Application Note

BQ41xxx Production Calibration Guide

ABSTRACT

This application note details manufacture testing, cell voltage calibration, BAT voltage calibration, PACK voltage calibration, current calibration (CC), and temperature calibration for the bq41xxx devices.

Table of Contents

Trademarks

1 Manufacture Testing

To improve the manufacture testing flow, the gas gauge device allows certain features to be toggled on or off through *ManufacturerAccess()* commands. For example, the PRE-CHG FET(), CHG FET(), DS FET(), Lifetime Data Collection(), Calibration() features. Enabling only the feature under test can simplify the test flow in production by avoiding any feature interference. These toggling commands only set the RAM data, meaning the conditions set by the these commands are cleared if a reset or seal is issued to the gauge. The *ManufacturingStatus()* keeps track of the status (enabled or disabled) of each feature.

The data flash *ManufacturingStatus* provides the option to enable or disable individual features for normal operation. Upon a reset or a seal command, the *ManufacturingStatus()* is re-loaded from data flash *ManufacturingStatus()*. This also means if an update is made to *ManufacturingStatus()* to enable or disable a feature, the gauge only takes the new setting if a reset or seal command is sent.

2 Calibration

The device has integrated routines that support calibration of current, voltage, and temperature readings, accessible after writing 0xF081 or 0xF082 to *ManufacturerAccess()* when the *ManufacturingStatus()*[CAL] bit is ON. While the calibration is active, the raw ADC data is available on *ManufacturerData()*. The device stops reporting calibration data on *ManufacturerData()* if any other MAC commands are sent or the device is reset or sealed.

Note

The *ManufacturingStatus()*)[CAL] bit must be turned OFF after calibration is completed. This bit is cleared at reset or after sealing.

ManufacturerAccess()	Description
0x002D	Enables/Disables *ManufacturingStatus()* [CAL]
0xF080	Disables raw ADC data output on *ManufacturerData()*
0xF081	Outputs raw ADC data of voltage, current, and temperature on *ManufacturerData()*
0xF082	Outputs raw ADC data of voltage, current, and temperature on *ManufacturerData()*. This mode enables an internal short on the coulomb counter inputs (SRP, SRN).

The *ManufacturerData()* output format is: ZZYYaaAAbbBBccCCddDDeeEEffFFggGGhhHHiiIjjJJkkKK,

where:

Value	Format	Description
ZZ	byte	8-bit counter, increments when raw ADC values are refreshed (every 250ms)
YY	byte	Output status *ManufacturerAccess()* = 0xF081: 1 *ManufacturerAccess()* = 0xF082: 2
AAaa	2's comp	Current (coulomb counter)
BBbb	2's comp	Cell voltage 1
CCcc	2's comp	Cell voltage 2
DDdd	2's comp	Cell voltage 3
EEee	2's comp	Cell voltage 4
FFff	2's comp	PACK voltage
Value	Format	Description
GGgg	2's comp	BAT Voltage
HHhh	2's comp	Cell current 1
IIii	2's comp	Cell current 2
JJjj	2's comp	Cell current 3
KKkk	2's comp	Cell current 4

2.1 Cell Voltage Calibration

Figure 2-1 illustrates cell voltage calibration.

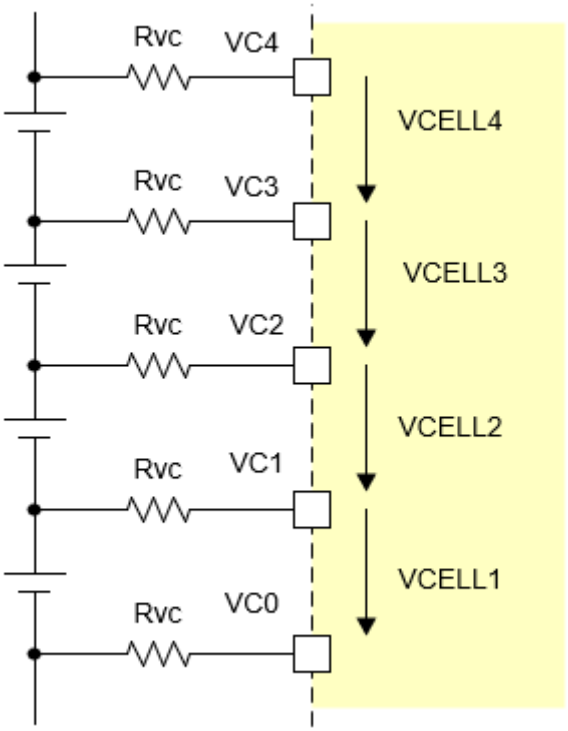

Figure 2-1. Cell Voltage Calibration

TI bq41zXXX devices have two processes to calibrate the cell voltage, which are Global Cell Gain Calibration and Individual Cell Gain Calibration. Global Cell Gain Calibration computes and applies a singular average gain value across all cells, while Individual Cell Gain Calibration will compute and apply individual gain values based on each cell.

For best accuracy, TI recommends to use the Individual Cell Gain Calibration process for calibrating the cell voltages.

For both processes, the first step is identical:

1. Apply known voltages in mV to the cell voltage inputs:
   - V_{CELL1} between VC1 pin and VSS pin
   - V_{CELL2} between VC2 pin and VC1 pin
   - V_{CELL3} between VC3 pin and VC2 pin
   - V_{CELL4} between VC4 pin and VC3 pin

Global Cell Gain Calibration

1. If *ManufacturerStatus()[CAL]* = 0, send 0x002D to *ManufacturerAccess()* to enable the *[CAL]* flag.
2. Send 0xF081 or 0xF082 to *ManufacturerAccess()* to enable raw cell voltage output on *ManufacturerData()*.
3. Poll *ManufacturerData()* until the 8-bit counter value increments by 2 before reading data.
4. Read the ADC conversion readings of cell voltages from *ManufacturerData()*:

   ADC_{CELL1} = BBbb of *ManufacturerData()*

   Is ADC_{CELL1} < 0x8000? If yes, use ADC_{CELL1}; otherwise, ADC_{CELL1} = –(0xFFFF – BBbb + 0x0001).
5. Average several readings for higher accuracy. Poll *ManufacturerData()* until ZZ increments, to indicate that updated values are available:

   ADC_{CELL1} = [ADC_{CELL1}(reading n) + … + ADC_{CELL1}(reading 1)]/n

6. Average all of the cells to create a single cell gain with the average all voltages:

$$Cell\ \ Gain\ \ =\ \ \frac{V_{cell1}+V_{cell2}+V_{cell3}+V_{cell4}}{ADC_{cell1}+ADC_{cell2}+ADC_{cell3}+ADC_{cell4}}\times 2^{16} \quad (1)$$

7. Write the new Cell Gain value to data flash.
8. Re-check voltage readings and if these are not accurate, repeat steps 4 – 8.
9. Send 0x002D to *ManufacturerAccess()* to clear the *[CAL]* flag if all calibration is complete.

Individual Cell Gain Calibration

1. If *ManufacturerStatus()[CAL]* = 0, send 0x002D to ManufacturerAccess() to enable the *[CAL]* flag.
2. Convert the known voltages being applied to each cell connection to hexadecimal.
3. Send the known voltage in hexadecimal (little endian) at each cell connection to the device by using the 0x0341 command to ManufacturerAccess(). The input must be as follows:**[Device Address + Start Address +Length + 0x0341 Command+ Applied Cell 1 Voltage + Applied Cell 2 Voltage + Applied Cell 3 Voltage + Applied Cell 4 Voltage]** If there are any cells not in use or not intended to be calibrated, input the expected voltage as 0x0000.

   Example: Calibrating a 4S pack where all known Cell Voltages are 4000mV (0x0FA0):
   0x0B 0x44 0x0A 0x41 0x03 0xA0 0x0F 0xA0 0x0F 0xA0 0x0F 0xA0 0x0F
4. Re-check voltage readings and if these are not accurate, repeat steps 1 – 3.
5. Send 0x002D to ManufacturerAccess() to clear the [CAL] flag if all calibration is complete.

2.2 BAT Voltage Calibration

BAT Voltage Calibration is shown in Figure 2-2.

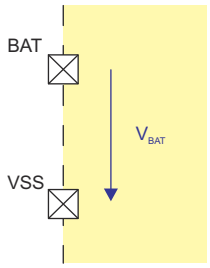

Figure 2-2. BAT Voltage Calibration

1. Apply known voltages in mV to the voltage input:
   - VBAT between VC4 pin and VSS pin
2. If *ManufacturerStatus()[CAL]* = 0, send 0x002D to *ManufacturerAccess()* to enable the *[CAL]* flag.
3. Send 0xF081 or 0xF082 to *ManufacturerAccess()* to enable raw cell voltage output on *ManufacturerData()*.
4. Poll *ManufacturerData()* until the 8-bit counter value increments by 2 before reading data.
5. Read ADC conversion readings of cell stack voltage from *ManufacturerData()*:
   - ADCBAT = GGgg of *ManufacturerData()*,
6. Average several readings for higher accuracy. Poll *ManufacturerData()* until ZZ increments to indicate that updated values are available:
   - ADCBAT = [ADCBAT(reading n) + … + ADCBAT(reading 1)]/n
7. Calculate gain value:

$$BAT\ Gain\ =\ \frac{V_{BAT}}{ADC_{BAT}}\times 2^{16} \quad (2)$$

8. Write the new BAT Gain value to data flash.
9. Re-check voltage readings and if these are not accurate, repeat steps 4 – 6.
10. Send 0x002D to *ManufacturerAccess()* to clear the *[CAL]* flag if all calibration is complete.

2.3 PACK Voltage Calibration

PACK voltage calibration is illustrated in Figure 2-3.

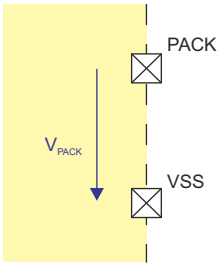

Figure 2-3. PACK Voltage Calibration

1. Apply known voltages in mV to the voltage input:
   - V_{PACK} between PACK pin and VSS pin
2. If *ManufacturerStatus()[CAL]* = 0, send 0x002D to *ManufacturerAccess()* to enable the [CAL] flag.
3. Send 0xF081 or 0xF082 to *ManufacturerAccess()* to enable raw cell voltage output on *ManufacturerData()*.
4. Poll *ManufacturerData()* until the 8-bit counter value increments by 2 before reading data.
5. Read ADC conversion readings of pack voltage from *ManufacturerData()* :
   - ADC_{PACK} = FFff of *ManufacturerData()*
6. Average several readings for higher accuracy. Poll *ManufacturerData()* until ZZ increments to indicate that updated values are available:
   - ADC_{PACK} = [ADC_{PACK}(reading n) + … + ADC_{PACK}(reading 1)]/n
7. Calculate gain value:

$$PACK\ Gain\ =\ \frac{V_{PACK}}{ADC_{PACK}}\ \times\ 2^{16} \tag{3}$$

8. Write the new PACK Gain value to data flash.
9. Re-check voltage readings and if these are not accurate, repeat steps 4 – 6.
10. Send 0x002D to *ManufacturerAccess()* to clear the *[CAL]* flag if all calibration is complete.

2.4 Current Calibration

A diagram of current calibration is shown in Figure 2-4.

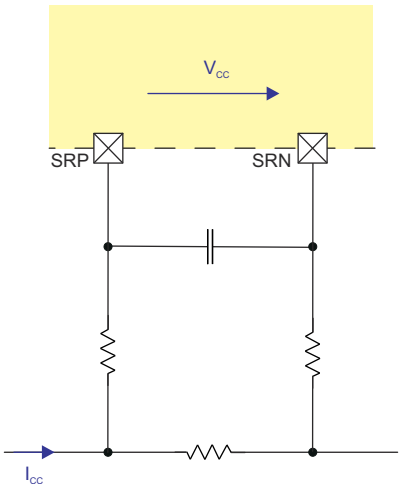

Figure 2-4. Current Calibration

2.4.1 CC Offset Calibration

Note

Due to hardware improvements in this device, CC Offset calibration is not necessary. Only run the CC Offset Calibration procedure if current is observed when no current is present.

1. Apply a known current of 0mA, and verify no current is flowing through the sense resistor connected between the SRP and SRN pins.
2. Externally short the SRN and SRP pins together for best results during CC offset calibration.
3. If *ManufacturerStatus()[CAL]* = 0, send 0x002D to *ManufacturerAccess()* to enable the [CAL] flag.
4. Send 0xF081 to *ManufacturerAccess()* to enable raw current output on *ManufacturerData()*.
5. Poll *ManufacturerData()* until ZZ increments by 2 before reading data.
6. Obtain the ADC conversion readings of current from *ManufacturerData()*:
   - ADC_{CC} = AAaa of *ManufacturerData()*

   Is ADC_{CC} < 0x8000? If yes, use ADC_{CC}; otherwise, ADC_{CC} = –(0xFFFF – AAaa + 0x0001).
7. Average several readings for higher accuracy. Poll *ManufacturerData()* until ZZ increments to indicate that updated values are available:
   - ADC_{CC} = [ADC_{CC}(reading n) + … + ADC_{CC}(reading 1)]/n
8. Read *Coulomb Counter Offset Samples* from data flash.
9. Calculate offset value:
   - CC offset = ADC_{CC} × (Coulomb Counter Offset Samples)
10. Write the new *CC Offset* value to data flash.
11. Re-check the current reading and if not accurate, repeat steps 1 – 10.
12. Send 0x002D to *ManufacturerAccess()* to clear the *[CAL]* flag if all calibration is complete.

2.4.2 Board Offset Calibration

Note

Due to hardware improvements in this device, Board Offset calibration is not necessary. Only run the Board Offset Calibration procedure if board offset current is observed.

1. Verify that Offset Calibration was performed first.
2. Apply a known current of 0mA, and verify no current is flowing through the sense resistor connected between the SRP and SRN pins.
3. If *ManufacturerStatus()[CAL]* = 0, send 0x002D to *ManufacturerAccess()* to enable the *[CAL]* flag.
4. Send 0xF081 to *ManufacturerAccess()* to enable raw current output on *ManufacturerData()*.
5. Poll *ManufacturerData()* until ZZ increments by 2 before reading data.
6. Obtain the ADC conversion readings of current from *ManufacturerData()*:
   - ADC_{CC} = AAaa of *ManufacturerData()*

   Is ADC_{CC} < 0x8000? If yes, use ADC_{CC}; otherwise, ADC_{CC} = –(0xFFFF – AAaa + 0x0001).
7. Average several readings for higher accuracy. Poll *ManufacturerData()* until ZZ increments to indicate that updated values are available:
   - ADC_{CC} = [ADC_{CC}(reading n) + … + ADCCC(reading 1)]/n
8. Read *Coulomb Counter Offset Samples* from data flash.
9. Calculate offset value:
   - Board offset = (ADC_{CC} – CC Offset) × Coulomb Counter Offset Samples
10. Write the new *Board Offset* value to data flash.
11. Re-check the current reading. If the reading is not accurate, repeat steps 1 – 10.
12. Send 0x002D to *ManufacturerAccess()* to clear the *[CAL]* flag if all calibration is complete.

2.4.3 CC Gain Calibration

1. Apply a known current (typically 1A to 2A), and verify ICC is flowing through the sense resistor connected between the SRP and SRN pins.
2. If *ManufacturerStatus()[CAL]* = 0, send 0x002D to *ManufacturerAccess()* to enable the [CAL] flag.
3. Send 0xF081 to *ManufacturerAccess()* to enable raw CC output on *ManufacturerData()*.
4. Poll *ManufacturerData()* until ZZ increments by 2 before reading data.
5. Read the ADC conversion readings of current from ManufacturerData():
   - ADC_{CC} = AAaa of *ManufacturerData()*

   Is ADC_{CC} < 0x8000? If yes, use ADC_{CC}; otherwise, ADC_{CC} = –(0xFFFF – AAaa + 0x0001).
6. Average several readings for higher accuracy. Poll *ManufacturerData()* until ZZ increments to indicate that updated values are available:
   - ADC_{CC} = [ADC_{CC}(reading n) + … + ADC_{CC}(reading 1)]/n
7. Read Coulomb Counter Offset Samples from data flash.
8. Calculate gain values:

$$CC\ \ Gain\ \ = \frac{I_{cc}}{ADC_{cc} - \frac{Board\ \ Offset\ \ +\ \ CC\ \ Offset}{Coulomb\ \ Counter\ \ Offset\ \ Samples}} \times 2^{16} \tag{4}$$

9. Write the new CC Gain and value to data flash.
10. Re-check the current reading. If the reading is not accurate, repeat steps 1 – 9.
11. Send 0x002D to *ManufacturerAccess()* to clear the *[CAL]* flag if all calibration is complete.

Note

Capacity Gain is no longer used in the BQ41xxx product family and must not be used or modified.

2.5 Temperature Calibration

Figure 2-5 illustrates temperature calibration.

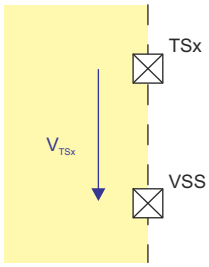

Figure 2-5. Temperature Calibration

2.5.1 Internal Temperature Sensor Calibration

1. Apply a known temperature in 0.1°C, and verify that temperature $Temp_{TINT}$ is applied to the device.
2. Read the TINT $offset_{old}$ from Internal Temp Offset.
3. Read the reported temperature from *DAStatus2()*:
   - TINT = AAaa of *DAStatus2()*Is TINT > 0? If yes, TINT = AAaa – 2732.
4. Calculate temperature offset:

$$TINT\ offset\ =\ TEMP_{TINT}\ -\ TINT\ +\ TINT\ offset_{old} \tag{5}$$

5. Write the new Internal Temp Offset value to data flash.
6. Re-check the *DAStatus2()* reading. If the reading is not accurate, repeat steps 1 – 5.

2.5.2 TS1–TS2–TS3–TS4 Calibration

1. Apply a known temperature in 0.1°C, and verify that temperature $TEMP_{TSx}$ is applied to the thermistor connected to the TSx pin. "TSx" refers to TS1, TS2, TS3, or TS4, whichever is applicable.
2. Read the TSx $offset_{old}$ from External x Temp Offset, where x is 1, 2 ,3, or 4.
3. Read the appropriate temperature from the DAStatus2() block as TSx.
4. Calculate the temperature offset:

$$TSx\ offset\ =\ TEMP_{TSX} - TSx\ +\ TSx\ offset_{old} \quad (6)$$

   Where x is 1, 2, 3, or 4.
5. Write the new External x Temp Offset (where x is 1, 2, 3, or 4) value to data flash.
6. Re-check the DAStatus2() reading. If the reading is not accurate, repeat steps 1 – 5.

3 References

- Texas Instruments, BQ41Z50 1-Series, 2-Series, 3-Series, and 4-Series Li-Ion Battery Pack Manager, datasheet.
- Texas Instruments, BQ41Z50 Technical Reference Manual, technical reference manual.
- Texas Instruments, BQ41Z50 Li-Ion Battery Pack Manager Evaluation Module EVM user's guide.

4 Revision History

Changes from Revision * (July 2024) to Revision A (April 2026)	Page
• Added '+' sign to value of ADCCELL1 in number 5 in the Cell Voltage Calibration section	4
• Deleted factor 'VCELL1' from Equation 1	4
• Added Individual Cell Gain Calibration section	4
• Added ' + J +' to '…' in step 6 of BAT Voltage Calibration section	5

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), 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 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, TI’s General Quality Guidelines, 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. Unless TI explicitly designates a product as custom or customer-specified, TI products are standard, catalog, general purpose devices.

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

Last updated 10/2025