秋月電子の 8文字 2行の　I2C LCD

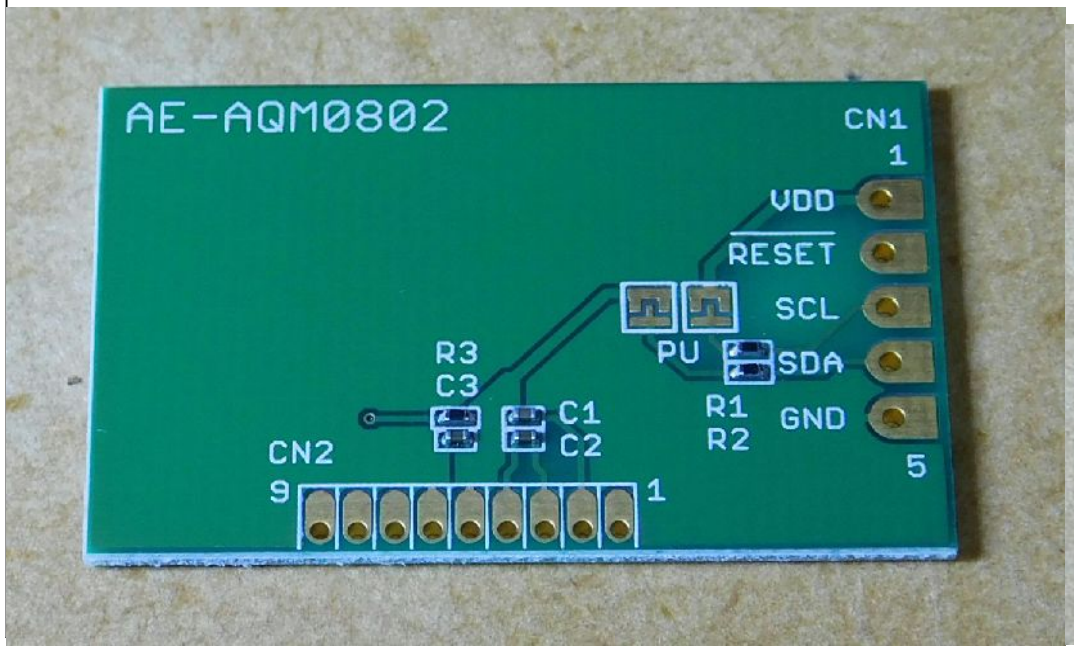
I2C　LCDは、9ピンの
接続端子が 出ています。
この向きから見て
左端が、1ピンです。

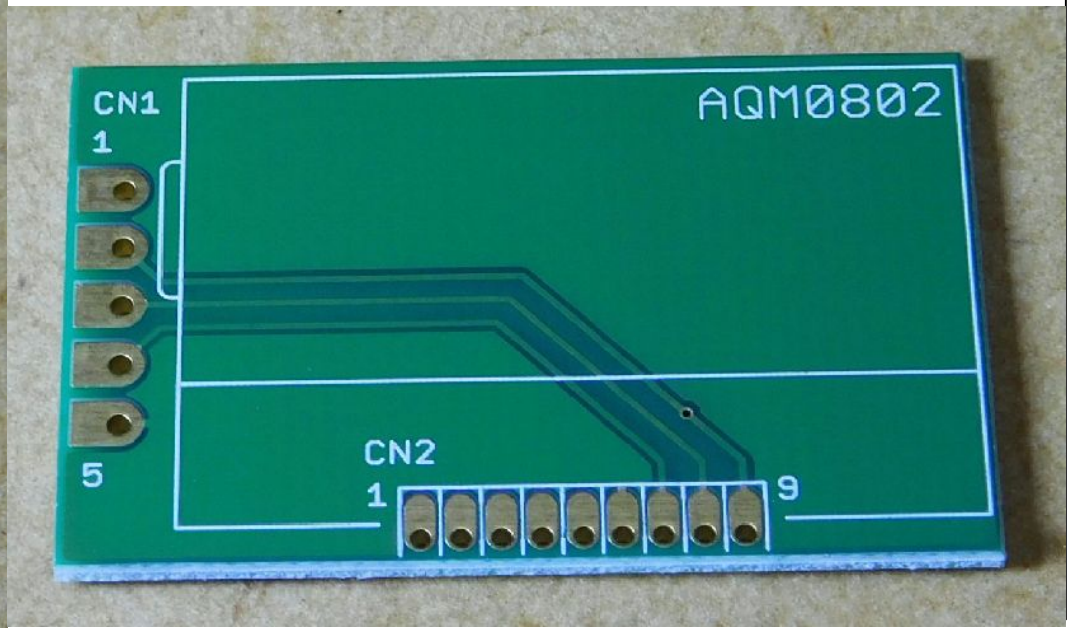
秋月電子の 8文字 2行の　I2C LCD用 ピッチ変換基板

ピッチ変換基板の表面と裏面

まあ、どちらが表面で、どちらが裏面と決まっては、無いでしょうが、仮に左側の小さい表面実装部品を実装した面を 表面と呼ぶ事にします。 表面には、10KΩの プルアップ抵抗を接続するパッドが、あります。 今回は、このプルアップ抵抗は、使いません。

右側の裏面は、AQM0802 LCDのピンを挿入する面です。 白い四角で LCDの外形を描いてあります。 CN2の左端のピンが、1ピンです。 左側面の CN1 は、1 ～ 5 の番号が付いています。 各ピンの信号名は、
1： VDD 、 2： /RESET 、 3： SCL
4： SDA 、 5： GND と なります。

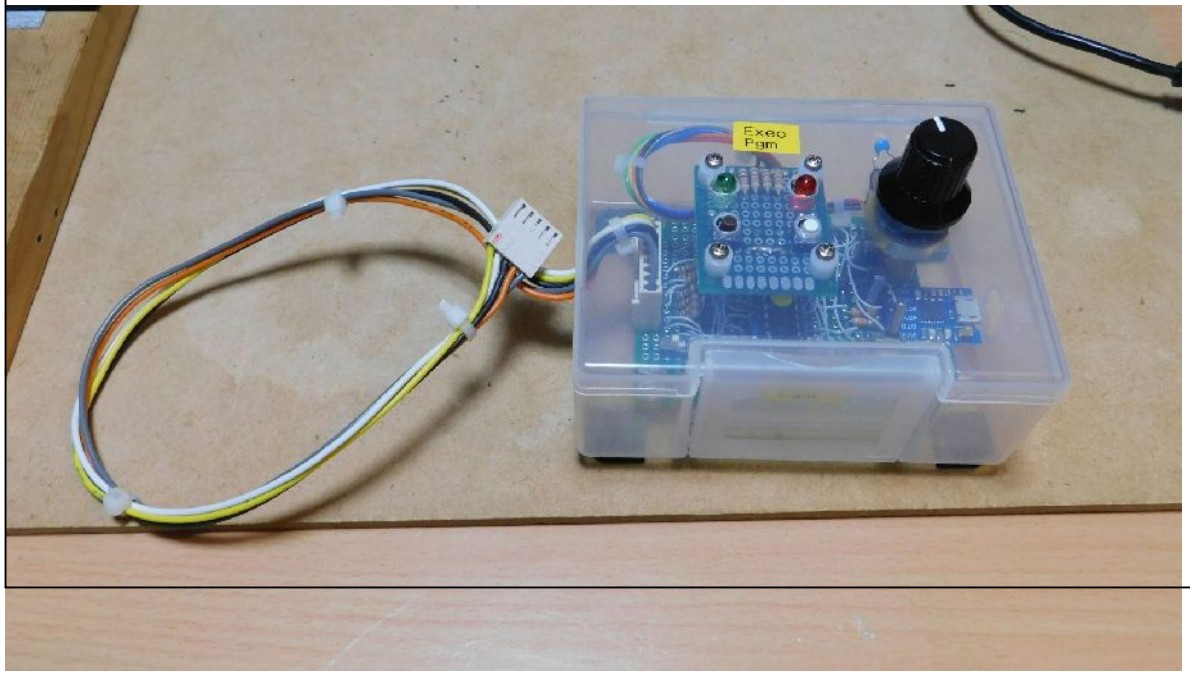




今回使用する I2Cマスタ側マイコン

今回、マスタに使うのは 036 の 動画にて作成した USB I2C 接続ユニットを 改造して使用する予定です。 上蓋に付いているボリュームと、LED、押しボタン基板は、外します。

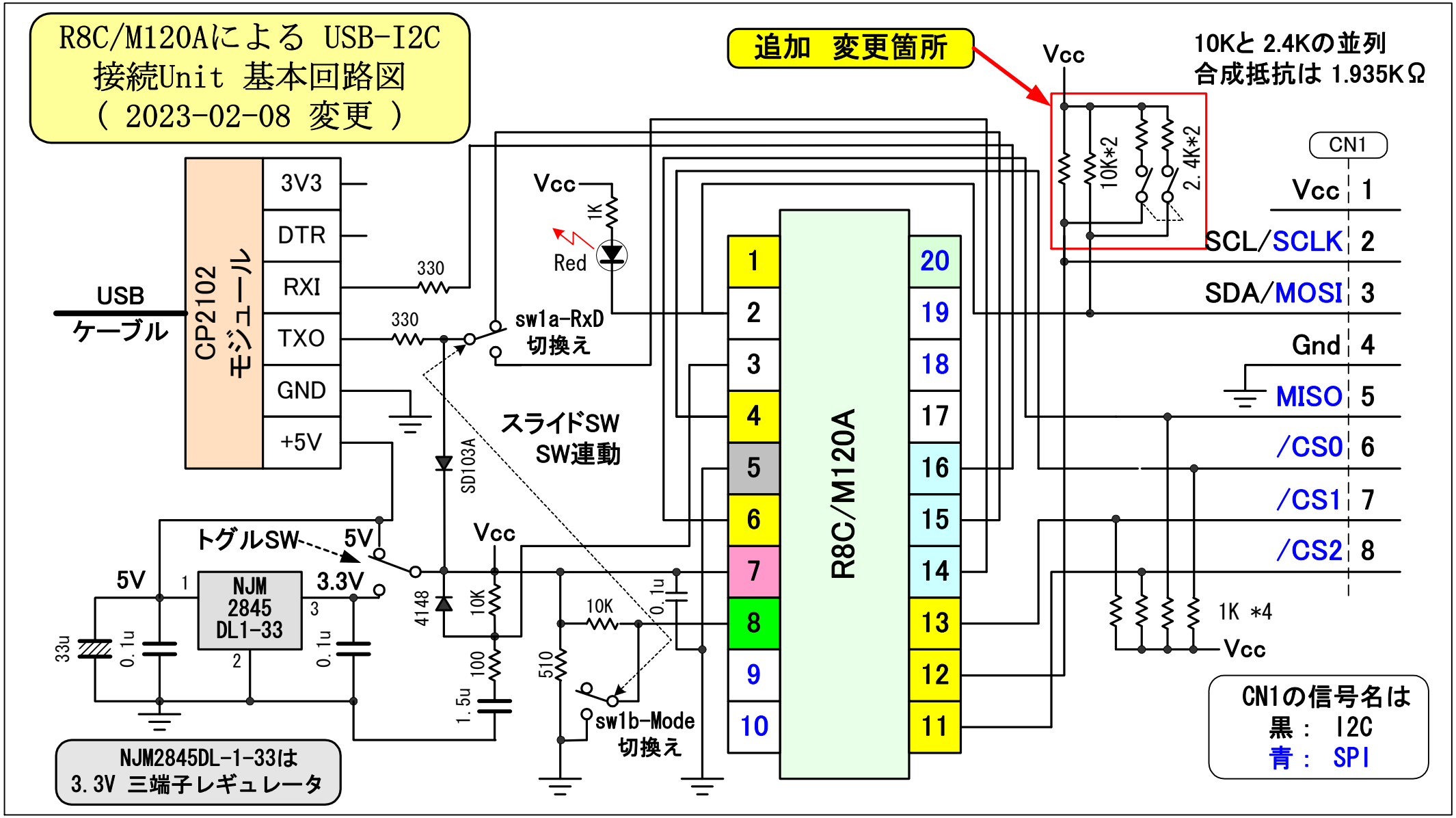
幸いに、SPI でも 使用できるように、信号線を余分に出していたので、それを /RESET信号に 割り当てる事が 出来ます。



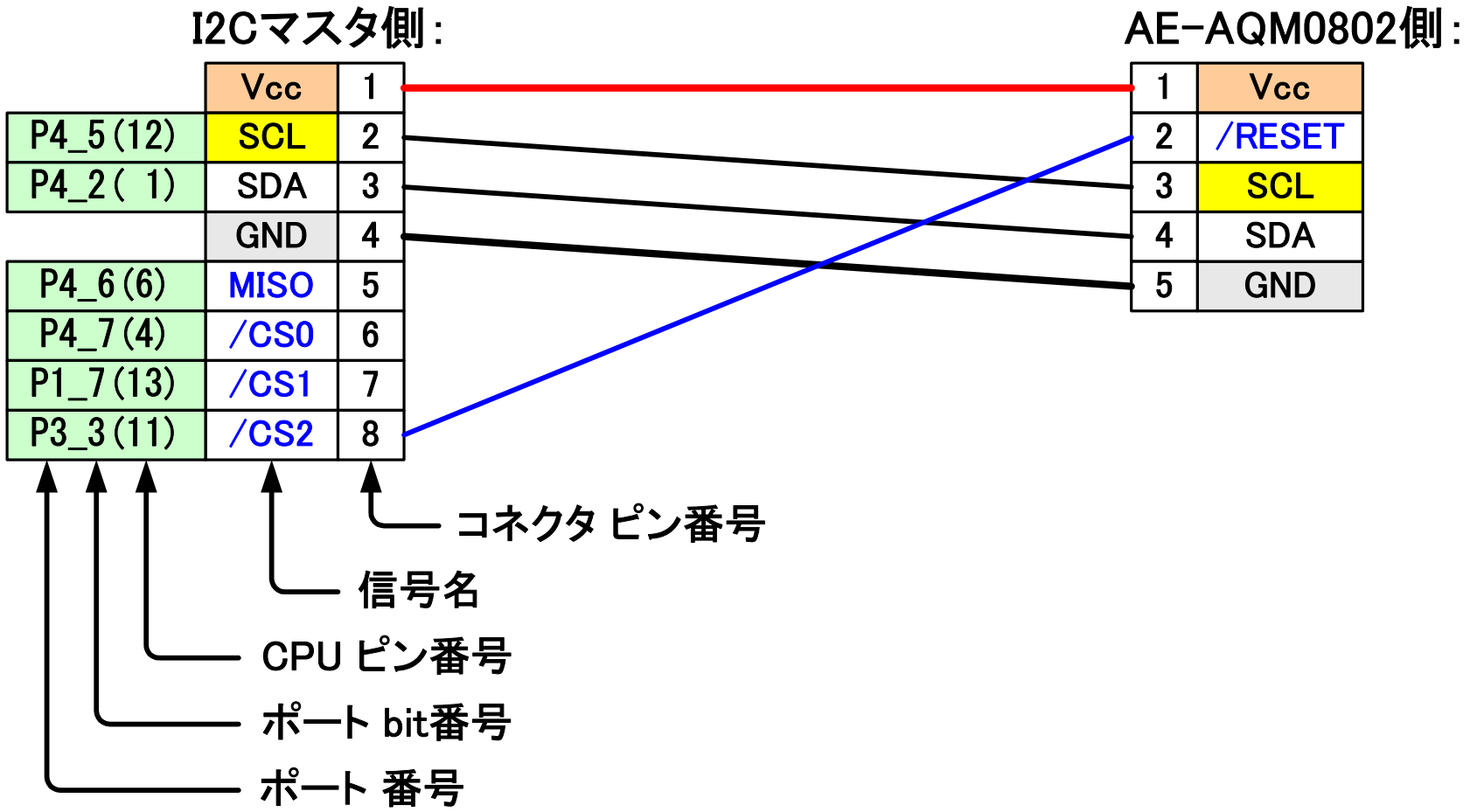
今回の 改造箇所は、元々のプルアップ抵抗が、線を長く引き出す関係もあり、1KΩに していました。 今回の LCDを使うには プルアップ抵抗を 10KΩにする必要があります。 で、場合によっては もう少し低い抵抗値にしたい場合もあると、思われるので、トグルスイッチで 10KΩと 2KΩを 切り替えられるようにします。

尚、今回の USB I2C 接続ユニットの改造作業は、申し訳ありませんが、動画撮影していません。

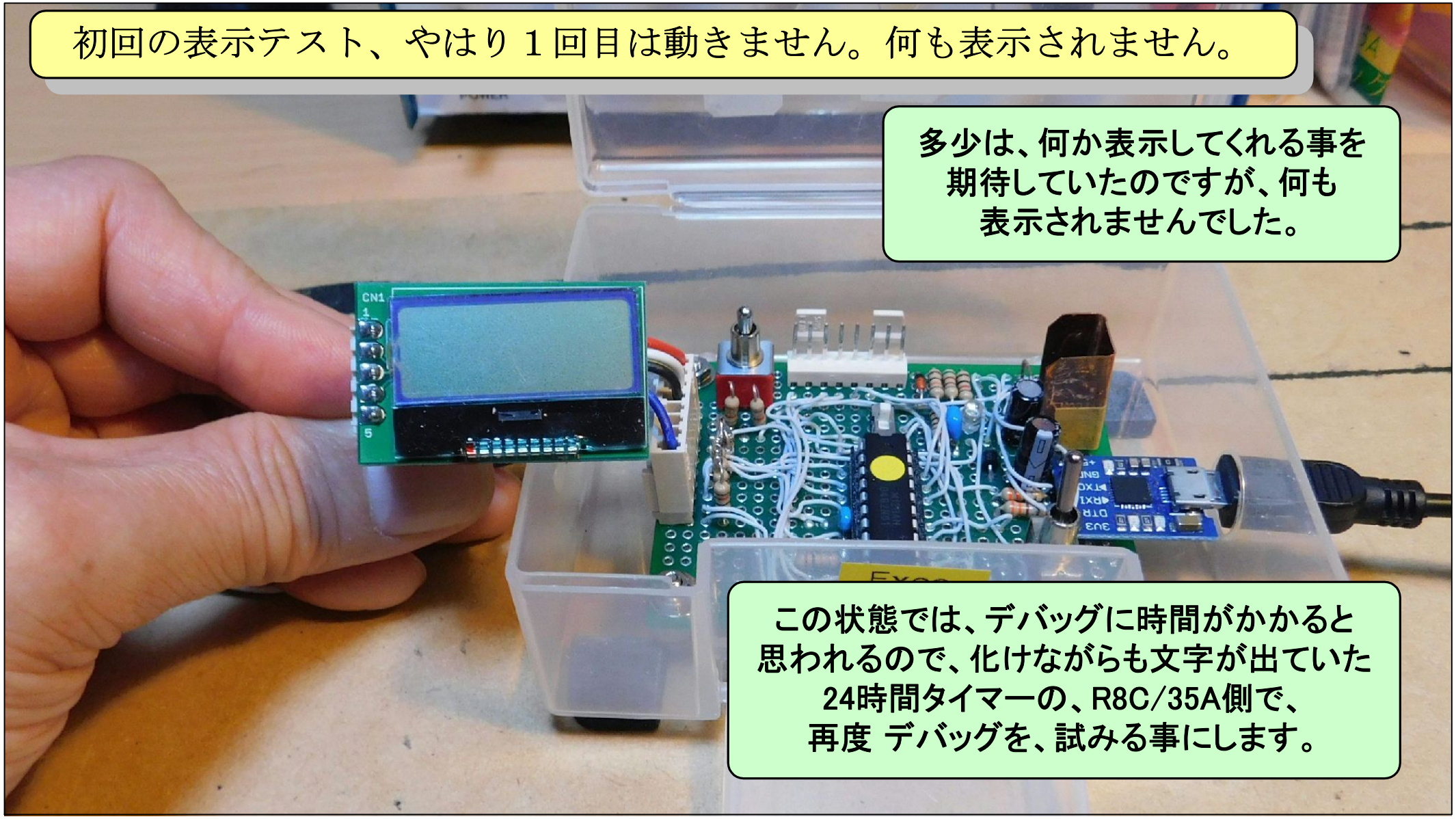
改造後の動画、及び、I2Cの動作確認の動画は、用意しています。


R8C/M120Aによる USB-I2C
接続Unit 基本回路図
(2023-02-08 変更)
追加 変更箇所
Vcc
10Kと 2.4Kの並列
合成抵抗は 1.935KΩ
10K*2
2.4K*2
CN1
Vcc 1
SCL/SCLK 2
SDA/MOSI 3
Gnd 4
MISO 5
/CS0 6
/CS1 7
/CS2 8
USB
ケーブル
CP2102
モジュール
3V3
DTR
RXI
TXO
GND
+5V
330
330
Vcc
1K
Red
sw1a-RxD
切換え
スライドSW
SW連動
SD103A
トグルSW
5V
Vcc
5V
NJM
2845
DL1-33
3.3V
33u
0.1u
0.1u
4148
10K
100
1.5u
10K
510
0.1u
sw1b-Mode
切換え
R8C/M120A
1
2
3
4
5
6
7
8
9
10
20
19
18
17
16
15
14
13
12
11
1K *4
Vcc
NJM2845DL-1-33は
3.3V 三端子レギュレータ
CN1の信号名は
黒： I2C
青： SPI

USB-I2C接続Unitと AE-AQM0802を接続するケーブル配線




USB-I2C接続Unitと AE-AQM0802を接続するケーブル


初回の表示テスト、やはり１回目は動きません。何も表示されません。
多少は、何か表示してくれる事を
期待していたのですが、何も
表示されませんでした。
この状態では、デバッグに時間がかかると
思われるので、化けながらも文字が出ていた
24時間タイマーの、R8C/35A側で、
再度 デバッグを、試みる事にします。

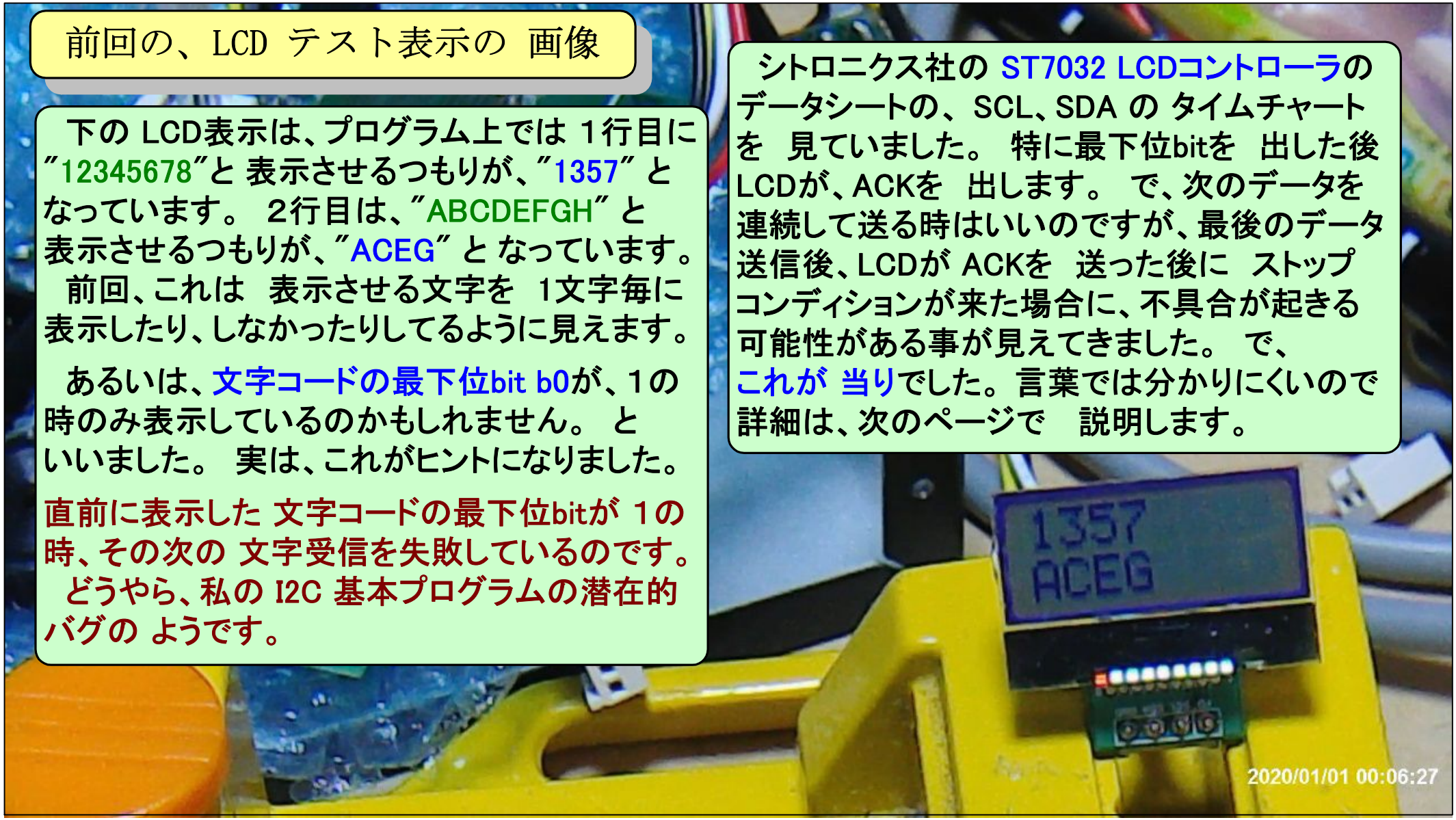
前回の、LCD テスト表示の 画像

下の LCD表示は、プログラム上では 1行目に"12345678"と 表示させるつもりが、"1357" となっています。 2行目は、"ABCDEFGH" と表示させるつもりが、"ACEG" と なっています。
前回、これは 表示させる文字を 1文字毎に表示したり、しなかったりしてるように見えます。

あるいは、文字コードの最下位bit b0が、1の時のみ表示しているのかもしれません。 といいました。 実は、これがヒントになりました。

直前に表示した 文字コードの最下位bitが 1の時、その次の 文字受信を失敗しているのです。
どうやら、私の I2C 基本プログラムの潜在的バグの ようです。

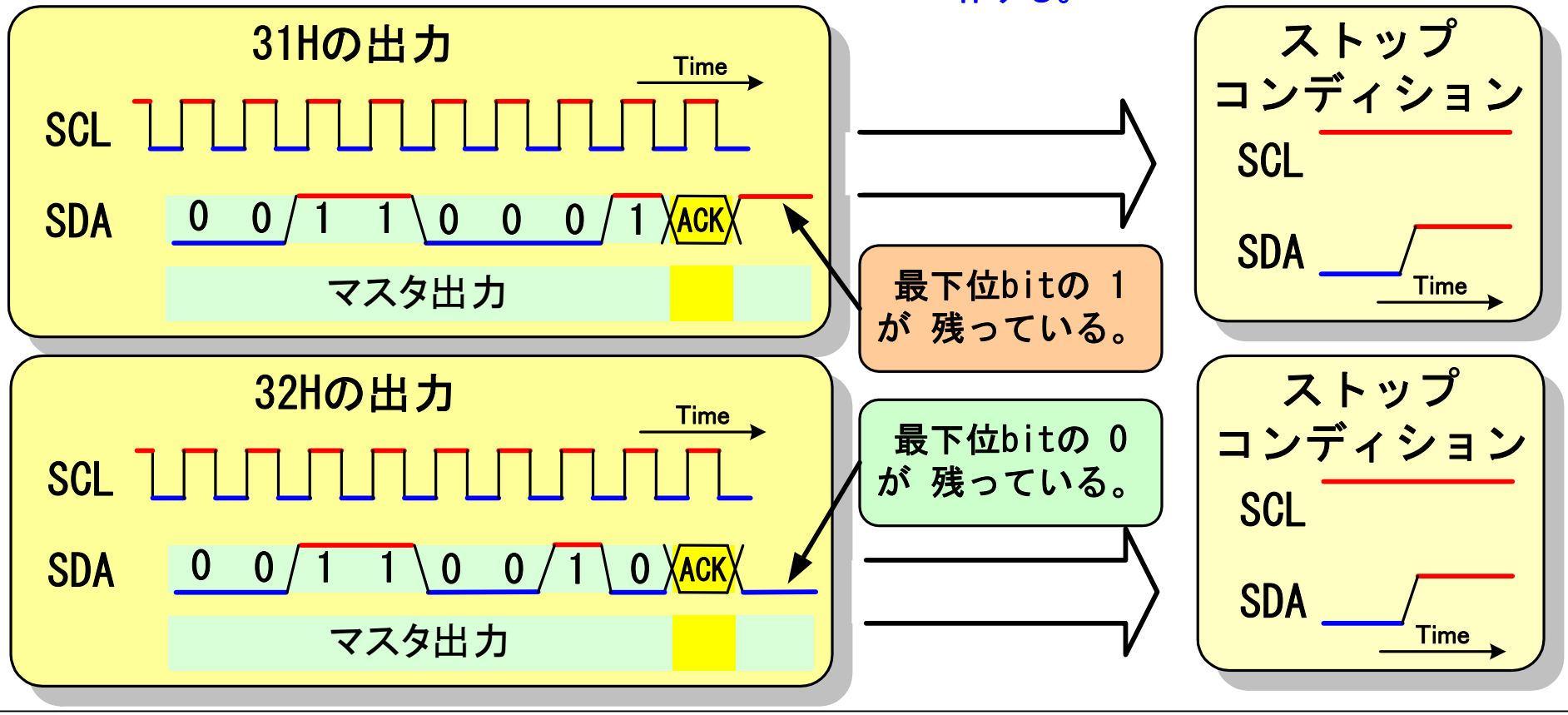
シトロニクス社の ST7032 LCDコントローラのデータシートの、SCL、SDA の タイムチャートを 見ていました。 特に最下位bitを 出した後LCDが、ACKを 出します。 で、次のデータを連続して送る時はいいのですが、最後のデータ送信後、LCDが ACKを 送った後に ストップコンディションが来た場合に、不具合が起きる可能性がある事が見えてきました。 で、これが 当りでした。言葉では分かりにくいので詳細は、次のページで 説明します。



出力データのコードに依存する不具合

文字コード '1' と '2' で、説明します。
'1' は 31Hは 2進数で 0011 0001 で、最下位bit = 1 です。
'2' は 32Hは 2進数で 0011 0010 で、最下位bit = 0 です。

31Hの場合は、ACKの後にマスタ出力に切り替えると、最下位bitの 1が 残っているため、ストップコンディションの SDAが、最初から Hiのためストップコンディションに失敗する。 最下位bitが 0 の場合は、ストップコンディションが、正常に動作する。



今回のトラブルに対する、修正箇所

```
;  スレーブの ACK/NAK 取り出し ( 400[KHz]未満、実測：376[KHz] )
;  返り値 ( ROL )：  ACK =0 ,  NAK = 80h
; ------------------------------------------------------
get_ack4    .macro
            .local   m01
            SDA_ROH             ; ROH(SDA=Hi)を P1に出力
            wait4_L             ; 短い Wait
            SCL_H               ; SCL = Hi
            SDA_IN              ; SDA (p4_2)を In Port に変更
            wait4_L             ; 短い Wait
            mov.w #0, r0        ; R0 = 0 の仮初期化
            btst  p4_2          ; p4_2 のビット判定
            jz    m01
            inc.b r0l
m01:
            SCL_L               ; SCL = Low
            wait4_L             ; ★  2023-02-08 僅かな時間待ち
            SDA_L               ; ★  2023-02-08 SDA=Low 出力に設定
            SDA_OUT             ; SDA (p4_2)を Out Port に変更
            .endm
```

左のソースは、i2c_tport_m120.incです。I2Cの I/Oポート寄りの処理をマクロ定義したファイルです。

赤の SDA_IN から SDA_OUT の間が、SDAが 入力ポートとなり、I2Cスレーブの ACKを読み込みます。 で、SDA_OUTを 行う直前で青の SDA_L を 行ってポート出力データを Lowに 設定します。
で、SDA_OUT で、SDAの信号線にLowを 出力します。

これにより、その後の ストップコンディションと、整合が、とれます。

```
void  i2c_init_lcd( void )
{
      p3_3 = 0;          // LCD AQM0802 /RESET ON ( 11pin )
      i2c_wait( 5 );     // 5ms 待つ
      p3_3 = 1;          // LCD AQM0802 /RESET OFF ( 11pin )

      i2c_set_adr( 0x3E );    // I2Cインタフェースに LCDのアドレスを設定

      i2c_wait( 100 );
      i2c_cmd_send( 0x38, 20 );
      i2c_cmd_send( 0x39, 20 );
      i2c_cmd_send( 0x14, 20 );

      i2c_cmd_send( 0x7A, 20 );      // 5V コントラスト調整
      i2c_cmd_send( 0x54, 20 );      // 5V = 0x54

//    i2c_cmd_send( 0x73, 20 );      // 3.3V コントラスト調整
//    i2c_cmd_send( 0x56, 20 );      // 3.3V = 0x54

      i2c_cmd_send( 0x6C, 20 );
      i2c_cmd_send( 0x38, 20 );
      i2c_cmd_send( 0x0C, 20 );
      i2c_cmd_send( 0x01, 30 );
}
```

左のソースは、i2c_oled_lcd.c内のLCD AQM0802の初期化処理です。 注意する必要があるのは、LCDに供給する電源電圧が、5Vか、3.3V 初期化処理のコマンドパラメータの一部が異なる事です。 赤い枠で囲ってある2ヶ所です。 コメントに 5V と付けてある2行が、5V用です。 コメントに 3.3V と 付けてある2行が 3.3V用です。 必要無い側を コメント化して下さい。

それと、上の方の青い枠のコードは、LCDに出力する /RESET信号です。 これも必要無ければ外しても動きます。

表示用テストプログラムの内容

```
void  lcd_loop_test( void )
{
      sio_print( "* Test - 0802 LCD." );          // パソコンへ シリアル出力
      i2c_init_lcd();                          // I2C LCD 初期化処理
      while( 1 )
      {
            i2c_lcd_print1( "23-02-08" );          // I2C LCD １行目 文字列表示
            i2c_lcd_print2( "01:23:45" );          // I2C LCD ２行目 文字列表示

            set_timer_10m1( 200 );             // ２秒 待ち
            while( get_timer_10m1() > 0 );

            i2c_lcd_print1( "ABCDEFGH" );          // I2C LCD １行目 文字列表示
            i2c_lcd_print2( "abcdefgh" );          // I2C LCD ２行目 文字列表示

            set_timer_10m1( 200 );             // ２秒 待ち
            while( get_timer_10m1() > 0 );
      }
}
```

では、何故 今まで OLEDや 別の I2C デバイスは、動かす事ができたのか。？

I2Cのドライバ処理内に、今まで潜在的にバグが 有ったにも関わらず、LCD以外のデバイスは、正常に動いていたのか。？
不思議に思われる方もいると 思います。

これは、あくまで 私の仮説ですが、他の I2C デバイスは、SCL = Hi 、SDA = Hi のアイドル状態が、ある程度以上の時間経過したら、I2C の インタフェース部分に アイドル状態に戻れというような、自動リセット機能を 持っているのかもしれません。 何らかの理由で I2Cバスがハングした時、マスターが I2Cバスをリセットして、アイドル状態に戻したら、スレーブ側も、アイドル状態を検知して自動的に、エラー解除して、アイドル状態に戻す機能を持っているのかも しれません。

今回の I2C LCD AQM0802は、バスがアイドル状態に戻った時に、自動的にエラー解除する機能が、無いのかも しれません。

正常に コマンドのやり取りをしている場合は問題ないですが、何らかの理由で I2Cバスがハングするような事が起こった場合、AQM0802は、エラー状態を 保持してしまうのかもしれません。 その、エラー状態を解除するために、/RESET信号を、付けたのかもしれません。

24時間タイマーマスタ側の R8C/35A においても、正常に LCD AQM0802 を アクセス出来るように なりました。

でも 今回は、LCD AQM0802のお陰で、I2C ドライバ処理の潜在的な バグ取りが、出来たので良かったです。