Application Note

BQ769X2 OTP Program Example Guide With Sealed Configuration

Jie Wu

ABSTRACT

BQ76942, which supports 3-series to 10-series battery cells, and BQ76952, which supports 3-series to 16-series battery cells, are battery monitor and protector devices used in battery packs. Those two devices integrate a one-time-programmable (OTP) memory. Writing initialize settings into OTP helps save initialize configuration before system power up. Users need to write *OTP* on the product line. This application note introduces a step-by-step hardware setup and software process which helps users save time to complete OTP configuration.

Table of Contents

List of Figures

Trademarks

1 Introduction

The BQ769X2 device can be used in Li-ion, Li-polymer, and LiFePO4 battery packs. The device features include monitor battery voltage with high accuracy, configurable protection function, host controlled cell balancing, internal temperature sensing, integrated OTP memory. For users who want to permanently configure the device parameters on the product line, OTP memory can be used to store some key parameters. This application note focuses on introducing OTP programming step-by-step and make the design process easier.

2 Hardware Setup

Before beginning OTP configuration, pay attention to the BQ769X2 power supply; the voltage between pack+ and pack- must be in the range of approximately 10V-12V.

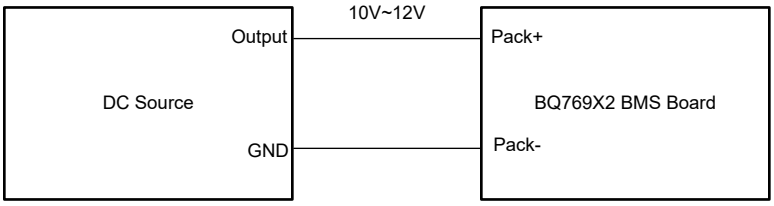

Figure 2-1. OTP Program Hardware Setup Block Diagram

3 CRC and Checksum Programming and Example Command

For the code example, which includes CRC calculation and checksum calculation in different commands. The following sections show a related software code example and application example.

3.1 CRC Calculation Example

Figure 3-1 shows the CRC software coding example.

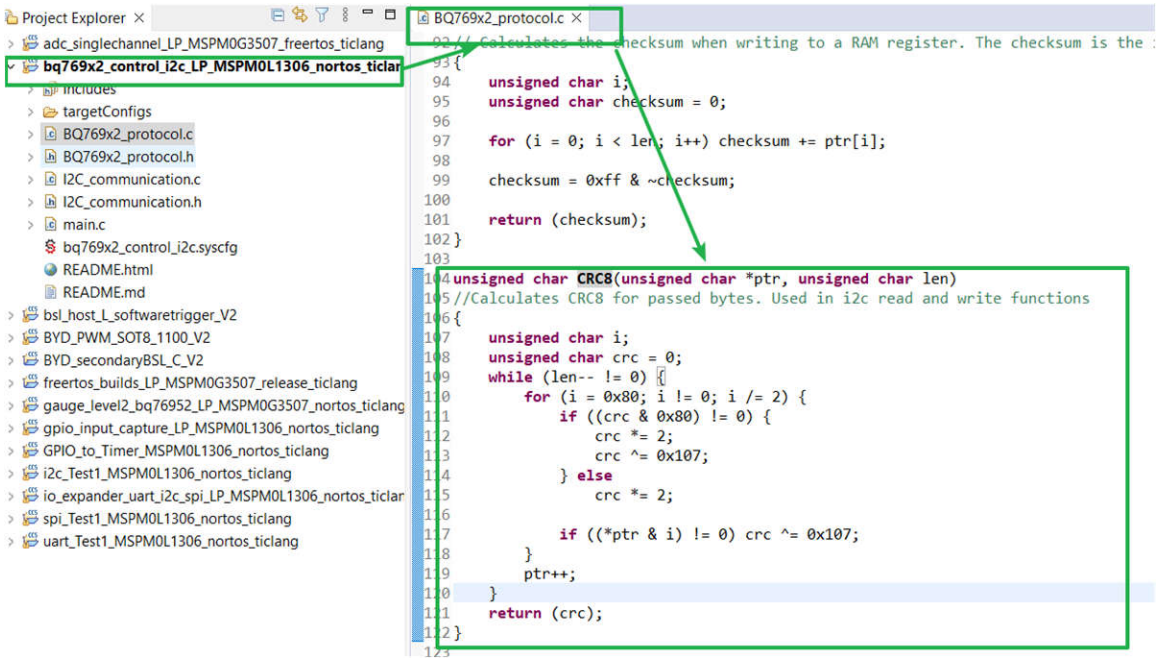

Figure 3-1. CRC Software Example Code

Figure 3-2 shows a write command format which includes a CRC calculation based on I2C communication. An example to write subcommand is 0x0022 (FET_ENABLE subcommand) with CRC calculation. The CRC for byte3 is computed for [0x10 0x3E 0x22]; the resulting CRC is 0x63. The CRC for the second byte [0x00] is 0x00. Related data sequence is 0x10 0x3E 0x22 0x63 0x00 0x00.

Byte0	Byte1	Byte2	Byte3	Byte4	Byte5
Device address	Subcommand/Data memory read or write	Register Address MSB	CRC calculation result(Byte0~Byte2)	Register Address LSB	CRC calculation result(Byte4)
0x10	0x3E	0x22	0x63	0x00	0x00

Figure 3-2. Write Command Format With CRC

Figure 3-3 shows a read command format which includes CRC. An example reads 0x14(VCell 1) with readback date 0x0B68. With CRC calculation, the CRC for the byte4 is computed for [0x10 0x14 0x11 0x68]; the resulting CRC is 0x33. The CRC for the second byte [0x0B] is 0x31. Related data sequence is 0x10 0x14 0x11 0x68 0x33 0x0B 0x31.

Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6
Device address	Register to be read	Read command	Read back data MSB	CRC calculation result(Byte0~Byte3)	Read back data LSB	CRC calculation result(Byte5)
0x10	0x14	0x11	0x68	0x33	0x0B	0x31

Figure 3-3. Read Command Format With CRC

3.2 Checksum Calculation Example

Figure 3-4 shows a software coding for checksum calculation.

```
unsigned char Checksum(unsigned char *ptr, unsigned char len)
// Calculates the checksum when writing to a RAM register. The checksum is the inverse of the sum of the bytes.
{
    unsigned char i;
    unsigned char checksum = 0;

    for (i = 0; i < len; i++) checksum += ptr[i];

    checksum = 0xff & ~checksum;

    return (checksum);
}
```

Figure 3-4. Software Coding for Checksum Calculation

Figure 3-5 shows an example for I2C command with Checksum format. An example is to write data 0x8C to register 0x9261, then write checksum data and length to 0x60/0x61. The checksum is calculated on the address and data (0x61 0x92 0x8C). The length also includes the two bytes for device address (0x10) and subcommand (0x3E), but do not include CRC bytes, for a total length of 5 in this case.

Data format is shown below:

0x10 0x3E 0x61 0xAD 0x92 0xF7 0x8C 0xAD;

0x10 0x60 0x80 0xDE 0x05 0x1B;

Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
Device address	Subcommand	Register Address MSB	CRC calculation result(Byte0~Byte2)	Register Address LSB	CRC calculation result(Byte4)	Data to write	CRC calculation result(Byte6)
0x10	0x3E	0x61	0xAD	0x92	0xF7	0x8C	0xAD
Byte0	Byte1	Byte2	Byte3	Byte4	Byte5		
Device address	Checksum buffer	Checksum result	CRC calculation result(Byte0~Byte2)	Data length	CRC calculation result(Byte4)		
0x10	0x60	0x80	0xDE	0x05	0x1B		

Figure 3-5. Command With Checksum And CRC

4 OTP Programming Example

4.1 Programming Steps for Configuration into Sealed Mode

1. Check whether OTP programming has already been done on the device by reading one of the programmed registers. When power is applied, registers either report the default values or the values programmed in OTP if OTP has been programmed. If OTP programming has not been done, then proceed to the next steps.
2. Read the 0x12 Battery Status[SEC1,SEC0] bits to verify the device is in FULL ACCESS mode (0x01).
3. If the device is in FULL ACCESS mode, then enter CONFIG_UPDATE mode - (Subcommand 0x0090). If not, unsealed device, then go back to step 2 to check if the device is in Full Access mode.
4. Configure the register settings in data memory.
5. Exit CONFIG_UPDATE mode - (Subcommand 0x0092).
6. Read the data memory registers to verify all parameters were written successfully.
7. Enter CONFIG_UPDATE mode.
8. Check the Battery Status[OTPB] bit is clear to verify OTP programming conditions are met.
9. Read OTP_WR_CHECK() (Subcommand 0x00A0). If this returns a value of 0x80, then OTP programming conditions are met.
10. If OTP_WR_CHECK indicates conditions are met, then send OTP_WRITE() subcommand (0x00A1).
11. Wait 100ms. Read from 0x40 to check if OTP programming was successful (0x80 indicates success).
12. Enter into sealed mode.
13. Exit CONFIG_UPDATE mode - (Subcommand 0x0092).

4.2 OTP Programming Flowchart

Figure 4-1 shows the OTP programing process that the device needs to enter to go into sealed mode.

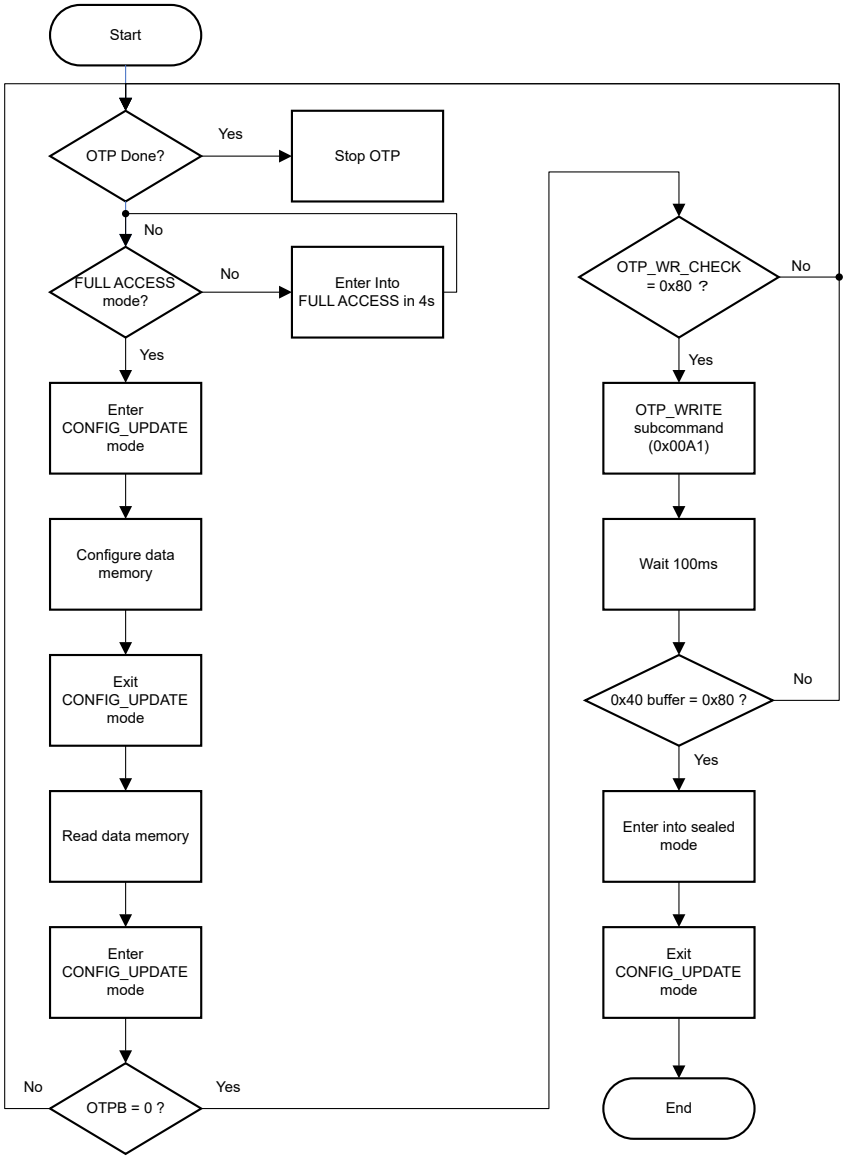

Figure 4-1. OTP Progress Flow Chart

4.3 OTP Step-by Step Command Example

1. Check whether OTP programming has already been done on the device by reading one of the programmed registers. When power is applied, registers either report the default values or the values programmed in OTP if OTP has been programmed. If OTP programming has not been done, then proceed to the following steps.
   a. Example: change the 0x9180 setting, then you can read the 0x9180 register and check if this register shows the default value. If the default value is shown, then go to the next step.
      i. 0x10 0x3E 0x80 Byte3(CRC) 0x91 Byte5(CRC).
      ii. 0x10 0x40 0x11 Byte3(Rdata) Byte4(CRC) Byte5(Rdata) Byte6(CRC).
2. Read the 0x12 Battery Status[SEC1,SEC0] bits to verify the device is in FULL ACCESS mode (0x01).
   a. Read 0x12 register, if sec1=0, sec0=1, then device is in full access mode. Go to the next step. If sec1=1, sec0=1, then the device is in sealed mode. Users need to enter into full access mode first.
      i. 0x10 0x12 0x11 Byte3(Rdata) Byte4(CRC) Byte5(Rdata) Byte6(CRC).
   b. Go into full access mode in 4 seconds before read data memory. An example is shown: Unseal_Key_Step1 0x1234; Unseal_Key_Step2 0x5678; FullAccess_Key_Step1 0x7856; FullAccess_Key_Step2 0x4321.
      i. 0x10 0x3E 0x34 Byte3(CRC) 0x12 Byte5(CRC); //Write Unseal_Key_Step1.
      ii. 0x10 0x3E 0x78 Byte3(CRC) 0x56 Byte5(CRC); //Write Unseal_Key_Step2.
      iii. 0x10 0x12 0x11 Byte3(Rdata) Byte4(CRC) Byte5(Rdata) Byte6(CRC); //Check Sec1 Sec0.
      iv. 0x10 0x3E 0x56 Byte3(CRC) 0x78 Byte5(CRC); //Write FullAccess_Key_Step1.
      v. 0x10 0x3E 0x21 Byte3(CRC) 0x43 Byte5(CRC); //Write FullAccess_Key_Step2.
      vi. 0x10 0x12 0x11 Byte3(Rdata) Byte4(CRC) Byte5(Rdata) Byte6(CRC); //Check Sec1 Sec0.
3. If the device is in FULL ACCESS mode, then enter CONFIG_UPDATE mode - (Subcommand 0x0090).
   a. Enter into CONFIG_UPDATE mode by writing subcommand 0x0090.
      i. 0x10 0x3E 0x90 Byte3(CRC) 0x00 Byte4(CRC)
4. Configure the register settings in data memory with CRC and Checksum.
   a. Configure data memory from 0x9180 to 0x925D which shows in the BQ76942 Technical Reference Manual in the *Data Memory* table.
      i. 0x10 0x3E 0x80 Byte3(CRC) 0x91 Byte5(CRC) Byte6(Wdata) Byte7(CRC) Byte8(Wdata) Byte8(CRC).
      ii. 0x10 0x60 Byte2(CheckSum) Byte3(CRC) Byte4(DataLength) Byte5(CRC).
      iii. 0x10 0x3E 0x82 Byte3(CRC) 0x91 Byte5(CRC) Byte6(Wdata) Byte7(CRC) Byte8(Wdata) Byte8(CRC).
      iv. 0x10 0x60 Byte2(CheckSum) Byte3(CRC) Byte4(DataLength) Byte5(CRC).
      v. 0x10 0x3E 0x5D Byte3(CRC) 0x92 Byte5(CRC) Byte6(Wdata) Byte7(CRC) Byte8(Wdata) Byte8(CRC).
      vi. 0x10 0x60 Byte2(CheckSum) Byte3(CRC) Byte4(DataLength) Byte5(CRC).
5. Exit CONFIG_UPDATE mode.
   a. Exit Config_Update mode by writing subcommand 0x0092.
      i. 0x10 0x3E 0X92 Byte3(CRC) 0x00 Byte5(CRC).
6. Read the data memory registers to verify all parameters were written successfully.
   a. Read data memory registers from 0x9180 to 0x925D which shows in the BQ76942 Technical Reference Manual in the *Data Memory* table.
      i. 0x10 0x3E 0x80 Byte3(CRC) 0x91 Byte5(CRC).
      ii. 0x10 0x40 0x11 Byte3(Rdata) Byte4(CRC) Byte5(Rdata) Byte6(CRC).
      iii. 0x10 0x3E 0x82 Byte3(CRC) 0x91 Byte5(CRC).
      iv. 0x10 0x40 0x11 Byte3(Rdata) Byte4(CRC) Byte5(Rdata) Byte6(CRC).
      v. 0x10 0x3E 0x5B Byte3(CRC) 0x92 Byte5(CRC).
      vi. 0x10 0x40 0x11 Byte3(Rdata) Byte4(CRC) Byte5(Rdata) Byte6(CRC).
      vii. 0x10 0x3E 0x5D Byte3(CRC) 0X92 Byte5(CRC).
      viii. 0x10 0x40 0x11 Byte3(Rdata) Byte4(CRC) Byte5(Rdata) Byte6(CRC).
7. Enter CONFIG_UPDATE mode.
   a. Enter into CONFIG_UPDATE mode by writing subcommand 0x0090.
      i. 0x10 0x3E 0x90 Byte3(CRC) 0x00 Byte5(CRC).
8. Check the Battery Status[OTPB] bit is clear to verify OTP programming conditions are met.

   a. Read 0x12 register to get OTPB bit status.
      i. 0x10 0x12 0x11 Byte3(Rdata) Byte4(CRC) Byte5(Rdata) Byte6(CRC).
9. Read OTP_WR_CHECK() (Subcommand 0x00A0). If this returns a value of 0x80, then OTP programming conditions are met.
   a. Write subcommand 0x00A0 and read from 0x40 to get the data of 0x00A0.
      i. 0X10 0X3E 0XA0 Byte3(CRC) 0X00 Byte5(CRC).
      ii. 0x10 0x40 0x11 0x80 Byte4(CRC).
10. If OTP_WR_CHECK indicates conditions are met, send OTP_WRITE() subcommand (0x00A1).
   a. Write subcommand 0x00A1 to send OTP_WRITE().
      i. 0x10 0x3E 0xA1 Byte3(CRC) 0X00 Byte5(CRC).
11. Wait 100ms. Read from 0x40 to check if OTP programming was successful (0x80 indicates success).
   a. Read back the 0x00A1 subcommand data from 0x40.
      i. 0x10 0x40 0x11 0x80 Byte4(CRC).
12. Enter into sealed mode.
   a. Write 0x0030 to go into sealed mode and read 0x12 register to make sure device enters into sealed mode.
      i. 0x10 0x3E 0x30 Byte3(CRC) 0x00 Byte5(CRC).
      ii. 0x10 0x12 0x11 Byte3(Rdata) Byte4(CRC) Byte5(Rdata) Byte6(CRC).
13. Exit CONFIG_UPDATE mode - (Subcommand 0x0092).
   a. 0x10 0x3E 0X92 Byte3(CRC) 0x00 Byte5(CRC).

5 Summary

This application note introduces the BQ769X2 OTP programming process step-by-step, and also introduces the hardware setup and example software command format. By referencing this application note, users can efficiently set up both hardware and software elements to complete OTP configuration in the product line.

6 References

1. Texas Instruments, BQ769x2 Calibration and OTP Programming Guide, application note
2. Texas Instruments, BQ76942 Technical Reference Manual, technical reference manual
3. Texas Instruments, BQ769X2 Software Development Guide, application note
4. Texas Instruments, BQ769X2 Control Based on MSPM0 Through I2C, application note
5. Texas Instruments, BQ76952: Can't leave SEALED mode, website

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