Application Note

BQ79616 Control Based on MSPM0 Through UART to CAN

Helic Chi, Joe Ji, and Gary Gao

ABSTRACT

As a general-purpose MCU product, MSPM0 has the characteristics of complete peripherals and high cost performance, making the BMS product design more flexible and low-cost. This application note is a guide for the preparation and usage of the sample code for the BQ79616-Q1 device paired with a LP-MSPM0G3507, including hardware connection, software, communication protocol, and test result between MSPM0 and BQ79616. Users can also port this code easily to other MSPM0 devices using the system configuration tool (SYSCONFIG). Demo code can be accessed from MSPM0 offline SDK or online SDK.

Table of Contents

List of Figures

List of Tables

Trademarks

LaunchPad™ is a trademark of Texas Instruments.
All trademarks are the property of their respective owners.

1 Introduction

Energy storage has been widely used electric vehicles, power grid, and renewable energy. BMS is the control unit that manages cells in the battery pack. Generally, BMS consists of four modules: power, AFE, communication, control, and monitoring, as shown in Figure 1-1. The auxiliary power component draws power from the battery pack to other parts of the BMS. AFE measures, balances and protects the integrated battery cells at the front end. The communication and control module obtain the measurement data of the AFE for calculation and control and send data to host PC.

This document uses BQ79616 EVM and LP-MSPM0G3507 to build a sample system, providing a design reference for the interaction between AFE and MCU in BMS application. Based on the demo code in MSPM0 SDK, the functions include BQ79616 initialization, BQ79616 data reading and sending data to the CAN bus.

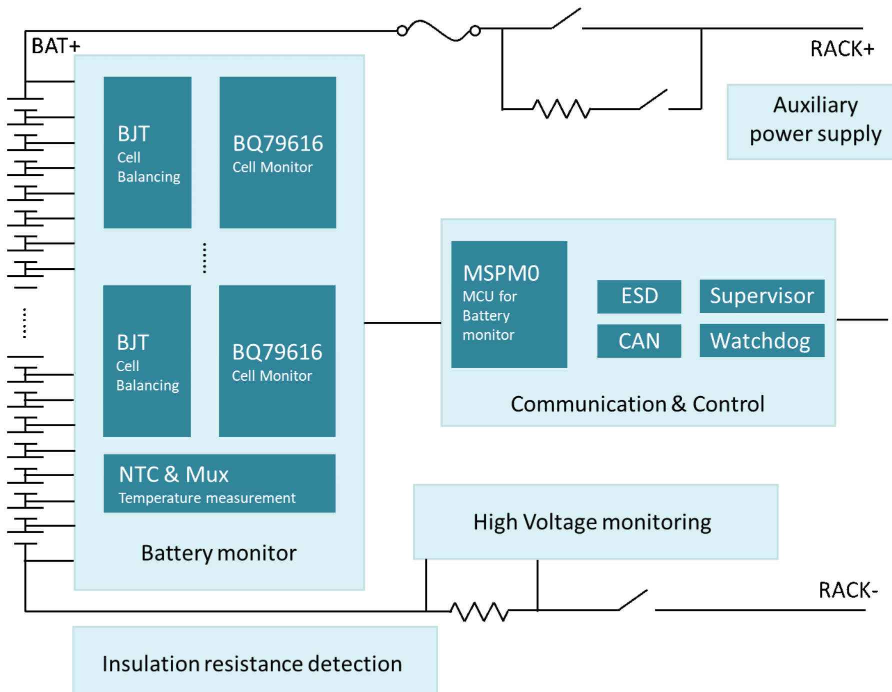

Figure 1-1. BMS System Block Diagram

2 Hardware Setup

Figure 2-1 shows the connection between BQ79616 EVMs, LP-MSPM0G3507, PC, CAN Analyzer and TCAN1046V-EVM.

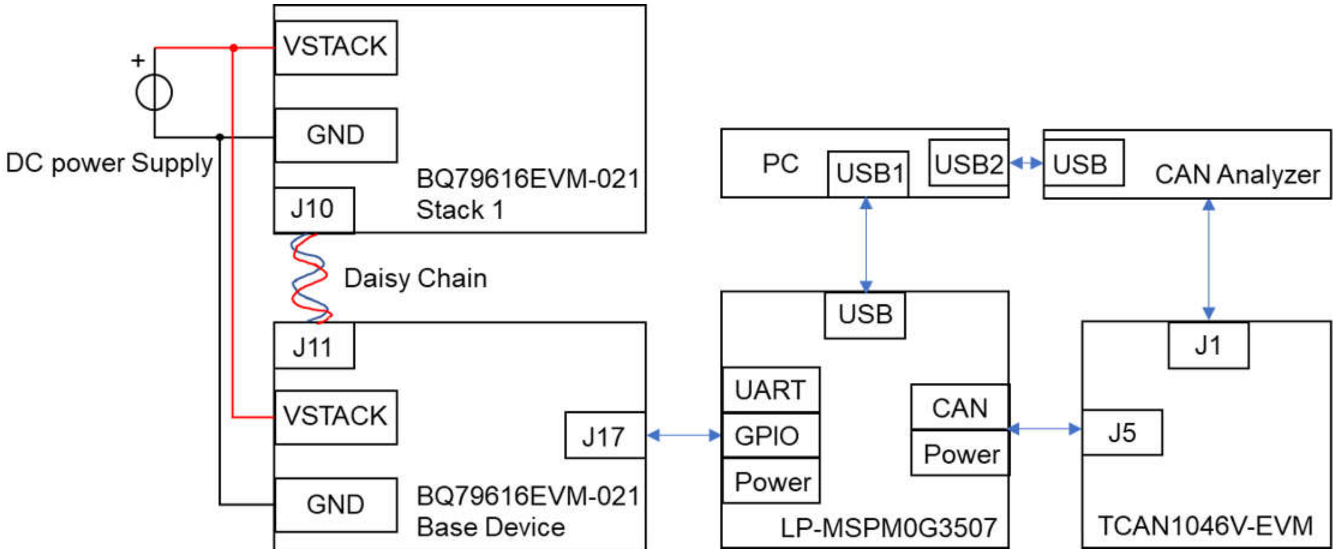

Figure 2-1. System Connection

2.1 LP-MSPM0G3507 Hardware Connection

For detailed information and schematics of LP-MSPM0G3507, see the *MSPM0G3507 LaunchPad Development Kit User's Guide*. LaunchPad™ pin configuration and pin connection with BQ79616 EVM, TCAN1046V-EVM are shown in Table 2-1.

Table 2-1. EVM Hardware Connection and Function

LaunchPad Pin Allocation		Target Device
PB0	Fault detection/GPIO	BQ79616 EVM J17-3
PA10	TX/UART	BQ79616 EVM J17-8
PA11	RX/UART	BQ79616 EVM J17-7
PA12	TX/CAN	TCAN1046EVM J5-6
PA13	RX/CAN	TCAN1046EVM J5-8
PB21	Enable CAN/GPIO	S2 on LaunchPad
GND	Power supply	BQ79616 EVM J17-5
3.3V	Power supply	BQ79616 EVM J17-6

BQ79616 EVM J17 jumper is shown in Figure 2-2.

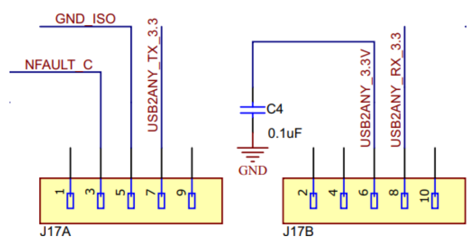

Figure 2-2. BQ79616 EVM J17

TCAN1046V-EVM J5 jumper allocation is shown in Figure 2-3, VIO and VCC are powered by LP-MSPM0G3507, to make communication properly, TCAN1046V-EVM and LP-MSPM0G3507 also share the same GND.

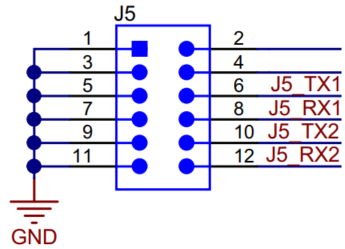

Figure 2-3. TCAN1046 EVM J5

2.2 BQ79616EVM Hardware Connection

For the hardware connection of BQ79616EVM-021, see the *BQ796xx-Q1 Evaluation Modules User's Guide*, in this EVM, and note that the resistor's ladder is used to simulate the voltage of single cell. To power up the EVM, DC voltage source needs to be connected to VSTACK and GND, at the same time, S1 and S2 need to be turned to right, shown in Figure 2-4. This demo can support BQ79616EVM stack using daisy chain. More information for the hardware connection can be found in the *Stacking BQ79616EVMs* section of the *BQ79616-Q1, BQ75614-Q1, BQ79656-Q1 Evaluation Modules User's Guide*. BQ79616 hardware design can refer to *BQ7961x-Q1 Design Recommendations for High Voltage Automotive BMS*. The online version is also available.

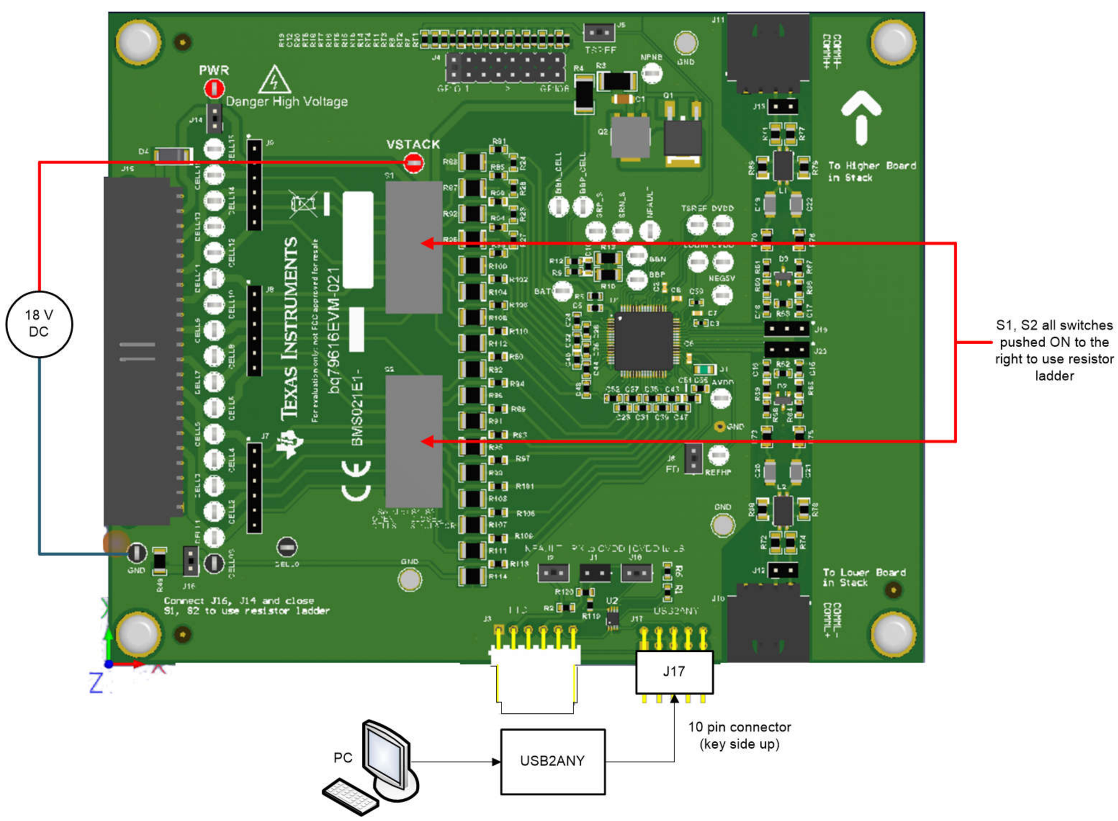

Figure 2-4. BQ79616EVM Setup Using DC Voltage With Resistor Ladder

2.3 TCAN1046VEVM Hardware Connection

TCAN1046VEVM hardware connection can be referred to TCAN1046V Evaluation Module User's Guide, TCAN1046VEVM's VCC is connected to LP-MAPM0G3507's 5V and VIO is connected to LP-MSPM0G3507's 3V3, CANH and CANL pins is connected to CAN analyzer.

3 Software Structure

In this section, the main contents include: Software function and flow diagram and Protocol description.

3.1 Project File Structure

The customer can easily use the System Configuration Tool to migrate this project from MSPM0G3507 to another MSPM0 device.

Key files in the project are list in Table 3-1.

Table 3-1. BQ79616 Project File Structure

File	Description
main.c	Program entrance
bq79616_uart_to_can.syscfg	MSPM0 peripherals initialization
BQ79616.cBQ79616.h	BQ79616 communication and control function
B0_Reg.h	Macro definition of BQ79616 register
UART.c UART.h	Low-level MSPM0 UART read/write function

3.2 Software Function and Flow Diagram

MSPM0 communicates with BQ79616 base device through universal asynchronous receiver/transmitter (UART), and BQ79616 stack devices communicate with BQ79616 base device through daisy chain.

There are two types of communication method that BQ79616 base device can communicates with MSPM0, one is PING mode and the other is UART mode. In PING mode, BQ79616 devices switch the working mode between WAKE, SLEEPtoACTIVE, SHUTDOWN, HW_RESET, according to the low-level duration time sent by MSPM0 UART Tx pin. And in UART mode, MSPM0 and BQ79616 communicate in normal UART mode, and MSPM0 configure different functions by changing BQ79616's registers through UART.

MSPM0 can control BQ79616's working mode in PING mode and uses UART to control BQ79616's register and read back BQ79616's data through UART. Functions Auto address, Reverse address, Passive balance, Fault report, and so forth are included in demo code.

MSPM0 initialize BQ79616, measure cell voltage and send voltage data through CAN to CAN analyzer, software flow diagram is shown in Figure 3-1.

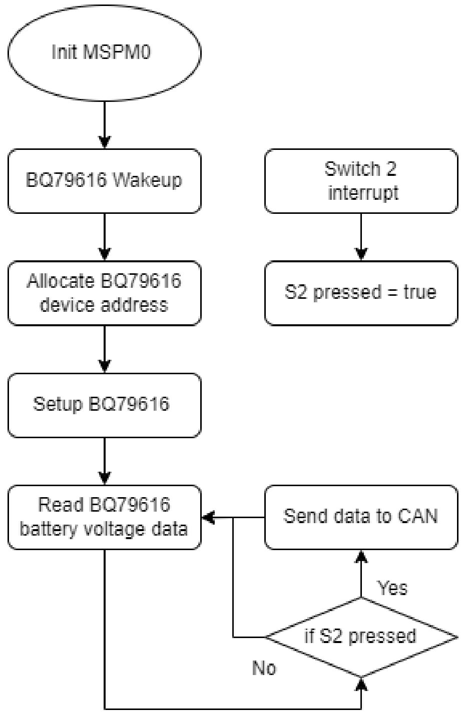

Figure 3-1. MSPM0 Software Diagram

3.3 Protocol Description

This section introduces the communication UART protocol between MSPM0 and BQ79616 devices, basic daisy chain structure knowledge and basic knowledge about UART Tx pin acting in PING mode.

3.3.1 UART and Daisy Chain

In a BQ79616 daisy chain communication structure, MSPM0 communicates directly with the BQ79616 base device through UART, and BA79616 base device communicates with BQ79616 stack devices through daisy chain.

The data structure transmitted between MSPM0 and BQ79616 is shown in Figure 3-2, each byte is transmitted in standard UART protocol, and UART baud rate is 1MHz. The data structure is defined below:

- Frame initialization (INIT, 1 byte)
- Device address (DEV ADR, 1 byte)
- Register address (REG ADR, 2 bytes)
- Data (DATA, various byte length)
- Cyclic redundancy check (CRC, 2 bytes)

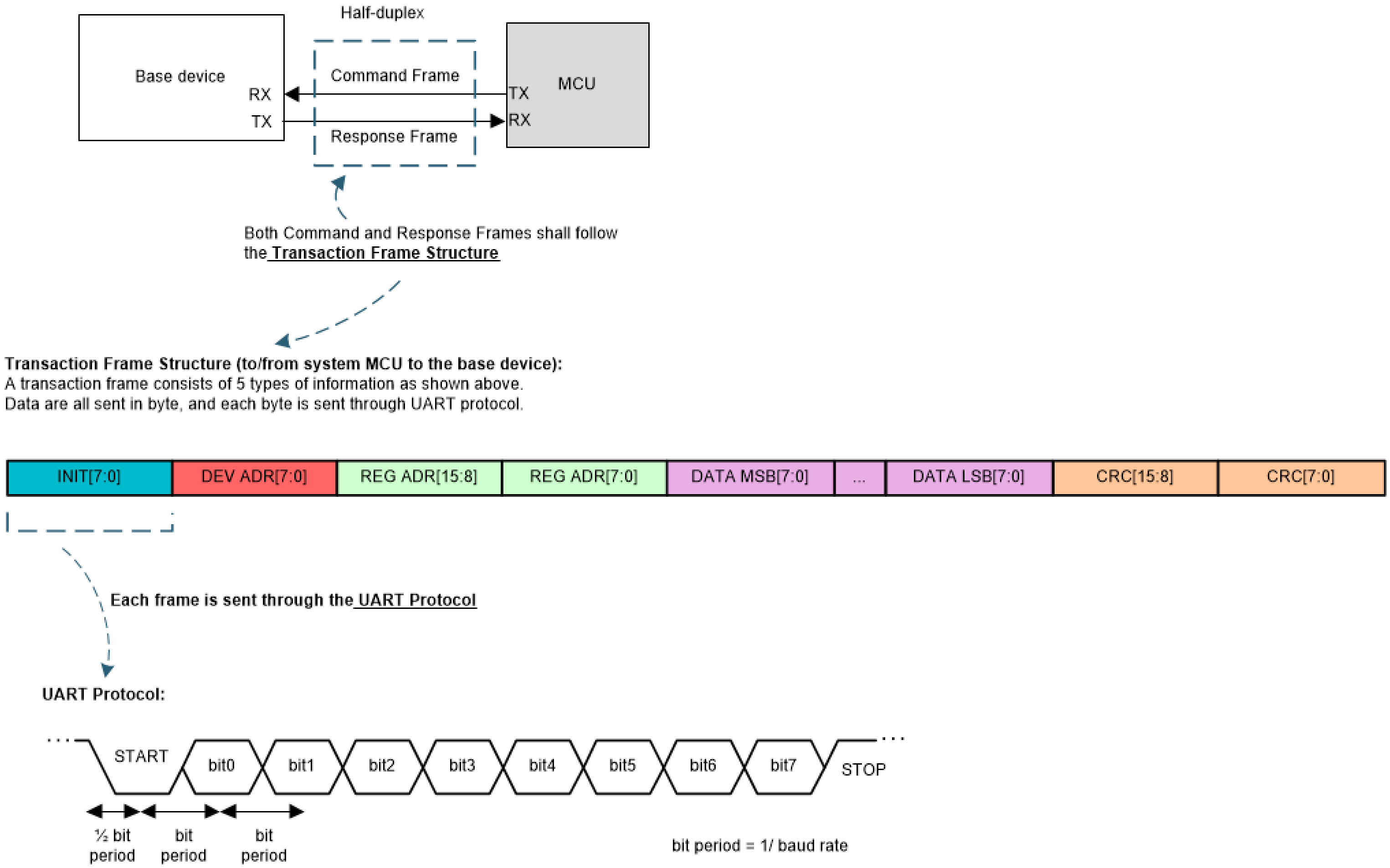

Figure 3-2. UART Communication to Host

DIR0_ADDR and DIR1_ADDR are two 6 bits device address registers, these two registers are used to recognize different BQ79616 devices connected through daisy chain. During the system initialization, MSPM0 executes BQ79616 auto address function, the base device's address is 0, the stack device's addresses starts from 1 and increases by one, the total BQ79616 quantity can be up to 64 pieces. To improve the communication robustness, the daisy chain transmits bit through COMH COML differential voltage signal. The voltage level is defined in Figure 3-3.

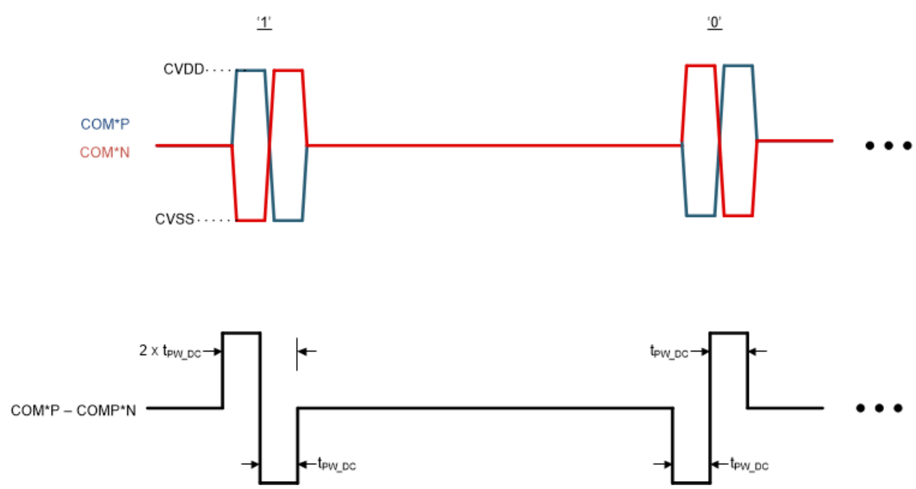

Figure 3-3. Daisy Chain Bit Definition

The daisy chain uses 13-bit binary to transmit one byte of data, and the content corresponds one-to-one with the UART byte sent or received by the base device. The data structure is defined in Figure 3-4.

- Preamble (half bit)
- SYNC [1:0]
- Start-Of-Frame (1 bit)
- Data [7:0]
- Byte Error BERR (1 bit)
- Post amble (half bit)

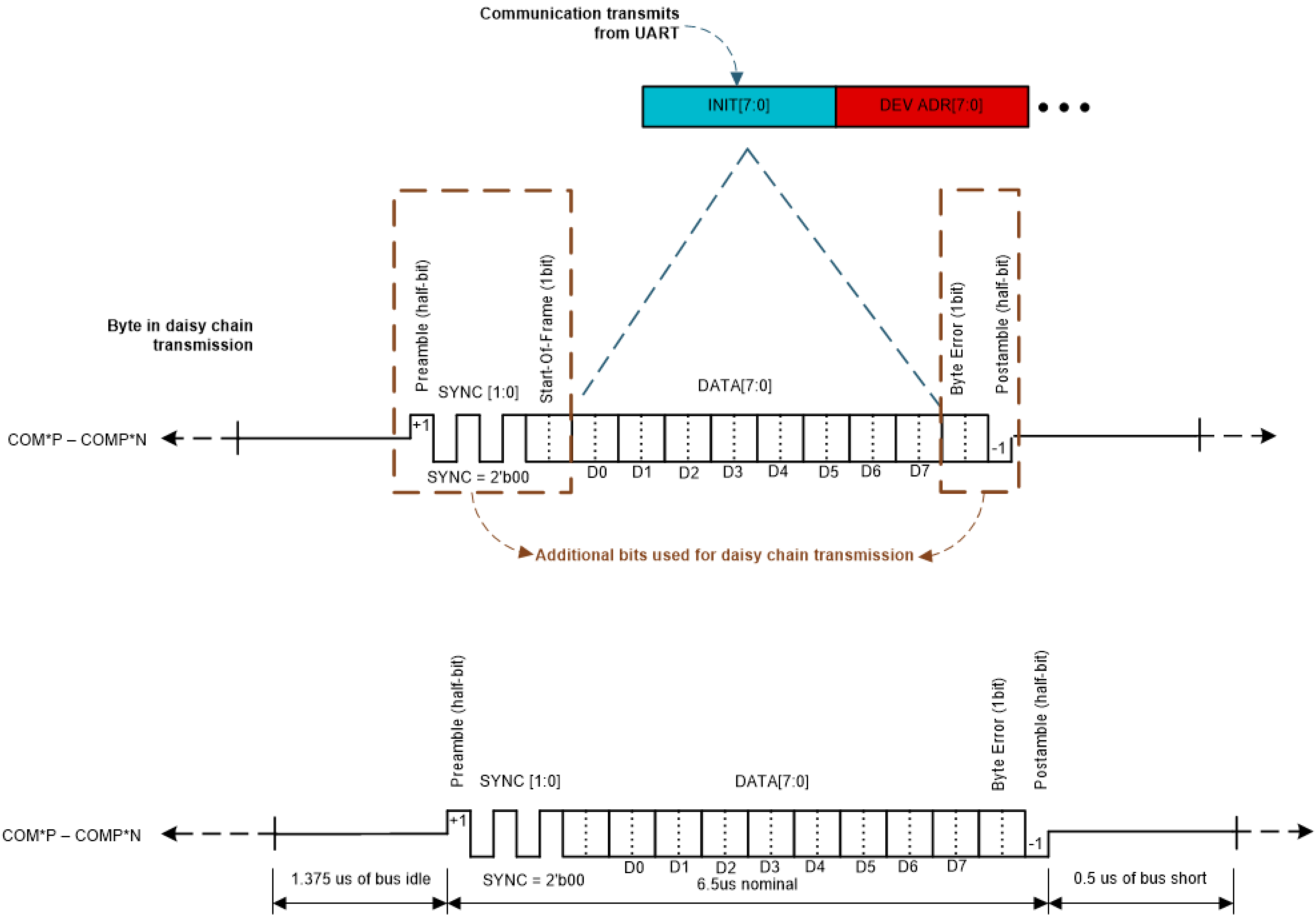

Figure 3-4. Daisy Chain Byte Definition

Based on UART and daisy chain communication protocol, communication between MSPM0 and BQ79616 stack devices can support the function shown in Table 3-2.

Table 3-2. Communication Function Between MSPM0 and BQ79616

Command	Description
Single device read	To read register from a single device (base or stack)
Single device write	To write register to a single device (base or stack)
Stack read	To read register from the stack devices only. The device must be configured as a stack device with COMM_CTRL[STACK_DEV] = 1 to respond to Stack Read commands.
Stack write	To write register for only the stack devices. The device must be configured as a stack device with COMM_CTRL[STACK_DEV] = 1 to respond to Stack Write commands.
Broadcast read	To read register for all of the devices in the daisy chain, including the base device.
Broadcast write	To write register for all of the devices in the daisy chain, including the base device.
Broadcast write reverse direction	To send a broadcast write in the reverse direction set by CONTROL1[DIR_SEL] bit. This command is intended to be used for switching the communication direction with the RING interface.

3.3.2 PING and TONG

In the noncommunicable conditions, such as in SHUTDOWN or SLEEP mode, or in the loss of communication situations when host need to send an instruction for a reset or power down as a communication recovery attempt, a Ping or Tone is a form of communication to the device for a specific action. For the detailed Ping and Tone description, see the *Ping and Tone* section of the *BQ79616 16-Series Battery Monitor, Balancer, and Integrated Hardware Protector Data Sheet*.

A ping is a specific high-low-high signal that MSPM0 send to BQ79616's Rx pin through UART or IO. Ping is used on the base device as only the base device is connected to MSPM0 which the UART Rx is accessible. The device detects different low times of the ping signal, which is shown in Figure 3-5.

The communication pings are referring to the WAKE ping, SLEEPtoACTIVE ping, SHUTDOWN ping, and HW_RESET ping. These pings instruct the BQ79616 to a specific power mode when normal communication is not available.

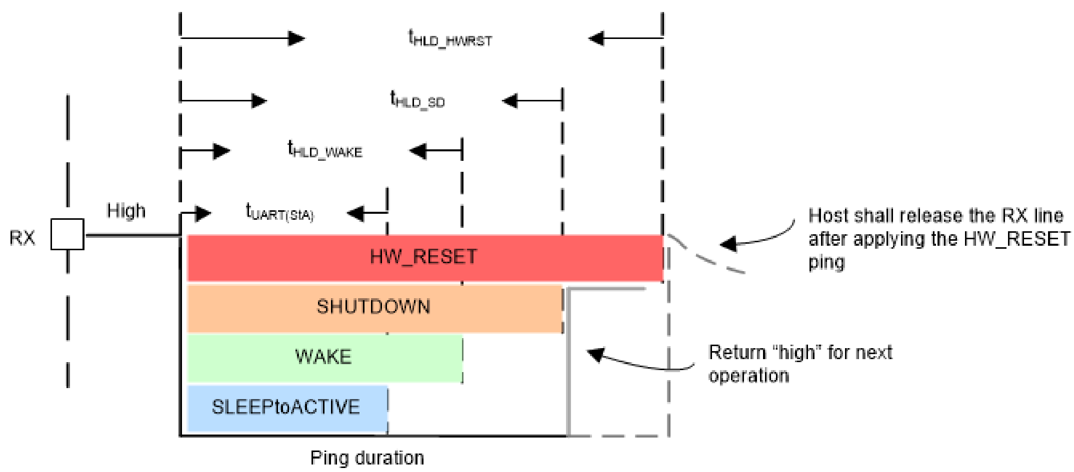

Figure 3-5. Communication Pings

A tone is a fixed number of couplets (pulses) with a specified polarity (all "+" or all "–") sent through the differential vertical interface COMH and COML ports. Tone is used on stack devices as only the COMH/L ports are accessible. The number of couplets for transmission is always greater than the number of couplets needed for detection.

4 System Test

In this section, the main contents include:

- How to setup the test environment with LP-MSPM0G3507, two BQ79616 EVMs, TCAN1046VEVM and CAN analyzer.
- Read voltage function test result in MSPM0's debug window and result sent through CAN in CAN analyzer PC window.

4.1 Test Setup

Here are the steps to set up the test environment. For the system diagram block, see Figure 2-1. Figure 4-1 is the actual test environment photo.

1. Based on Figure 4-1, Figure 2-1, connect LP-MSPM0G3507 with two BQ79616 EVMs, TCAN1046VEVM and CAN analyzer. Both BQ79616 EVMs need to be powered, here we use a 18V DC power supply.
2. Connect LP-MSPM0G3507 to PC using USB cable for debug purpose, and connect CAN Analyzer to PC to receive the CAN data.
3. Import the BQ79616 demo code to CCS from the latest MSPM0-SDK, compile and start debugging on LP-MSPM0G3507.
4. Open CAN Analyzer data receive window and CCS debug console window.
5. Press S2 button (PB21) on LP-MSPM0G3507, to set the *gTXMsg* flag to true, and MSPM0 starts to send battery voltage data through debug printf port and CAN interface. The debug printf port send the data through XDS110 to CCS debug console window.

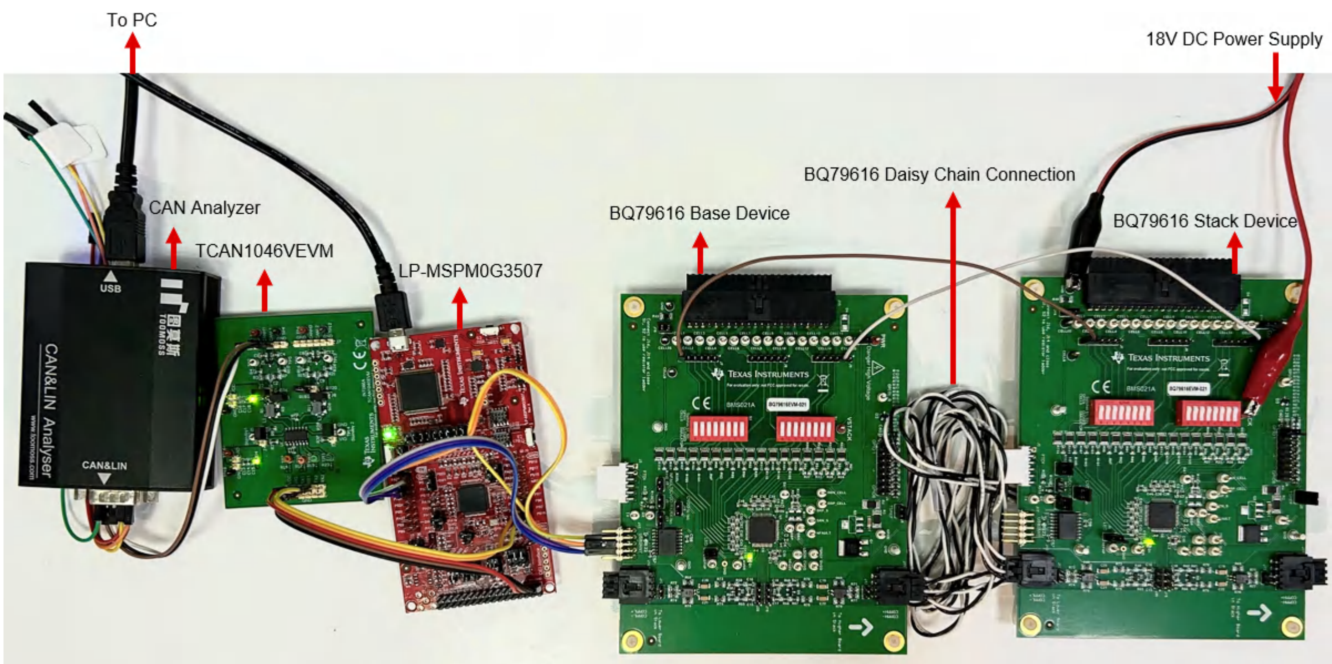

Figure 4-1. Hardware Setup

For the CAN bit timing settings, this demo code uses standard CAN, running at 500kbps and sampling point is 87.5%. Figure 4-2 is the MSPM0 CAN bit timing setting in System Configuration Tool. And in some CAN Analyzer host PC software, four CAN bit timing parameters (NBRP, NTSEG1 Tq, NTSEG2 Tq, NSJW Tq) need to be increased by 1 based on MSPM0 CAN bit timing settings to work properly, as shown in Figure 4-3.

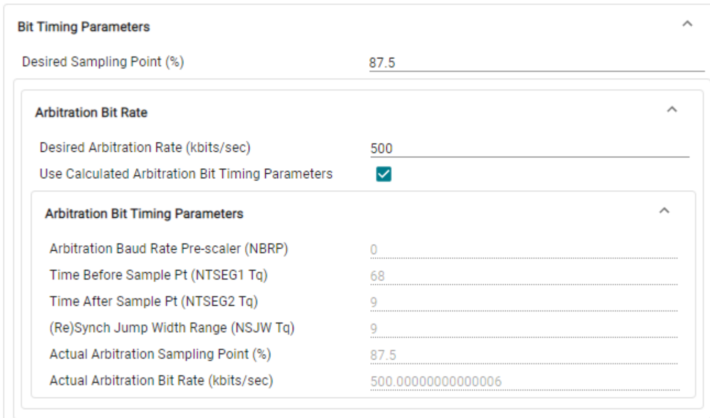

Figure 4-2. MSPM0 CAN Bit Timing Parameters Settings

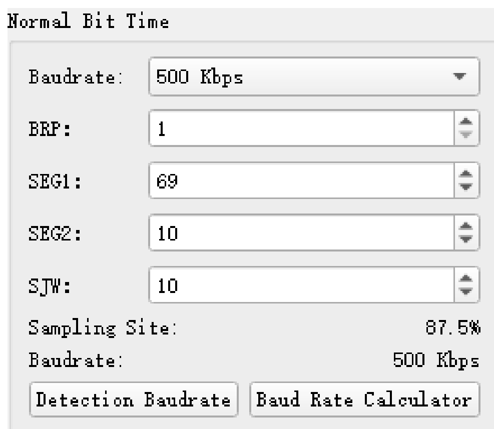

Figure 4-3. CAN Analyzer CAN Bit Timing Parameters Settings

4.2 Read Voltage

Read Voltage is a function that implement in main while, when S2 is pressed, MSPM0 starts reading voltage from BQ79616 devices and send these data to debug port and CAN interface.

Figure 4-4 and Figure 4-5 is a group test result that shown in CCS debug console windows and host PC CAN analyzer window.

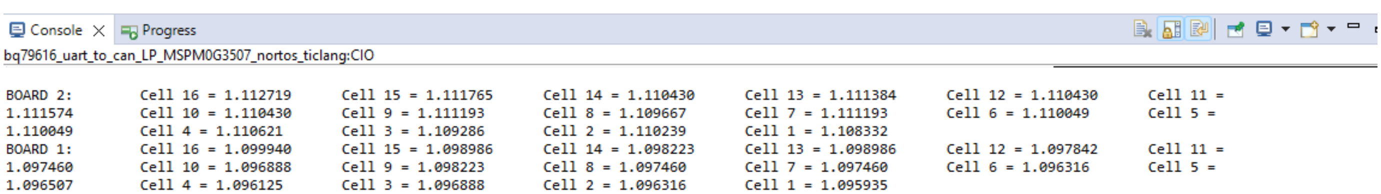

Figure 4-4. CCS Console printf Result

Line Num	ID(Hex)	Length	数据(Hex)	Timestamp	Dir	Frame Type	Frame Format	CAN Type
1	0x4	8	16 C0 16 BC 16 B4 16 B9	0.000000	Receive	Standard	Normal	CAN
2	0x4	8	16 B4 16 B9 16 B4 16 B8	0.010040	Receive	Standard	Normal	CAN
3	0x4	8	16 B1 16 B8 16 B2 16 B2	0.015000	Receive	Standard	Normal	CAN
4	0x4	8	16 B5 16 AE 16 B4 16 A9	0.031000	Receive	Standard	Normal	CAN
5	0x4	8	16 7C 16 78 16 74 16 77	0.041040	Receive	Standard	Normal	CAN
6	0x4	8	16 72 16 6F 16 6C 16 74	0.046000	Receive	Standard	Normal	CAN
7	0x4	8	16 6F 16 6F 16 6A 16 6A	0.062000	Receive	Standard	Normal	CAN
8	0x4	8	16 68 16 6D 16 6A 16 68	0.072040	Receive	Standard	Normal	CAN

Receive Num: 8 Send Num: 0 Error Num: 0

Figure 4-5. CAN Analyzer Result

5 Summary

This document introduces the BMS software design based on LP-MSPM0G3507 and BQ79616 EVM, hardware and software setup, and the test result. Customers can easily setup the BQ79616 test environment based on this document and demo code provided in MSPM0 SDK, for both hardware connection and software settings.

6 References

- Texas Instruments, *BQ79616 16-Series Battery Monitor, Balancer, and Integrated Hardware Protector Data Sheet*
- Texas Instruments, *BQ79616-Q1, BQ75614-Q1, BQ79656-Q1 Evaluation Modules User's Guide*
- Texas Instruments, *MSPM0G350x Mixed-Signal Microcontrollers With CAN-FD Interface Data Sheet*
- Texas Instruments, *MSPM0 G-Series 80-MHz Microcontrollers Technical Reference Manual*
- Texas Instruments, *MSPM0G3507 LaunchPad Development Kit User's Guide (LP-MSPM0G3507)*

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265