ27542-G1, 0x0000 or 0x0060 is returned.

15.1.1.5 RESET_DATA: 0x0005

Instructs the fuel gauge to return the number of resets performed to addresses 0x00 and 0x01.

15.1.1.6 PREV_MACWRITE: 0x0007

Instructs the fuel gauge to return the previous *Control()* subcommand written to addresses 0x00 and 0x01. The value returned is limited to less than 0x0020.

15.1.1.7 CHEM_ID: 0x0008

Instructs the fuel gauge to return the chemical identifier for the Impedance Track configuration to addresses 0x00 and 0x01.

15.1.1.8 BOARD_OFFSET: 0x0009

Instructs the fuel gauge to perform the board offset calibration. During board offset calibration the *CONTROL_STATUS [BCA]* bit is set.

15.1.1.9 CC_OFFSET: 0x000A

Instructs the fuel gauge to perform the coulomb counter offset calibration. During calibration the *CONTROL_STATUS [CCA]* bit is set.

15.1.1.10 DF_VERSION: 0x000C

Instructs the fuel gauge to return the data flash version stored in ***DF Config Version*** to addresses 0x00 and 0x01.

15.1.1.11 SET_FULLSLEEP: 0x0010

Instructs the fuel gauge to set the *CONTROL_STATUS [FULLSLEEP]* bit to 1. The gauge enters the FULLSLEEP power mode after the transition to the SLEEP power state is detected. In FULLSLEEP mode less power is consumed by disabling an oscillator circuit used by the communication engines. For HDQ communication one host message is dropped, while the oscillator restarts if communication is initiated during FULLSLEEP and may need to be repeated. For I^2C communications, the first I^2C message incurs a 6- to 8-ms clock stretch while the oscillator is started and stabilized. A communication to the device in FULLSLEEP forces the part back to the SLEEP mode.

15.1.1.12 SET_HIBERNATE: 0x0011

Instructs the fuel gauge to force the CONTROL_STATUS [HIBERNATE] bit to 1. This will allow the gauge to enter the HIBERNATE power mode after the transition to SLEEP power state is detected and the required conditions are met. The [HIBERNATE] bit is automatically cleared upon exiting from HIBERNATE mode.

15.1.1.13 CLEAR_HIBERNATE: 0x0012

Instructs the fuel gauge to force the CONTROL_STATUS [HIBERNATE] bit to 0. This prevents the gauge from entering the HIBERNATE power mode after the transition to SLEEP power state is detected unless *Voltage()* is less than Hibernate V. It can also be used to force the gauge out of HIBERNATE mode.

15.1.1.14 SET_SHUTDOWN: 0x0013

Sets the CONTROL_STATUS [SHUTDWN] bit to 1, thereby enabling the SE pin to change state. The Impedance Track algorithm controls the setting of the SE pin, depending on whether the conditions are met for fuel gauge SHUTDOWN or not.

15.1.1.15 CLEAR_SHUTDOWN: 0x0014

Clears the *CONTROL_STATUS [SHUTDN_EN]* bit to 0. The fuel gauge aborts the SHUTDOWN sequence.

15.1.1.16 SET_HDQINTEN: 0x0015

Instructs the fuel gauge to set the ***CONTROL_STATUS [HDQIntEn]*** bit to 1. This enables the HDQ Interrupt function. When this subcommand is received, the device will detect any of the interrupt conditions and assert the interrupt at 1-s intervals until the ***CLEAR_HDQINTEN*** command is received or the count of ***HDQHostIntrTries*** has lapsed (default 3).

15.1.1.17 CLEAR_HDQINTEN: 0x0016

Instructs the fuel gauge to set the ***CONTROL_STATUS [HDQIntEn]*** bit to 0. This disables the HDQ Interrupt function.

15.1.1.18 STATIC_CHEM_CHKSUM: 0x0017

Instructs the fuel gauge to calculate chemistry checksum as a 16-bit unsigned integer sum of all static chemistry data. The most significant bit (MSB) of the checksum is masked yielding a 15-bit checksum. This checksum is compared with value stored in the data flash ***Static Chem DF Checksum***. If the value matches, the MSB is cleared to indicate pass. If it does not match, the MSB is set to indicate failure. The checksum can verify the integrity of the chemistry data stored internally.

Note

The ***Static Chem DF Checksum*** is programmed by the Chemistry programming tool.

15.1.1.19 ALL_DF_CHKSUM: 0x0018

Instructs the fuel gauge to calculate data flash checksum as a 16-bit unsigned integer sum of all data flash excluding device-specific variables. The most significant bit (MSB) of the checksum is masked yielding a 15-bit checksum. This checksum is compared with the value stored in the data flash ***ALL DF Checksum***. If the value matches, the MSB is cleared to indicate pass. If it does not match, the MSB is set to indicate failure. The checksum can verify the integrity of the data flash stored internally.

15.1.1.20 STATIC_DF_CHKSUM: 0x0019

Instructs the fuel gauge to calculate static data flash checksum as a 16-bit unsigned integer sum of static data flash excluding device specific variables. The most significant bit (MSB) of the checksum is masked yielding a 15-bit checksum. This checksum is compared with value stored in the data flash ***Static DF Checksum***. If the value matches, the MSB is cleared to indicate pass. If it does not match, the MSB is set to indicate failure. The checksum can verify the integrity of the static data flash stored internally.

15.1.1.21 SYNC_SMOOTH: 0x001E

This synchronizes *RemCapSmooth()* and *FCCSmooth()* with *RemCapTrue()* and *FCCTrue()*.

15.1.1.22 SEALED: 0x0020

Instructs the fuel gauge to transition from UNSEALED state to SEALED state. The fuel gauge should always be set to SEALED state for use in customer's end equipment as it prevents spurious writes to most standard commands and blocks access to most data flash.

Note

If the gauge is sealed, it will always return to the SEALED state after POR even if the gauge is unsealed prior to a POR. If the SREC of a sealed gauge is extracted and then programmed it into another gauge, the other gauge will also power up in the SEALED state. The only way to permanently restore the UNSEALED state is to re-flash the gauge with an unsealed SREC.

15.1.1.23 IT ENABLE: 0x0021

Forces the fuel gauge to begin the Impedance Track algorithm, sets Bit 2 of ***UpdateStatus*** and causes the *[VOK]* and *[QEN]* flags to be set in the *CONTROL_STATUS* register. *[VOK]* is cleared if the voltages are not suitable for a Qmax update. Once set, *[QEN]* cannot be cleared. This command is only available when the fuel gauge is UNSEALED and is typically enabled at the last step of production after system test is completed.

15.1.1.24 IMAX_INT_CLEAR: 0x0023

Clears an Imax interrupt that is presently asserted on the (SE or HDQ) pin. The command is only applicable if the Imax feature is enabled in ***Pack Configuration D [IMAXEN]***.

15.1.1.25 CAL_ENABLE: 0x002D

Toggles entry into/exit out of CALIBRATION mode

15.1.1.26 RESET: 0x0041

Instructs the fuel gauge to perform a full reset. This command is only available when the fuel gauge is UNSEALED.

15.1.1.27 EXIT_CAL: 0x0080

Instructs the fuel gauge to execute raw measurement data collection for host-managed calibration of the fuel gauge

15.1.1.28 ENTER_CAL: 0x0081

Instructs the fuel gauge to cease raw measurement data collection for host-managed calibration of the fuel gauge

15.1.1.29 OFFSET_CAL: 0x0082

Instructs the fuel gauge to perform offset calibration when in CALIBRATION mode (*CONTROL_STATUS[CALMODE]* = 1)

15.1.2 AtRate(): 0x02 and 0x03

The *AtRate()* is a read-write function that reads or sets the load value used in computing load-compensated capacity in the Impedance Track algorithm when ***Load Mode*** = 0 or 1 and ***Load Select*** = 5. For configurations employing ***Load Mode*** = 0, the *AtRate()* register expects the host to write values in terms of mA. With a ***Load Mode*** of 1, the fuel gauge will expect units of mWh or cWh, depending on the setting for ***Design Energy Scale***. The *AtRate()* value is a signed integer, with negative values interpreted as a discharge current value.

15.1.3 UnfilteredSOC(): 0x04 and 0x05

This read-only function returns an unsigned integer value of the predicted remaining battery capacity expressed as a percentage of *UnfilteredFCC()*, with a range of 0 to 100%.

15.1.4 Temperature(): 0x06 and 0x07

This read-only function returns an unsigned integer value of the battery temperature in units of 0.1°K measured by the fuel gauge and is used for the fuel gauging algorithm. It reports either the *InternalTemperature()* or the external thermistor temperature depending on the setting of the ***[TEMPS]*** bit in ***Pack Configuration***.

15.1.5 Voltage(): 0x08 and 0x09

This read-only function returns an unsigned integer value of the measured cell-pack voltage in mV with a range of 0 to 6000 mV.

15.1.6 Flags(): 0x0A and 0x0B

This read-only function returns the contents of the gas-gauge status register, depicting the current operating status.

Table 15-4. Flags Bit Definitions

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
High Byte	RSVD	RSVD	BATHI	BATLOW	CHG_INH	RSVD	FC	RSVD
Low Byte	CHG_SUS	RSVD	RSVD	IMAX	CHG	SOC1	SOCF	DSG

High Byte

RSVD = Reserved

RSVD = Reserved

BATHI = Battery High bit indicating a high battery voltage condition. See the parameters in Table 5-11, *Battery High Set Voltage Threshold, Time, and Clear,* for threshold settings.

BATLOW = Battery Low bit indicating a low battery voltage condition. See the parameters in Table 5-11, *Battery Low Set Voltage Threshold, Time, and Clear,* for threshold settings.

CHG_INH = Charge Inhibit indicates that temperature is < ***T1 Temp*** or > ***T4 Temp*** while charging is not active. True when set.

RSVD = Reserved

FC = Full-charged state is detected. FC is set when charge termination is reached and ***FC Set %*** = –1 or State of Charge is larger than ***FC Set %*** and ***FC Set %*** is not –1. True when set.

RSVD = Reserved

Low Byte

CHG_SUS = Charge Suspend indicates that temperature is < ***T1 Temp*** or > ***T5 Temp*** while charging is active. True when set.

RSVD = Reserved

RSVD = Reserved

IMAX = Indicates that the computed *Imax()* value has changed enough to signal an interrupt. True when set.
CHG = (Fast) charging allowed. True when set.
SOC1 = State-of-Charge Threshold 1 ***(SOC1 Set)*** reached. True when set.
SOCF = State-of-Charge Threshold Final ***(SOCF Set %)*** reached. True when set.
DSG = Discharging detected. True when set.

15.1.7 NomAvailableCapacity(): 0x0C and 0x0D

This read-only command pair returns the uncompensated (less than C/20 load) battery capacity remaining. Units are mAh.

15.1.8 FullAvailableCapacity(): 0x0E and 0x0F

This read-only command pair returns the uncompensated (less than C/20 load) capacity of the battery when fully charged. Units are mAh. *FullAvailableCapacity()* is updated at regular intervals, as specified by the IT algorithm.

15.1.9 RemainingCapacity(): 0x10 and 0x11

This read-only command pair returns battery capacity remaining. The filtered or the raw battery capacity is returned based on the ***[SmoothEn]*** bit in ***Pack Configuration C***.

***[SmoothEn]* Bit Setting**	**Description**
[SmoothEn] = 1	*RemainingCapacity()* = Filtered compensated battery capacity remaining (*FilteredRM()*)
[SmoothEn] = 0	*RemainingCapacity()* = Compensated battery capacity remaining (*UnfilteredRM()*)

15.1.10 FullChargeCapacity(): 0x12 and 0x13

This read-only command pair returns battery capacity remaining. The filtered or the raw battery capacity is returned based on the ***[SmoothEn]*** bit in ***Pack Configuration C***.

***[SmoothEn]* Bit Setting**	**Description**
[SmoothEn] = 1	*FullChargeCapacity()* = Filtered compensated of fully charged battery (*FilteredFCC()*)
[SmoothEn] = 0	*FullChargeCapacity()* = Compensated capacity of fully charged battery (*UnfilteredFCC()*)

15.1.11 AverageCurrent(): 0x14 and 0x15

This read-only command pair returns a signed integer value that is the average current flow through the sense resistor. It is updated every second in NORMAL mode and every 20 seconds in SLEEP and FULLSLEEP modes. Units are mA.

15.1.12 TimeToEmpty(): 0x16 and 0x17

This read-only function returns an unsigned integer value of the predicted remaining battery life at the present rate of discharge, in minutes. A value of 65,535 indicates battery is not being discharged.

15.1.13 FilteredFCC(): 0x18 and 0x19

This read-only command pair returns the modified full charge capacity based on the SOC smoothing algorithm. The value is modified during battery charge and is increased or decreased to achieve SOC convergence by the time end of charge is reached. For reporting of the capacity that can be extracted from a fully charged battery, the host should always refer to *UnfilteredFCC()*. Units are mAh. *FilteredFCC()* is updated at regular intervals, as specified by the IT algorithm.

15.1.14 SafetyStatus(): 0x1A and 0x1B

This read-only function returns the status of numerous firmware-based safety protections. Internal short, tab disconnect, overtemperature in charge, overtemperature in discharge, overvoltage, and undervoltage are all supported protections.

Table 15-5. Safety Status Bit Definitions

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
High Byte	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
Low Byte	RSVD	RSVD	ISD	TDD	OTC	OTD	RSVD	RSVD

High Byte

RSVD = Bits 7:0 are reserved.

Low Byte

RSVD = Reserved

RSVD = Reserved

ISD = Internal Short condition is detected. True when set.

TDD = Tab Disconnect condition is detected. True when set.

OTC = Overtemperature in charge condition detected. True when set.

OTD = Overtemperature in discharge condition detected. True when set.

RSVD = Reserved

RSVD = Reserved

15.1.15 UnfilteredFCC(): 0x1C and 0x1D

This read-only command pair returns the compensated capacity of the battery when fully charged. Units are mAh. *UnFilteredFCC()* is updated at regular intervals, as specified by the IT algorithm.

15.1.16 Imax(): 0x1E and 0x1F

This read-only function returns the maximum discharge current that the battery can support for ***Max Current Pulse Duration*** time without prematurely dropping to empty (that is, 0%). It is useful for systems that need to dynamically scale applied load for extended runtime at low states of charge.

15.1.17 UnfilteredRM(): 0x20 and 0x21

This read-only command pair returns the compensated battery capacity remaining. Units are mAh.

15.1.18 FilteredRM(): 0x22 and 0x23

This read-only command pair returns the filtered, compensated battery capacity remaining. Units are mAh.

15.1.19 BTPSOC1Set(): 0x24 and 0x25

This read-write function is used to dynamically update the BTP threshold for detecting *RemainingCapacity()* decreasing below the programmed value in the discharge direction.

15.1.20 BTPSOC1Clear(): 0x26 and 0x27

This read-write function is used to dynamically update the BTP threshold for detecting *RemainingCapacity()* increasing above the programmed value in the charge direction.

15.1.21 InternalTemperature(): 0x28 and 0x29

This read-only function returns an unsigned integer value of the measured internal temperature of the device in units of 0.1°K as measured by the fuel gauge.

15.1.22 CycleCount(): 0x2A and 0x2B

This read-only function returns an unsigned integer value of the number of cycles the battery has experienced with a range of 0 to 65,535. One cycle occurs when accumulated discharge ≥ ***CC Threshold***.

15.1.23 StateOfCharge(): 0x2C and 0x2D

This read-only function returns an unsigned integer value of the predicted *RemainingCapacity()* expressed as a percentage of *FullChargeCapacity()*, with a range of 0 to 100%. The *StateOfCharge()* can be filtered or unfiltered since *RemainingCapacity()* and *FullChargeCapacity()* can be filtered or unfiltered based on ***[SmoothEn]*** bit selection in ***Pack Configuration C***.

15.1.24 StateOfHealth(): 0x2E and 0x2F

0x2E SOH percentage: This read-only function returns an unsigned integer value, expressed as a percentage of the ratio of predicted *FCC(25°C, SOH Load I)* over the *DesignCapacity()*. The *FCC(25°C, SOH Load I)* is the calculated full charge capacity at 25°C and the SOH current rate specified by *SOH Load I*. The range of the returned SOH percentage is 0x00 to 0x64, indicating 0 to 100%, correspondingly.

15.1.25 ChargingVoltage(): 0x30 and 0x31

This read-only function returns the recommended charging voltage output from the JEITA charging profile. It is updated automatically based on the present temperature range.

15.1.26 ChargingCurrent(): 0x32 and 0x33

This read-only function returns the recommended charging current output from the JEITA charging profile. It is updated automatically based on the present temperature range.

15.1.27 PassedCharge(): 0x34 and 0x35

This signed integer indicates the amount of charge passed through the sense resistor since the last IT simulation in mAh.

15.1.28 DOD0(): 0x36 and 0x37

This unsigned integer indicates the depth of discharge during the most recent OCV reading. The reported value is scaled to an integer value per DOD0 = DOD(OCV, Temperature) × 2^{14} and has a range of 0 to 16384.

15.1.29 SelfDischargeCurrent(): 0x38 and 0x39

This read-only command pair returns the signed integer value that estimates the battery self-discharge current.

15.2 Extended Data Commands

Extended commands offer additional functionality beyond the standard set of commands. They are used in the same manner; however, unlike standard commands, extended commands are not limited to 2-byte words. The number of command bytes for a given extended command ranges in size from single to multiple bytes, as specified in Table 15-6. For details on the SEALED and UNSEALED states, see Section 16.2.2, *Access Modes*.

Table 15-6. Extended Commands

Name	Command Code	Unit	SEALED Access(1) (2)	UNSEALED Access(1) (2)
PackConfiguration()	0x3A and 0x3B	Hex	R	R
DesignCapacity()	0x3C and 0x3D	mAh	R	R
DataFlashClass() (2)	0x3E	NA	NA	RW
DataFlashBlock() (2)	0x3F	NA	RW	RW
BlockData()/Authenticate() (3)	0x40 to 0x53	NA	RW	RW
BlockData()/AuthenticateCheckSum() (3)	0x54	NA	RW	RW
BlockData()	0x55 to 0x5F	NA	R	RW
BlockDataCheckSum()	0x60	NA	RW	RW
BlockDataControl()	0x61	NA	NA	RW
DODatEOC()	0x62 and 0x63	NA	R	R
Qstart()	0x64 and 0x65	mAh	R	R
FastQmax()	0x66 and 0x67	mAh	R	R
Reserved	0x68 to 0x6C	NA	R	R
Reserved	0x6E and 0x6F	NA	R	R
Reserved	0x70 and 0x71	NA	R	R
Reserved	0x72 and 0x73	NA	R	R

Table 15-6. Extended Commands (continued)

Name	Command Code	Unit	SEALED Access(1) (2)	UNSEALED Access(1) (2)
AveragePower()	0x76 and 0x77	mW or cW	R	R

(1) SEALED and UNSEALED states are entered via commands to *Control()* 0x00 and 0x01.
(2) In SEALED mode, data flash cannot be accessed through commands 0x3E and 0x3F.
(3) The *BlockData()* command area shares functionality for accessing general data flash and for using Authentication. See Chapter 9, *Authentication*, for more details.

15.2.1 PackConfiguration(): 0x3A and 0x3B

SEALED and UNSEALED Access: This command returns the value stored in ***Pack Configuration*** and is expressed in hex value.

15.2.2 DesignCapacity(): 0x3C and 0x3D

SEALED and UNSEALED Access: This command returns the value stored in ***Design Capacity*** and is expressed in mAh. This is intended to be the theoretical or nominal capacity of a new pack, but has no bearing on the operation of the fuel gauge functionality.

15.2.3 DataFlashClass(): 0x3E

This command sets the data flash class to be accessed. The subclass ID to be accessed must be entered in hexadecimal.

SEALED Access: This command is not available in SEALED mode.

15.2.4 DataFlashBlock(): 0x3F

UNSEALED Access: This command sets the data flash block to be accessed. When 0x00 is written to *BlockDataControl()*, *DataFlashBlock()* holds the block number of the data flash to be read or written. Example: Writing a 0x00 to *DataFlashBlock()* specifies access to the first 32-byte block and a 0x01 specifies access to the second 32-byte block, and so on.

SEALED Access: This command directs which data flash block is accessed by the *BlockData()* command. Writing a 0x00 to *DataFlashBlock()* specifies the *BlockData()* command to transfer authentication data. Issuing a 0x01 or 0x02 instructs the *BlockData()* command to transfer ***Manufacturer Info Block A or B***, respectively.

15.2.5 BlockData(): 0x40 Through 0x5F

This command range is used to transfer data for data flash class access. This command range is the 32-byte data block used to access ***Manufacturer Info Block A*** or ***B***. ***Manufacturer Info Block A*** is read-only for the SEALED access. UNSEALED access is read-write.

15.2.6 BlockDataCheckSum(): 0x60

The host system must write this value to inform the device that new data is ready for programming into the specified data flash class and block.

UNSEALED Access: This byte contains the checksum on the 32 bytes of block data read from or written to data flash. The least-significant byte of the sum of the data bytes written must be complemented ([255 – x] for x the 8-bit summation of the *BlockData()* (0x40 to 0x5F) on a byte-by-byte basis) before being written to 0x60.

SEALED Access: This byte contains the checksum for the 32 bytes of block data written to ***Manufacturer Info Block A***. The least-significant byte of the sum of the data bytes written must be complemented ([255 – x], for x the 8-bit summation of the *BlockData()* (0x40 to 0x5F) on a byte-by-byte basis) before being written to 0x60.

15.2.7 BlockDataControl(): 0x61

UNSEALED Access: This command controls DATA FLASH ACCESS mode. The value determines the data flash to be accessed. Writing 0x00 to this command enables *BlockData()* to access general data flash.

SEALED Access: This command is not available in SEALED mode.

15.2.8 DODatEOC(): 0x62 and 0x63

UNSEALED and SEALED Access: This command reports DOD at the end of charge (EOC).

15.2.9 Qstart(): 0x64 and 0x65

UNSEALED and SEALED Access: This command reports Qstart.

15.2.10 FastQmax(): 0x66 and 0x67

UNSEALED and SEALED Access: This command reports Fast Qmax.

15.2.11 Reserved—0x68 to 0x6C

15.2.12 Reserved—0x6E and 0x6F

15.2.13 Reserved—0x70 and 0x71

15.2.14 Reserved—0x72 and 0x73

15.2.15 AveragePower(): 0x76 and 0x77

UNSEALED and SEALED Access: This read-word function returns an unsigned integer value of the average power of the current discharge. It is negative during discharge and positive during charge. A value of 0 indicates that the battery is not being discharged. The value is reported in units of mW (***Design Energy Scale*** = 1) or cW (***Design Energy Scale*** = 10).

15.2.16 AN_COUNTER: 0x79

UNSEALED and SEALED Access: This command reports AN_COUNTER.

15.2.17 AN_CURRENT_LSB: 0x7A

UNSEALED and SEALED Access: This command reports AN_CURRENT_LSB.

15.2.18 AN_CURRENT_MSB: 0x7B

UNSEALED and SEALED Access: This command reports AN_CURRENT_MSB.

15.2.19 AN_VCELL_LSB: 0x7C

UNSEALED and SEALED Access: This command reports AN_VCELL_LSB.

15.2.20 AN_VCELL_MSB: 0x7D

UNSEALED and SEALED Access: This command reports AN_VCELL_MSB.

15.2.21 AN_TEMP_LSB: 0x7E

UNSEALED and SEALED Access: This command reports AN_TEMP_LSB.

15.2.22 AN_TEMP_MSB: 0x7F

UNSEALED and SEALED Access: This command reports AN_TEMP_MSB.

This page intentionally left blank.

Chapter 16

Data Flash Summary

16.1 Introduction

The data flash is a non-volatile memory that contains initialization, default, cell status, calibration, configuration, and user information.

16.2 Data Flash Interface

16.2.1 Accessing the Data Flash

The data flash can be accessed in several different ways, depending in which mode the BQ27542-G1 fuel gauge is operating and what data is accessed.

Commonly accessed data flash memory locations, frequently read by a system, are conveniently accessed through specific instructions, already described in Chapter 15, *Data Commands*. These commands are available when the fuel gauge is either in UNSEALED or SEALED modes.

Most data flash locations, however, are only accessible in UNSEALED mode by use of the evaluation software or by data flash block transfers. These locations must be optimized and/or fixed during the development and manufacture processes. They become part of a golden image file and can then be written to multiple battery packs. Once established, the values generally remain unchanged during end-equipment operation.

To access data flash locations individually, the block containing the desired data flash location(s) must be transferred to the command register locations where they can be read to the system or changed directly. This is accomplished by sending the set-up command *BlockDataControl()* (0x61) with data 0x00. Up to 32 bytes of data can be read directly from the *BlockData()* (0x40 to 0x5F), externally altered, then rewritten to the *BlockData()* command space. Alternatively, specific locations can be read, altered, and rewritten if their corresponding offsets are used to index into the *BlockData()* command space. Finally, the data residing in the command space is transferred to data flash, once the correct checksum for the whole block is written to *BlockDataChecksum()* (0x60).

Occasionally, a data flash class is larger than the 32-byte block size. In this case, the *DataFlashBlock()* command designates in which 32-byte block the desired locations reside. The correct command address is then given by 0x40 + offset *modulo* 32. For example, to access ***Terminate Voltage*** in the ***Gas Gauging*** class, *DataFlashClass()* is issued 80 (0x50) to set the class. Because the offset is 67, it must reside in the third 32-byte block. Hence, *DataFlashBlock()* is issued 0x02 to set the block offset, and the offset used to index into the *BlockData()* memory area is 0x40 + 67 *modulo* 32 = 0x40 + 0x03 = 0x43.

Reading and writing subclass data are block operations up to 32 bytes in length. If during a write the data length exceeds the maximum block size, then the data is ignored.

None of the data written to memory is bounded by the fuel gauge—the values are not rejected by the fuel gauge. Writing an incorrect value may result in hardware failure due to firmware program interpretation of the invalid data. The written data is persistent, so a power-on reset does not resolve the fault.

16.2.2 Access Modes

The fuel gauge provides three security modes (FULL ACCESS, UNSEALED, and SEALED) that control data flash access permissions according to Table 16-1. The **Data Flash** column refers to those data flash locations that are accessible to the user. The **Manufacturer Information** column refers to the two 32-byte blocks.

Table 16-1. Data Flash Access

Security Mode	Data Flash	Manufacturer Information
FULL ACCESS	RW	RW
UNSEALED	RW	RW
SEALED	None	R (A); RW (B)

Although FULL ACCESS and UNSEALED modes appear identical, only the FULL ACCESS mode allows the fuel gauge to write access mode transition keys stored in the ***Security*** class.

16.2.3 Sealing or Unsealing Data Flash

The fuel gauge implements a key-access scheme to transition between SEALED, UNSEALED, and FULL ACCESS modes. Each transition requires that a unique set of two keys be sent to the fuel gauge via the *Control()* command. The keys must be sent consecutively, with no other data being written to the *Control()* register in between. To avoid conflict, the keys must be different from the codes presented in the CNTL DATA column of Table 15-2, *Control() Subcommands*.

When in SEALED mode the *CONTROL_STATUS [SS]* bit is set, but when the Unseal Keys are correctly received by the fuel gauge, the *[SS]* bit is cleared. When the Full Access Keys are correctly received, the *CONTROL_STATUS [FAS]* bit is cleared.

Both ***Unseal Key*** and ***Full Access Key*** have two words and are stored in data flash. The first word is Key 0 and the second word is Key 1. The order of the keys sent to fuel gauge is Key 1 followed by Key 0. The order of the bytes for each key entered through the *Control()* command is the reverse of what is read from the part.

Example:

If the Unseal Key = 0x56781234

key 1 = 0x1234

key 2 is 0x5678

Control() must supply 0x3412 and 0x7856 to unseal the part. The ***Unseal Key*** and the ***Full Access Key*** can only be updated when in FULL ACCESS mode.

Every time a SEAL command is sent, a control flag location inside the gauge's information flash is changed from the default (0xFFFF) value. This location is checked for the default value every time the gauge powers up.

Note

If the gauge is sealed, it will always return to the SEALED state after POR even if the gauge is unsealed prior to a POR. If the SREC of a sealed gauge is extracted and then programmed it into another gauge, the other gauge will also power up in the SEALED state. The only way to permanently restore the UNSEALED state is to reflash the gauge with an unsealed SREC.

Figure 16-1 shows the interactions between the various modes inside the gauge and the specific value of the *CONTROL_STATUS[SS]* and *CONTROL_STATUS[FAS]* bits.

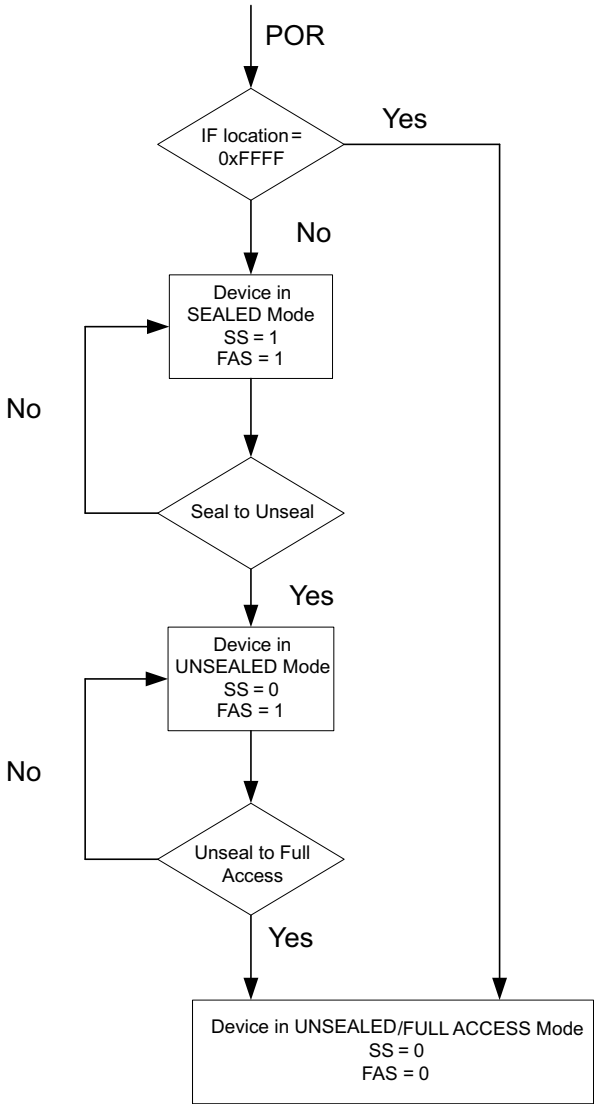


Figure 16-1. Gauge Mode State Diagram

16.3 Data Flash Summary Tables

Table 16-3 through Table 16-4 summarize the data flash locations, including their default, minimum, and maximum values, that are available to the user.

Table 16-2. Data Type Decoder

Type	Min Value	Max Value
F4	±9.8603 × 10^{-39}	±5.707267 × 10^{37}
H1	0x00	0xFF
H2	0x00	0xFFFF
H4	0x00	0xFFFF FFFF
I1	–128	127
I2	–32768	32767
I4	−2,147,483,648	2,147,483,647
Sx	1-byte string	X-byte string
U1	0	255
U2	0	65535
U4	0	4,294,967,295

Table 16-3. Data Flash Summary

Class	Subclass	Subclass ID	Offset	Type	Name	Min	Max	Default	Unit
Calibration	Data	104	0	F4	CC Gain	1.00E-01	4.00E+01	0.47095	mΩ
Calibration	Data	104	4	F4	CC Delta	2.98E+04	1.19E+06	559500	mΩ
Calibration	Data	104	8	I2	CC Offset	–32768	32767	–1200	—
Calibration	Data	104	10	I1	Board Offset	–128	127	0	Counts
Calibration	Data	104	11	I1	Int Temp Offset	–128	127	0	0.1°C
Calibration	Data	104	12	I1	Ext Temp Offset	–128	127	0	0.1°C
Calibration	Data	104	13	I1	Pack V Offset	–128	127	0	mV
Calibration	Current	107	0	U1	Filter	0	255	239	Num
Calibration	Current	107	1	U1	Deadband	0	255	5	mA
Calibration	Current	107	2	U1	CC Deadband	0	255	17	Counts
Configuration	Charge	34	0	I2	Charging Voltage	4000	4600	4350	mV
Configuration	Charge Termination	36	0	I2	Taper Current	0	1000	100	mA
Configuration	Charge Termination	36	2	I2	Min Taper Capacity	0	1000	25	mAh
Configuration	Charge Termination	36	4	I2	Taper Voltage	0	1000	100	mV
Configuration	Charge Termination	36	6	U1	Current Taper Window	0	60	40	s
Configuration	Charge Termination	36	7	I1	TCA Set %	–1	100	–1	%
Configuration	Charge Termination	36	8	I1	TCA Clear %	–1	100	98	%
Configuration	Charge Termination	36	9	I1	FC Set %	–1	100	–1	%
Configuration	Charge Termination	36	10	I1	FC Clear %	–1	100	98	%
Configuration	Charge Termination	36	11	I2	DODatEOC Delta T	0	1000	50	0.1°C
Configuration	JEITA	39	0	I1	T1 Temp	–128	127	0	°C
Configuration	JEITA	39	1	I1	T2 Temp	–128	127	10	°C
Configuration	JEITA	39	2	I1	T3 Temp	–128	127	45	°C
Configuration	JEITA	39	3	I1	T4 Temp	–128	127	50	°C
Configuration	JEITA	39	4	I1	T5 Temp	–128	127	60	°C
Configuration	JEITA	39	5	I1	Temp Hys	–128	127	1	°C
Configuration	JEITA	39	6	I2	T1-T2 Chg Voltage	0	4600	4350	mV

Table 16-3. Data Flash Summary (continued)

Class	Subclass	Subclass ID	Offset	Type	Name	Min	Max	Default	Unit
Configuration	JEITA	39	8	I2	T2-T3 Chg Voltage	0	4600	4350	mV
Configuration	JEITA	39	10	I2	T3-T4 Chg Voltage	0	4600	4300	mV
Configuration	JEITA	39	12	I2	T4-T5 Chg Voltage	0	4600	4250	mV
Configuration	JEITA	39	14	U1	T1-T2 Chg Current	0	100	50	%
Configuration	JEITA	39	15	U1	T2-T3 Chg Current	0	100	80	%
Configuration	JEITA	39	16	U1	T3-T4 Chg Current	0	100	80	%
Configuration	JEITA	39	17	U1	T4-T5 Chg Current	0	100	80	%
Configuration	Registers	64	0	H2	Pack Configuration	0	ffff	297f	Flag
Configuration	Registers	64	2	H1	Pack Configuration B	0	ff	87	Flag
Configuration	Registers	64	3	H1	Pack Configuration C	0	ff	b9	Flag
Configuration	Registers	64	4	H1	Pack Configuration D	0	ff	57	Flag
Configuration	HDQ	64	5	U1	Host Interrupt Tries	0	255	3	Num
Configuration	Power	68	0	I2	Flash Update OK Voltage	0	5000	2800	mV
Configuration	Power	68	2	I2	Sleep Current	0	100	15	mA
Configuration	Power	68	9	I2	Hibernate Current	–32768	32767	8	mA
Configuration	Power	68	11	I2	Hibernate Voltage	0	32767	2550	mV
Configuration	Power	68	13	U1	FS Wait	0	255	0	s
Configuration	Lifetime Data	59	0	I2	Lifetime Max Temp	-600	1400	0	0.1°C
Configuration	Lifetime Data	59	2	I2	Lifetime Min Temp	-600	1400	500	0.1°C
Configuration	Lifetime Data	59	4	I2	Lifetime Max Pack Voltage	0	32767	2800	mV
Configuration	Lifetime Data	59	6	I2	Lifetime Min Pack Voltage	0	32767	5000	mV
Configuration	Lifetime Data	59	8	I2	Lifetime Max Chg Current	–32768	32767	0	mA
Configuration	Lifetime Data	59	10	I2	Lifetime Max Dsg Current	–32768	32767	0	mA
Configuration	Lifetime Temp Samples	59	12	U2	LT Flash Cnt	0	32767	0	Num
Configuration	Lifetime Resolution	66	0	U1	LT Temp Res	0	255	10	0.1°C
Configuration	Lifetime Resolution	66	1	U1	LT V Res	0	255	25	mV
Configuration	Lifetime Resolution	66	2	U1	LT Cur Res	0	255	100	mA
Configuration	Lifetime Resolution	66	3	U2	LT Update Time	0	65535	60	s
Configuration	Safety	2	0	I2	OT Chg	0	1200	550	0.1°C
Configuration	Safety	2	2	U1	OT Chg Time	0	60	5	s
Configuration	Safety	2	3	I2	OT Chg Recovery	0	1200	500	0.1°C
Configuration	Safety	2	5	I2	OT Dsg	0	1200	600	0.1°C
Configuration	Safety	2	7	U1	OT Dsg Time	0	60	5	s
Configuration	Safety	2	8	I2	OT Dsg Recovery	0	1200	550	0.1°C
Configuration	Manufacturer Data	56	0	H2	Pack Lot Code	0	ffff	0	Hex
Configuration	Manufacturer Data	56	2	H2	PCB Lot Code	0	ffff	0	Hex

Table 16-3. Data Flash Summary (continued)

Class	Subclass	Subclass ID	Offset	Type	Name	Min	Max	Default	Unit
Configuration	Manufacturer Data	56	4	H2	Firmware Version	0	ffff	0	Hex
Configuration	Manufacturer Data	56	6	H2	Hardware Revision	0	ffff	0	Hex
Configuration	Manufacturer Data	56	8	H2	Cell Revision	0	ffff	0	Hex
Configuration	Manufacturer Data	56	10	H2	DF Config Version	0	ffff	0	Hex
Configuration	Integrity Data	57	6	H2	All DF Checksum	0	7fff	0	Hex
Configuration	Integrity Data	57	8	H2	Static Chem DF Checksum	0	7fff	7c23	Hex
Configuration	Integrity Data	57	10	H2	Static DF Checksum	0	7fff	0	Hex
Configuration	Data	48	0	I2	Design Voltage	2000	5000	3800	mV
Configuration	Data	48	8	U2	Cycle Count	0	65535	0	Num
Configuration	Data	48	10	I2	CC Threshold	100	32767	900	mAh
Configuration	Data	48	12	I2	Design Capacity	0	14500	1000	mAh
Configuration	Data	48	14	I2	Design Energy	0	32767	3800	mWh
Configuration	Data	48	16	I2	SOH Load I	–32768	0	-400	mA
Configuration	Data	48	18	U1	TDD SOH Percent	0	100	80	%
Configuration	Data	48	19	U2	ISD Current	1	32767	10	hourrate
Configuration	Data	48	21	U1	ISD I Filter	0	255	127	Num
Configuration	Data	48	22	U1	Min ISD Time	0	255	7	hour
Configuration	Data	48	23	U1	Design Energy Scale	1	10	1	Num
Configuration	Discharge	49	0	U2	SOC1 Set Threshold	0	65535	150	mAh
Configuration	Discharge	49	2	U2	SOC1 Clear Threshold	0	65535	175	mAh
Configuration	Discharge	49	4	U2	SOCF Set Threshold	0	65535	75	mAh
Configuration	Discharge	49	6	U2	SOCF Clear Threshold	0	65535	100	mAh
Configuration	Discharge	49	8	I2	BL Set Volt Threshold	0	5000	2500	mV
Configuration	Discharge	49	10	U1	BL Set Volt Time	0	60	2	s
Configuration	Discharge	49	11	I2	BL Clear Volt Threshold	0	5000	2600	mV
Configuration	Discharge	49	13	I2	BH Set Volt Threshold	0	5000	4500	mV
Configuration	Discharge	49	15	U1	BH Volt Time	0	60	2	s
Configuration	Discharge	49	16	I2	BH Clear Volt Threshold	0	5000	4400	mV
Gas Gauging	Current Thresholds	81	0	I2	Dsg Current Threshold	0	2000	60	mA
Gas Gauging	Current Thresholds	81	2	I2	Chg Current Threshold	0	2000	75	mA
Gas Gauging	Current Thresholds	81	4	I2	Quit Current	0	1000	40	mA
Gas Gauging	Current Thresholds	81	6	U2	Dsg Relax Time	0	65535	60	s
Gas Gauging	Current Thresholds	81	8	U1	Chg Relax Time	0	255	60	s
Gas Gauging	Current Thresholds	81	9	I2	Max IR Correct	0	1000	400	mV
Gas Gauging	State	82	0	I2	Qmax Cell 0	0	14500	1000	mAh
Gas Gauging	State	82	2	H1	Update Status	0	6	0	Hex
Gas Gauging	State	82	3	I2	V at Chg Term	0	5000	4350	mV
Gas Gauging	State	82	5	I2	Avg I Last Run	–32768	0	-299	mA
Gas Gauging	State	82	7	I2	Avg P Last Run	–32768	0	–1131	pwr
Gas Gauging	State	82	9	I2	Delta Voltage	0	32767	2	mV
Gas Gauging	State	82	11	I2	T Rise	0	32767	50	Num

Table 16-3. Data Flash Summary (continued)

Class	Subclass	Subclass ID	Offset	Type	Name	Min	Max	Default	Unit
Gas Gauging	State	82	13	I2	T Time Constant	0	32767	1000	Num
Gas Gauging	IT Cfg	80	0	U1	Load Select	0	6	1	Num
Gas Gauging	IT Cfg	80	1	U1	Load Mode	0	1	1	Num
Gas Gauging	IT Cfg	80	17	U1	Max Res Factor	0	255	15	Num
Gas Gauging	IT Cfg	80	18	U1	Min Res Factor	0	255	5	Num
Gas Gauging	IT Cfg	80	20	U2	Ra Filter	0	1000	800	Num
Gas Gauging	IT Cfg	80	22	I2	Res V Drop	0	32767	50	mV
Gas Gauging	IT Cfg	80	39	U1	Fast Qmax Start DOD %	0	100	92	%
Gas Gauging	IT Cfg	80	40	U1	Fast Qmax End DOD %	0	100	96	%
Gas Gauging	IT Cfg	80	41	I2	Fast Qmax Start Volt Delta	0	4200	200	mV
Gas Gauging	IT Cfg	80	43	U2	Fast Qmax Current Threshold	0	1000	4	hourrate
Gas Gauging	IT Cfg	80	61	U1	Qmax Capacity Err	0	100	15	0.10%
Gas Gauging	IT Cfg	80	62	U1	Max Qmax Change	0	255	30	%
Gas Gauging	IT Cfg	80	64	I2	Terminate Voltage	2800	3700	3000	mV
Gas Gauging	IT Cfg	80	66	I2	Term V Delta	0	4200	200	mV
Gas Gauging	IT Cfg	80	69	U2	ResRelax Time	0	65535	500	s
Gas Gauging	IT Cfg	80	73	I2	User Rate-mA	0	32767	0	mA
Gas Gauging	IT Cfg	80	75	I2	User Rate-Pwr	0	32767	0	pwr
Gas Gauging	IT Cfg	80	77	I2	Reserve Cap-mAh	0	14500	0	mAh
Gas Gauging	IT Cfg	80	79	I2	Reserve Energy	0	32767	0	egy
Gas Gauging	IT Cfg	80	84	I2	Max DeltaV	0	32767	200	mV
Gas Gauging	IT Cfg	80	86	I2	Min DeltaV	0	32767	0	mV
Gas Gauging	IT Cfg	80	88	U1	Max Sim Rate	0	255	1	hourrate
Gas Gauging	IT Cfg	80	89	U1	Min Sim Rate	0	255	20	hourrate
Gas Gauging	IT Cfg	80	90	I2	Ra Max Delta	0	32767	54	mΩ
Gas Gauging	IT Cfg	80	92	I2	Trace Resistance	0	32767	0	mΩ
Gas Gauging	IT Cfg	80	94	I2	Downstream Resistance	0	32767	0	mΩ
Gas Gauging	IT Cfg	80	96	U1	Qmax Max Delta %	0	100	5	%
Gas Gauging	IT Cfg	80	97	U1	Qmax Bound %	0	255	130	%
Gas Gauging	IT Cfg	80	98	U2	DeltaV Max Delta	0	65535	10	mV
Gas Gauging	IT Cfg	80	100	I2	Max Res Scale	0	32767	5000	Num
Gas Gauging	IT Cfg	80	102	I2	Min Res Scale	0	32767	200	Num
Gas Gauging	IT Cfg	80	104	U1	Fast Scale Start SOC	0	100	10	%
Gas Gauging	IT Cfg	80	105	U1	Fast Scale Load Select	0	6	3	Num
Gas Gauging	IT Cfg	80	106	I2	Charge Hys V Shift	0	2000	40	mV
Gas Gauging	IT Cfg	80	108	I1	RaScl OCV Rst Temp Thresh	0	127	15	°C
Gas Gauging	IT Cfg	80	109	I2	Max Allowed Current	0	32767	8500	mA
Gas Gauging	IT Cfg	80	111	U1	Max Current Pulse Duration	0	255	10	s
Gas Gauging	IT Cfg	80	112	I2	Max Current Interrupt Step	–32768	32767	500	mA
Gas Gauging	IT Cfg	80	116	U2	Predict Outside Temp Time	0	65535	2000	s
Gas Gauging	IT Cfg	80	118	U1	Termination Voltage Valid Time	0	255	2	s

Table 16-3. Data Flash Summary (continued)

Class	Subclass	Subclass ID	Offset	Type	Name	Min	Max	Default	Unit
Security	Codes	112	0	H2	Sealed to Unsealed MSB	0	ffff	3672	Hex
Security	Codes	112	2	H2	Sealed to Unsealed LSB	0	ffff	414	Hex
Security	Codes	112	4	H2	Unsealed to Full MSB	0	ffff	ffff	Hex
Security	Codes	112	6	H2	Unsealed to Full LSB	0	ffff	ffff	Hex
Security	Codes	112	8	H2	Authen Key3 MSB	0	ffff	123	Hex
Security	Codes	112	10	H2	Authen Key3 LSB	0	ffff	4567	Hex
Security	Codes	112	12	H2	Authen Key2 MSB	0	ffff	89ab	Hex
Security	Codes	112	14	H2	Authen Key2 LSB	0	ffff	cdef	Hex
Security	Codes	112	16	H2	Authen Key1 MSB	0	ffff	fedc	Hex
Security	Codes	112	18	H2	Authen Key1 LSB	0	ffff	ba98	Hex
Security	Codes	112	20	H2	Authen Key0 MSB	0	ffff	7654	Hex
Security	Codes	112	22	H2	Authen Key0 LSB	0	ffff	3210	Hex
System Data	Manufacturer Info	58	0	H1	Block A 0	0	ff	0	Hex
System Data	Manufacturer Info	58	1	H1	Block A 1	0	ff	0	Hex
System Data	Manufacturer Info	58	2	H1	Block A 2	0	ff	0	Hex
System Data	Manufacturer Info	58	3	H1	Block A 3	0	ff	0	Hex
System Data	Manufacturer Info	58	4	H1	Block A 4	0	ff	0	Hex
System Data	Manufacturer Info	58	5	H1	Block A 5	0	ff	0	Hex
System Data	Manufacturer Info	58	6	H1	Block A 6	0	ff	0	Hex
System Data	Manufacturer Info	58	7	H1	Block A 7	0	ff	0	Hex
System Data	Manufacturer Info	58	8	H1	Block A 8	0	ff	0	Hex
System Data	Manufacturer Info	58	9	H1	Block A 9	0	ff	0	Hex
System Data	Manufacturer Info	58	10	H1	Block A 10	0	ff	0	Hex
System Data	Manufacturer Info	58	11	H1	Block A 11	0	ff	0	Hex
System Data	Manufacturer Info	58	12	H1	Block A 12	0	ff	0	Hex
System Data	Manufacturer Info	58	13	H1	Block A 13	0	ff	0	Hex
System Data	Manufacturer Info	58	14	H1	Block A 14	0	ff	0	Hex
System Data	Manufacturer Info	58	15	H1	Block A 15	0	ff	0	Hex
System Data	Manufacturer Info	58	16	H1	Block A 16	0	ff	0	Hex
System Data	Manufacturer Info	58	17	H1	Block A 17	0	ff	0	Hex
System Data	Manufacturer Info	58	18	H1	Block A 18	0	ff	0	Hex
System Data	Manufacturer Info	58	19	H1	Block A 19	0	ff	0	Hex
System Data	Manufacturer Info	58	20	H1	Block A 20	0	ff	0	Hex
System Data	Manufacturer Info	58	21	H1	Block A 21	0	ff	0	Hex

Table 16-3. Data Flash Summary (continued)

Class	Subclass	Subclass ID	Offset	Type	Name	Min	Max	Default	Unit
System Data	Manufacturer Info	58	22	H1	Block A 22	0	ff	0	Hex
System Data	Manufacturer Info	58	23	H1	Block A 23	0	ff	0	Hex
System Data	Manufacturer Info	58	24	H1	Block A 24	0	ff	0	Hex
System Data	Manufacturer Info	58	25	H1	Block A 25	0	ff	0	Hex
System Data	Manufacturer Info	58	26	H1	Block A 26	0	ff	0	Hex
System Data	Manufacturer Info	58	27	H1	Block A 27	0	ff	0	Hex
System Data	Manufacturer Info	58	28	H1	Block A 28	0	ff	0	Hex
System Data	Manufacturer Info	58	29	H1	Block A 29	0	ff	0	Hex
System Data	Manufacturer Info	58	30	H1	Block A 30	0	ff	0	Hex
System Data	Manufacturer Info	58	31	H1	Block A 31	0	ff	0	Hex
System Data	Manufacturer Info	58	32	H1	Block B 0	0	ff	0	Hex
System Data	Manufacturer Info	58	33	H1	Block B 1	0	ff	0	Hex
System Data	Manufacturer Info	58	34	H1	Block B 2	0	ff	0	Hex
System Data	Manufacturer Info	58	35	H1	Block B 3	0	ff	0	Hex
System Data	Manufacturer Info	58	36	H1	Block B 4	0	ff	0	Hex
System Data	Manufacturer Info	58	37	H1	Block B 5	0	ff	0	Hex
System Data	Manufacturer Info	58	38	H1	Block B 6	0	ff	0	Hex
System Data	Manufacturer Info	58	39	H1	Block B 7	0	ff	0	Hex
System Data	Manufacturer Info	58	40	H1	Block B 8	0	ff	0	Hex
System Data	Manufacturer Info	58	41	H1	Block B 9	0	ff	0	Hex
System Data	Manufacturer Info	58	42	H1	Block B 10	0	ff	0	Hex
System Data	Manufacturer Info	58	43	H1	Block B 11	0	ff	0	Hex
System Data	Manufacturer Info	58	44	H1	Block B 12	0	ff	0	Hex
System Data	Manufacturer Info	58	45	H1	Block B 13	0	ff	0	Hex
System Data	Manufacturer Info	58	46	H1	Block B 14	0	ff	0	Hex
System Data	Manufacturer Info	58	47	H1	Block B 15	0	ff	0	Hex
System Data	Manufacturer Info	58	48	H1	Block B 16	0	ff	0	Hex
System Data	Manufacturer Info	58	49	H1	Block B 17	0	ff	0	Hex
System Data	Manufacturer Info	58	50	H1	Block B 18	0	ff	0	Hex
System Data	Manufacturer Info	58	51	H1	Block B 19	0	ff	0	Hex
System Data	Manufacturer Info	58	52	H1	Block B 20	0	ff	0	Hex
System Data	Manufacturer Info	58	53	H1	Block B 21	0	ff	0	Hex
System Data	Manufacturer Info	58	54	H1	Block B 22	0	ff	0	Hex
System Data	Manufacturer Info	58	55	H1	Block B 23	0	ff	0	Hex

Table 16-3. Data Flash Summary (continued)

Class	Subclass	Subclass ID	Offset	Type	Name	Min	Max	Default	Unit
System Data	Manufacturer Info	58	56	H1	Block B 24	0	ff	0	Hex
System Data	Manufacturer Info	58	57	H1	Block B 25	0	ff	0	Hex
System Data	Manufacturer Info	58	58	H1	Block B 26	0	ff	0	Hex
System Data	Manufacturer Info	58	59	H1	Block B 27	0	ff	0	Hex
System Data	Manufacturer Info	58	60	H1	Block B 28	0	ff	0	Hex
System Data	Manufacturer Info	58	61	H1	Block B 29	0	ff	0	Hex
System Data	Manufacturer Info	58	62	H1	Block B 30	0	ff	0	Hex
System Data	Manufacturer Info	58	63	H1	Block B 31	0	ff	0	Hex
Ra Tables	Ra0 Table	88	0	H2	Ra Flag	0	ffff	ff55	—
Ra Tables	Ra0 Table	88	2	I2	Ra 0	0	32767	272	2^{-10} Ω
Ra Tables	Ra0 Table	88	4	I2	Ra 1	0	32767	316	2^{-10} Ω
Ra Tables	Ra0 Table	88	6	I2	Ra 2	0	32767	374	2^{-10} Ω
Ra Tables	Ra0 Table	88	8	I2	Ra 3	0	32767	507	2^{-10} Ω
Ra Tables	Ra0 Table	88	10	I2	Ra 4	0	32767	360	2^{-10} Ω
Ra Tables	Ra0 Table	88	12	I2	Ra 5	0	32767	330	2^{-10} Ω
Ra Tables	Ra0 Table	88	14	I2	Ra 6	0	32767	389	2^{-10} Ω
Ra Tables	Ra0 Table	88	16	I2	Ra 7	0	32767	345	2^{-10} Ω
Ra Tables	Ra0 Table	88	18	I2	Ra 8	0	32767	352	2^{-10} Ω
Ra Tables	Ra0 Table	88	20	I2	Ra 9	0	32767	367	2^{-10} Ω
Ra Tables	Ra0 Table	88	22	I2	Ra 10	0	32767	374	2^{-10} Ω
Ra Tables	Ra0 Table	88	24	I2	Ra 11	0	32767	397	2^{-10} Ω
Ra Tables	Ra0 Table	88	26	I2	Ra 12	0	32767	455	2^{-10} Ω
Ra Tables	Ra0 Table	88	28	I2	Ra 13	0	32767	808	2^{-10} Ω
Ra Tables	Ra0 Table	88	30	I2	Ra 14	0	32767	1182	2^{-10} Ω
Ra Tables	Ra0x Table	89	0	H2	Ra Flag	0	ffff	ffff	-
Ra Tables	Ra0x Table	89	2	I2	Ra 0	0	32767	272	2^{-10} Ω
Ra Tables	Ra0x Table	89	4	I2	Ra 1	0	32767	316	2^{-10} Ω
Ra Tables	Ra0x Table	89	6	I2	Ra 2	0	32767	374	2^{-10} Ω
Ra Tables	Ra0x Table	89	8	I2	Ra 3	0	32767	507	2^{-10} Ω
Ra Tables	Ra0x Table	89	10	I2	Ra 4	0	32767	360	2^{-10} Ω
Ra Tables	Ra0x Table	89	12	I2	Ra 5	0	32767	330	2^{-10} Ω
Ra Tables	Ra0x Table	89	14	I2	Ra 6	0	32767	389	2^{-10} Ω
Ra Tables	Ra0x Table	89	16	I2	Ra 7	0	32767	345	2^{-10} Ω
Ra Tables	Ra0x Table	89	18	I2	Ra 8	0	32767	352	2^{-10} Ω
Ra Tables	Ra0x Table	89	20	I2	Ra 9	0	32767	367	2^{-10} Ω
Ra Tables	Ra0x Table	89	22	I2	Ra 10	0	32767	374	2^{-10} Ω
Ra Tables	Ra0x Table	89	24	I2	Ra 11	0	32767	397	2^{-10} Ω
Ra Tables	Ra0x Table	89	26	I2	Ra 12	0	32767	455	2^{-10} Ω
Ra Tables	Ra0x Table	89	28	I2	Ra 13	0	32767	808	2^{-10} Ω
Ra Tables	Ra0x Table	89	30	I2	Ra 14	0	32767	1182	2^{-10} Ω

Table 16-4. Data Flash to EVSW Conversion

Class	Subclass ID	Subclass	Offset	Name	Data Type	Data Flash Default	Data Flash Unit	EVSW Default	EVSW Unit	Data Flash to EVSW Conversion
Calibration	104	Data	0	CC Gain	F4	0.9536	number	5.00	mΩ	4.768/DF
			4	CC Delta	F4	11.19e5	number	5.0737	mΩ	5677445/DF
			8	CC Offset	I2	1432	mA	6.8736	mA	DF × 0.0048
			10	Board Offset	I1	88	µA	0.66	µA	DF × 0.0075

Chapter 17

Factory Calibration

The BQ27542-G1 fuel gauge requires factory calibration. The gauge performs only a limited number of calibration functions. The other functions must be performed by a host system using commands provided by the gauge for this purpose. The following sections provide detailed information, including flowcharts, of the various calibration sequences.

17.1 General I²C Command Information

In the following flowcharts, all I²C functions take three arguments. Write command arguments:

- Address
- Data
- Wait time in ms

Read command arguments:

- Address
- Number of bytes read
- Wait time in ms

17.2 Calibration

17.2.1 Method

The calibration method is broken up into the following sections. The first four sequences are subroutines to be used in the main calibration sequences.

- Section 17.3, *Enter CALIBRATION Mode*
- Section 17.4, *Exit CALIBRATION Mode*
- Section 17.7, *Obtain Raw Calibration Data*
- Section 17.11, *Floating Point Conversion*
- Section 17.5, *CC Offset*
- Section 17.6, *Board Offset*
- Section 17.8, *Current Calibration*
- Section 17.9, *Voltage Calibration*
- Section 17.10, *Temperature Calibration*

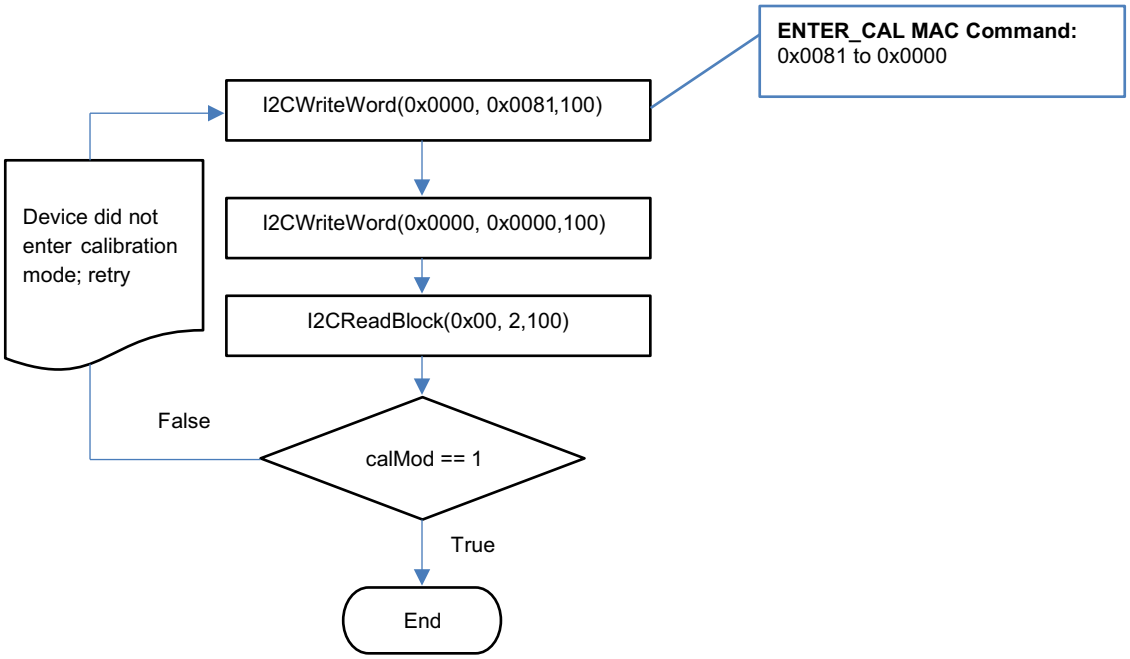
17.2.2 Sequence

Perform the following calibration sequence during battery pack manufacturing process:

1. Perform CC Offset
2. Perform Board Offset
3. Perform Current Calibration
4. Perform Voltage Calibration
5. Perform Temperature Calibration
6. Write calibration results to data flash

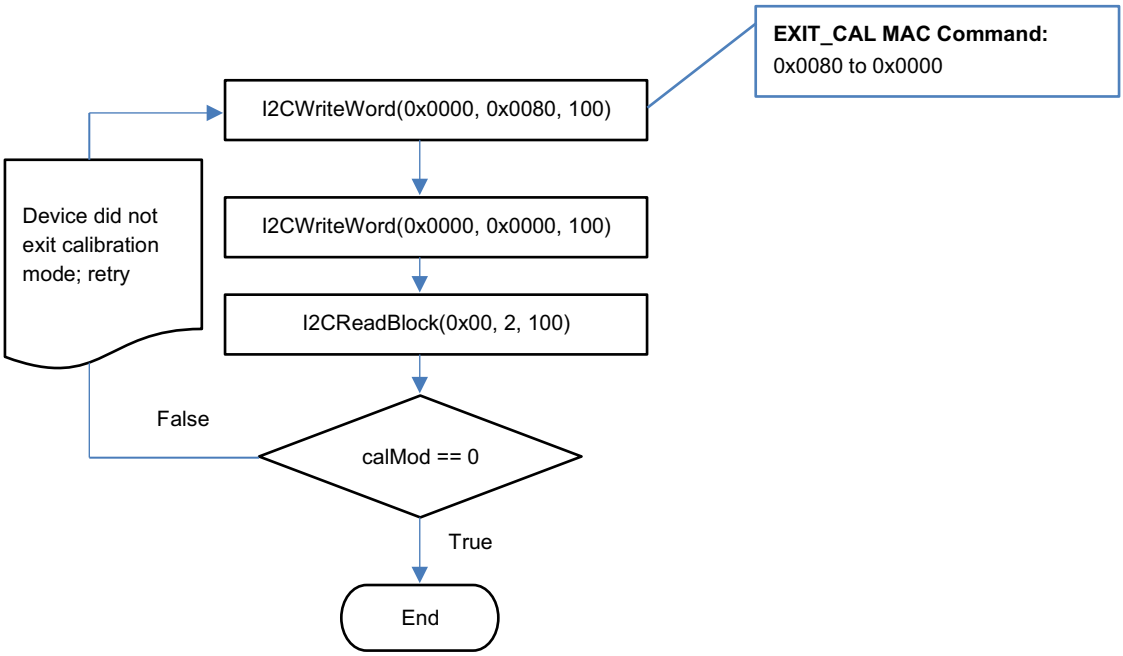
17.3 Enter CALIBRATION Mode

This sequence puts the gauge into CALIBRATION mode. These steps must be performed when gauge is in UNSEALED mode.



17.4 Exit CALIBRATION Mode

This sequence takes gauge out of CALIBRATION mode. These steps must be performed when gauge is in UNSEALED mode.



17.5 CC Offset

Use MAC commands for ***CC Offset*** calibration. The host system does not need to write information to the data flash (DF). See Section 15.1.1.1 for the description of the *CONTROL_STATUS[CCA]* bit. The host system needs to make sure the fuel gauge is UNSEALED.

Note

While the device is calibrating the ***CC Offset***, the host system must not read the *CONTROL_STATUS* register at a rate greater than once every 0.5 seconds.

Note

The step labeled **Enter CALIBRATION Mode** refers to Section 17.3, *Enter CALIBRATION Mode*.

The step labeled **Exit CALIBRATION Mode** refers to Section 17.4, *Exit CALIBRATION Mode*.

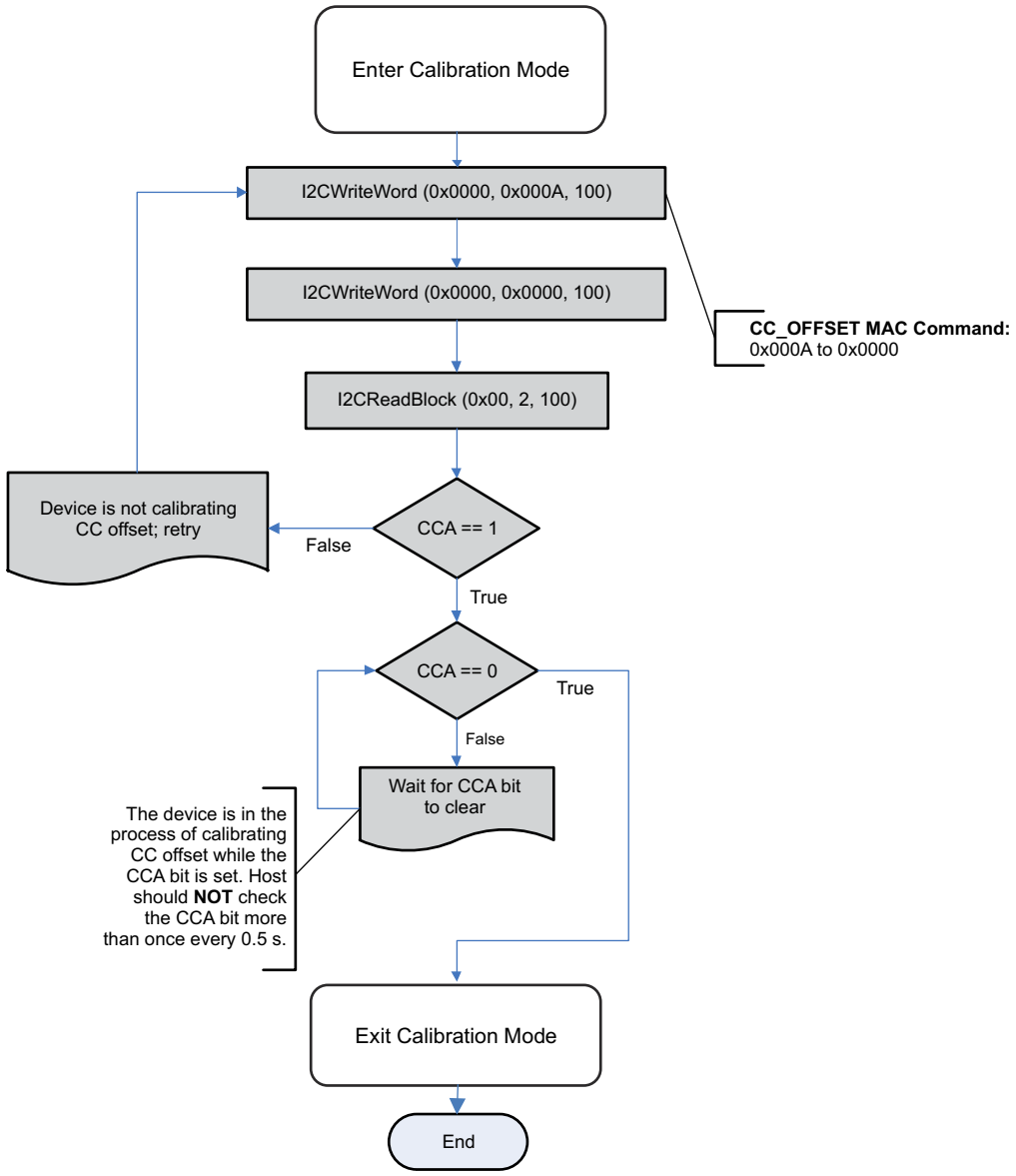


Figure 17-1. CC Offset Flow

17.6 Board Offset

Use MAC commands for ***Board Offset*** calibration. The host system does not need to write information to the DF. The host system needs to make sure the fuel gauge is UNSEALED. See Section 15.1.1.1 for the description of the *CONTROL_STATUS[CCA]* and *[BCA]* bits. Note that calculating the ***Board Offset*** will also calculate the ***CC Offset***; therefore, it is not necessary to go through the ***CC Offset*** calibration process if the ***Board Offset*** calibration process is implemented.

Note

While the device is calibrating the ***CC Offset***, the host system should not read the *CONTROL_STATUS()* register at a rate greater than once every 0.5 seconds.

Note

The step labeled **Enter CALIBRATION Mode** refers to Section 17.3, *Enter CALIBRATION Mode*.

The step labeled **Exit CALIBRATION Mode** refers to Section 17.4, *Exit CALIBRATION Mode*.

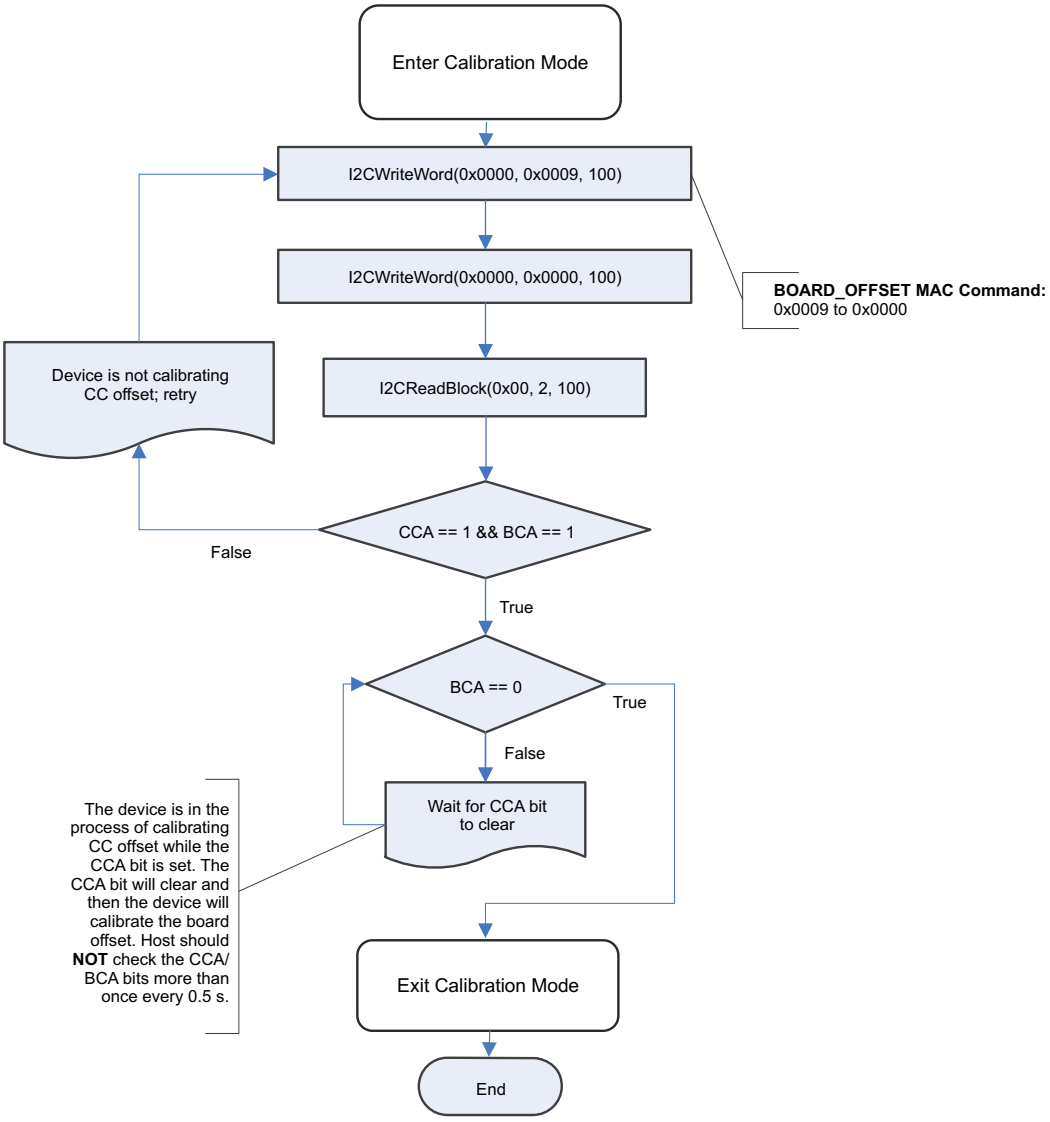


Figure 17-2. Board Offset Flow

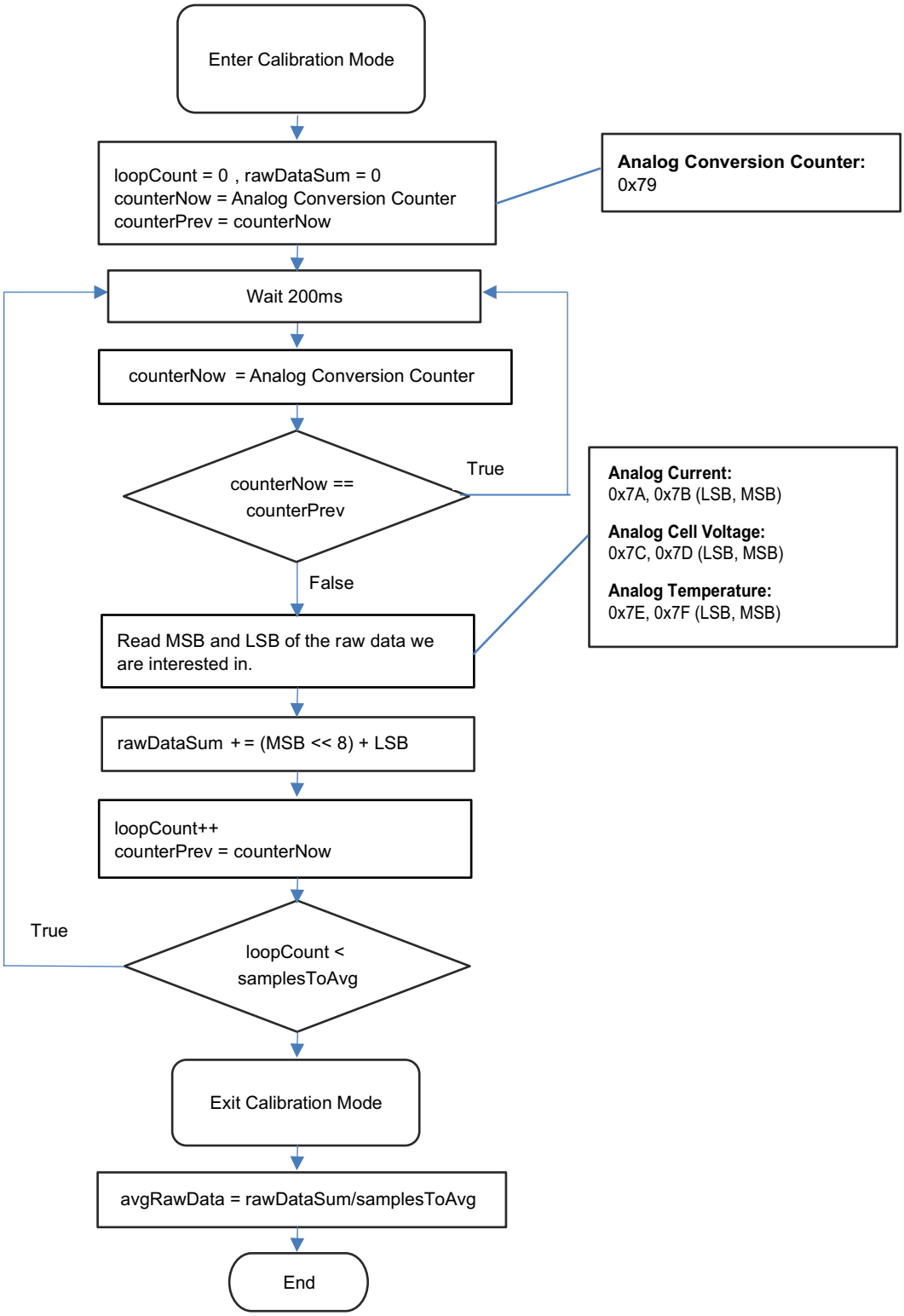
17.7 Obtain Raw Calibration Data

The following flowchart demonstrates how the host system obtains the raw data to calibrate current, voltage, and temperature. The host system uses this flow in conjunction with the Current, Voltage, and Temperature flows described in this appendix. It is recommended that the host system samples the raw data multiple times at a rate of once per second to obtain an average of the raw current, voltage, and temperature. The host system needs to make sure the fuel gauge is UNSEALED.

Note

The step labeled **Enter CALIBRATION Mode** refers to Section 17.3, *Enter CALIBRATION Mode*.

The step labeled **Exit CALIBRATION Mode** refers to Section 17.4, *Exit CALIBRATION Mode*.



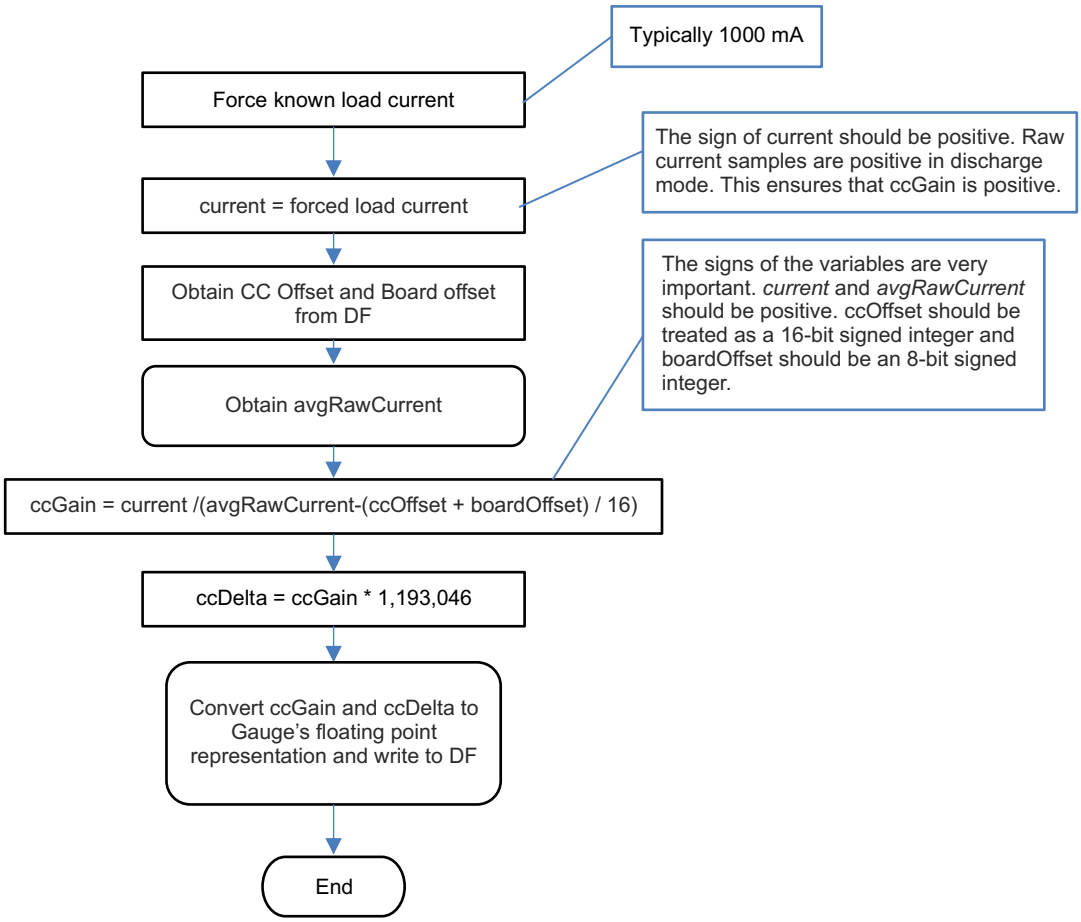
17.8 Current Calibration

The ***CC Gain*** and ***CC Delta*** are two calibration parameters of concern for current calibration. A known load, typically 1000 mA, is applied to the device during this process. Details on converting the ***CC Gain*** and ***CC Delta*** to floating point format are in Section 17.11, *Floating Point Conversion*. The host system needs to ensure the fuel gauge is UNSEALED.

Note

The step labeled **Obtain avgRawCurrent** refers to Section 17.7, *Obtain Raw Calibration Data*.

The step labeled **Convert ccGain and ccDelta to Gauge's floating point representation and write to DF** refers to Section 17.11, *Floating Point Conversion*.

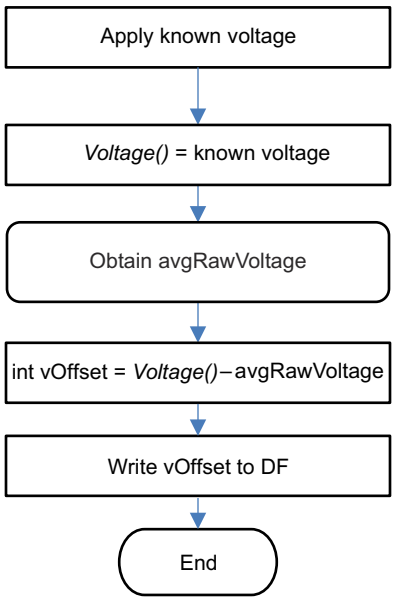


17.9 Voltage Calibration

A known voltage must be applied to the device for voltage calibration. The calculated voltage offset must be written to the corresponding location in DF. The voltage offset is represented by an integer that is a single byte in size and can be written to the appropriate location in DF without any intermediate steps. The host system needs to ensure the fuel gauge is UNSEALED.

Note

The step labeled **Obtain avgRawVoltage** refers to Section 17.7, *Obtain Raw Calibration Data*.

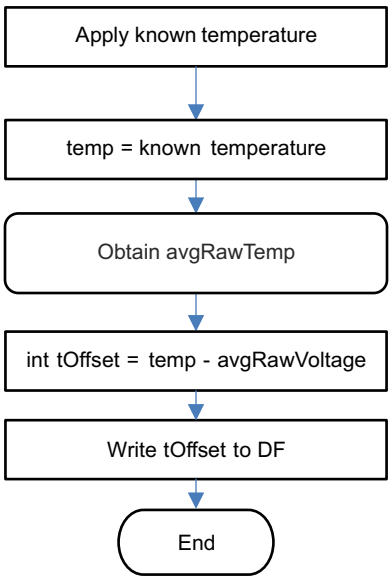


17.10 Temperature Calibration

A known temperature must be applied to the device for temperature calibration. The calculated temperature offset is written to the corresponding location in DF. The temperature offset is represented by an integer that is a single byte in size and can be written to the appropriate location in DF without any intermediate steps. The host system needs to ensure the fuel gauge is unsealed.

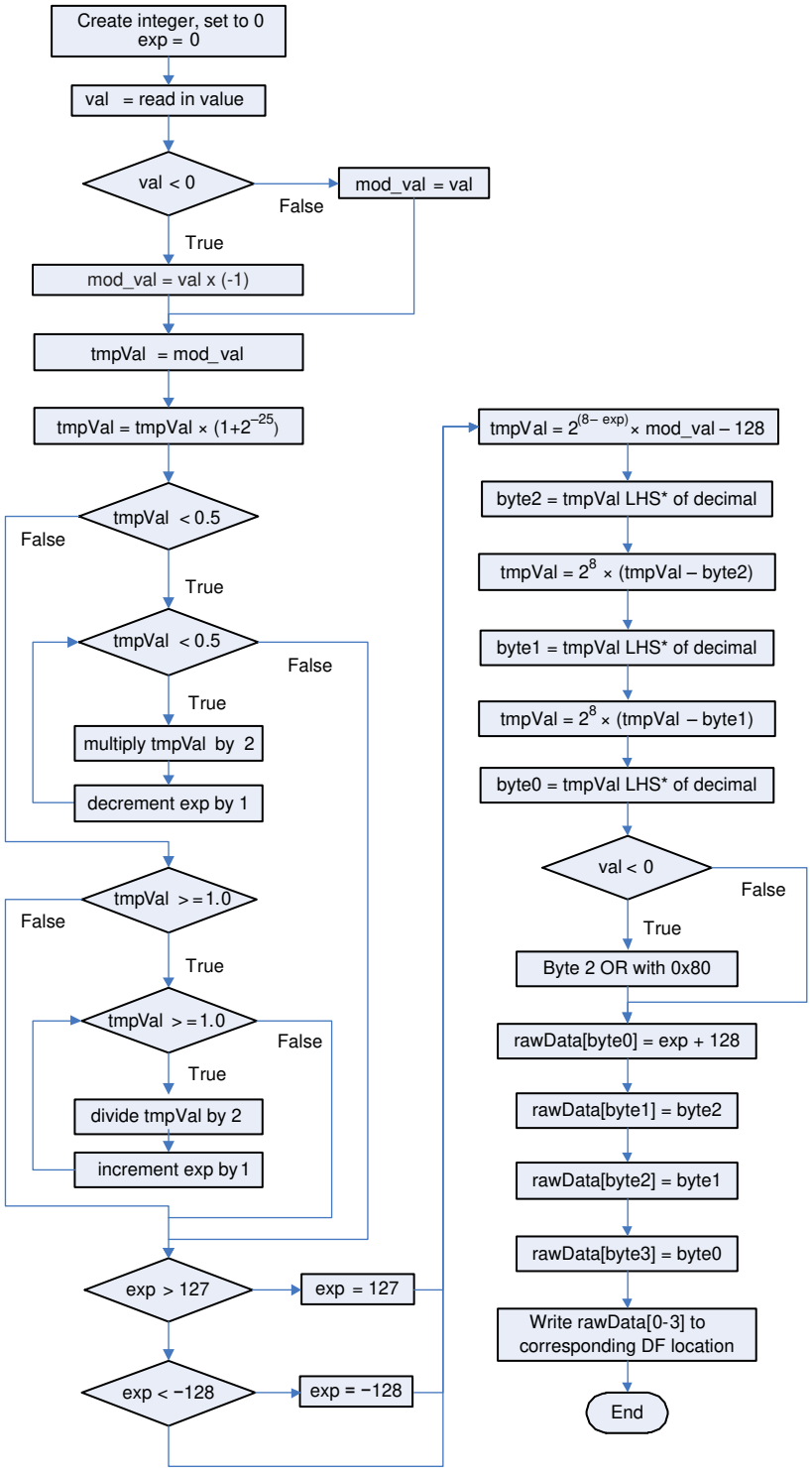
Note

The step labeled **Obtain avgRawTemp** refers to Section 17.7, *Obtain Raw Calibration Data*.



17.11 Floating Point Conversion

This section details how to convert the floating point ***CC Gain*** and ***CC Delta*** values to the format understood by the gauge.



* LHS is an abbreviation for Left Hand Side. This refers to truncating the floating point value by removing anything to the right of the decimal point.

This page intentionally left blank.

Chapter 18

Updating the BQ27542-G1 Firmware

The BQ275xx gas gauges are flash-based and can be updated, if necessary. An example would be a pack-side device implementation in which the data flash image needs updating, but I^2C is no longer accessible and HDQ communication must make the update. This section provides all the information necessary to implement a system firmware that sends the necessary I^2C or HDQ commands to update the firmware or data flash in the BQ27542-G1. This section must be used along with a file of .BQFS type for firmware plus data flash updates or a file of .DFFS type for data-flash-only updates.

Note

To get sample C code that can interact with the gauge (read/write registers and commands) as well as configure the gauge, go to this link: http://processors.wiki.ti.com/index.php/Linux/Android_Software_Solutions_for_TI_Single-cell_Gas_Gauges.

The BQtool utility enables configuration of the fuel gauge by using either the FlashStream method (parsing a .gm.fs file) or by using a .gg.csv file that includes the parameter locations.

18.1 Data Flash Stream (.DFFS)/BMS Data Flash Stream (.BQFS) Files

With Battery Management Studio (BQStudio), users can generate specific instruction files called DFFS or BQFS files, which contain a series of I^2C or HDQ commands that a host can send to a device to program the flash memory in the gauge. The commands in these files are largely ROM commands.

File Type	Description
BQFS	Can be used to program Instruction and data flash
DFFS	Can be used to program data flash

Both of these files are ASCII text files that contain commands and data. Each line of the file represents one command and potentially 96 bytes of data, as described in the following text. No row contains more than 96 data bytes. The first two characters of each row represent the command, followed by a ":".

"W:" — Indicates that the row is a command to write one or more bytes of data.

"C:" — Indicates that the row is a command to Read and Compare one or more bytes of data.

"X:" — Indicates that the row is a command to wait a given number of milliseconds before proceeding.

White space is used to separate fields within the files. Each row contains only *one* of the four commands. The commands discussed in this section can be implemented by a system that can perform multibyte operations for I^2C or single-byte operations for I^2C.

Figure 18-1 shows a typical DFFS file snippet generated from BQStudio.

```
;-----------------------------------------------------------
;Verify Existing Firmware Version
;-----------------------------------------------------------
W: AA 00 01 00
C: AA 00 20 05
W: AA 00 02 00
C: AA 00 29 03
;-----------------------------------------------------------
;Unseal device
;-----------------------------------------------------------
W: AA 00 14 04
W: AA 00 72 36
W: AA 00 FF FF
W: AA 00 FF FF
X: 1000
;-----------------------------------------------------------
;Go To ROM Mode
;-----------------------------------------------------------
W: AA 00 00 0F
X: 1000
W: 16 00 03 00 00
W: 16 64 03 00
X: 20
C: 16 66 00
W: 16 00 02 00 00 00 EA FF 33 06 FA 33 8C FA 33 39 FE 33 41 FE
W: 16 64 C9 34
X: 2
C: 16 66 00
W: 16 00 02 01 00 00 61 A2 0E BD A5 0E 13 A4 0E CA FF 3A FE A1
W: 16 64 89 2E
X: 2
```

Figure 18-1. BQStudio DFFS File Snippet

18.2 Write Command

The write command "W:" instructs the I^{2}C master to write one or more bytes to a given I^{2}C address and given register address. The I^{2}C address format used throughout this document is based on an 8-bit representation of the address. The format of this sequence is:

```
"W: I2CAddr RegAddr Byte0 Byte1 Byte2…".
```

For example, the following:

```
W: AA 55 AB CD EF 00
```

indicates that the I^{2}C master writes the byte sequence 0xAB 0xCD 0xEF 0x00 to register 0x55 of the device addressed at 0xAA.

More precisely, it indicates to write the following data to the device address 0xAA:

0xAB to register 0x55

0xCD to register 0x56

0xEF to register 0x57

0x00 to register 0x58

18.3 Read and Compare Command

The Read and Compare command is formatted identically to the write command. The data presented with this command should match the data read exactly or the operation could cease with an error indication to the user. The .gm.fs file contains no information about program flow or decision making. If a Read and Compare command results in data that does not match the expected values, then the interpreting program needs to handle the next step itself. It should not continue with further commands, but would typically go back to the beginning of the .gm.fs file and try again several times before giving up.

The format of this sequence is:

```
"C: i2cAddr RegAddr Byte0 Byte1 Byte2".
```

An example of this command is as follows:

```
C: AA 55 AB CD EF 00
```

This example expects the master to read back 4 bytes from the register address 0x55 of the device, addressed at 0xAA, and then compare the data to the values given on the line command in this same order as 0xAB, 0xCD, 0xEF, and 0x00.

18.4 Wait Command

The wait command indicates that the host waits a minimum of the given number of milliseconds before continuing to the next row of the FlashStream file. A wait command is typically used to allow the fuel gauge processor to complete a process before proceeding to the next command in the file.

For example, the following:

```
X: 200
```

indicates that the I^2C master must wait at least 200 ms before continuing.

18.5 Firmware Updating Flow

The basic programming flow expected for users to update the firmware and/or data flash of a BQ275xx device is summarized in Figure 18-2. To call on the commands given in the BQFS of DFFS file, ensure that the target device is in ROM mode. While in ROM mode, the target device responds to the I^2C address of 0x16 (8 bit) or 0x0B (7 bit) if using I^2C. From here, the 8-bit I^2C address reference is used. To enter ROM mode, 0x0F00 must be written to register address 0x00 of the target device if in the I^2C mode, or 0x00 into register 0x00 and 0x0F into register 0x01 if in the HDQ mode. The I^2C address of the device is 0xAA while it is in NORMAL gas gauge mode (default). This step of the process is not part of the BQFS or DFFS command sequence. It is up to the user to provide the Enter ROM mode command before using the BQFS or DFFS file (Figure 18-3 and Figure 18-4).

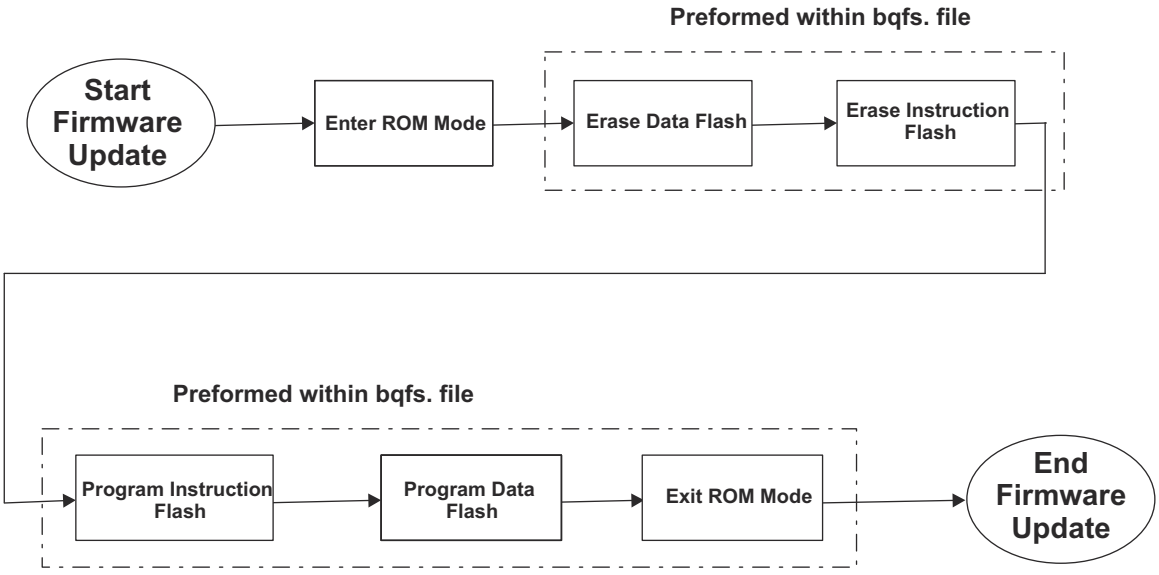


Figure 18-2. Basic Firmware Programming Flow

Before entering ROM mode, it must be confirmed that the device is not sealed from a firmware perspective. To verify that the device is not sealed, read ***Control Status*** by first writing 0x00 into register 0x00 and 0x01, and then read back register 0x01. If Bit 5 is set, then sending the Unseal keys is required. If Bit 6 is set, then it requires sending the Full Access Unseal key. These keys are sent by writing the respective keys into the ***Control*** register. The keys are 32-bit each. If both keys are required, then they must be entered in the order of Unseal key first and then Full Access Unseal key. For any 32-bit key, no intermittent communication is allowed other than the actual communication to write the keys.

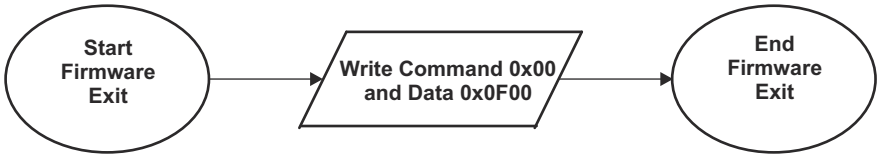


Figure 18-3. Command to Enter ROM Mode in I^2C

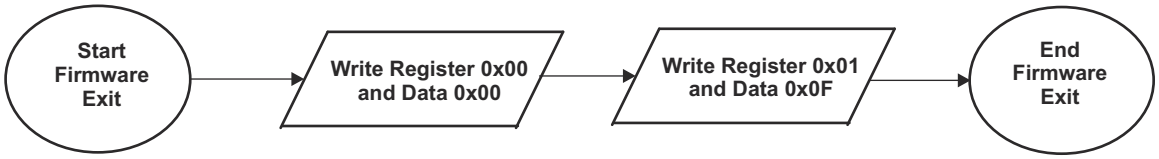


Figure 18-4. Commands to Enter ROM Mode in HDQ

In I^2C, it is evident when operating in firmware or ROM mode based on the I^2C slave address. If the device sends an Acknowledge signal to the address 0xAA, the device is indicating that it is in FIRMWARE mode; if it acknowledges the address 0x16, then it is in ROM mode. Unfortunately, HDQ does not support slave addressing. The following process can indicate in what mode the gas gauge IC is operating:

1. Try to modify register address 0x04. It must be "read only" in FIRMWARE mode.
   a. x = ReadByte(0x04)
   b. WriteByte(0x04, ~x) // Write back different data
   c. y = ReadByte(0x04)
   d. if (x == y) then ~'not in ROM mode'
2. If Step 1 indicates that ROM mode is required, then proceed with the BQFS or DFFS sequence.

Once in ROM mode, the user test setup must be able to open the BQFS or DFFS file and perform each of the I²C or HDQ transactions in strictly the same order as given with the FlashStream file. Once completing all commands within the FlashStream file, the user test system sends the Exit ROM mode procedure (Figure 18-5). At least a 250-ms delay must occur before attempting to proceed with I²C or HDQ communication to the target device using the I²C address 0xAA after exiting ROM Mode.

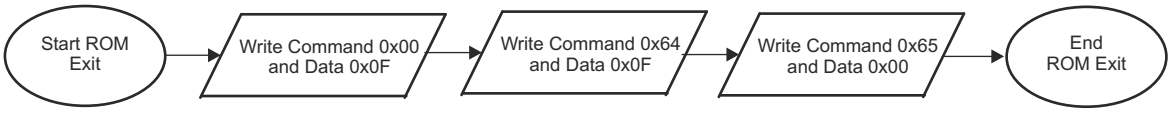


Figure 18-5. ROM Mode Exit Sequence in I²C or HDQ

18.6 Debugging BQFS Reader and Programmer

Some development effort is necessary for a user to implement a program that can open a BQFS and give out I²C or HDQ commands based on the BQFS format. During the development phase, TI recommends that a user have a dedicated prototype board that allows easy replacement of target devices. This is because during the debugging of the program, it is most likely that devices become useless.

The BQFS sequence is created so as to minimize the chances of hindering any devices in the program debug process and also during the actual production run. During debug, once a target device is placed in ROM mode, the program developer must not exit ROM mode unless the full process is completed successfully. Exiting ROM mode while the firmware or data flash is not fully programmed can place the device in an unrecoverable mode.

The BQFS file contains some sequences defined as Read and Compare. When implementing these commands within the user's program, these commands must be used as checkpoints with which the program developer decides to continue with the process or actually start from the beginning of the BQFS sequence. The user must decide how many attempts to make before considering a device unprogrammable.

18.7 Creating a BQFS and DFFS Containing User-Specific DFI

The BQFS file provided with this document is based on the latest firmware revision for a given BQ275xx device and its default data flash configuration. A user can have one of two situations in a production environment:

1. The situation requires programming a specific DFI and retaining the current version of firmware, or
2. The situation requires programming firmware and also a specific DFI. It is not time-effective for developers to use a BQFS that programs the default firmware and data flash, and then reprograms the data flash to a custom DFI.

Also, a tool is available that allows converting a senc file into a version that programs custom DFI from the beginning. It is necessary to have a DFI file to use this tool and an original senc file. A senc file is a file that actually is used to reprogram the firmware of a BQ275xx device by using the evaluation software. These files are input to the BQFS Update Tool and after executing creates a BQFS and DFFS file that contains the desired firmware with the data flash configuration specific to a user's application (Figure 18-6).

The user has the option to create a BQFS and DFFS based on I²C (default) or HDQ. The Update Tool is called from a command screen (DOS) by running the FlashStream.exe file. The command structured for the tool is displayed when calling the FlashStream.exe. The associated files used with the Update Tool must be within the same directory as the tool.

The *Going to Production With the BQ2750x Application Report* (SLUA449) describes how to create a DFI for the first time. The *Updating Firmware With the BQ2750x and EVM Application Report* (SLUA453) describes how to update a DFI that was created with an older version of firmware and now must have a DFI that is compatible with the later version firmware without requiring to go through the whole process described in SLUA449.

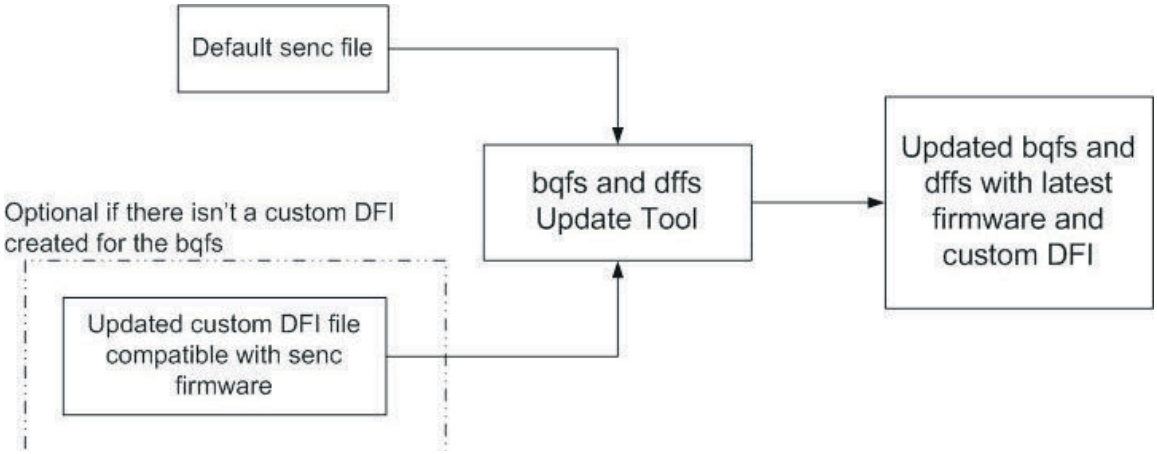


Figure 18-6. BQFS and DFFS Update Tool Process

Chapter 19

Impedance Track Gauge Configuration

19.1 Introduction

The gauge uses a combination of open circuit voltage (OCV) measurements and coulomb counting to determine the absolute state of charge (SOC).

There are typically five necessary steps, as follows, to complete prior to using a gauge with a battery:

1. Determine the ChemID.
2. Perform the learning cycle.
3. Test the gauge and optimize.
4. Finalize the golden file.
5. Program and test the PCB.

19.2 Determining ChemID

A proper ChemID or profile for a given battery must be determined for the Impedance Track algorithm.

To determine ChemID for a given battery, do the following:

1. Look up the cell/pack in the TI database to see if there is an existing ChemID already available. To do this, use BQStudio (http://www.ti.com/tool/BQStudio) with the latest chemistry plugin (http://www.ti.com/lit/zip/sluc564).
2. If an existing ChemID cannot be found, use the *Gauging Parameter Calculator Chemistry* matching tool (GPCCHEM) to submit logs and get a closely matched ChemID. For more details, see the *Simple Guide to Chemical ID Selection Tool (GPC) Application Note* (SLVA725).
3. If there is no match from either (1) or (2), cells can be sent to TI to characterize and generate a custom ChemID.

19.3 Learning Cycle

For the most optimum gauging, the pack/cell must undergo a learning cycle with the gauge. The gauge indicates the progress of the learning cycle with the following control flag bits:

1. ***VOK***: This bit tracks when the gauge measures the battery's voltage. This bit is normally set when charging/discharging starts, and is cleared when discharging stops; the gauge has detected that the battery voltage has stabilized, and the gauge has taken the OCV measurement. It is a good way to track when OCV measurements occur.
2. ***RUP_DIS***: If set, the gauge cannot determine its current state and calculations, and will not update the battery resistance tables with its presently measured data. It is cleared when a good OCV measurement is taken, as the gauge acknowledges the absolute state of charge based on this value.
3. ***FC***: This flag indicates whether or not the gauge detects the battery as "full."
4. ***Update Status***: This data flash value contains the current status of the learning cycle.

Figure 19-1 shows the voltage and current.

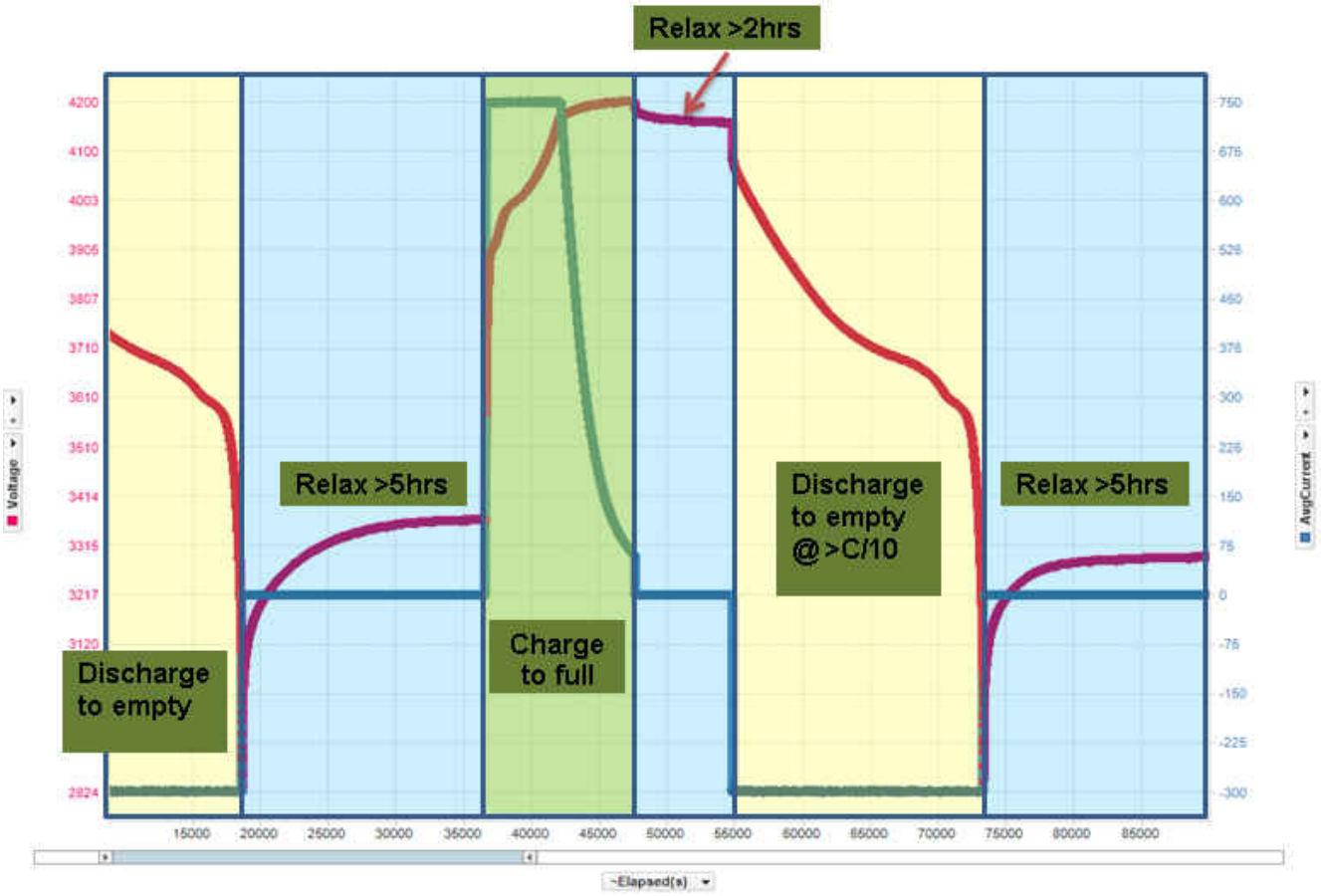


Figure 19-1. Voltage/Current

Figure 19-2 shows the value of the control flag bits and the ***Update Status*** for each state.

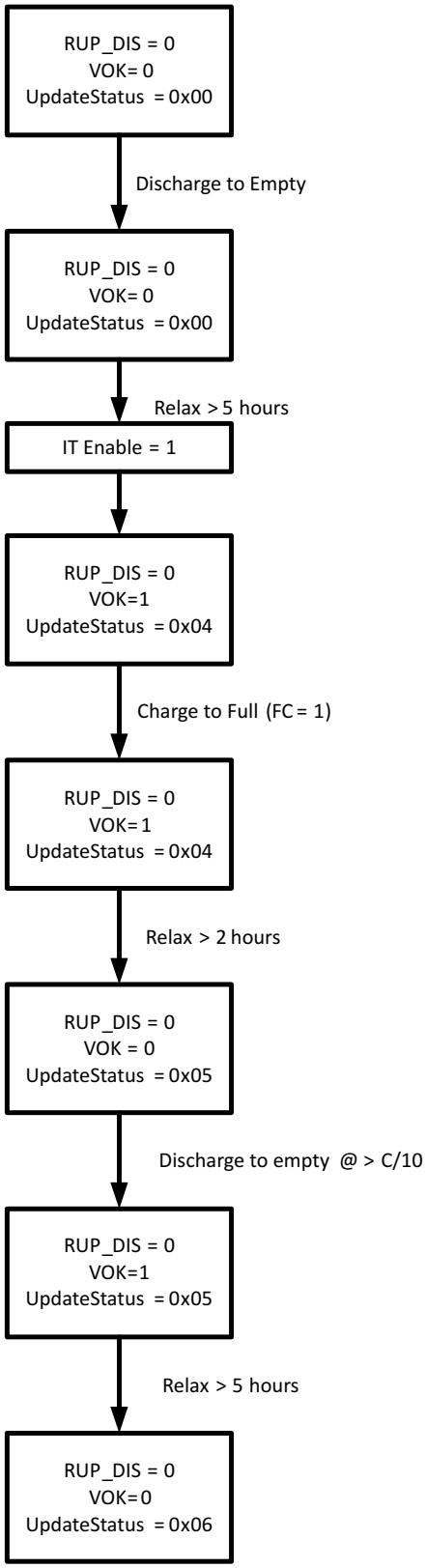


Figure 19-2. Update Status Learning Cycle Diagram

19.4 Common Problems Seen During the Learning Cycle

To diagnose a failed learning cycle, it is critical to set BQStudio to log the data RAM during the cycle every ~5–10 seconds. It is also advisable to auto-export the data flash on a less frequent basis (every ~1–10 minutes). This way, all of the information is collected to determine the point of failure.

Tips for the learning cycle are as follows:

1. Ensure that the battery is at a low SOC and relaxed when running IT_ENABLE. This is typically not a problem, but it is important.
2. During the charge, ensure that the gauge detects the "full charge" condition. If after the learning cycle, ***UPDATE_STATUS*** has not been updated to 1 (that is, it is still 0), then this may be the problem. The gauge detects the full charge condition with three criteria:
   a. Battery voltage is within 0.1 V of the "charging voltage," as defined in BQStudio.
   b. Battery current is below "taper current," as defined in BQStudio.
   c. Battery current stays below this "taper current" and above the "quit current" for over 40 seconds.

   This means that the battery must be charging with a significant current below the "taper current" for almost a minute. If the charger cuts off before or just after the current drops below the "taper current," as indicated in BQStudio, then the gauge does not detect the "full" condition.
3. When the battery is fully charged, wait long enough for the ***VOK*** bit to be cleared. This is generally two hours. Logging data RAM shows if this has occurred. ***UPDATE_STATUS*** will stay 0 if this does not happen. If ***VOK*** is never set, then this means that it did not start with an empty battery, or did not fully charge the battery, or the charge cycle was disturbed in some way (see #7) for charging advice.
4. When discharging starts, ensure that the ***VOK*** bit is set. The gauge has a data flash parameter "DSG Threshold" that determines the minimum current needed to enter the "discharge" state. If it discharges with less than this current, the ***VOK*** bit will not be cleared. The gauge will never go into the "discharge" state, and the Ra tables will never be updated. ***UPDATE_STATUS*** will be '1'. Either decrease "DSG Threshold" in the data flash, or increase the discharge rate.
5. During the discharge, the Resistance table is *never* updated. If during a learning cycle the data flash log shows that the Resistance Table never changed, this indicates that the discharge load was too light. The gauge needs to measure a significant voltage drop across the internal battery impedance before it can measure the impedance. If the load is too light, the measurement fails, and it will never get any Resistance Table updates.
6. During discharge, the Resistance Table may update for a while, and then stop. When this happens, RUP_DIS is set. This indicates that the Chemistry ID choice is incorrect. This means that the gauge has measured a resistance value that does not make sense (negative). A chemistry cycling is needed to identify the correct chemistry profile.
7. General Charge/Discharge Profile Advice: Most learning cycles fail because there is something incorrect with the charge/discharge profile. The following are suggestions:
   a. Ensure charger cutoff upon charge completion: Most users do not have a battery cycling automation setup, so a bench power supply is used to change a battery overnight. This is not recommended. The system does follow a CC/CV profile, but there is no cutoff. Therefore, when the gauge recognizes a full charge and tries to take an OCV measurement, it actually measures the supply voltage and disrupts the system.
   b. CC/CV charge profile: In line with the above, ensure that a CC/CV profile charger with reasonable values is used. C/2 Fast charge rate, C/100 to C/10 taper current.
   c. Continuous charge profile: While not strictly necessary, it is advisable to ensure that the charging profile is continuous. If the charging cycle stops, then the cycle is *may* fail. If the battery discharges for any reason during this time, the cycle will fail.
   d. After Charge, relax at least two hours with NO load/charger: Wait long enough to see ***VOK***. Two hours is generally enough.
   e. Discharge C/5 Constant Current: Make sure to use a C/5 current. This is preferred, and there can be some error. However, if the current drifts too high or too low, the cycle can fail. Too low is around C/10, too high is around C/3 to C/2. Smaller cells (<800 mAH) are much less forgiving in this regard.

f. Continuous discharge: This is absolutely necessary. If the discharge ever stops before reaching the terminate voltage, the cycle will fail.
g. Termination voltage: Make sure that when terminate voltage is reached, the load is removed. Let the cell relax. If the load kept attached to the battery and allows the battery voltage to drop well below terminate, then not only does the learning cycle fail, but it also damages the battery.

19.5 Test Gauge and Optimize

Once learning is completed, the gas gauge can be tested in an actual application environment to start optimizing the performance across temperature and load corners. TI offers the GPCRA0 tool as part of the Gauge Parameter Calculator to help users. More details can be found in the *Simple Guide to GPC Golden GG Maker Tool Application Note* (SLUUBC9). For very low temperature operations typically below 5°C system temperature distribution can deviate from lab conditions and so resistance temperature compensation parameters need to be adjusted. TI offers the GPCRB tool as part of the Gauge Parameter Calculator to help achieving this. More details can be found in the *Golden GG Maker and Resistance Temperature Application Note* (SLUUBD0).

19.6 Finalize Golden File

After all the optimizations have been performed and validation done, the golden file can be created and exported from BQStudio.

19.7 Program and Test the PCB

Once the final golden file (.gg) is ready, it can be programmed to the battery pack PCB and then tested directly with the battery pack.

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from Revision A (May 2016) to Revision B (December 2022)	Page
• Changed Standard Commands	79

Changes from Revision * (May 2015) to Revision A (May 2016)	Page
• Added or deleted content throughout to streamline the document	9
• Changed the organization of this document	11
• Changed Chapter 1	11
• Changed Chapter 2	13
• Changed Section 3.1.3	16
• Deleted Power Subclass	18
• Changed the register bits and bit descriptions in Section 4.2.1	19
• Changed Section 6.3.1	38
• Changed Section 6.4	39
• Changed Section 6.6	42
• Changed Section 6.9.2	46
• Changed Section 6.9.3	46
• Changed Section 6.9.4	46
• Changed Section 9.3	63
• Changed Section 10.2	65
• Changed Chapter 12	71
• Changed Chapter 13	73
• Changed Table 13-1	73
• Changed Table 13-2	73
• Changed Table 13-3	73
• Added Section 14.4	76
• Changed DEVICE_TYPE in Table 15-2	80
• Changed Section 15.1.1.11	82
• Changed Section 15.1.1.19	83
• Changed Section 15.1.1.20	83
• Changed *Section 15.1.1.22*	83
• Added Section 15.1.14, *SafetyStatus(): 0x1A and 0x1B*	85
• Changed equation in Section 16.2.1	91
• Changed *Section 16.2.3*	92
• Changed Section 17.5	104
• Changed Section 17.6	105
• Added *Chapter 18*	113
• Added *Chapter 19*	119

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265