参考資料

Application Note

BQ25856-Q1 によるスーパーキャパシタの状態の測定

Ethan Galloway, Michael Bradbourne

概要

スーパーキャパシタは、通常のコンデンサまたはバッテリに比べて、最高の電力密度を実現します。高い電力密度と低 ESR の組み合わせにより、スーパーキャパシタの普及がますます進んでいます。スーパーキャパシタは、自動車のコンピュータや自動車のドアに電源バックアップを提供したり、サーバーに補助電源を供給したりするために使用されます。しかしながら、スーパーキャパシタは時間の経過とともに静電容量が減少し、等価直列抵抗 (ESR) が増加します。これらのコンデンサが必要なときに機能できるようにするためには、コンデンサの電流 ESR と静電容量を把握していることが重要です。BQ2575X および BQ2585X-Q1 IC ファミリは、マイコンと組み合わせることで、スーパーキャパシタの ESR と静電容量のおおよその測定値を提供し、コンデンサの寿命終了 (EOL) を推定するのに役立ちます。また、これらの IC は、充電電流と放電電流を制限し、出力電圧を精密に制御することで、スーパーキャパシタの寿命を延ばすのにも役立ちます。このアプリケーション ノートでは、BQ25856-Q1 と MSPM0L2228 マイコンを使用して、スーパーキャパシタの ESR と静電容量を概算する方法の概要とサンプル コードを紹介します。

目次

商標

1 概要

スーパーキャパシタは、バックアップ システムとしてますます普及しています。スーパーキャパシタは、医療機器、自動車のドア、車載コンピュータなどの最終製品にバックアップ電力を供給できます。スーパーキャパシタは、サーバーに補助的な電力を供給することもできます。

スーパーキャパシタは、バッテリ テクノロジーに比べてはるかに長い寿命と高い電力密度を実現しますが、経年劣化の影響を考慮する必要があります。スーパーキャパシタの経年劣化が進むと ESR は増加し、静電容量は減少します。これらの要因が相まって、スーパーキャパシタの利用可能な総エネルギーと電力を減少させます。

これらのコンデンサは、アプリケーションに応じて信頼性が高く、長寿命であることが求められます。このアプリケーション ノートでは、スーパーキャパシタの静電容量と ESR を迅速に測定する方法について説明します。ESR と静電容量に関する情報により、コンデンサの寿命終了 (EOL) に達する前にスーパーキャパシタを交換することができます。

2 スーパーキャパシタの寿命を延ばす方法

スーパーキャパシタの機能寿命は、電圧、充電電流、機械的ストレス、動作温度に依存します。多くのスーパーキャパシタメーカーは、静電容量が公称値の 80% に低下するか、または等価直列抵抗 (ESR) が公称値の 200% に増加すると、コンデンサが EOL に達したと判断しています。スーパーキャパシタが EOL に達した時期を判断するには、各スーパーキャパシタのメーカーにお問い合わせください。

このアプリケーション ノートでは主に、スーパーキャパシタの ESR と静電容量の測定方法に焦点を当てています。とはいえ、スーパーキャパシタの寿命を縮める可能性のあるいくつかの要因と、BQ2585X-Q1 を使用して寿命の低下を緩和する方法を以下に紹介します。

2.1 電圧の影響

スーパーキャパシタに過電圧を印加すると、スーパーキャパシタの合計寿命が短くなります。スーパーキャパシタをより低い電圧で充電すると、スーパーキャパシタの動作寿命を延長できます。BQ2575X と BQ2585X-Q1 の充電コントローラは、帰還抵抗の電圧精度が 0.5% であり、スーパーキャパシタを推奨されるフル充電電圧まで高精度で充電できます。

2.2 電流の影響

スーパーキャパシタへの大量の電流シンクまたはソースは、内部の発熱を引き起こします。スーパーキャパシタの仕様を上回る電流をシンクまたはソースすると、デバイスの寿命が短くなります。BQ2575X と BQ2585X-Q1 ファミリは、IBAT_REV レジスタを経由してスーパーキャパシタの合計ソース電流を制限するための放電電流制限機能を備えています。BQ2575X および BQ2585X-Q1 ファミリは、スーパーキャパシタに流れ込む電流を正確に制御するための充電電流設定も備えています。

2.3 温度の影響

温度が上昇すると、スーパーキャパシタの寿命は大幅に短くなります。スーパーキャパシタの温度要件については、スーパーキャパシタ メーカーにお問い合わせください。BQ2575X および BQ2585X-Q1 は JEITA に準拠しており、サーミスタと連携して、スーパーキャパシタの温度が上昇したときの充電電流または充電電圧を低減できます。

3 スーパーキャパシタの ESR と静電容量

3.1 コンデンサの静電容量と ESR の推定

スーパーキャパシタとコンデンサは、図 3-1 に示すように、抵抗とコンデンサとしてモデル化できます。

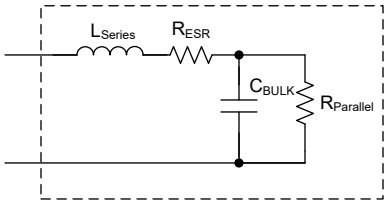

図 3-1. コンデンサ モデル

静電容量と ESR は、スーパーキャパシタの健全性を示す決定的な特性であり、時間とともに変化するため、このアプリケーション ノートではこれらの値の測定に焦点を当てています。

コンデンサの電流を定義する式を式 1 に示します。

$$i = C\frac{dV}{dt} \tag{1}$$

BQ2585X-Q1 および BQ2575X ファミリの部品は、定電流源として機能できます。定電流源を使用するということは、電圧の変化が時間に対して直線的であることを意味します。式 1 は式 2 のように書き換えて静電容量を計算できます。BQ2585X-Q1 IC は、オンボード ADC で電圧の変化を測定できます。充電開始後の初期電圧と最終電圧を測定してから、充電を無効にします。電圧の変化は式 3 次で求められます。

$$C = \frac{i\Delta t}{\Delta V} \tag{2}$$

$$\Delta V = V_{Final} - V_{Initial} \tag{3}$$

静電容量は、式 4 で計算できます。

$$C = \frac{i\Delta t}{V_{Final} - V_{Initial}} \tag{4}$$

既知の電流を使用して、既知の時間にわたって電圧の変化を測定することで、上記の式を使用してスーパーキャパシタの静電容量を計算できます。静電容量を時間の経過とともに追跡することで、コンデンサが間もなく EOL に達するかどうかを判定できます。

ESR も、時間の経過とともに追跡する必要があります。コンデンサの ESR での電圧降下はオームの法則で定義され、コンデンサ ESR の抵抗は \ で求めることができます。

$$R_{ESR} = \frac{V_{ESR}}{i} \tag{5}$$

デバイスがスーパーキャパシタを充電するために電流を供給している場合、コンデンサから測定される電圧は、コンデンサの電圧と、コンデンサの ESR (V_{ESR}) での電圧降下という 2 つの電圧の合計です。このコンデンサは、V_{ESR} を相殺するために、シンク電流もソース電流もなしで測定できます。ここでは、最終電圧を充電終了直前の電圧として定義します。終了電圧とは、電流が流れていない状態でのスーパーキャパシタの電圧のことです。終了電圧には電流がないため、ESR の両端に電圧はありません。

2 つの電圧測定値を差し引くと、スーパーキャパシタの ESR における電圧降下が得られます。

$$V_{Final} - V_{Ending} = \left(V_{ESR} + V_{CAP}\right) - \left(V_{CAP}\right) = V_{ESR} \tag{6}$$

式 6 の結果を式 5 に代入し、充電電流の設定値が分かれば、スーパーキャパシタの ESR を求めることができます。また、スーパーキャパシタの寿命全体にわたって ESR を追跡し、スーパーキャパシタが寿命終了に達したかどうかを推定することもできます。

テスト中に電流が一定に維持され、コンデンサがフル充電されないことを確認するため、以下の式を使用して、テスト後のコンデンサの終了電圧を計算できます。

$$V_{Ending} = \frac{1}{C}\int_0^t i \, \mathrm{d}\, t + V_{Start} \tag{7}$$

充電電流「i」は一定の値であるため、式 7 は以下に書き換えることができます。

$$V_{Final} = \frac{i\Delta t}{C} + V_{Start} \tag{8}$$

この式を使って、BQ2585X-Q1 IC が定電流モードに維持され、テストでコンデンサが完全に充電されないことを確認できます。また、BQ2585X-Q1 には、IC が定電圧モードか定電流モードかを示すステータス レジスタもあります。テスト後にステータス レジスタを読み取ることで、IC が定電流モードを維持していたことを確認できます。

以下のフロー チャートは、BQ25856-Q1 を使用してスーパーキャパシタの状態を測定する方法を示しています。

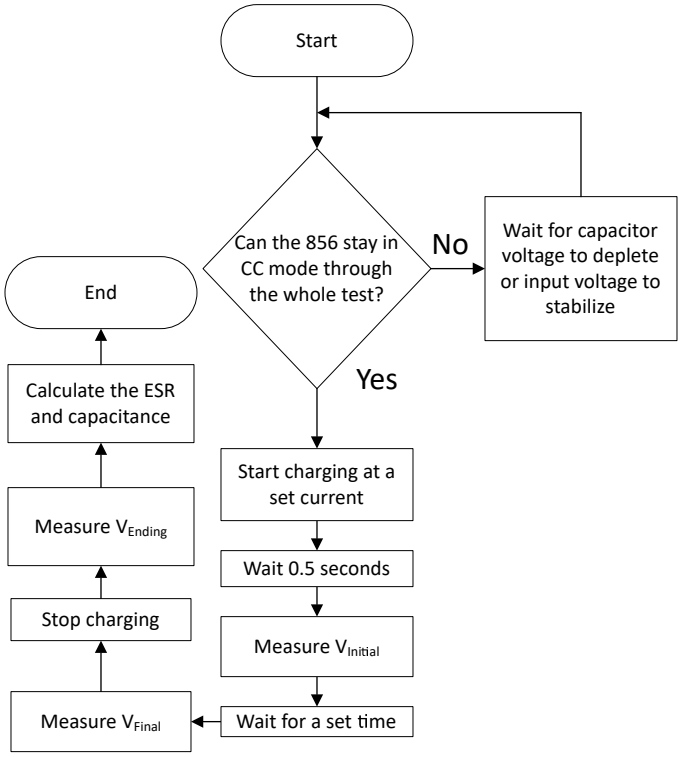

図 3-2. スーパーキャパシタの状態測定フロー チャート

3.2 BQ25856-Q1 によるスーパーキャパシタの状態の測定

以下の例では、BQ25856-Q1 IC を MSPM0L2228 マイコンと組み合わせて、直列接続した 8 個のスーパーキャパシタの静電容量と ESR を測定します。マイコン ホスト コントローラを搭載した BQ25856-Q1 は、以下の手順で ESR と静電容量を測定できます。

1. コンデンサの電圧を測定します
2. 合計充電時間と充電電流を計算して、充電プロセス全体を通じて BQ25856-Q1 が定電流モードにとどまるようにします
3. コンデンサの充電を開始します
4. 充電開始後の $V_{Initial}$ を測定します
5. 充電終了直前に V_{Ending} を測定します
6. コンデンサの充電を停止します
7. 充電電流がないときの V_{Final} を測定します=
8. $V_{Initial}$、V_{Ending}、および 2 つの測定間の時間を使用して静電容量を計算します
9. 充電電流 V_{Ending}、V_{Final} で ESR を計算します。

以下の結果は、以下の条件でテストされています。

- V_{AC} = 10V
- I_{CHG} = 1A
- t_{pulse} = 1s
- $C_{OUT\_nominal}$ = 10F x 8S1P
- $R_{ESR\_nominal}$ = 35 mΩ x 8S1P

図 3-3 では、BQ25856-Q1 はコンデンサを充電しながら静電容量と ESR を測定するのに約 3 秒かかります。必要に応じて、測定時間を短縮することも可能です。

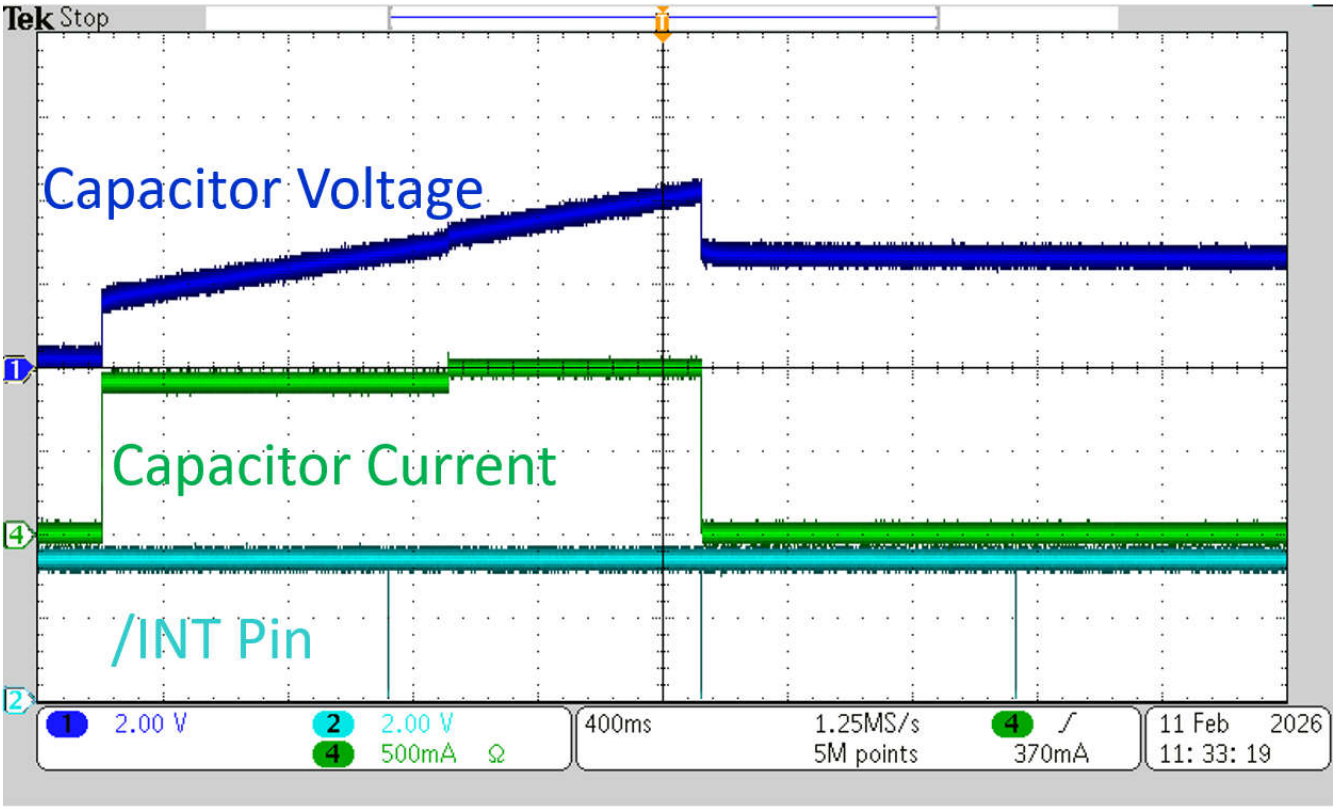

図 3-3. サンプルの測定波形

図 3-3 に、測定コードの実装を示す波形のサンプルを示します。デバイスは、チャネル 4 でプローブされている I_{CHG} 電流でコンデンサの充電を開始します。IC から観測されたコンデンサ電圧を、チャネル 1 でプローブします。ADC 測定が発生するたびに、デバイスの / INT ピンから割り込みがアサートされます。/INT ピンはチャネル 2 で監視され、コードのタイミングを提示します。チャネル 2 の最初のパルスは、$V_{Initial}$ を決定するための最初の ADC 測定値を示しています。その後、タイマが開始され、充電は 1 秒間実行されます。チャネル 2 の 2 番目のパルスは、V_{Ending} を決定するための 2 回目の ADC 測定値を示しています。ADC の測定後、充電は直ちに無効化され、コンデンサ電圧が安定するまでの間にもう 1 つの待機期間があります。チャネル 2 の 3 番目のパルスは、V_{Ending} を決定するための最後の ADC 測定値を示しています。式 5 と式 4 を用いることで、BQ25856-Q1 の ADC 測定値、BQ25856-Q1 の設定値から得られる既知の充電電流、およびマイコン タイマから得られる既知の充電時間を使用して、ESR と静電容量を求めることができます。

表 3-1. ADC 電圧測定

ADC 読み取り	電圧 (V)
初期電圧	2.586
ピーク電圧	4.000
最終電圧	2.454

表 3-2. スーパーキャパシタのパラメータの計算

パラメータ	結果
直列接続した 8 個のコンデンサの静電容量	0.71F
直列接続された 8 個のコンデンサの ESR	1.55 Ω
コンデンサあたりの平均静電容量	5.68F
コンデンサあたりの平均 ESR	194mΩ

測定が行われた後、コンデンサが充電され、バックアップ電源を供給する準備ができるまでフル充電サイクルが再開されます。計算された静電容量と ESR を時間の経過とともに追跡することで、コンデンサの寿命終了に達したタイミングを判定できます。このテストでは、コンデンサは古く摩耗しており、ESR が公称値よりもはるかに大きく、静電容量が元の値の 70% まで低下したため、寿命終了に達しました。計算された静電容量が公称値の 80% を下回るか、ESR が公称値の 200% を超えると、コンデンサは EOL に達します。

3.3 逆方向モードでの静電容量と ESR の推定

特定の条件が満たされている場合、逆方向モードで BQ25856-Q1 を使用して静電容量と ESR を測定することもできます。BQ25856-Q1 は、IC が逆方向モードの間に、コンデンサからの電流をドレインするために VAC 側に負荷を必要とします。また、BQ25856-Q1 は主に逆方向モードでは定電圧源として機能します。これは、電力供給の際、スーパーキャパシタに線形電流が流れないことを意味します。

コンデンサからの合計電流を測定する方法はいくつかあります。まず、システムにコンデンサからの平均ソース電流を測定する方法がある場合、平均電流に時間を乗じた値を、次の積分に代入できます: $\int_0^t i \, \mathrm{d}\, t$ 。これにより、クーロン単位の値が得られます。充電量に対するコンデンサの式は次のとおりです。

$$C = \frac{q}{V} \tag{9}$$

コンデンサの式は、次のように書き換えることができます。

$$\frac{\int_0^t i \, \mathrm{d}\, t}{V_{Ending} - V_{Start}} = C \tag{10}$$

式 10 では、コンデンサの ESR での電圧降下は考慮されていません。TI では、「V_{Ending}」と「V_{Start}」のコンデンサ電流がほぼ等しくなるようにすることを推奨しています。

ESR は、既知の時間における 2 つの異なるコンデンサ電流間の電圧の差を測定することで計算できます。

$$\frac{V_{C2} - V_{C1} - \frac{1}{C}\int_0^t i \, \mathrm{d}\, t}{(i_2 - i_1)} = R_{ESR} \tag{11}$$

式 11 において、「V_{C1}」と「V_{C2}」は、コンデンサのソース電流が「i_1」と「i_2」の場合に測定されたコンデンサ電圧です。「t」は「V_{C1}」と「V_{C2}」の測定間隔です。 静電容量「C」は式 10 を用いて計算できます。

または、マイコンが平均電流を計算できない場合、IBAT_REV レジスタを使用してコンデンサから流れ出る電流を 5A、10A、15A、20A に制限できます。コンデンサのソース電流が制限されている場合、電流は線形になり、セクション 3.1 の式を使用して静電容量と ESR を計算できます。VAC への負荷が入力に十分な負荷をかけることで、IBAT_REV がアクティブになり、放電電流が制限されることを確認してください。

4 まとめ

スーパーキャパシタは、電解コンデンサやバッテリに比べて大きな利点があります。スーパーキャパシタは、バッテリに比べて電力密度がかなり高く、寿命がかなり長くなります。また、スーパーキャパシタは電解コンデンサよりも多くのエネルギーを蓄積することができます。しかしながら、スーパーキャパシタも経年劣化し、使用可能な電力とエネルギーは時間とともに減少します。BQ2585X-Q1 および BQ2575X の IC ファミリは、スーパーキャパシタの ESR と静電容量を推定できます。この情報は、スーパーキャパシタの状態を追跡するのに役立ち、この情報を使用してコンデンサの寿命終了を推定できます。

5 付録 - BQ25858-Q1 向けマイコン コード サンプル

```
extern volatile bool timer_exp;

int main(void)
{
    SYSCFG_DL_init();                                   // Initialize the I2C and the timer via SuperCapHealth.sysconfig


    __NVIC_EnableIRQ(I2C_0_INST_INT_IRQN);              // Initialize the I2C interrupt routine for I2C communication

    __NVIC_EnableIRQ(TIMER_0_INST_INT_IRQN);            // Initialize the timer interrupt routine.
    timer_exp = 0;                                      // Initialize the timer flag.

    // I2C Settings
    uint8_t DIS_WD[1] = {0x05};                         // Disable Watchdog timer
    uint8_t EN_CHG[1] = {0xC9};                         // 0x09 for BQ25858-Q1
    uint8_t DIS_CHG[1] = {0xC8};                        // 0x08 for BQ25858-Q1
    uint8_t DIS_PRECHG[1] = {0x06};                     // Disable Precharge and Termination
    uint8_t EN_ADC[1] = {0xE0};                         // One shot conversion mode to control when ADC Reads
    uint8_t VOUT_ADC[1] = {0xEF};                       // Set only the VOUT ADC to read (shortens conversion time)
    uint8_t DIS_INT_MASK[3] = {0x7F, 0xFF, 0xFF};       // Set only the ADC interrupt

    // Current Parameter
    uint8_t CHG_CURRENT[2] = {0x50, 0x00};              //  1 A IOUT setting
    volatile uint16_t ICHG = 0x0000;                    // 16 bit variable for charge current
    volatile double_t current_conv = 0.05;              // 50 mA per bit on BQ2585X-Q1 Register
    volatile double_t current = 0.0;                    // Current value for ESR and CAP calculations
    // Determine Charge Current in AMPERES
    ICHG = (((uint16_t)CHG_CURRENT[1] << 8) | (uint16_t)CHG_CURRENT[0]) >> 2;
    current = (double_t)ICHG * current_conv;

    // Voltage Parameter
    uint8_t VBAT[2] = {0x00, 0x00};                     // Variable to read ADC Conversion
    volatile unsigned int adc_result0 = 0;              // 16 bit variable for ADC Conversion result
    volatile unsigned int adc_result1 = 0;
    volatile unsigned int adc_result2 = 0;
    volatile double_t volt_conv = 0.002;                // 2 mV per bit on BQ2585X-Q1 ADC
    volatile double_t voltage0 = 0;                     // Voltage value for ESR and CAP calculations
    volatile double_t voltage1 = 0;
    volatile double_t voltage2 = 0;

    // Time parameter
    volatile double_t charge_time = 1.0;                // 1000 ms charge time

    //Results
    volatile double_t ESR = 0.0;                        // ESR Result
    volatile double_t CAP = 0.0;                        // Capacitance result
```

```
// Disable Watchdog to avoid registers being overwritten
I2C_Write(BQ2585X_TIMER_CONTROL, DIS_WD, 1);

// Disable Charge to set proper settings
I2C_Write(BQ2585X_CHARGER_CONTROL, DIS_CHG, 1);

// Set 1 A IOUT for measurement calculations
I2C_Write(BQ2585X_CHARGE_CURRENT_LIMIT_LSB, CHG_CURRENT, 2);

// Disable Precharge and Termination
I2C_Write(BQ2585X_PRECHARGE_AND_TERMINATION_CONTROL, DIS_PRECHG, 1);

// Set only VOUT_ADC for fast ADC operation
I2C_Write(BQ2585X_ADC_CHANNEL_CONTROL, VOUT_ADC, 1);

// Set only ADC interrupts
I2C_Write(BQ2585X_MASK_1, DIS_INT_MASK, 3);

while (1) {
    // Enable the charge sequence
    I2C_Write(BQ2585X_CHARGER_CONTROL, EN_CHG, 1);
    // Start timer to remove EN_CHG delay
    DL_TimerG_startCounter(TIMER_0_INST);
    // Wait for the 1s timer to finish
    while (!timer_exp);
    // Reset timer flag for next timer cycle
    timer_exp = 0;

    // Perform ADC conversion for initial voltage
    I2C_Write(BQ2585X_ADC_CONTROL, EN_ADC, 1);
    // Wait for ADC conversion to finish
    while (DL_GPIO_readPins(BQ_INT_PORT, BQ_INT_PIN_0_PIN));
    // Read ADC conversion results
    I2C_Read(BQ2585X_VBAT_ADC_LSB, VBAT, 2);
    // Store ADC conversion results for later calculations
    adc_result0 = (VBAT[1] << 8) | (VBAT[0]);

    // Start the timer
    DL_TimerG_startCounter(TIMER_0_INST);
```

```
        // Wait for the 1 s timer to finish
        while (!timer_exp);

        // Perform ADC conversion for max voltage
        I2C_Write(BQ2585X_ADC_CONTROL, EN_ADC, 1);
        // Wait for ADC conversion to finish
        while (DL_GPIO_readPins(BQ_INT_PORT, BQ_INT_PIN_0_PIN));
        // Disable charge
        I2C_Write(BQ2585X_CHARGER_CONTROL, DIS_CHG, 1);
        // Read ADC conversion results
        I2C_Read(BQ2585X_VBAT_ADC_LSB, VBAT, 2);
        // Store ADC conversion results for later calculations
        adc_result1 = (VBAT[1] << 8) | VBAT[0];
        // Reset timer flag for next timer cycle
        timer_exp = 0;


        // Start timer to let output voltage settle
        DL_TimerG_startCounter(TIMER_0_INST);
        // Wait for the 1 s timer to finish
        while (!timer_exp);
        // Reset timer flag for next timer cycle
        timer_exp = 0;


        // Perform ADC conversion for final voltage
        I2C_Write(BQ2585X_ADC_CONTROL, EN_ADC, 1);
        // Wait for ADC conversion to finish
        while (DL_GPIO_readPins(BQ_INT_PORT, BQ_INT_PIN_0_PIN));
        // Read ADC conversion results
        I2C_Read(BQ2585X_VBAT_ADC_LSB, VBAT, 2);
        // Store ADC conversion results for later calculations
        adc_result2 = (VBAT[1] << 8) | (VBAT[0]);

        // Convert ADC values to a voltage
        voltage0 = (double_t)adc_result0 * volt_conv;
        voltage1 = (double_t)adc_result1 * volt_conv;
        voltage2 = (double_t)adc_result2 * volt_conv;

        // Calculations for Capacitance and ESR

        CAP = (current * charge_time) / (voltage1 - voltage0);
        ESR = (voltage1 - voltage2) / current;
    }
}
```

```
int main(void) {
  SYSCFG_DL_init(); // Initialize the I2C and the timer via SuperCapHealth.sysconfig
  __NVIC_EnableIRQ(I2C_0_INST_INT_IRQN); // Initialize the I2C interrupt routine for I2C
communication
  __NVIC_EnableIRQ(TIMER_0_INST_INT_IRQN); // Initialize the timer interrupt routine.
  timer_exp = 0; // Initialize the timer flag.
  // I2C Settings
  uint8_t DIS_WD[1] = {0x05}; // Disable Watchdog timer
  uint8_t EN_CHG[1] = {0xC9}; // 0x09 for BQ25858-Q1
  uint8_t DIS_CHG[1] = {0xC8}; // 0x08 for BQ25858-Q1
  uint8_t DIS_PRECHG[1] = {0x06}; // Disable Precharge and Termination
  uint8_t EN_ADC[1] = {0xE0}; // One shot conversion mode to control when ADC Reads
  uint8_t VOUT_ADC[1] = {0xEF}; // Set only the VOUT ADC to read (shortens conversion time)
  uint8_t DIS_INT_MASK[3] = {0x7F, 0xFF, 0xFF}; // Set only the ADC interrupt for the best results
  // Current Parameter
  uint8_t CHG_CURRENT[2] = {0x50, 0x00}; // 1 A IOUT setting
  volatile uint16_t ICHG = 0x0000; // 16 bit variable for charge current volatile
  double_t current_conv = 0.05; // 50 mA per bit on BQ2585X-Q1 Register
  volatile double_t current = 0.0; // Current value for ESR and CAP calculations
  // Determine Charge Current in AMPERES
  ICHG = (((uint16_t)CHG_CURRENT[1] << 8) | (uint16_t)CHG_CURRENT[0]) >> 2;
  current = (double_t)ICHG * current_conv;
  // Voltage Parameter
  uint8_t VBAT[2] = {0x00, 0x00};
  // Variable to read ADC Conversion
  volatile unsigned int adc_result0 = 0; // 16 bit variable for ADC Conversion result
  volatile unsigned int adc_result1 = 0;
```

```
  volatile unsigned int adc_result2 = 0;
  volatile double_t volt_conv = 0.002; // 2 mV per bit on BQ2585X-Q1 ADC
  volatile double_t voltage0 = 0; // Voltage value for ESR and CAP calculations
  volatile double_t voltage1 = 0;
  volatile double_t voltage2 = 0;
  // Time parameter
  volatile double_t charge_time = 1.0; // 1000 ms charge time
  //Results
  volatile double_t ESR = 0.0; // ESR Result
  volatile double_t CAP = 0.0; // Capacitance result
  // Disable Watchdog to avoid registers being overwritten
  I2C_Write(BQ2585X_TIMER_CONTROL, DIS_WD, 1);
  // Disable Charge to set proper settings
  I2C_Write(BQ2585X_CHARGER_CONTROL, DIS_CHG, 1);
  // Set 1 A IOUT for measurement calculations
  I2C_Write(BQ2585X_CHARGE_CURRENT_LIMIT_LSB, CHG_CURRENT, 2);
  // Disable Precharge and Termination
  I2C_Write(BQ2585X_PRECHARGE_AND_TERMINATION_CONTROL, DIS_PRECHG, 1);
  // Set only VOUT_ADC for best operation
  I2C_Write(BQ2585X_ADC_CHANNEL_CONTROL, VOUT_ADC, 1);
  // Set only ADC interrupts for best operation
  I2C_Write(BQ2585X_MASK_1, DIS_INT_MASK, 3);
  while (1) {
    // Enable the charge sequence
    I2C_Write(BQ2585X_CHARGER_CONTROL, EN_CHG, 1);
    // Start timer to remove EN_CHG delay
    DL_TimerG_startCounter(TIMER_0_INST);
    // Wait for the 1 s timer to finish
    while (!timer_exp);
    // Reset timer flag for next timer cycle
    timer_exp = 0;
    // Perform ADC conversion for initial voltage
    I2C_Write(BQ2585X_ADC_CONTROL, EN_ADC, 1);
    // Wait for ADC conversion to finish
    while (DL_GPIO_readPins(BQ_INT_PORT, BQ_INT_PIN_0_PIN));
    // Read ADC conversion results
    I2C_Read(BQ2585X_VBAT_ADC_LSB, VBAT, 2);
    // Store ADC conversion results for later calculations
    adc_result0 = (VBAT[1] << 8) | (VBAT[0]);
    // Start the timer
    DL_TimerG_startCounter(TIMER_0_INST);
    // Wait for the 1 s timer to finish
    while (!timer_exp);
    // Perform ADC conversion for max voltage
    I2C_Write(BQ2585X_ADC_CONTROL, EN_ADC, 1);
    // Wait for ADC conversion to finish
    while (DL_GPIO_readPins(BQ_INT_PORT, BQ_INT_PIN_0_PIN));
    // Disable charge
    I2C_Write(BQ2585X_CHARGER_CONTROL, DIS_CHG, 1);
    // Read ADC conversion results
    I2C_Read(BQ2585X_VBAT_ADC_LSB, VBAT, 2);
    // Store ADC conversion results for later calculations
    adc_result1 = (VBAT[1] << 8) | VBAT[0];
    // Reset timer flag for next timer cycle
    timer_exp = 0;
    // Start timer to let output voltage settle
    DL_TimerG_startCounter(TIMER_0_INST);
    // Wait for the 1 s timer to finish
    while (!timer_exp);
    // Reset timer flag for next timer cycle
    timer_exp = 0;
    // Perform ADC conversion for final voltage
    I2C_Write(BQ2585X_ADC_CONTROL, EN_ADC, 1);
    // Wait for ADC conversion to finish
    while (DL_GPIO_readPins(BQ_INT_PORT, BQ_INT_PIN_0_PIN));
    // Read ADC conversion results
    I2C_Read(BQ2585X_VBAT_ADC_LSB, VBAT, 2);
    // Store ADC conversion results for later calculations
    adc_result2 = (VBAT[1] << 8) | (VBAT[0]);
    // Convert ADC values to a voltage
    voltage0 = (double_t)adc_result0 * volt_conv;
    voltage1 = (double_t)adc_result1 * volt_conv;
    voltage2 = (double_t)adc_result2 * volt_conv;
    // Calculations for Capacitance and ESR
    CAP = (current * charge_time) / (voltage1 - voltage0);
    ESR = (voltage1 - voltage2) / current;
  }
}
```

6 参考資料

1. Wiley Advanced、『スーパーキャパシタの多くの死：劣化、経年劣化、および性能低下』、Web ページ。
2. テキサス インスツルメンツ、『BQ25856-Q1：車載用、スタンドアロン / I2C 制御、1 ～ 14 セル双方向昇降圧バッテリ充電コントローラ』、データシート。
3. テキサス インスツルメンツ、『BQ25856-Q1 によるコンデンサ バックアップ回路』、アプリケーション ノート。

重要なお知らせと免責事項

TI は、技術データと信頼性データ (データシートを含みます)、設計リソース (リファレンス デザインを含みます)、アプリケーションや設計に関する各種アドバイス、Web ツール、安全性情報、その他のリソースを、欠陥が存在する可能性のある「現状のまま」提供しており、商品性および特定目的に対する適合性の黙示保証、第三者の知的財産権の非侵害保証を含むいかなる保証も、明示的または黙示的にかかわらず拒否します。

これらのリソースは、TI 製品を使用する設計の経験を積んだ開発者への提供を意図したものです。(1) お客様のアプリケーションに適した TI 製品の選定、(2) お客様のアプリケーションの設計、検証、試験、(3) お客様のアプリケーションに該当する各種規格や、その他のあらゆる安全性、セキュリティ、規制、または他の要件への確実な適合に関する責任を、お客様のみが単独で負うものとします。

上記の各種リソースは、予告なく変更される可能性があります。これらのリソースは、リソースで説明されている TI 製品を使用するアプリケーションの開発の目的でのみ、TI はその使用をお客様に許諾します。これらのリソースに関して、他の目的で複製することや掲載することは禁止されています。TI や第三者の知的財産権のライセンスが付与されている訳ではありません。お客様は、これらのリソースを自身で使用した結果発生するあらゆる申し立て、損害、費用、損失、責任について、TI およびその代理人を完全に補償するものとし、TI は一切の責任を拒否します。

TI の製品は、 TI の販売条件 、 TI の総合的な品質ガイドライン 、 ti.com または TI 製品などに関連して提供される他の適用条件に従い提供されます。TI がこれらのリソースを提供することは、適用される TI の保証または他の保証の放棄の拡大や変更を意味するものではありません。 TI がカスタム、またはカスタマー仕様として明示的に指定していない限り、TI の製品は標準的なカタログに掲載される汎用機器です。

お客様がいかなる追加条項または代替条項を提案する場合も、TI はそれらに異議を唱え、拒否します。

最終更新日：2025 年 10 月