# RTC(時計Unit)に 関わる 補足情報

前回、左のRTCを使用しましたが 左のRX8900に 限らずマイコンに付ける時計Unit の 使い方として マイコンの主電源が OFF状態の時も、RTCの時刻計時機能を 維持するため、電池や スーパーキャパシタ等のバックアップ電源機能を持ちます。 左上の Unit の場合、1Fのスーパーキャパシタを満充電状態にして、時刻を設定して、時計Unitを I2Cバスから切り離して、1週間後に接続して時刻を読み出すと正常に時刻を刻んでいました。

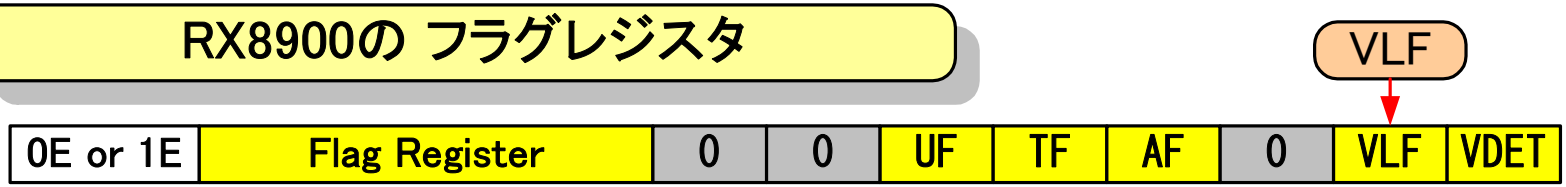
更に長い期間 放置した実験は、した事がないので 最大どのくらい持つかは分かりません。
一応、一週間は持つようです。

で、今回の補足情報は、マイコンの 電源ON時の、初期化処理において、RTCは、計時動作を続けている場合は、初期化してはならない。
という事です。

せっかく RTCが、正確な時刻で計時動作を続けているのに、初期化して 0年 1月 1日 0時 0分 0秒に初期しては、時刻を 計時する意味が ないという事です。 しかし、かなり長期のバックアップで、電池やスーパーキャパシタが、干上がっていた場合は、RTCも停止しているので、初期化処理が必要になります。

で、今回使用した RTC RX8900には、フラグレジスタに VLFフラグ Voltage Low Flag があります。 このフラグを最初に読み取り、0 ( 計時動作 継続中 )で あれば、初期化する必要は、ありません。 1 に なっていた場合は、初期化する必要が あります。

# RX8900の フラグレジスタ




UF: Update Flag 時刻更新終了時にセットされます。0 を書き込む事で リセット します。 1 は、書き込み出来ません。

TF: Timer Flag タイマーカウンタが、ゼロになった時 1 になります。 0 を書き込む事で リセットします。 1 は、書き込み出来ません。

AF: Alarm Flag アラームが発生すると 1 になります。0 を書き込む事で リセット します。 1 は、書き込み出来ません。

VLF: Voltage Low Flag 本フラグは、以下の2要因で セットされます。

1) ICの電源電圧が VLOW電圧を下回った時

2) 水晶発振が 約 10ms以上 停止したとき

VLFビットが、1 を示している場合は、全てのレジスタデータの初期化を行って下さい。 VLF は 0 を書き込むまで、1 を 保持します。

RESETビットには、影響は受けません。 VLOW電圧の検出は常時監視です。

VDET: ICの電源電圧が、VDET電圧よりも低下したことを検出して温度補償回路の停止を検出保持します。 VDET=1 の時は温度補正が停止した履歴を示し 0 を 書き込むまで保持します。 RESETビットには、影響は受けません。

以下のソースは、RTC_RX8900_Sub.c 内の 初期化処理 init_rx8900関数の 先頭部分です。
フラグレジスタの アドレスは、0Eh です。 因みに I2Cデバイスアドレス SA_8900 は、32h です。
フラグレジスタを 読み込み b1 の VLFフラグが、ゼロであるか調べます。
ゼロであれば、時刻の計時処理が 正常に行われている事になり、
初期化処理は行わずに リターンします。 よって、アプリ側では init_rx8900初期化関数を、一律 呼び出して かまいません。

```
BYTE init_rx8900( void )
{
    BYTE sts, sts2, reg;
    BYTE dt[8];

    set_timer_10m1( 30 );              // Wait 0.3秒
    while( get_timer_10m1() > 0 );

    reg = 0x0E;         // Flag Register Address
    sts = get_rtc_reg( SA_8900, reg ) & 0x3b;  // Flag読込み
    sts2 = sts & 0x02;                 // VLF 絞り込み
    if( sts2 == 0 )  return  0;        // 何もせずに 戻る
```

因みに、このVLFの確認処理は、バージョン 08 の IOCSから 反映しています。

## RTC RX8900の　電源OFF時の癖？

USBシリアルモジュールから、電源を供給している時、USBケーブルをパソコンから、引き抜いても、マイコンボードの電源LEDが　点灯しているので、ビックリされる方が いると思いますので、説明しておきます。

RX8900は、外から電源を供給している時、外部電源供給端子と　バックアップ電源端子をPチャネル MOS-FETにて、ショートさせています。　バックアップの スーパーキャパシタ等に充電するためだと思います。

ただし、外部電源と　バックアップ電源をショートさせると、外部電源を切断した際に、バックアップ側から、電流が流れ出るのですぐに、停電した事が検出 出来ません。

よって、定期的に　一瞬FETのスイッチを切って外部電源の 電圧を確認します。　そして停電を検出した場合、FETのスイッチを切ったままにして、バックアップモードに移行します。

その停電確認の定期的に確認する周期がハッキリ分かりませんが、すぐに電源が切れる場合もあれば、数秒ほどバックアップ電源から電流が　流れ出てから切れる場合もあるようです。

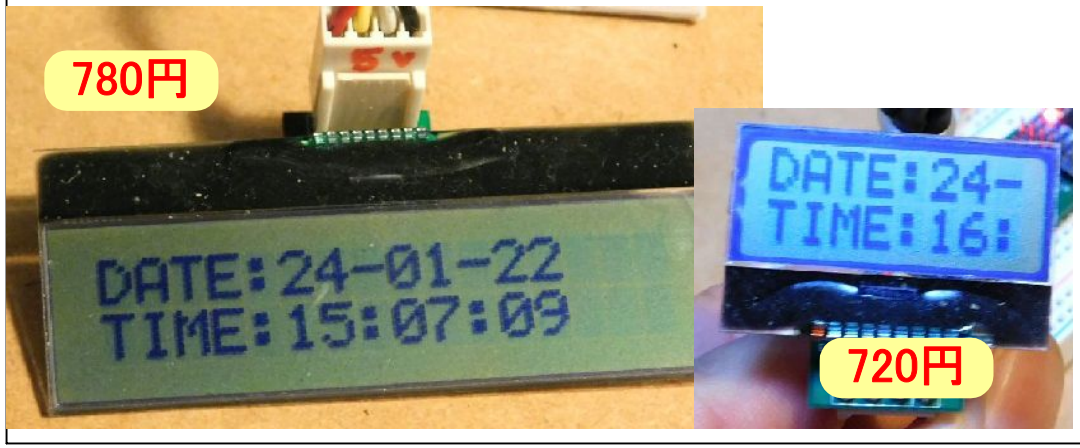
もしかして私が、RTC基板に 10uFのコンデンサを　外部電源側に付けていたので、10uFのコンデンサの放電時間も影響している事が考えられます。

この現象については、検討しておきます。

## バックライト無し 16x2LCD ドライバ処理

前回は、有機EL表示器 SO1602を 表示デバイスに使用しました。 緑色の視認性のいい表示器です。 I2Cから使う上でも、安定して通信を行う事が出来ます。 緑色文字表示の物で 1個 1,580円で、価格は やや高いです。

今回は、液晶表示器で、バックライト無しで16文字 2行表示の AQM1602Aを 使用してみます。 価格は、足ピン変換基板付きで 780円です。 8文字2行もあります。 足ピン変換基板付きキットが、720円です。







今回作成した AQM製 LCDドライバモジュールの名前は、i2c_lcd_aqm_sub.c です。

実際、ソフトを組んだ感じとしては、一応動きますが、やや癖が あります。左下の画像は、既にDATE、TIMEを表示してますが、どちらも同じソフトで表示できます。 8文字2行の表示器で 8文字を超える文字列を表示すると左端から、8文字の表示が でます。 9文字目以降は範囲外で出ません。 という事です。 それとは別に このLCDは、5Vでも 3.3Vでも動作させる事が出来ますが、初期化処理にて、電源電圧により 一部変えないといけないコマンドが あります。 文字表示のコントラストの設定値を 変える必要があります。 この設定変更を 行わないと文字表示が見えなくなるか、非常 に薄くかすれる、または文字表示部分が、黒く 四角く表示され文字が見えないという現象が 発生します。

因みに、OLED SO1602は、設定の変更なしで5Vでも 3.3Vでも 表示出来ます。

## AQM LCD用ドライバ i2c_lcd_aqm_sub.c

AQM LCD用の ドライバのソースファイル i2c_lcd_aqm_sub.c では、初期化処理 aqm_lcd_init関数に 引数が 有ります。

```
//**********************************
//**  AQM LCD 初期化                **
//** ------------------------------ **
//**  sw : 使用電源電圧選択         **
//**         =0 : 5V                **
//**         =1 : 3.3V              **
//**********************************
void  aqm_lcd_init( BYTE sw )
{
```

aqm_lcd_init関数の引数 sw が、0 で あれば 電源 5V用で、1 で あれば 電源 3.3V用です。

アプリから主に使用する関数は、以下の3つです。

```
void  aqm_lcd_init( BYTE sw );
                  // AQM LCD 初期化
void  aqm_lcd_txprn1( char *tx );
                  // １行目に 文字列表示
void  aqm_lcd_txprn2( char *tx );
                  // ２行目に 文字列表示
```

上記、プロトタイプ宣言は、R8CM12_IOCS.h に追加されています。 ドライバルーチンの使い方は、SO1602の時と 殆ど同じです。

表示画面のチラつきは、有機EL表示器に比べ液晶表示器は 殆ど無いように見えます。

液晶表示器は、文字表示の応答速度が、やや遅いようで、目立たないのかもしれません。

この i2c_lcd_aqm_sub.c も、IOCS¥I2C_Device フォルダーに 追加しておきます。

視聴者の皆様へ

大変　お待たせしました。
　ここから、I2Cの　I/O　エクスパンダーの話になります。

まずは、今回使用する　MCP23017の事についての説明ですが
過去に、RX220マイコンの　SPI周辺回路を使って MCP23S17を
使った実験を行いました。

　インタフェースが　I2Cか　SPIかの違いだけで、その他の部分
は、全く同じなので、過去の動画の一部分を使い
I/Oエクスパンダー MCP23017の説明を　行います。

# I/O Exp MCP23017 周辺回路

MCP23S17( SPI仕様 )と MCP23017( I2C仕様 )の ピン アサインの異なる部分を 示します。( 青表示 )

足ピンの 11 ～ 14 までの 4ピン部分が、SPI仕様と、I2C仕様で異なります。

I2Cでは、11ピンと 14ピンが NC 未接続となっています。

ついでに、デバイスアドレス設定の A2 A1 A0 の A2は、GND 接続で、A1と A0 に 0 ～ 3 の アドレスが 設定出来る様に DIPスイッチを 付けました。

それと この MCP23x17は 別途 RESET信号が 必要です。