*Application Note*

# *BQ79616-Q1 软件设计参考*

*Taylor Vogt*

**摘要**

本应用手册概述了 BQ79616-Q1 器件和主机系统之间的基本通信。这包括单一 BQ79616-Q1 器件或 BQ79616-Q1 器件栈的通信。为了给用户简单演示器件的基本通信，报告中包含了自动寻址和反向寻址等示例。这些信息旨在简要介绍 *BQ79616-Q1、BQ79614-Q1、BQ79612-Q1 汽车类功能安全合规型 16/14/12 节串联电池监测器、平衡器和集成式硬件保护器* 数据表中所述的通信信息。

本文档中使用的通信方式通过一系列十六进制字节值来表示。实际的器件通信使用标准 UART（通用异步接收器/发送器）格式发送。

## 内容



## 表格清单



## 商标

所有商标均为其各自所有者的财产。



English Document: SLVAE86
Copyright © 2023 Texas Instruments Incorporated

# 1 命令帧

使用命令帧读写寄存器是与 BQ79616-Q1 进行几乎所有基本通信的基础。所有读写命令全部按照命令帧顺序以十六进制格式提供。

## 1.1 结构

### 1.1.1 初始化字节

**表 1-1. 初始化字节**

| 类型 | 值 |
|---|---|
| 单器件读 | 0x80 |
| 单器件写 | 0x90 |
| 栈读 | 0xA0 |
| 栈写 | 0xB0 |
| 广播读 | 0xC0 |
| 广播写 | 0xD0 |
| 广播写反向 | 0xE0 |

### 1.1.2 器件 ID 地址

仅适用于单器件读/写。1 个字节，例如 0x02。

### 1.1.3 寄存器地址

2 个字节，例如 0x0306。

### 1.1.4 数据

对于读取：（请求的字节数 - 1），最多请求 128 个字节。对于写入：要写入的数据字节，最多写入 8 个字节。

### 1.1.5 CRC

两个字节，由 CRC-16-IBM 多项式生成器计算出。更多信息，请参阅数据表 CRC 部分。

## 1.2 命令帧模板表

下列表格提供了单器件读取/写入、栈读取/写入和广播读取/写入的命令帧格式模板。有关命令帧字节级的详细信息，请参阅 *BQ79616-Q1、BQ79614-Q1、BQ79612-Q1 汽车类功能安全合规型 16/14/12 节串联电池监测器、平衡器和集成式硬件保护器* 数据表的“命令和响应协议”部分。

**表 1-2. 单器件读命令帧**

| | 数据 | 说明 |
|---|---|---|
| 初始化字节 | 0x80 | 始终为 0x80 |
| 器件 ID 地址 | 0x00 | 本例中进行寻址的是器件地址 0 |
| 寄存器地址 | 0x0215 | 从地址 0x215 开始 |
| 数据 | 0x0B | 发回 12 个字节的数据（寄存 0x215 至 0x220 的内容） |
| CRC | 0xCB49 | |



English Document: SLVAE86
Copyright © 2023 Texas Instruments Incorporated

表 1-3. 单器件写命令帧

| | 数据 | 说明 |
|---|---|---|
| 初始化字节 | 0x93 | 向单一器件写入 4 个数据字节（0x90 用于 1 个字节的读数据） |
| 器件 ID 地址 | 0x00 | 本例中进行寻址的是器件地址 0 |
| 寄存器地址 | 0x0100 | 从地址 0x100 开始 |
| 数据 | 0x02B778BC | 向寄存器 0x100-0x103 写入 4 个字节 |
| CRC | 0x9A8C | |

表 1-4. 栈读命令帧

| | 数据 | 说明 |
|---|---|---|
| 初始化字节 | 0xA0 | 始终为 0xA0 |
| 器件 ID 地址 | -- | 栈读过程中不发送地址字节 |
| 寄存器地址 | 0x0215 | 从地址 0x215 开始 |
| 数据 | 0x02B778BC | 从栈中的每个器件发回 12 个字节的数据（0x215 至 0x220 的寄存器内容） |
| CRC | 0xCCB3 | |

表 1-5. 栈写命令帧

| | 数据 | 说明 |
|---|---|---|
| 初始化字节 | 0xB3 | 向栈器件写入 4 个字节 |
| 器件 ID 地址 | -- | 栈写过程中不发送地址字节 |
| 寄存器地址 | 0x0100 | 从地址 0x100 开始 |
| 数据 | 0x02B778BC | 依次向寄存器 0x100-0x103 和栈中的所有器件写入 4 个字节 |
| CRC | 0x0A35 | |

表 1-6. 广播读命令帧

| | 数据 | 说明 |
|---|---|---|
| 初始化字节 | 0xC0 | 始终为 0xC0 |
| 器件 ID 地址 | -- | 广播模式下不发送地址字节 |
| 寄存器地址 | 0x0215 | 从地址 0x215 开始 |
| 数据 | 0x0B | 发回 12 个字节的数据（寄存 0x215 至 0x220 的内容） |
| CRC | 0xD2B3 | |

表 1-7. 广播写命令帧

| | 数据 | 说明 |
|---|---|---|
| 初始化字节 | 0xD3 | 向所有器件写入 4 个字节 |
| 器件 ID 地址 | -- | 广播模式下不发送地址字节 |
| 寄存器地址 | 0x0100 | 从地址 0x100 开始 |
| 数据 | 0x02B778BC | 依次向寄存器 0x100-0x103 和所有器件写入 4 个字节 |
| CRC | 0x336A | |


Copyright © 2023 Texas Instruments Incorporated

## 1.3 ReadReg 和 WriteReg 函数

使用 BQ79616 示例代码时，ReadReg 和 WriteReg 充当 TMS570 LaunchPad 和 BQ79616 之间的主要通信包装器函数。CRC 由这些函数自动计算并附加。

### 1.3.1 ReadReg

ReadReg 函数的基本结构如下：

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

Device_Address、#_Data_Bytes 和 ms_Before_Time_Out 是整数，而 Incoming_Data_Byte_Array 和 Register_Address 是带前缀“0x”的十六进制值。Device_Address 在广播/栈读操作中会被忽略。

例如：

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

此行会从器件 nDev_ID 的寄存器 0x0306 读取 12 个字节的数据，并将其存储在名为 bFrame 的本地字节数组中（在微控制器上）。数据包类型为单器件读。

### 1.3.2 WriteReg

WriteReg 函数的基本结构如下：

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

Device_Address 和 #_Data_Bytes 是整数，而 Register_Address 和数据是十六进制值（带前缀“0x”）。Device_Address 在广播和栈写操作中会被忽略。

例如：

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

此行会将 1 字节数据写入设备 nDev_ID 的寄存器 0x0306。发送的数据为 0x01。数据包的类型为单器件写。

### 1.3.3 示例代码中可用的数据包类型

下表提供了可用于 ReadReg 和 WriteReg 函数的各种数据包类型：

**表 1-8. 数据包类型**

| 帧能指 | 数据包类型 |
|---|---|
| FRMWRT_SGL_W | 单器件写 |
| FRMWRT_SGL_R | 单器件读 |
| FRMWRT_STK_W | 栈写 |
| FRMWRT_STK_R | 栈读 |
| FRMWRT_ALL_W | 广播写 |
| FRMWRT_ALL_R | 广播读 |

# 2 快速入门指南

若要快速开始测量，只需阅读本指南的“唤醒序列”、“自动寻址”和“读取电池电压”部分。



English Document: SLVAE86
Copyright © 2023 Texas Instruments Incorporated

# 3 唤醒序列

微控制器将唤醒 ping 应用至 BQ79616-Q1 器件的 RX 行。此 ping 为低电平有效，且低电平时间为 2.5ms。若要唤醒器件：

1. 发送唤醒 ping ( 如上所述 )
2. 在任何进一步通信之前，等待 ( 约 10ms 关断至活动过渡 + 约 600us 唤醒传播 ) x number_of_devices。

# 4 自动寻址

以下部分介绍了器件栈的标准方向自动寻址。

## 4.1 步骤

1. 虚拟广播写入 OTP_ECC_TEST=0x00 以同步 DLL ( 延迟锁相环 )
2. 通过广播写入 CONTROL1=0x01 来启用自动寻址模式。
3. 环路通过电路板总数，设置每个电路板的 DIR0_ADDR
4. 广播写入一切为栈器件优先 (COMM_CTRL=0x02)
5. **如果总共 1 个电路板：**将器件设置为基底和栈顶 (COMM_CTRL=0x01) **或者：**分别设置栈顶器件和基底器件 ( 基底器件 COMM_CTRL=0x00，顶器件 COMM_CTRL=0x03 )
6. 虚拟广播读取 OTP_ECC_TEST 以同步 DLL

## 4.2 三个器件的示例命令

```
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
```

第一个广播写命令帧 (D0 03 4C 00 FC 24) 说明：

- D0 = 广播写入一个字节
- 034C = 寄存器地址
- 00 = 写入值 0x00
- FC24 = CRC

第一个单器件写命令帧 (90 00 03 08 00 13 DD) 说明：

- 90 = 单器件写入一个字节
- 00 = 器件地址
- 0308 = 寄存器地址
- 00 = 写入值 0x00
- 13DD = CRC

第一个广播读命令帧 (C0 03 4C 00 F8 E4) 说明：

- C0 = 广播读
- 034C = 寄存器地址
- 00 = 读取一个字节的数据
- F8E4 = CRC


English Document: SLVAE86
Copyright © 2023 Texas Instruments Incorporated

# 5 读取电芯电压

## 5.1 步骤

1. 将所有已用电芯设置为活动状态。示例：对于 16 节电芯，ACTIVE_CELL=0x0A
2. 设置所需的运行模式，然后启动 ADC。示例：对于连续运行，ADC_CTRL1=0x06
3. 等待所需的循环时间（每个循环 192us，加上写入 ADC_CTRL1 寄存器产生的任何重新计时延迟）
4. 循环读取适当的电芯测量寄存器。示例：从 VCELL16_HI 读取到 VCELL1_LO

## 5.2 三个器件的示例命令

```
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 转换为电压

将 16 位 ADC 值转换为实际电压：

1. 将 16 位值从二进制补码格式转换为 16 位十进制值
2. 乘以 ADC 分辨率 (190.73uV/LSB)

# 6 电池平衡

以下示例适用于简单的自动平衡控制设置。有关更先进的电池平衡技术，请参阅数据表的“电池平衡”部分。

## 6.1 步骤

1. 确保已为所需数量的通道设置 ACTIVE_CELL。
2. 使用 CB_CELL*_CTRL 寄存器设置电芯均衡计时器，以便为要均衡的所需通道选择计时器。仅具有非零值的通道将被均衡。
3. 使用 BAL_CTRL1[DUTY2:0] 位设置用于在奇偶电芯之间切换的占空比。
4. 可选：对于所有通道，将 VCB_DONE_THRESH 寄存器设置为所需的停止电压。目前，器件会在电芯低于此阈值时停止均衡。然后，设置 OVUV_CTRL = 0x05 以循环运行 OVUV 比较器。注意：针对电芯均衡完成时（以及均衡开始之前）设置 OV_THRESH 和 UV_THRESH 也是不错的选择。
5. 通过设置 BAL_CTRL2 = 0x03，选择自动均衡控制并开始均衡

## 6.2 示例命令

```
D0 00 03 0A B8 13                                  //Step 1
D7 03 18 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

以下示例适用于 OV 和 UV 保护器的连续循环运行。

## 7.1 步骤

1. 确保已为所需数量的通道设置 ACTIVE_CELL。
2. 通过分别写入寄存器 OV_THRESH 和 UV_THRESH，为所有 VC 通道设置 OV 和 UV 阈值。
3. 将 OVUV 模式设置为循环，并通过写入 OVUV_CTRL = 0x03，设置 OVUV_GO 以启动 OVUV 保护器。



English Document: SLVAE86
Copyright © 2023 Texas Instruments Incorporated

# 8 OTUT

以下示例适用于 OT 和 UT 保护器的连续循环运行。

## 8.1 步骤

1. 确保已为所需数量的通道设置 ACTIVE_CELL。
2. 通过写入 CONTROL2 = 0x01 启用 TSREF。
3. 等待 6 ms 以使 TSREF 完全启用。
4. 通过写入 OTUT_THRESH[OT_THR4:0 and UT_THR2:0]，为所有 GPIO 输入写入 OT 和 UT 阈值。
5. 通过将 GPIO_CONF1 写入 GPIO_CONF4 寄存器并将每个所需的 GPIO 引脚设置为 ADC 和 OTUT 输入，配置要检测的 GPIO 引脚
6. 将 OTUT 模式设置为循环，并通过写入 OTUT_CTRL = 0x03，设置 OTUT_GO 以启动 OTUT 保护器。

# 9 反向寻址

此示例提供有关对整个菊花链进行反向寻址的详细信息。请注意，当双向寻址完成后，主机能够在更换方向时跳过自动寻址。

## 9.1 步骤

1. 对于单一器件，可写入 CONTROL1 = 0x80，以为基底器件设置 DIR_SEL = 1。
2. 发送广播写反向 CONTROL1 = 0x80，以更改其余器件的方向。这种命令类型应仅用于用户更改菊花链通信方向这一场景。请勿将其用于其他命令。
3. 现在已更改所有器件上的方向，接下来请执行上述自动寻址序列，但应使用 DIR1_ADDR 而不是 DIR0_ADDR，并且 CONTROL1 = 0x81，而不是 0x01（保持启用反向）。此外，确保更新栈顶的 COMM_CTRL 寄存器。

## 9.2 三个器件的示例命令

```
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
```



English Document: SLVAE86
Copyright © 2023 Texas Instruments Incorporated

# 10 修订历史记录

| Changes from Revision A (December 2020) to Revision B (August 2023) | Page |
|---|---|
| • 更新了通篇内的文献名称，使技术文档表述更清楚 | 1 |

| Changes from Revision * (September 2019) to Revision A (December 2020) | Page |
|---|---|
| • 更新了整个文档中的表格、图和交叉参考的编号格式 | 1 |



English Document: SLVAE86
Copyright © 2023 Texas Instruments Incorporated

# 重要声明和免责声明

TI“按原样”提供技术和可靠性数据（包括数据表）、设计资源（包括参考设计）、应用或其他设计建议、网络工具、安全信息和其他资源，不保证没有瑕疵且不做出任何明示或暗示的担保，包括但不限于对适销性、某特定用途方面的适用性或不侵犯任何第三方知识产权的暗示担保。

这些资源可供使用 TI 产品进行设计的熟练开发人员使用。您将自行承担以下全部责任：(1) 针对您的应用选择合适的 TI 产品，(2) 设计、验证并测试您的应用，(3) 确保您的应用满足相应标准以及任何其他功能安全、信息安全、监管或其他要求。

这些资源如有变更，恕不另行通知。TI 授权您仅可将这些资源用于研发本资源所述的 TI 产品的应用。严禁对这些资源进行其他复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。您应全额赔偿因在这些资源的使用中对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，TI 对此概不负责。

TI 提供的产品受 TI 的销售条款或 ti.com 上其他适用条款/TI 产品随附的其他适用条款的约束。TI 提供这些资源并不会扩展或以其他方式更改 TI 针对 TI 产品发布的适用的担保或担保免责声明。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

邮寄地址：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2023，德州仪器 (TI) 公司