Application Note

BQ79616-Q1 Software Design Reference

Taylor Vogt

ABSTRACT

This application note provides an outline for the basic communications between the BQ79616-Q1 device and a host system. This includes communications for a single BQ79616-Q1 device or a stack of BQ79616-Q1 devices. Examples, such as auto-addressing and reverse-addressing, are included to provide the user with simple demonstrations of the basic communications of the device. The information is meant to provide an overview of the communications information outlined in the *BQ79616-Q1, BQ79614-Q1, BQ79612-Q1 Functional Safety-Compliant Automotive 16S/14S/12S Battery Monitor, Balancer and Integrated Hardware Protector* data sheet.

The communications used in this document are presented in a series of hexadecimal byte values. The actual device communications are sent in standard UART (universal asynchronous receiver-transmitter) format.

Table of Contents

List of Tables

Trademarks

1 Command Frames

Reading and writing registers using command frames underlies nearly all basic communication with the BQ79616-Q1. All read and write commands are provided in hexadecimal format, in command frame order.

1.1 Structure

1.1.1 Initialization Byte

Table 1-1. Initialization Byte

Type	Value
Single device read	0x80
Single device write	0x90
Stack read	0xA0
Stack write	0xB0
Broadcast read	0xC0
Broadcast write	0xD0
Broadcast write reverse direction	0xE0

1.1.2 Device ID Address

For single device read/write only. 1 byte, for example, is 0x02.

1.1.3 Register Address

2 bytes, for example, is 0x0306.

1.1.4 Data

For reads: (Number of bytes requested - 1), with a maximum of 128 bytes requested. For writes: the data bytes to be written, with a maximum of eight bytes written.

1.1.5 CRC

Two bytes, calculated using the CRC-16-IBM generator polynomial. See the datasheet CRC section for more information.

1.2 Command Frame Template Tables

Command frame format templates are provided in the following tables for single device read/write, stack read/write, and broadcast read/write. For bit-level detail on the command frames, see the "Command and Response Protocol" section of the *BQ79616-Q1, BQ79614-Q1, BQ79612-Q1 Functional Safety-Compliant Automotive 16S/14S/12S Battery Monitor, Balancer and Integrated Hardware Protector* data sheet.

Table 1-2. Single Device Read Command Frame

	Data	Comments
Initialization Byte	0x80	Always 0x80
Device ID Address	0x00	Device address 0 is addresssed in this case
Register Address	0x0215	Start with address 0x215
Data	0x0B	Send 12 bytes worth of data back (register contents from 0x215 to 0x220)
CRC	0xCB49	

Table 1-3. Single Device Write Command Frame

	Data	Comments
Initialization Byte	0x93	Writing four data bytes to a single device (0x90 for 1 byte of data read)
Device ID Address	0x00	Device address 0 is addressed in this case
Register Address	0x0100	Start with address 0x100
Data	0x02B778BC	Write 4 bytes to registers 0x100-0x103
CRC	0x9A8C	

Table 1-4. Stack Read Command Frame

	Data	Comments
Initialization Byte	0xA0	Always 0xA0
Device ID Address	--	No address byte is sent in stack read
Register Address	0x0215	Start with address 0x215
Data	0x02B778BC	Send 12 bytes worth of data back (register contents from 0x215 to 0x220) from each device in the stack
CRC	0xCCB3	

Table 1-5. Stack Write Command Frame

	Data	Comments
Initialization Byte	0xB3	Writing 4 bytes to the stack devices
Device ID Address	--	No address byte is sent in stack write
Register Address	0x0100	Start with address 0x100
Data	0x02B778BC	Write 4 bytes to registers 0x100-0x103 to all devices in stack
CRC	0x0A35	

Table 1-6. Broadcast Read Command Frame

	Data	Comments
Initialization Byte	0xC0	Always 0xC0
Device ID Address	--	No address byte is sent in broadcast mode
Register Address	0x0215	Start with address 0x215
Data	0x0B	Send 12 bytes worth of data back (register contents from 0x215 to 0x220)
CRC	0xD2B3	

Table 1-7. Broadcast Write Command Frame

	Data	Comments
Initialization Byte	0xD3	Writing four bytes to all the devices
Device ID Address	--	No address byte is sent in broadcast mode
Register Address	0x0100	Start with address 0x100
Data	0x02B778BC	Write four bytes to registers 0x100-0x103 to all devices
CRC	0x336A	

1.3 ReadReg and WriteReg Functions

When using the BQ79616 sample code, ReadReg and WriteReg act as the primary communication wrapper functions between the TMS570 LaunchPad and BQ79616. CRC is automatically calculated and appended by these functions.

1.3.1 ReadReg

The basic structure for the ReadReg function is as follows:

```
#_of_Read_Bytes = ReadReg(Device_Address, Register_Address, Incoming_Data_Byte_Array, #_Data_Bytes,
ms_Before_Time_Out, Packet_Type)
```

Device_Address, #_Data_Bytes, and ms_Before_Time_Out are integers while Incoming_Data_Byte_Array and Register_Address are hex values with the prefix "0x". Device_Address is ignored for broadcast/stack reads.

For example:

```
nRead = ReadReg(nDev_ID, 0x0306, bFrame, 12, 0, FRMWRT_SGL_R);
```

This line reads 12 bytes of data from register 0x0306 of the device nDev_ID and stores it in a local byte array (on the microcontroller) called bFrame. The packet type is a single device read.

1.3.2 WriteReg

The basic structure for the WriteReg function is as follows:

```
#_of_Sent_Bytes = WriteReg(Device_Address, Register_Address, Data, #_Data_Bytes, Packet_Type)
```

Device_Address and #_Data_Bytes are integers, while Register_Address and Data are hex values (with the prefix "0x"). Device_Address is ignored for broadcast and stack writes.

For example:

```
nSent = WriteReg(nDev_ID, 0x0306, 0x01, 1, FRMWRT_SGL_NR);
```

This line writes to register 0x0306 of the device nDev_ID with one byte of data. The data sent is 0x01. The type of packet is a single device write.

1.3.3 Packet Types Available in Sample Code

The following table provides the various packet types available for use in the ReadReg and WriteReg functions:

Table 1-8. Packet Types

Frame Signifier	Packet Type
FRMWRT_SGL_W	Single Device Write
FRMWRT_SGL_R	Single Device Read
FRMWRT_STK_W	Stack Write
FRMWRT_STK_R	Stack Read
FRMWRT_ALL_W	Broadcast Write
FRMWRT_ALL_R	Broadcast Read

2 Quick Start Guide

To get started with measurements quickly, all that is required is the "Wake Sequence", "Auto-Addressing", and "Read Cell Voltages" sections of this guide.

3 Wake Sequence

The wake ping is applied by the microcontroller to the RX line of the BQ79616-Q1 device. This ping is an active low, and it has a low time of 2.5 ms. To wake the device:

1. Send a wake ping (as described above)
2. Wait (approximately10ms shutdown to active transition + approximately 600us propagation of wake) x number_of_devices, before any further communications.

4 Auto-Addressing

The following section discusses the standard direction auto-address of the device stack.

4.1 Steps

1. Dummy broadcast write OTP_ECC_TEST=0x00 to sync the DLL (delay-locked loop)
2. Enable auto addressing mode by broadcast writing CONTROL1=0x01.
3. Loop through the total number of boards setting the DIR0_ADDR of each board
4. Broadcast write everything as a stack device first (COMM_CTRL=0x02)
5. **IF 1 board total:** Set device as base and top of stack (COMM_CTRL=0x01) **ELSE:** Set top of stack and base device separately (base device COMM_CTRL=0x00, top device COMM_CTRL=0x03)
6. Dummy broadcast read OTP_ECC_TEST to sync the DLL

4.2 Example Commands for ThreeDevices

```
D0 03 4C 00 FC 24       //Step 1 (from above description)
D0 03 09 01 0F 74       //Step 2
D0 03 06 00 CB 44       //Step 3 (device 0)
D0 03 06 01 0A 84       //Step 3 (device 1)
D0 03 06 02 4A 85       //Step 3 (device 2)
D0 03 08 02 4E E5       //Step 4
90 00 03 08 00 13 DD    //Step 6 (base device)
90 02 03 08 03 52 64    //Step 6 (top of stack)
C0 03 4C 00 F8 E4       //Step 7
```

Explanation of first broadcast write command frame (D0 03 4C 00 FC 24):

- D0 = broadcast write of one byte
- 034C = register address
- 00 = write value 0x00
- FC24 = CRC

Explanation of first single device write command frame (90 00 03 08 00 13 DD):

- 90 = single device write of one byte
- 00 = device address
- 0308 = register address
- 00 = write value 0x00
- 13DD = CRC

Explanation of first broadcast read command frame (C0 03 4C 00 F8 E4):

- C0 = broadcast read
- 034C = register address
- 00 = read one byte of data
- F8E4 = CRC

5 Read Cell Voltages

5.1 Steps

1. Set all used cells to active. Example: For 16 cells, ACTIVE_CELL=0x0A
2. Set the desired run mode, and start the ADC. Example: For continuous run, ADC_CTRL1=0x06
3. Wait the required round-robin time (192us per round robin, plus any reclocking delays from writing the ADC_CTRL1 register)
4. Loop read the appropriate cell measurement registers. Example: Read from VCELL16_HI to VCELL1_LO

5.2 Example Commands for ThreeDevices

```
D0 00 03 0A B8 13                    //Step 1
D0 03 0D 06 4C 76                    //Step 2
delay [192us + (5us x TOTALBOARDS)] //Step 3
C0 05 68 1F 42 2D                    //Step 4
```

5.3 Convert to Voltages

To convert 16-bit ADC values to actual voltages:

1. Convert the 16 bit value from two's complement form to a 16 bit decimal value
2. Multiply by the ADC resolution (190.73uV/LSB)

6 Cell Balancing

The following example is for a simple, automatic balancing control setup. For more advanced cell balancing techniques, please see the “Cell Balancing” section of the datasheet.

6.1 Steps

1. Make sure ACTIVE_CELL has been set up for the desired number of channels.
2. Set cell balancing timers using CB_CELL*_CTRL registers to choose the timers for the desired channels to balance. Only channels with nonzero values will be balanced.
3. Set the duty cycle used to switch between even and odd cells using BAL_CTRL1[DUTY2:0] bits.
4. OPTIONAL: Set VCB_DONE_THRESH register to the desired stop voltage for all channels. Now the device stops balancing if a cell reaches below this threshold. Then set OVUV_CTRL=0x05 to run OVUV comparators in round robin. NOTE: It is also a good idea to set OV_THRESH and UV_THRESH for when cell balancing finishes (and for before balancing starts).
5. Choose auto-balance control, and start balancing by setting BAL_CTRL2 = 0x03

6.2 Example Commands

```
D0 00 03 0A B8 13                                  //Step 1
D7 03 18 02 02 02 02 02 02 02 02 14 BE             //Step 2
D7 03 20 02 02 02 02 02 02 02 02 27 7F             //Step 2
D0 03 2E 01 14 84                                  //Step 3
D0 03 2A 08 D6 42                                  //Step 4
D0 03 2C 05 14 27                                  //Step 4
D0 03 2F 03 94 D5                                  //Step 5
```

7 OVUV

The following example is for a continuous, round-robin run of the OV and UV protectors.

7.1 Steps

1. Make sure ACTIVE_CELL has been set up for the desired number of channels.
2. Set OV and UV thresholds for all VC channels by writing to registers OV_THRESH and UV_THRESH, respectively.
3. Set OVUV mode to round robin, and set OVUV_GO to begin the OVUV protector by writing OVUV_CTRL = 0x03.

8 OTUT

The following example is for a continuous, round-robin run of the OT and UT protectors.

8.1 Steps

1. Make sure ACTIVE_CELL has been set up for the desired number of channels.
2. Enable TSREF by writing CONTROL2 = 0x01.
3. Wait 6 ms for TSREF to fully enable.
4. Write OT and UT thresholds for all GPIO inputs by writing OTUT_THRESH[OT_THR4:0 and UT_THR2:0].
5. Configure which GPIO pin to sense by writing the GPIO_CONF1 to GPIO_CONF4 registers with each desired GPIO pin set as ADC and OTUT inputs
6. Set OTUT mode to round robin, and set OTUT_GO to begin the OTUT protector by writing OTUT_CTRL = 0x03.

9 Reverse Addressing

This example provides details for reverse addressing the entire daisy chain. Note that once addressing in both directions is complete, the host can skip auto-addressing when changing directions.

9.1 Steps

1. For a single device, write CONTROL1 = 0x80 to set DIR_SEL=1 for the base device.
2. Send broadcast write reverse direction CONTROL1 = 0x80 to change direction for the rest of the devices. This command type should only ever be used for this one scenario, where the user is changing the direction of the daisy chain communications. Do not use this for other commands.
3. Now that the direction has been changed on all devices, do the standard auto-addressing sequence above, but with DIR1_ADDR instead of DIR0_ADDR, and with CONTROL1=0x81 instead of 0x01 (to keep the reverse direction enabled). Make sure to also update the COMM_CTRL register for top of stack.

9.2 Example Commands for ThreeDevices

```
90 00 03 09 80 13 ED    //Step 1
E0 03 09 80 C0 14       //Step 2
//Step 3 begin normal auto address sequence, but for DIR1_ADDR
D0 03 4C 00 FC 24       //sync DLL with dummy write
D0 03 09 81 0E D4       //enter auto-address mode, BUT KEEP REVERSE DIRECTION
D0 03 07 00 CA D4       //give each device its DIR1_ADDR address
D0 03 07 01 0B 14
D0 03 07 02 4B 15
D0 03 08 02 4E E5       //Set everything as stack device first
90 00 03 08 00 13 DD    //set base device as base
90 02 03 08 03 52 64    //set top of stack as top of stack
C0 03 4C 00 F8 E4       //dummy read to sync DLL
```

10 Revision History

Changes from Revision A (December 2020) to Revision B (August 2023)	Page
• Updated publication throughout improving technical clarity	1

Changes from Revision * (September 2019) to Revision A (December 2020)	Page
• Updated the numbering format for tables, figures, and cross-references throughout the document	1

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