

Calibration and Data Flash Programming the bq33100 Supercapacitor Manager

PMP - Battery Monitoring Solutions

ABSTRACT

This application report presents a strategy for high-speed, economical calibration and data flash programming of the bq33100 advanced gas gauge. VB6 code examples are provided.

Contents

1	Introduction	2
2	Writing the Data Flash Image to Each Target Device	3
	2.1 Preparing the Data Flash Image Pack	3
	2.2 Reading and Saving the Data Flash Image	3
	2.3 Writing the Data Flash Image to Each Target Device	4
3	Calibrating the bq33100	5
4	Writing Pack-Specific Data Flash Locations	5

1 Introduction

The methods in this document are presented as VB6 (Visual Basic 6) functions. These functions were copied directly from working code. In order to read from and write to the data flash, they use four types of SMBus read and write functions. These may be duplicated in any software environment that has SMBus communication capabilities. As used in this document, each read/write function is designed for communication with a battery device, so the device address (0x16) is omitted for clarity.

1. WriteSMBusInteger() has two arguments – the SMBus command and a signed integer. Internally, this function separates the integer into two bytes for transmission by the SMBus write-word protocol.
2. WriteSMBusByteArray() has three arguments – the SMBus command, the array of bytes, and an integer specifying the length of the byte array. Internally, this function separates the byte array into separate bytes for transmission by the SMBus write-block protocol.
3. WriteSMBusCommand() has only one argument – the SMBus command.
4. ReadSMBusByteArray() has three arguments– the SMBus command, the returned array of bytes, and the returned length of the byte array. It is internally implemented with the SMBus read-block protocol.

Also used in these functions is a simple delay routine called DoDelay. VB6 code for this procedure is provided in [Section 4](#).

Error handling is not implemented in this sample code because requirements are unique and varied. Also, constants are hard-coded into the functions to improve clarity rather than documenting them in code elsewhere, as in typical coding practice.

A good strategy for bq33100 production is an eight-step process flow:

1. Power the device with a cell simulator.
2. Write the Data Flash Image to the device. This file can be built per [Section 2.2](#)
3. Update any individual flash locations, such as serial number, lot code, and date.
4. Perform any desired protection tests.
5. Connect the cells.
6. Perform additional desired protection tests.
7. Send 0x0021 to Manufacturer Access 0x00 command, enabling Lifetime and Permanent Fail functions.
8. Seal the pack.

In this document, the second and third steps are examined in detail.

2 Writing the Data Flash Image to Each Target Device

2.1 Preparing the Data Flash Image Pack

The bq33100 ICs are shipped preprogrammed with default parameter values. Create the data flash image used for every production pack by assembling a supercapacitor pack with the default firmware, and set the data flash constants for the application using the evaluation software. This includes the number of serial cells, design capacity, and charging parameters, to name a few. Using the available application notes and datasheet, insure the device is working correctly for your application. For low volume production, program additional packs by simply replacing the .gg file from within the evaluation software. The following procedure is useful only for mass production, where programming speed is important.

2.2 Reading and Saving the Data Flash Image

Note that this step only needs to be done once for a given project.

```
Function SaveDataFlashImageToFile(sFileName As String) As Long
```

```

    Dim iNumberOfRows As Integer
    Dim lError As Long
    Dim yRowData(32) As Byte
    Dim yDataFlashImage(&H400) As Byte
    Dim iRow As Integer
    Dim iIndex As Integer
    Dim iLen As Integer
    Dim iFileNumber As Integer

    '// FOR CLARITY, WITHOUT USING CONSTANTS
    '// 0x400 is the data flash size.
    '0x400 \ 32 = 32 rows
    iNumberOfRows = &H400 \ 32

    '// PUT DEVICE INTO ROM MODE
    lError = WriteSMBusInteger(&H0, &HF00)
    DoDelay 0.01

    '// READ THE DATA FLASH, ROW BY ROW
    For iRow = 0 To iNumberOfRows -1

        '// Set the address for the row. &H9 (0x09) is the ROM mode command.
        '// 0x200 is the row number where data flash starts.
        '// Multiplication by 32 gives the actual physical address where each row starts
        lError = WriteSMBusInteger(&H9, (&H200 + iRow) * 32)

        '// Read the row. &HC (0x0c) is the ROM mode command.
        lError = ReadSMBusByteArray(&HC, yRowData, iLen)

        '//Copy this row into its place in a big byte array
        For iIndex = 0 To 32 -1
            yDataFlashImage((iRow * 32) + iIndex) = yRowData(iIndex)
        Next iIndex
    Next iRow

    '// WRITE DATA FLASH IMAGE TO FILE
    iFileNumber = FreeFile
    Open sFileName For Binary Access Write As #iFileNumber
    Put #iFileNumber, , yDataFlashImage
    Close #iFileNumber

    '// EXECUTE GAS GAUGE PROGRAM
    lError = WriteSMBusCommand(&H8)
End Function
```

2.3 Writing the Data Flash Image to Each Target Device

The following method takes about 2 seconds to write the entire data flash:

CAUTION

If power is interrupted during this process, the device may become unusable.

```

Function WriteDataFlashImageFromFile(sFileName As String) As Long Dim lError As Long
  Dim iFileNumber As Integer
  Dim iNumberOfRows As Integer
  Dim iRow As Integer
  Dim iIndex As Integer
  Dim yRowData(32) As Byte
  Dim yDataFlashImage(&H400) As Byte

  '// READ THE FLASH IMAGE FROM THE FILE INTO A GLOBAL BYTE ARRAY
  iFileNumber = FreeFile
  Open sFileName For Binary Access Read As #iFileNumber
  Get #iFileNumber, , yDataFlashImage
  Close #iFileNumber

  '// FOR CLARITY, WITHOUT USING CONSTANTS
  iNumberOfRows = &H400 \ 32 '32 Rows

  '// PUT DEVICE INTO ROM MODE
  lError = WriteSMBusInteger(&H0, &HF00)
  DoDelay 0.01

  '// ERASE DATA FLASH, ROWS ARE ERASED IN PAIRS
  For iRow = 0 To iNumberOfRows -1 Step 2
    lError = WriteSMBusInteger(&H11, iRow)
    DoDelay 0.04
  Next iRow

  '// WRITE EACH ROW
  For iRow = 0 To iNumberOfRows -1
    '// Set the row to program into the first element of the 33 byte array
    yRowData(0) = iRow

    '// Copy data from the full array to the row array
    For iIndex = 0 To 31
      yRowData(iIndex + 1) = yDataFlashImage((iRow * 32) + iIndex)
    Next iIndex

    '// Write the row. Length is 33 because first byte is row number
    lError = WriteSMBusByteArray(&H10, yRowData, 32 + 1)
    DoDelay 0.02
  Next iRow

  '// EXECUTE GAS GAUGE PROGRAM
  lError = WriteSMBusCommand(&H8)
End Function

```

3 Calibrating the bq33100

Individual calibration for each bq33100-based supercapacitor pack is not recommended. The preferred technique is to manually calibrate ten units using the Evaluation Software calibration screen, then find average values for mass production calibration constants.

4 Writing Pack-Specific Data Flash Locations

The third step is to fine tune the data flash for each pack, giving it a unique identity. In the following example, the pack Serial Number is written using subclass and offset information found in the bq33100 data sheet. Modifications to single data flash locations normally require a block read of the 32-byte data flash page, updating the desired element of the block, and then writing it back to the device. The procedure is documented in the product data sheet.

```
Function WritePackSerialNumber(iSerialNumber As Integer) As Long
    Dim lError As Long
    Dim yData(32) As Byte
    Dim iLen As Integer

    '// SET THE SUBCLASS TO 48 (FOUND IN PRODUCT Technical Reference)
    lError = WriteSMBusInteger(&H77, 48)

    '// READ THE PAGE
    lError = ReadSMBusByteArray(&H78, yData(), iLen)

    '// REPLACE THE TWO BYTES AT OFFSET 4 (FOUND IN product Data Sheet) WITH NEW S/N
    yData(4) = (iSerialNumber And &HFF00) \ 256 '// modify MS byte
    yData(5) = iSerialNumber And &HFF '// modify LS byte

    '// WRITE THE PAGE BACK TO FLASH
    lError = WriteSMBusByteArray(&H78, yData(), iLen)

    '// FLASH WRITES ARE SLOW
    DoDelay 0.1
End Function
```

```
Sub DoDelay(fWaitTime As Single)
    Dim vTime As Variant
    vTime = Timer
    While Timer < (vTime + fWaitTime)
        '// fix midnight problem
        If Timer < vTime Then Exit Sub
        '// Yield to various Windows events while the delay is in progress
        DoEvents
    Wend
End Sub
```

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products

Audio	www.ti.com/audio
Amplifiers	amplifier.ti.com
Data Converters	dataconverter.ti.com
DLP® Products	www.dlp.com
DSP	dsp.ti.com
Clocks and Timers	www.ti.com/clocks
Interface	interface.ti.com
Logic	logic.ti.com
Power Mgmt	power.ti.com
Microcontrollers	microcontroller.ti.com
RFID	www.ti-rfid.com
OMAP Mobile Processors	www.ti.com/omap
Wireless Connectivity	www.ti.com/wirelessconnectivity

Applications

Automotive and Transportation	www.ti.com/automotive
Communications and Telecom	www.ti.com/communications
Computers and Peripherals	www.ti.com/computers
Consumer Electronics	www.ti.com/consumer-apps
Energy and Lighting	www.ti.com/energy
Industrial	www.ti.com/industrial
Medical	www.ti.com/medical
Security	www.ti.com/security
Space, Avionics and Defense	www.ti.com/space-avionics-defense
Video and Imaging	www.ti.com/video

TI E2E Community Home Page

e2e.ti.com

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2012, Texas Instruments Incorporated